SISTEM MANAJEMEN DATABASE

Dr. Budi Raharjo, S.Kom., M.Kom., MM.

YAYASAN PRIMA AGUS TEKNIK

SISTEM MANAJEMEN DATABASE

Dr. Budi Raharjo, S.Kom., M.Kom., MM.

BIODATA PENULIS

Dr. Budi Raharjo, S.Kom, M.Kom, MM lahir di Semarang, tanggal 22 Februari 1985. Beliau adalah Alumni dari Universitas Bina Nusantara (BINUS University) Jakarta dan juga alumni Universitas Kristen Satya wacana (UKSW) Salatiga. Dr. Budi Raharjo telah menjadi Dosen pada Universitas STEKOM pada mata kuliah Kepemimpinan (Leadership), mata kuliah Pengantar Akuntansi, Manajemen Proses, Manajemen Akuntansi dan Manajemen Resiko Bisnis. Selain sebagai dosen Universitas STEKOM, Dr. Budi Raharjo, M.Kom, MM juga mempunyai bisnis sendiri dalam bidang perhotelan dan juga sebagai wirausaha dalam bidang pemasok unggas (ayam) beku, ke berbagai kota besar, khususnya Jakarta dan sekitarnya.

Pengalaman beliau berwirausaha menjadi bekal utama dalam penulisan buku ajar yang diterbitkan oleh Yayasan Prima Agus Teknik (YPAT) Semarang. Oleh sebab itu bukunya berisi langkah langkah praktis yang mudah diikuti oleh para mahasiswa, saat mahasiswa mengikuti proses perkuliahan pada Universitas Sains dan Teknologi Komputer (Universitas STEKOM). Jabatan struktural yang di embannya saat ini adalah Wakil Rektor 1 (Akademik) Universitas STEKOM Semarang.

YAYASAN PRIMA AGUS TEKNIK

YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-5734-08-8 (PDF)

9 786235 734088

SISTEM
MANAJEMEN DATABASE

Dr. Budi Raharjo, S.Kom., M.Kom., MM.

YAYASAN PRIMA AGUS TEKNIK

SISTEM MANAJEMEN DATABASE

Penulis :
Dr. Budi Raharjo, S.Kom., M.Kom., MM.

ISBN : 9 786235 734088

Editor :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Penyunting :
Dr. Joseph Teguh Santoso, M.Kom.

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom

Penebit :
Yayasan Prima Agus Teknik

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

Hak cipta dilindungi undang-undang
Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara apapun tanpa ijin dari penulis

KATA PENGANTAR

Puji syukur kami panjatkan atas kehadirat Tuhan karena buku yang berjudul "Sistem Manejemen Database" dapat terselesaikan. Informasi merupakan hal yang sangat berharga, informasi dapat disimpan berupa data, yang nantinya akan diolah didalam sebuah Sistem. *Database Magement System* atau yang disingkat DBMS merupakan perangkat lunak yang dibangun untuk mengelola data dalam jumlah yang sangat besar. DBMS juga dirancang untuk memanipulasi data secara lebih mudah. Penyimpanan data berupa *database* memiliki banyak keunggulan dibanding penyimpanan data berupa *Flatfile*, antara lain perfoma yang lebih baik, Integritas, Independensi, Sentralisasi dan Security yang akan dibahas mendetail dalam buku ini.

Dalam buku ini juga akan dijelaskan tentang pengolahan database, seperti membuat tabel, bentuk – bentuk normalisasi dalam *database*, dan lainnya. Perumpamaan DBMS dapat berupa *Database Hierarcy*, *Data Network*, Dan *Data Relational*, yang nantinya akan dijelaskan dalam buku ini. Pada saat sekarang ini Administrasi Database Manajement System selain dilakukan sengan bahasa SQL, yang nantinya akan dijelaskan dalam buku ini.

Buku ini juga memberikan tentang cara mengolah database agar mudah diaplikasikan kedalam beberapa Aplikasi secara langsung. Konkurensi atau bisa disebut dengan Sistem database, memiliki 2 teknik utama yang mengijinkan transaksi untuk berjalan dengan aman dalam subjek paralel untuk *constraint* tertentu, yaitu *locking* dan metode *timestamp* tertentu. *Locking* dan *timestamping* adalah teknik konservatif yang menyebabkan transaksi ditunda dalam kasus mereka konflik dengan transaksi lain dalam selang waktu sesuai dengan arus transaksi yang terjadi. Metode optimistik, juga akan dibahas dalam buku ini. Pengendalian dalam sistem Database memiliki beberapa Protokol / Aturan antara lain Protokol Berbasis penguncian (*Lock*), Protokol Berbasis – Pembatasan waktu, Protokol berbasis Validasi dan Penanganan *Deadlock*. Buku ini juga menyediakan latihan soal juga disertakan jawaban yang diletakkan pada bagian akhir buku ini untuk memperjelas pemahaman bagi para mahasiswa. Diharapkan buku ini dapat berguna bagi mahasiswa dalam memahami tentang *Database Management System.*

Semarang, November 2021
Penulis

Dr. Budi Raharjo, S.Kom., M.Kom., MM.

DAFTAR ISI

BAB 1
SISTEM MANAJEMEN DATABASE

1.1 DATA

Data adalah fakta mentah atau terisolasi dari mana informasi yang dibutuhkan dihasilkan. Data adalah potongan informasi yang berbeda, biasanya diformat dengan cara khusus.

Tabel 1.1 Data Potongan Informasi

Dalam Fikiran Pemberi Pekerjaan	Dalam Pandangan Karyawan Penjualan
Identitas Nama Karyawan Departemen Tempat Tanggal Lahir Kualifikasi	Nama_Pelanggan Rekening_Pelanggan Alamat Kota Nomor_Telepon Negara

Arsitektur Data Tiga Lapisan

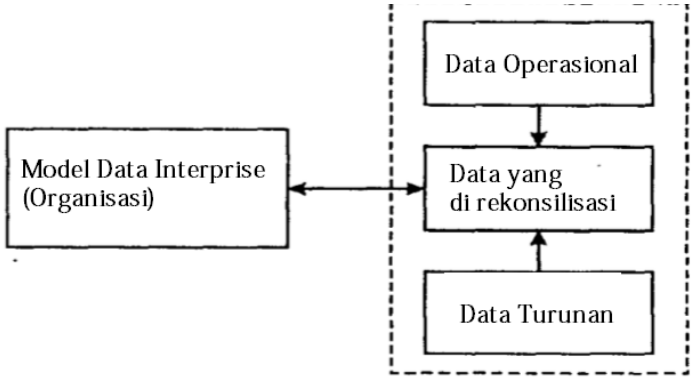
Data diatur dalam struktur berlapis berikut:

- Data Operasiona
- Data yang Direkonsiliasi
- Data Turunan

Data operasional: Data operasional disimpan di berbagai sistem operasi di seluruh sistem organisasi (baik internal maupun eksternal).

Data yang direkonsiliasi*:* Data yang direkonsiliasi disimpan di Data warehouse organisasi dan di penyimpanan data operasional. Mereka adalah data terperinci dan terkini, yang dimaksudkan sebagai sumber tunggal dan otoritatif untuk semua aplikasi pendukung keputusan.

Data Turunan: Data turunan disimpan di masing-masing data mart. Data yang diturunkan dipilih, diformat dan dikumpulkan untuk aplikasi pendukung keputusan pengguna akhir.



Gambar 1.1 Arsitektur Data 3 lapisan.

1.2 INFORMASI

Data dan informasi terkait erat dan sering digunakan secara bergantian. Informasi adalah data yang diproses, diorganisasikan atau diringkas. Ini dapat didefinisikan sebagai kumpulan data terkait yang ketika disatukan, mengkomunikasikan pesan yang bermakna dan berguna kepada penerima yang menggunakannya, untuk membuat keputusan atau untuk menafsirkan data untuk mendapatkan makna.

Data diproses untuk menciptakan informasi yang berarti bagi penerimanya. Misalnya, dari sudut pandang tenaga penjual, kita mungkin ingin mengetahui saldo saat ini dari pelanggan Mis Waterhouse Ltd. atau mungkin kita mungkin meminta saldo rata-rata saat ini dari semua pelanggan di Indonesia. Jawaban atas pertanyaan semacam itu adalah informasi. Dengan demikian, informasi melibatkan komunikasi dan penerimaan pengetahuan atau kecerdasan. Ini mengurangi ketidakpastian mengungkapkan alternatif tambahan atau membantu menghilangkan yang tidak relevan atau buruk, mempengaruhi individu dan mensimulasikannya ke dalam tindakan.

1.3 DATA WAREHOUSE

Data warehouse adalah kumpulan data yang dirancang untuk mendukung manajemen dalam proses pengambilan keputusan. Ini adalah kumpulan data yang berorientasi pada subjek, terintegrasi, bervariasi waktu, tidak dapat didata, yang digunakan untuk mendukung proses pengambilan keputusan manajemen dan intelijen bisnis. Ini berisi berbagai macam data yang menyajikan gambaran yang koheren tentang kondisi bisnis pada satu titik waktu. Ini adalah jenis database unik yang berfokus pada intelijen bisnis, data eksternal, dan data varian waktu. Pergudangan data adalah proses, di mana organisasi mengekstrak makna dan pengambilan keputusan informasi dari aset informasi mereka melalui penggunaan Data warehouse.

1.4 *DATA DICTIONARY* (KAMUS DATA)

Data Dictionay adalah sistem manajemen Database mini yang mengelola metadata. Ini adalah gudang informasi tentang database yang mendokumentasikan elemen data dari database. Kamus data merupakan bagian integral dari sistem manajemen database dan menyimpan metadata atau informasi tentang database, nama atribut dan definisi untuk setiap tabel dalam database. Kamus data biasanya merupakan bagian dari katalog sistem yang dihasilkan untuk setiap database. Sistem kamus data yang berguna biasanya menyimpan dan mengelola jenis informasi berikut:

- Deskripsi skema database.
- Informasi rinci tentang desain database fisik, seperti struktur penyimpanan, jalur akses dan ukuran file dan catatan.
- Deskripsi pengguna database, tanggung jawab dan hak akses mereka.
- Deskripsi tingkat tinggi dari transaksi & aplikasi database dan hubungan pengguna dengan terjemahan.

- Hubungan antara transaksi database dan item data yang direferensikan oleh mereka. Ini berguna dalam menentukan transaksi mana yang terpengaruh ketika definisi data tertentu diubah.

Catatan

Catatan adalah kumpulan bidang atau item data yang terkait secara logis, dengan setiap bidang memproses sejumlah byte dan memiliki tipe data tetap. Sebuah record terdiri dari nilai-nilai untuk setiap field. Pengelompokan item data dapat dicapai melalui cara yang berbeda untuk membentuk catatan yang berbeda untuk tujuan yang berbeda. Catatan ini diambil atau diperbarui menggunakan program.

1.5 FILE

File adalah kumpulan dari urutan record yang saling berhubungan. Dalam banyak kasus, semua record dalam sebuah file memiliki tipe record yang sama (setiap record memiliki format yang sama). Jika setiap record dalam file memiliki ukuran yang sama persis dalam byte, file tersebut dikatakan terdiri dari record dengan panjang tetap. Jika record yang berbeda dalam file memiliki ukuran yang berbeda, file tersebut dikatakan dibuat dari record dengan panjang variabel.

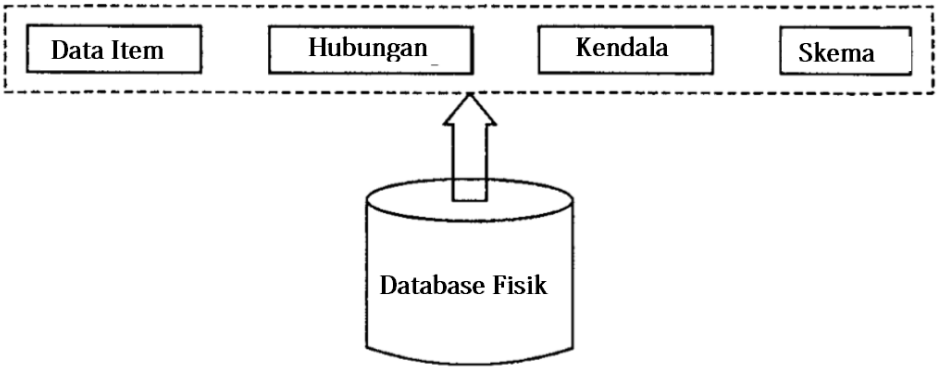
1.6 DATABASE

Database didefinisikan sebagai kumpulan data yang terkait secara logis yang disimpan bersama yang dirancang untuk memenuhi kebutuhan informasi suatu organisasi. Database lebih lanjut dapat didefinisikan sebagai, itu:

- adalah kumpulan data yang saling terkait yang disimpan bersama tanpa redundansi yang berbahaya atau tidak perlu.
- melayani banyak aplikasi di mana setiap pengguna memiliki pandangannya sendiri tentang data. Data ini dilindungi dari akses yang tidak sah oleh mekanisme keamanan dan akses bersamaan ke data disediakan dengan mekanisme pemulihan.
- menyimpan data independen dari program dan perubahan dalam struktur penyimpanan data atau strategi akses tidak memerlukan perubahan dalam mengakses program atau kueri.

Sebuah database terdiri dari empat komponen berikut seperti yang ditunjukkan pada gambar.

- Data item
- Hubungan
- Kendala
- Skema



Gambar 1.2 Database Fisik

1.7 SISTEM MANAJEMEN DATABASE

Sistem Manajemen Database adalah kumpulan data yang saling terkait dan serangkaian program untuk mengakses data tersebut.

Aplikasi pada Sistem Database

Berikut ini adalah aplikasi database.

- Perbankan
- Maskapai penerbangan
- Universitas
- Kereta Api
- Keuangan
- Penjualan
- Telekomunikasi
- Sistem penggajian
- Manufaktur
- Sumber daya manusia

Fungsi dan Pelayanan DBMS

Sebuah DBMS melakukan beberapa fungsi penting yang menjamin integritas dan konsistensi data dalam database.

1. ***Data Storage Management/Manajemen Penyimpanan Data***: DBMS menciptakan struktur kompleks yang diperlukan untuk penyimpanan data dalam database fisik. Ini menyediakan mekanisme untuk pengelolaan penyimpanan permanen data.
2. ***Manajemen Transaksi:*** Transaksi adalah serangkaian operasi Database, yang dilakukan oleh program aplikasi, yang mengakses atau mengubah isi Database. Oleh karena itu, DBMS harus menyediakan mekanisme untuk memastikan bahwa semua pembaruan yang terkait dengan transaksi tertentu dilakukan atau tidak ada satu pun yang dibuat.
3. ***Layanan Integritas***: Integritas database mengacu pada kebenaran dan konsistensi data yang disimpan dan sangat penting dalam sistem database berorientasi transaksi. Oleh karena itu, DBMS harus menyediakan untuk memastikan bahwa baik data dalam database dan perubahan data mengikuti aturan tertentu. Ini meminimalkan

redundansi data dan memaksimalkan konsistensi data. Hubungan data yang disimpan dalam kamus data digunakan untuk menegakkan integritas data. Berbagai jenis mekanisme dan batasan integritas dapat didukung untuk membantu memastikan bahwa nilai data dalam database valid, bahwa operasi yang dilakukan pada nilai tersebut valid dan database tetap dalam keadaan konsisten.

4. ***Manajemen Pencadangan dan Pemulihan***: DBMS menyediakan mekanisme untuk berbagai jenis kegagalan. Ini mencegah hilangnya data. Mekanisme pemulihan DBMS, memastikan bahwa database dikembalikan ke keadaan yang konsisten setelah transaksi gagal atau dibatalkan karena sistem crash, kegagalan media, kesalahan perangkat keras atau perangkat lunak, kegagalan daya, dan sebagainya.
5. ***Concurrency Control Services:*** Karena DBMS mendukung berbagi data di antara banyak pengguna, mereka harus menyediakan mekanisme untuk mengelola akses bersamaan ke database. DBMS memastikan bahwa database disimpan dalam keadaan yang konsisten dan integritas data dipertahankan. Ini memastikan bahwa database diperbarui dengan benar ketika banyak pengguna memperbarui database secara bersamaan.
6. ***Data Manipulation Management/Manajemen Manipulasi Data:*** DBMS melengkapi pengguna dengan kemampuan untuk mengambil, memperbarui dan menghapus data yang ada dalam database atau untuk menambahkan data baru ke database. Ini termasuk komponen prosesor DML untuk menangani bahasa manipulasi data (DML).
7. ***Data Dictionary/Manajemen Katalog Sistem:*** DBMS menyediakan kamus data atau fungsi katalog sistem di mana deskripsi item data disimpan dan dapat diakses oleh pengguna. Katalog sistem atau kamus data adalah Database sistem, yang merupakan tempat penyimpanan informasi yang menjelaskan data dalam Database. Ini adalah data tentang data atau metadata. Misalnya, DBMS akan berkonsultasi dengan katalog sistem untuk memverifikasi bahwa tabel yang diminta ada dan pengguna yang mengeluarkan permintaan memiliki hak akses yang diperlukan.
8. ***Otorisasi/Manajemen Keamanan*** : DBMS melindungi database terhadap akses yang tidak sah, baik disengaja atau tidak disengaja. Ini melengkapi mekanisme untuk memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses database. Ini menciptakan sistem keamanan yang memberlakukan keamanan pengguna dan privasi data dalam database. Aturan keamanan menentukan pengguna mana yang dapat mengakses database, item data mana yang dapat diakses setiap pengguna, dan operasi data mana (menambah, menghapus, dan memodifikasi) yang dapat dilakukan pengguna.
9. ***Utility Services/Layanan Utilitas:*** DBMS menyediakan satu set layanan utilitas yang digunakan oleh DBA dan perancang Database untuk membuat, mengimplementasikan, memantau, dan memelihara Database. Layanan utilitas ini membantu DBA untuk mengelola database secara efektif.
10. ***Akses Database dan Antarmuka Pemrograman Aplikasi***: Semua DBMS menyediakan antarmuka untuk memungkinkan aplikasi menggunakan layanan DBMS. Mereka

menyediakan akses data melalui bahasa kueri terstruktur (SQL). Bahasa query DBMS berisi dua komponen:

a. Bahasa definisi data*/data definition language* (DDL)
b. Bahasa manipulasi data*/data manipulation language* (DML)

mendefinisikan struktur di mana data disimpan dan DML memungkinkan pengguna akhir untuk mengekstrak data dari database. DBMS juga menyediakan akses data ke pemrogram aplikasi melalui bahasa prosedural seperti C, C++, Java dan lain-lain.

11. ***Data Independence Services/Layanan Independensi Data:*** DBMS harus mendukung independensi program dari struktur database yang sebenarnya.
12. ***Data Definition Services/Layanan Definisi Data:*** DBMS menerima definisi data seperti skema eksternal, skema konseptual, skema internal, dan semua pemetaan terkait dalam bentuk sumber. Ini mengubahnya ke bentuk objek yang sesuai menggunakan komponen prosesor DDL untuk masing-masing dari berbagai bahasa definisi data (DDL).

Database versus Sistem File

File adalah urutan record.

- Semua record dalam sebuah file memiliki tipe record yang sama.
- Sistem pemrosesan file didukung oleh sistem operasi konvensional. Sistem menyimpan catatan permanen dalam berbagai file, dan memerlukan program aplikasi yang berbeda untuk mengekstrak catatan dari file yang sesuai dan menambahkan catatan ke file yang sesuai.

Kelebihan dari Sistem Pemrosesan File

Meskipun sistem pemrosesan file sekarang sebagian besar sudah usang, berikut adalah keuntungan dari sistem pemrosesan file.

- Ini memberikan perspektif sejarah yang berguna tentang cara menangani data.
- Karakteristik sistem berbasis file membantu dalam pemahaman keseluruhan kompleksitas desain sistem database.
- Memahami masalah dan pengetahuan tentang batasan yang melekat pada sistem berbasis file membantu menghindari masalah yang sama saat merancang sistem Database dan dengan demikian menghasilkan transisi yang mulus.

Kekurangan Sistem Pemrosesan File

1. ***Upaya Pemrograman Berlebihan*** : Program aplikasi baru sering kali membutuhkan kumpulan definisi file yang sama sekali baru. Meskipun file yang ada mungkin berisi beberapa data yang diperlukan, aplikasi seringkali membutuhkan sejumlah item data lainnya. Akibatnya, programmer harus mengkode ulang definisi item data yang dibutuhkan dari file yang ada serta definisi semua item data baru. Jadi dalam sistem berorientasi file, ada saling ketergantungan yang berat antara program dan data.
2. ***Inkonsistensi Data:*** Redundansi Data juga menyebabkan inkonsistensi data, karena format data mungkin tidak konsisten atau nilai data mungkin tidak lagi sesuai atau keduanya.
3. ***Berbagi data terbatas*** : Ada peluang berbagi data terbatas dengan sistem berorientasi file tradisional. Setiap aplikasi memiliki file pribadinya sendiri dan

pengguna memiliki sedikit kesempatan untuk berbagi data di luar aplikasi mereka sendiri. Untuk mendapatkan data dari beberapa file yang tidak kompatibel dalam sistem yang terpisah akan membutuhkan upaya pemrograman yang besar.

4. ***Kontrol data yang buruk:*** Sistem berorientasi file bersifat desentralisasi, tidak ada kontrol terpusat pada tingkat elemen data (bidang). Bisa jadi sangat umum untuk bidang data memiliki beberapa nama yang ditentukan oleh berbagai departemen organisasi dan tergantung pada file yang ada di dalamnya. Hal ini dapat menyebabkan arti yang berbeda dari data yang diajukan dalam konteks yang berbeda, dan sebaliknya, arti yang sama untuk bidang yang berbeda. Hal ini menyebabkan kontrol data yang buruk, mengakibatkan kebingungan besar.
5. ***Kemampuan manipulasi data yang tidak memadai:*** Karena sistem berorientasi file tidak menyediakan koneksi yang kuat antara data dalam file yang berbeda dan oleh karena itu kemampuan manipulasi datanya sangat terbatas.
6. ***Redundansi Data (atau duplikasi):*** Aplikasi dikembangkan secara independen dalam sistem pemrosesan file yang mengarah ke file duplikat yang tidak direncanakan. Duplikasi adalah pemborosan karena membutuhkan ruang penyimpanan tambahan dan perubahan dalam satu file harus dilakukan secara manual di semua file. Hal ini juga mengakibatkan hilangnya integritas data. Ada juga kemungkinan bahwa item data yang sama mungkin memiliki nama yang berbeda dalam file yang berbeda, atau nama yang sama dapat digunakan untuk item data yang berbeda dalam file yang berbeda.
7. ***Masalah atomisitas:*** Atomisitas berarti semua operasi transaksi tercermin dalam database atau tidak ada sama sekali, yaitu, jika semuanya bekerja dengan benar tanpa kesalahan, maka semuanya akan dikomit ke database. Jika ada bagian dari transaksi yang gagal, seluruh transaksi akan dibatalkan. Transfer dana harus atomik - itu harus terjadi secara keseluruhan atau tidak sama sekali. Sulit untuk memastikan atomisitas dalam sistem pemrosesan file konvensional.
8. ***Masalah keamanan :*** Masalah keamanan dalam pemrosesan file adalah orang yang tidak berwenang dapat mengambil, mengubah, menghapus, atau memasukkan data ke dalam sistem file. Tapi ini tidak mungkin di DBMS.
9. ***Masalah integritas :*** Nilai data yang disimpan dalam database harus memenuhi jenis kendala konsistensi tertentu. Pengembang memberlakukan batasan ini dalam sistem dengan menambahkan kode yang sesuai di berbagai program aplikasi. Ketika kendala baru ditambahkan, sulit untuk mengubah program untuk menegakkannya. Masalahnya diperparah ketika kendala melibatkan beberapa item data untuk file yang berbeda.
10. ***Ketergantungan Data Program :*** Deskripsi file (struktur fisik, penyimpanan file data dan catatan) didefinisikan dalam setiap program aplikasi yang mengakses file tertentu.
11. ***Isolasi data :*** Karena data tersebar di berbagai file, dan file mungkin dalam format yang berbeda, menulis program aplikasi baru untuk mengambil data yang sesuai menjadi sulit.

12. ***Kesulitan dalam mengakses data:*** Lingkungan pemrosesan file konvensional tidak memungkinkan data yang dibutuhkan diambil dengan cara yang nyaman dan efisien seperti DBMS. Sistem pengambilan data yang lebih baik harus dikembangkan untuk penggunaan umum.
13. ***Anomali akses serentak :*** Untuk meningkatkan kinerja sistem secara keseluruhan dan memperoleh waktu respons yang lebih cepat, banyak sistem memungkinkan banyak pengguna untuk memperbarui data secara bersamaan. Dalam lingkungan seperti itu, interaksi pembaruan bersamaan dapat menghasilkan data yang tidak konsisten.

Kelebihan DBMS

1. ***Mengontrol redundansi data:*** Redundansi data, menyimpan data yang sama beberapa kali menyebabkan beberapa masalah. Pertama, ruang penyimpanan terbuang ketika data yang sama disimpan berulang kali. Kedua, file yang mewakili data yang sama mungkin menjadi tidak konsisten. Ini mungkin terjadi karena pembaruan diterapkan ke beberapa file tetapi tidak untuk yang lain. Sebagian besar DBMS menyediakan fasilitas untuk mengontrol redundansi data menggunakan konsep normalisasi dan kunci.
2. ***Membatasi akses yang tidak sah:*** Ketika beberapa pengguna mengakses database, oleh karena itu beberapa pengguna tidak akan diizinkan untuk mengakses (!. Semua informasi dalam database. DBMS harus menyediakan subsistem keamanan dan otorisasi, yang ditentukan oleh Data Base Administor (DBA). DBMS kemudian harus memberlakukan pembatasan ini secara otomatis.Misalnya, data Perbankan sering dianggap rahasia, dan karenanya hanya orang yang berwenang yang diizinkan untuk mengakses data tersebut.
3. ***Menyediakan backup dan Recovery :*** Sebuah DBMS harus menyediakan fasilitas untuk recovery dari hardware atau software. Subsistem pencadangan dan pemulihan DBMS bertanggung jawab atas pemulihan. Misalnya, jika sistem komputer gagal di tengah program pembaruan yang kompleks, subsistem pemulihan bertanggung jawab dan memastikan bahwa database dipulihkan ke keadaan sebelum program mulai dijalankan. Alternatifnya, subsistem pemulihan memastikan bahwa program dilanjutkan dari titik di mana ia terganggu sehingga efek penuhnya dicatat dalam database.
4. ***Menyediakan Beberapa Antarmuka*** :Pengguna Karena banyak jenis pengguna dengan berbagai tingkat pengetahuan teknis menggunakan database, DBMS harus menyediakan berbagai antarmuka pengguna. Ini termasuk bahasa kueri untuk pengguna biasa. Antarmuka bahasa pemrograman untuk pemrogram aplikasi, formulir dan kode perintah untuk pengguna parameter dan antarmuka pengguna grafis untuk pengguna yang berdiri sendiri.
5. ***Inforcing Integritas kendala :*** Integritas data berarti bahwa data yang terkandung dalam database akurat dan konsisten. Integritas berarti batasan, yaitu aturan konsistensi yang tidak boleh dilanggar oleh sistem Database. Sebagian besar aplikasi database memiliki batasan integritas tertentu yang harus dimiliki untuk data. Sebuah DBMS harus menyediakan kemampuan untuk mendefinisikan dan menegakkan

kendala ini. Jenis batasan integritas paling sederhana yang menentukan tipe data untuk setiap item data.

6. ***Akses data yang efisien:*** DBMS menggunakan berbagai teknik canggih untuk menyimpan dan mengambil data secara efisien. Fitur ini sangat penting jika data disimpan di perangkat penyimpanan eksternal.
7. ***Peningkatan berbagi data: Karena, sistem Database adalah penyimpanan data terpusat milik seluruh organisasi,*** sistem Database dapat dibagikan oleh semua pengguna yang berwenang. Program aplikasi yang ada dapat berbagi data dalam database. Selanjutnya, program aplikasi baru dapat dikembangkan pada data yang ada di database untuk berbagi data yang sama dan hanya menambahkan data yang saat ini tidak disimpan. Oleh karena itu, lebih banyak pengguna dan aplikasi dapat berbagi lebih banyak data.
8. ***Peningkatan keamanan:*** Keamanan Database adalah perlindungan Database dari pengguna yang tidak berwenang. Database administrator (DBA) memastikan bahwa prosedur akses yang tepat diikuti, termasuk skema otentikasi yang tepat untuk akses ke DBMS dan pemeriksaan tambahan sebelum mengizinkan akses ke data sensitif. DBA dapat menentukan nama pengguna dan kata sandi untuk mengidentifikasi orang yang berwenang menggunakan database.
9. ***Konsistensi data yang ditingkatkan:*** Jika redundansi dihapus atau dikendalikan, kemungkinan data yang tidak konsisten juga dihapus dan dikendalikan. Dalam sistem database, inkonsistensi seperti itu dihindari sampai batas tertentu dengan membuat mereka tahu ke DBMS. DBMS memastikan bahwa setiap perubahan yang dibuat ke salah satu dari dua entri dalam database secara otomatis diterapkan ke yang lain juga. Proses ini dikenal sebagai menyebarkan pembaruan.
10. ***Kemandirian data program:*** Dalam lingkungan Database, memungkinkan perubahan pada satu tingkat Database tanpa mempengaruhi tingkat lainnya. Perubahan ini diserap oleh pemetaan antar level dengan pendekatan database, metedata disimpan di lokasi sentral yang disebut repository. Properti sistem data ini memungkinkan data organisasi berubah tanpa mengubah program aplikasi yang memproses data.
11. ***Peningkatan kualitas data:*** Sistem database menyediakan sejumlah alat dan proses untuk meningkatkan kualitas data.
12. ***Menyediakan penyimpanan persisten untuk objek program dan struktur data:*** Database dapat digunakan untuk menyediakan penyimpanan persisten untuk objek program dan struktur data. Ini adalah salah satu alasan utama untuk sistem database berorientasi objek. Penyimpanan persisten dari objek program dan struktur data merupakan fungsi penting dari sistem database.
13. ***Mewakili hubungan yang kompleks antara data:*** Sebuah database dapat mencakup berbagai jenis data yang saling terkait dalam banyak cara. Sebuah DBMS harus memiliki kemampuan untuk mewakili berbagai hubungan yang kompleks antara data serta untuk mengambil dan memperbarui data dengan mudah dan efisien.
14. ***Mengizinkan inferensi dan tindakan menggunakan aturan:*** Beberapa sistem database menyediakan kemampuan untuk mendefinisikan aturan duduksi untuk

menyimpulkan informasi baru dari fakta database yang disimpan. Sistem seperti ini disebut sistem Database deduktif. Fungsionalitas yang lebih kuat disediakan oleh sistem database aktif, yang menyediakan aturan aktif yang dapat secara otomatis memulai tindakan ketika peristiwa dan kondisi tertentu terjadi.

15. ***Ketersediaan informasi terkini untuk semua pengguna:*** DBMS membuat database tersedia untuk semua pengguna. Segera setelah pembaruan satu pengguna diterapkan ke database, semua pengguna lain dapat segera melihat pembaruan ini. Ketersediaan informasi terkini ini sangat penting untuk banyak aplikasi pemrosesan transaksi, seperti sistem reservasi, database perbankan, dan dimungkinkan oleh subsistem kontrol konkurensi dan pemulihan DBMS.
16. ***Fleksibilitas:*** Mungkin perlu untuk mengubah struktur database sebagai perubahan persyaratan. DBMS modern memungkinkan jenis perubahan evolusioner tertentu pada struktur database tanpa mempengaruhi data yang disimpan dan program aplikasi yang ada.
17. ***Peningkatan konkurensi:*** DBMS mengelola akses database konkuren dan mencegah masalah hilangnya informasi atau hilangnya integritas.
18. ***Keseimbangan persyaratan yang saling bertentangan*** : DBA menyelesaikan persyaratan yang saling bertentangan dari berbagai pengguna dan aplikasi. DBA dapat menyusun sistem untuk menyediakan layanan keseluruhan yang terbaik bagi organisasi. DBA dapat memilih struktur file dan metode akses terbaik untuk mendapatkan kinerja optimal untuk operasi respons-kritis, sementara mengizinkan aplikasi yang kurang penting untuk terus menggunakan database.

Kekurangan DBMS

Berikut ini kelemahan DBMS.

1. ***Kompleksitas Pencadangan dan Pemulihan*** : Agar Database bersama terpusat menjadi akurat dan tersedia setiap saat, prosedur yang komprehensif diperlukan untuk dikembangkan dan digunakan untuk menyediakan salinan cadangan data dan untuk memulihkan Database ketika terjadi kerusakan. DBMS modern biasanya mengotomatiskan lebih banyak tugas pencadangan dan pemulihan daripada sistem berorientasi file.
2. ***Peningkatan biaya instalasi dan manajemen:*** Perangkat lunak DBMS yang besar dan kompleks memiliki biaya awal yang tinggi. Ini membutuhkan tenaga terlatih untuk menginstal dan mengoperasikan dan juga memiliki biaya pemeliharaan dan dukungan tahunan yang besar. Perangkat lunak Database tambahan mungkin diperlukan untuk memberikan keamanan dan untuk memastikan pemutakhiran data bersama yang tepat secara bersamaan.
3. ***Biaya perangkat keras tambahan:*** Biaya instalasi DBMS sangat bervariasi, tergantung pada lingkungan dan fungsionalitas, ukuran perangkat keras dan biaya pemeliharaan perangkat keras dan perangkat lunak tahunan yang berulang.
4. ***Kebutuhan tenaga kerja baru dan khusus*** : Karena perubahan yang cepat dalam teknologi database dan kebutuhan bisnis organisasi, kebutuhan organisasi untuk mempekerjakan, melatih kembali tenaganya secara teratur untuk merancang dan

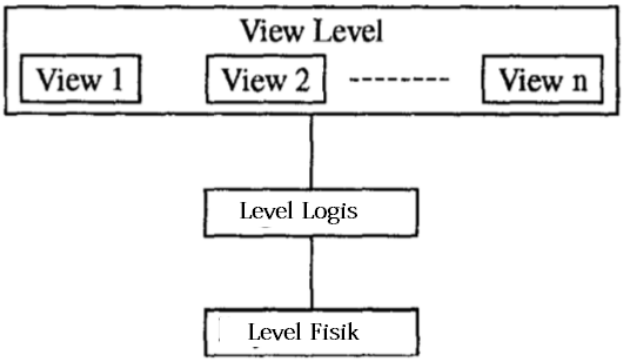
mengimplementasikan database, menyediakan layanan administrasi database dan mengelola staf orang baru. Oleh karena itu, organisasi perlu mempertahankan tenaga terampil khusus.

5. ***Peningkatan kompleksitas:*** DBMS multi-pengguna menjadi perangkat lunak yang sangat kompleks karena fungsionalitas yang diharapkan darinya. Menjadi penting bagi perancang Database, pengembang, administrator Database, dan pengguna akhir untuk memahami fungsionalitas ini secara maksimal.
6. ***Masalah yang terkait dengan sentralisasi***: Sentralisasi berarti bahwa data dapat diakses dari satu sumber yang disebut database.
7. ***Ukuran DBMS yang besar:*** Kompleksitas yang besar dan fungsionalitas yang luas membuat DBMS menjadi perangkat lunak yang sangat besar. Ini menempati banyak gigabyte ruang disk penyimpanan dan membutuhkan sejumlah besar memori utama untuk berjalan secara efisien.

1.8 ABSTRAKSI DATA

Ada tiga tingkat abstraksi data.

1. **Level Fisik:** Level fisik dari abstraksi data menggambarkan bagaimana data sebenarnya disimpan.
2. **Level Logis** : Level logis dari abstraksi data menggambarkan data "apa" yang disimpan dalam database dan hubungan apa yang ada di antara data tersebut.
   Jadi tingkat logis menggambarkan seluruh database.
   Administrator Database, yang harus memutuskan informasi apa yang akan disimpan dalam Database, menggunakan tingkat abstraksi logis.



Gambar 1.3 Tingkat Abstraksi Data

3. **Level tampilan:** Level tampilan abstraksi data hanya menjelaskan sebagian dari keseluruhan database. Tingkat tampilan abstraksi menyederhanakan interaksi mereka dengan sistem. Sistem dapat menyediakan banyak tampilan untuk database yang sama.

***View of Data*:** Sebuah sistem database adalah kumpulan file terintegrasi dan satu set program yang memungkinkan pengguna untuk mengakses dan memodifikasi file-file ini.

1.9 INSTANCES DAN SKEMA

Instances: Kumpulan informasi yang disimpan dalam database pada saat tertentu disebut instance dari database.

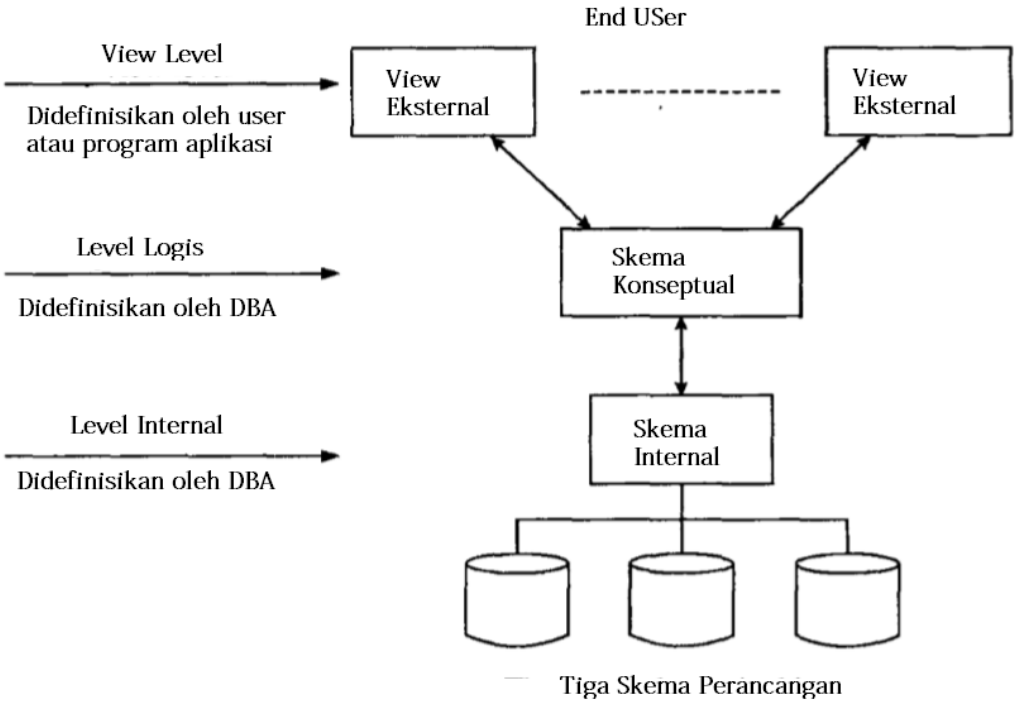
Skema: Desain keseluruhan database disebut skema database.

Artinya, deskripsi database disebut skema database yang ditentukan selama desain database dan diharapkan tidak sering berubah. Skema database dapat dipartisi sesuai dengan tingkat abstraksi.

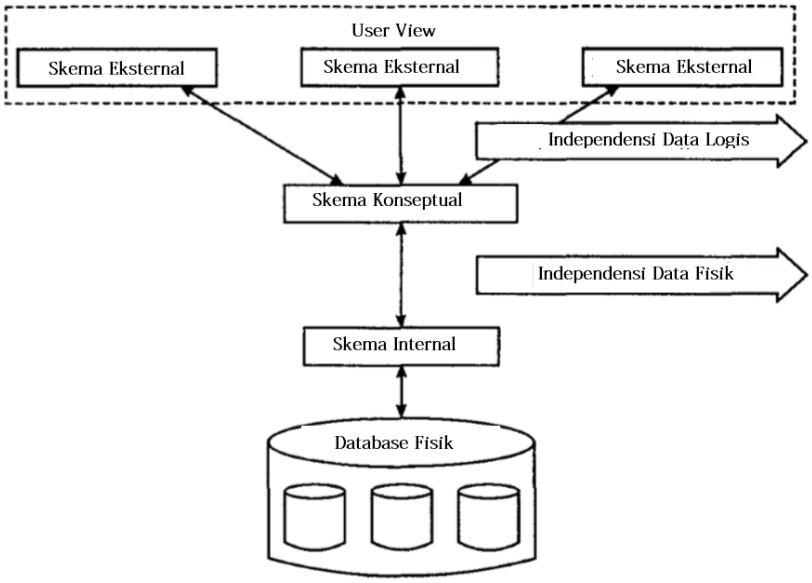
a. **Skema Fisik atau Internal:** Skema fisik, menjelaskan struktur penyimpanan fisik database. Tingkat internal berkaitan dengan kegiatan berikut:
   1. Alokasi ruang penyimpanan untuk data dan penyimpanan.
   2. Rekam deskripsi untuk penyimpanan dengan ukuran yang disimpan untuk item data.
   3. Penempatan Rekam.
   4. Teknik kompresi data dan enkripsi data.

   Skema ini menggunakan model data fisik dan menjelaskan detail lengkap penyimpanan data dan jalur akses untuk database.

b. **Skema konseptual atau logis:** Skema logis menggambarkan struktur seluruh database untuk komunitas pengguna. Skema konseptual menggambarkan entitas, tipe data, hubungan, operasi pengguna, dan batasan serta menyembunyikan detail struktur penyimpanan fisik.



Gambar 1.4 Skema Database



Gambar 1.5 Skema Database

Tingkat konseptual berkaitan dengan aktivitas berikut:

1. Semua entitas, atributnya, dan hubungannya.
2. Kendala pada data.
3. Informasi semantik tentang data.
4. Informasi keamanan
5. Pemeriksaan untuk menjaga konsistensi dan integritas data.

c. **Skema Eksternal:** Setiap skema eksternal menjelaskan bagian dari database yang diminati oleh kelompok pengguna tertentu dan menyembunyikan sisa database dari kelompok pengguna tersebut.

1.10 INDEPENDENSI DATA

Kemampuan untuk mengubah skema pada satu tingkat sistem Database tanpa harus mengubah skema pada tingkat berikutnya yang lebih tinggi disebut "Independensi Data". Ada dua jenis independensi data:

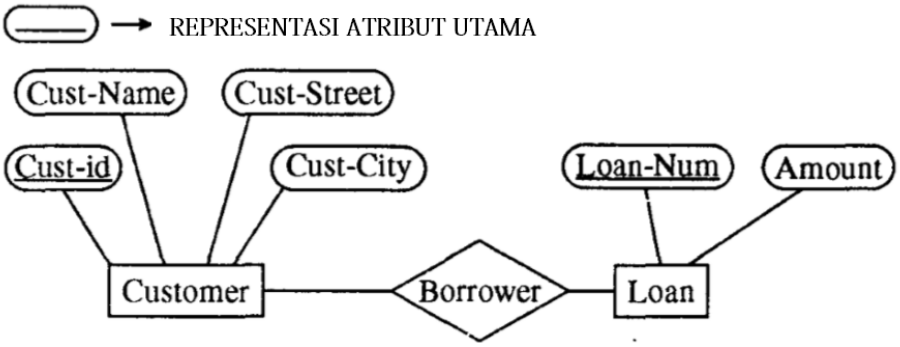
1. **Independensi Data Fisik:** Independensi data fisik adalah kemampuan untuk mengubah skema internal tanpa harus mengubah skema konseptual. *Contoh* : Dengan membuat struktur akses tambahan untuk meningkatkan kinerja pengambilan atau pembaruan.
2. **Logical Data Independence :** Logical Data Independence adalah kemampuan untuk mengubah skema konseptual tanpa harus mengubah program aplikasi (skema eksternal). *Contoh* : Kita dapat mengubah skema konseptual untuk memperluas database dengan menambahkan tipe record atau item data. ATAU untuk mengurangi database dengan menghapus item data.

Model Entity-Relationship (E-R)

Model data E-R didasarkan pada persepsi dunia nyata yang terdiri dari kumpulan objek dasar yang disebut entitas dan hubungan di antara objek-objek tersebut. Keseluruhan

struktur logika database dapat direpresentasikan secara grafis dengan diagram ER. Diagram E-R dibangun dari komponen-komponen berikut.

- **Rectangle**: yang mewakili himpunan entitas
- **Elips**: yang mewakili atribut
- **Diamond**: yang mewakili hubungan antara himpunan entitas.
- **Line**: yang menghubungkan atribut ke kumpulan entitas dan kumpulan entitas ke hubungan.
- **Double Ellips** : yang mewakili atribut multinilai.
- **Dashed Ellipses** : yang menunjukkan atribut turunan.
- **Double Line :** yang mewakili total partisipasi suatu entitas dalam himpunan hubungan. Dobel
- **Rectangle** : yang mewakili himpunan entitas yang lemah.



Gambar 1.6 Representasi Atribut Utama

Kelebihan

1. **Representasi relasional lurus ke depan:** Setelah merancang diagram ER untuk aplikasi database, representasi relasional dari model database menjadi relatif lurus ke depan.
2. **Konversi Mudah untuk ER ke model data lain:** Konversi dari diagram ER ke jaringan atau model data hierarkis dapat dengan mudah dilakukan.
3. **Representasi grafis** untuk pemahaman yang lebih baik.

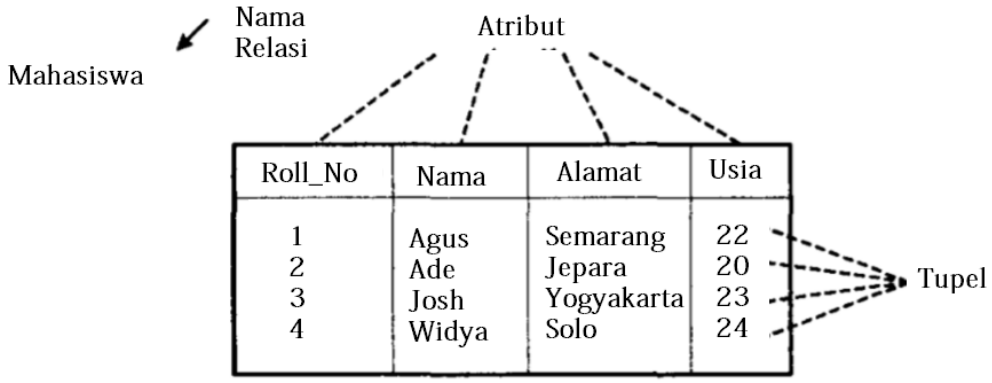
Kekurangan

1. **Tidak ada standar industri untuk notasi:** Tidak ada notasi standar industri untuk mengembangkan diagram ER.
2. **Populer untuk desain tingkat tinggi:** Model data ER adalah desain database tingkat tinggi.

Model Relasional

Model relasional menggunakan kumpulan tabel untuk mewakili data dan hubungan di antara data tersebut. Tabel adalah kumpulan baris dan kolom. Setiap kolom memiliki nama yang unik. Setiap baris dalam tabel mewakili kumpulan nilai data terkait. Dalam model relasional, baris disebut tuple, header kolom disebut atribut dan tabel disebut relasi.

Tabel 1.2 Model Relasional



Kelebihan Model Relasional

1. **Kesederhanaan**: Model data relasional bahkan lebih sederhana dari model hierarkis dan jaringan. Ini membebaskan desainer dari detail penyimpanan data fisik yang sebenarnya, sehingga memungkinkan mereka untuk berkonsentrasi pada tampilan logis dari database.
2. **Independensi struktural :** Model data relasional tidak bergantung pada sistem akses data navigasi. Perubahan struktur database tidak mempengaruhi akses data.
3. Kemudahan desain, implementasi, pemeliharaan, dan penggunaan.
4. Kemampuan kueri yang fleksibel dan kuat.

Kekurangan

1. **Overhead perangkat keras:** Model data relasional membutuhkan perangkat keras komputasi dan perangkat penyimpanan data yang lebih kuat untuk melakukan tugas yang ditetapkan RDBMS.
2. Mudah untuk merancang kemampuan yang mengarah ke desain yang buruk.

Model Data Berorientasi Objek

Model data berorientasi objek IS didasarkan pada paradigma bahasa pemrograman berorientasi objek. Paradigma berorientasi objek didasarkan pada Enkapsulasi data dan kode yang terkait dengan suatu objek menjadi satu kesatuan, pewarisan dan identitas objek.

Contoh:

```
Kelas karyawan
        {
        nama string;
        alamat string;
        int gaji;
        int gaji tahunan ( );
        };
```

Kelebihan Model Data Berorientasi Objek

1. **Akses data yang ditingkatkan:** Model data berorientasi objek mewakili hubungan secara eksplisit, mendukung akses navigasi dan asosiatif ke informasi. Ini meningkatkan kinerja akses data.
2. **Peningkatan produktivitas:** Ini menyediakan fitur-fitur canggih seperti pewarisan, polimorfisme, dan pengikatan dinamis yang memungkinkan pengguna untuk membuat objek dan memberikan solusi tanpa menulis kode khusus objek. Fitur ini meningkatkan produktivitas para pengembang aplikasi database secara signifikan.
3. Menggabungkan pemrograman berorientasi objek dengan teknologi database.
4. Mampu menangani berbagai macam tipe data.

Kekurangan

1. **Tidak ada definisi yang tepat:** Sulit untuk memberikan definisi yang tepat tentang apa yang terdapat pada DBMS berorientasi objek.
2. **Sulit untuk dipertahankan :** Definisi objek perlu diubah secara berkala dan migrasi database yang ada agar sesuai dengan definisi objek baru dengan perubahan kebutuhan informasi organisasi.
3. **Tidak cocok untuk semua aplikasi:** Model data berorientasi objek digunakan jika ada kebutuhan untuk mengelola hubungan kompleks di antara objek data.
4. **Overhead perangkat keras:** Model data relasional membutuhkan perangkat keras komputasi dan penyimpanan data yang lebih kuat.

Model Data Obyek-Relasional

Model data relasional objek, menggabungkan fitur model rasional dan berorientasi objek. Model ini menyediakan sistem tipe kaya database berorientasi objek, menggabungkan dengan hubungan sebagai dasar untuk penyimpanan data. Sistem Database relasional objek menyediakan jalur migrasi yang mulus bagi pengguna Database relasional yang ingin menggunakan fitur berorientasi objek.

Model Hirarki

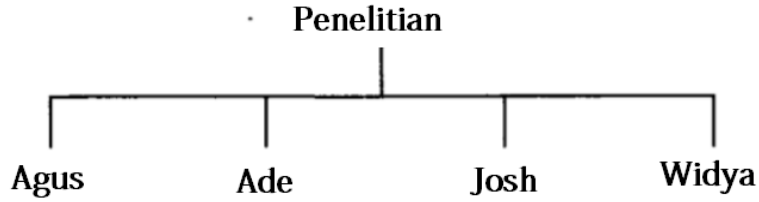
Model hierarki menyediakan dua konsep penataan data utama.

- Catatan
- Hubungan orang tua-anak.

Catatan : Catatan adalah kumpulan nilai bidang yang memberikan informasi tentang entitas atau instance hubungan. ***Relasi Parent-child :*** Tipe relasi parent-child adalah relasi 1 : N antara dua tipe record.

- Tipe record pada sisi 1 disebut tipe parent reecprd dan tipe record pada sisi N disebut tipe record anak tipe PCR
- Instance tipe PCR terdiri dari satu record dari tipe record induk dan sejumlah record (nol atau lebih) dari tipe record anak.

Misalnya, Departemen :



Gambar 1.7 Contoh Instance

Keuntungan Model Data Hirarki

Berikut adalah keuntungan dari model data hierarkis.

1. **Kesederhanaan:** Hubungan antara berbagai lapisan secara logis sederhana dan desain Database hierarkis sederhana.
2. **Berbagi data :** Karena semua data disimpan dalam database yang sama, berbagi data menjadi praktis. .
3. **Keamanan data: Ini** adalah model database pertama yang menawarkan keamanan data yang disediakan dan diterapkan oleh DBMS.
4. **Integritas data:** Dalam hubungan induk/anak, selalu ada tautan antara segmen induk dan segmen turunannya di bawahnya.
5. **Efficiency:** Model ini sangat efisien ketika database berisi volume data yang besar dalam hubungan one-lo-many (1:m) dan ketika pengguna membutuhkan sejumlah besar transaksi.

Independensi data: DBMS menciptakan lingkungan di mana independensi data dapat dipertahankan.

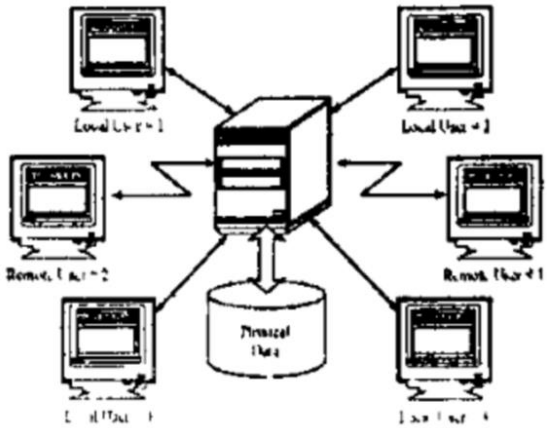
Kekurangan

1. **Kompleksitas implementasi:** Meskipun Database hierarkis secara konseptual sederhana, mudah dirancang dan tidak ada masalah ketergantungan data, namun cukup rumit untuk diterapkan.
2. **Batasan implementasi :** Banyak dari hubungan umum tidak mengkonfirmasi format hubungan satu-ke-banyak yang diperlukan oleh model Database hierarkis.
3. **Tidak fleksibel:** Database hierarkis tidak memiliki fleksibilitas. Perubahan dalam relasi atau segmen baru sering kali menghasilkan tugas manajemen sistem yang sangat kompleks.
4. Kurangnya independensi struktural.
5. Kompleksitas pemrograman aplikasi.

Model Data Jaringan

Data dalam model Jaringan mewakili kumpulan catatan dan hubungan antar data diwakili oleh tautan, yang dapat dilihat sebagai petunjuk. Catatan dalam database diatur sebagai kumpulan grafik arbitrer.

Sebagai contoh,



Gambar 1.8 Model Data Jaringan

Keuntungan Model Data Jaringan

- **Kesederhanaan:** Mirip dengan model data hierarkis, model jaringan juga sederhana dan mudah dirancang.
- **Akses data superior:** Akses dan fleksibilitas data lebih unggul daripada yang ditemukan dalam model data hierarkis.
- **Memfasilitasi lebih banyak jenis hubungan :** Model jaringan memfasilitasi dalam menangani hubungan satu-ke-banyak (1:m) dan banyak-ke-banyak (n:m), yang membantu dalam pemodelan situasi kehidupan nyata.
- **Integritas Database:** Model jaringan menegakkan integritas Database dan tidak mengizinkan anggota ada tanpa pemilik
- **Independensi data:** Model data jaringan menyediakan independensi data yang cukup dengan setidaknya mengisolasi sebagian program dari detail penyimpanan fisik yang kompleks.

Kekurangan

- **Kompleksitas sistem:** Karena model jaringan menyediakan mekanisme akses navigasi ke data di mana data mengakses satu catatan pada satu waktu. Mekanisme ini membuat implementasi sistem menjadi sangat kompleks.
- **Tidak adanya independensi struktural:** Sulit untuk membuat perubahan dalam database jaringan.
- **Tidak ramah pengguna:** Model data jaringan bukanlah desain untuk sistem yang mudah digunakan dan merupakan sistem yang sangat berorientasi pada keterampilan.
- Penggunaan pointer mengarah ke struktur yang kompleks, yang membuat pemetaan data terkait menjadi sangat sulit.

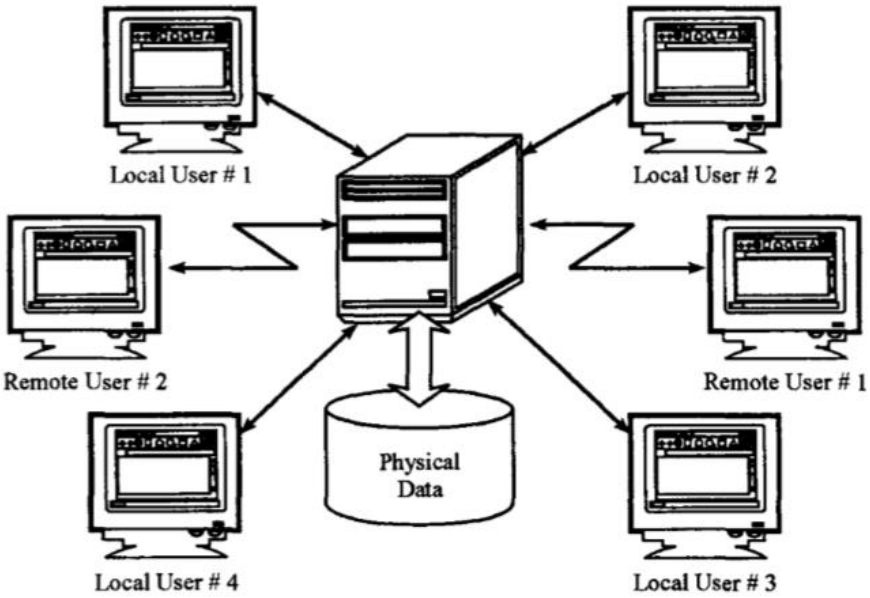
1.11 JENIS SISTEM DATABASE

DMBS dapat diklasifikasikan menurut jumlah pengguna, lokasi situs database.

1. Berdasarkan jumlah pengguna:
    - DBMS pengguna tunggal

- DBMS multi-pengguna

2. Berdasarkan lokasi situs:
   - DBMS Terpusat
   - DBMS Terdistribusi
   - DBMS paralel
   - DBMS Client/Server



Gambar 1.9 DMBS

Sistem Database Terpusat

Sistem database terpusat terdiri dari satu prosesor bersama dengan perangkat penyimpanan data terkait dan periferal lainnya. Secara fisik terbatas pada satu lokasi. Pengelolaan sistem dan datanya dikendalikan secara terpusat dari siapa pun atau situs pusat. Sebuah DBMS terpusat jika data disimpan di satu situs komputer. Sebuah DBMS terpusat dapat mendukung banyak pengguna, tetapi DBMS dan database itu sendiri berada sepenuhnya di satu situs komputer.

Kelebihan Sistem Database Terpusat :

- Sebagian besar fungsi seperti pembaruan, pencadangan, kueri, akses kontrol, dan sebagainya, lebih mudah dilakukan dalam sistem Database terpusat.
- Ukuran Database dan komputer tempat ia berada tidak perlu menentukan apakah Database terletak di pusat.

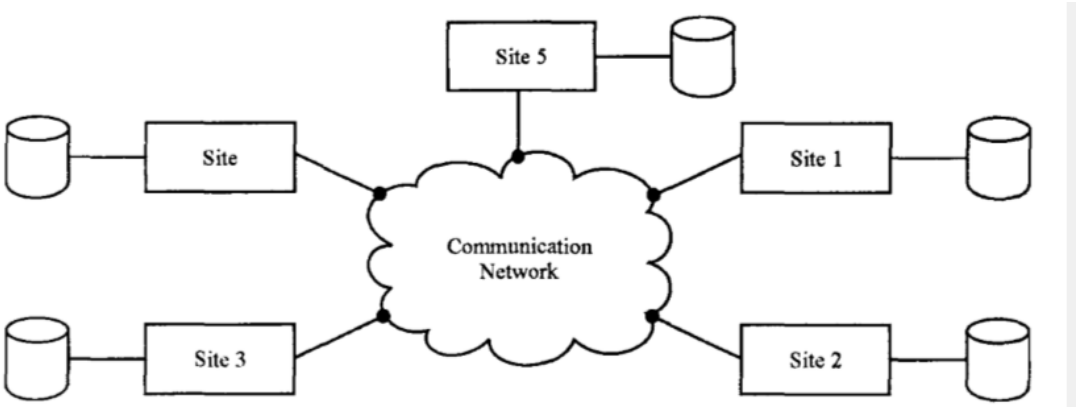
Kekurangan *Sistem Database Terpusat :*

- Ketika komputer situs pusat atau sistem Database mati, maka setiap pengguna diblokir dari menggunakan sistem hingga sistem kembali.
- Biaya komunikasi dari terminal ke lokasi pusat bisa mahal.

Sistem Database Terdistribusi

Database terdistribusi adalah kumpulan beberapa database yang saling berhubungan secara logis yang didistribusikan melalui Jaringan Komputer. Sistem manajemen Database terdistribusi adalah sistem perangkat lunak yang mengelola Database terdistribusi sambil

membuat distribusi transparan kepada pengguna. Dalam sistem database terdistribusi, data didistribusikan di berbagai database yang berbeda.



Gambar 1.10 Jaringan Sistem Database Terdistribusi

Ini dikelola oleh berbagai perangkat lunak DBMS yang berbeda yang berjalan pada berbagai mesin komputasi yang berbeda yang didukung oleh berbagai sistem operasi yang berbeda. Mesin-mesin ini didistribusikan secara geografis dan dihubungkan bersama oleh berbagai jaringan komunikasi. Dalam sistem database terdistribusi, satu aplikasi dapat beroperasi pada data yang didistribusikan secara geografis pada mesin yang berbeda. Jadi, dalam sistem Database terdistribusi, data mungkin didistribusikan pada komputer yang berbeda sedemikian rupa sehingga data untuk satu bagian disimpan di satu komputer dan data untuk bagian lain disimpan di komputer lain. Setiap mesin dapat memiliki data dan aplikasinya sendiri. Namun, pengguna di satu komputer dapat mengakses data yang disimpan di serveral komputer lain. Oleh karena itu, setiap mesin akan bertindak sebagai server untuk beberapa pengguna dan klien untuk yang lain. Detil lebih lanjut tentang sistem database terdistribusi diberikan di Unit-So

Keuntungan Sistem Database Terdistribusi

1. Pengelolaan data terdistribusi dengan tingkat transparansi yang berbeda.
2. Peningkatan Keandalan dan Ketersediaan.
3. Pemrosesan kueri terdistribusi.
4. Peningkatan kinerja.
5. Peningkatan skalabilitas
6. Evaluasi paralel.
7. Pemulihan Database terdistribusi.
8. Manajemen Data yang Direplikasi
9. Keamanan
10. Transparansi jaringan.
11. Ini memberikan efisiensi yang lebih besar dan kinerja yang lebih baik.
12. Transparansi replikasi.

Kekurangan Sistem Database Terdistribusi

1. Masalah teknis menghubungkan mesin yang berbeda.
2. Biaya dan kompleksitas perangkat lunak.
3. Kesulitan dalam kontrol integritas data.

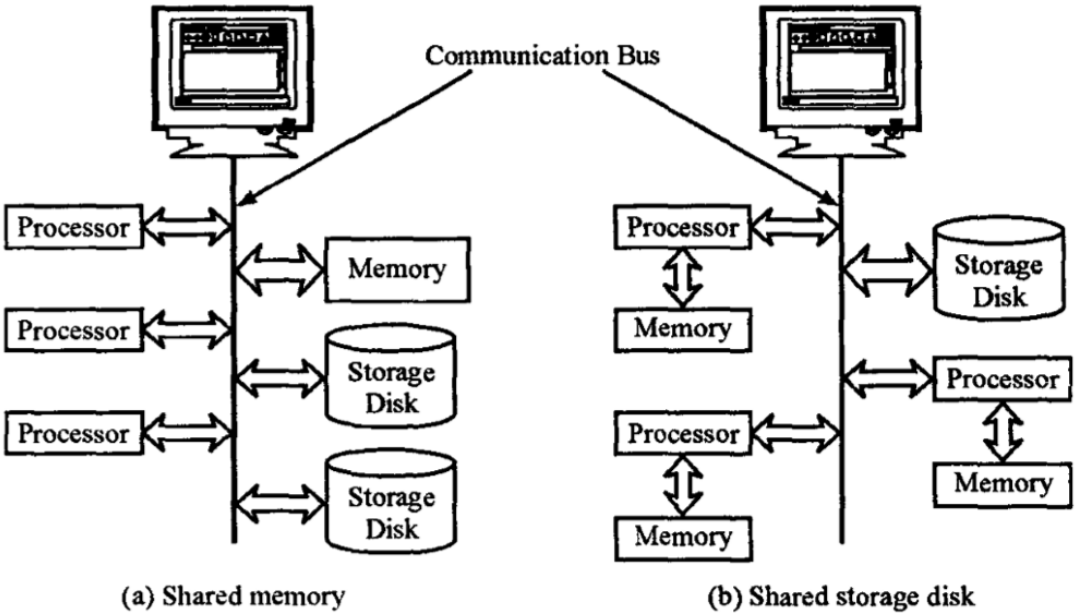
4. Memproses overhead.
5. Kegagalan jaringan komunikasi.
6. Pemulihan dari kegagalan lebih kompleks.

Sistem Database Paralel

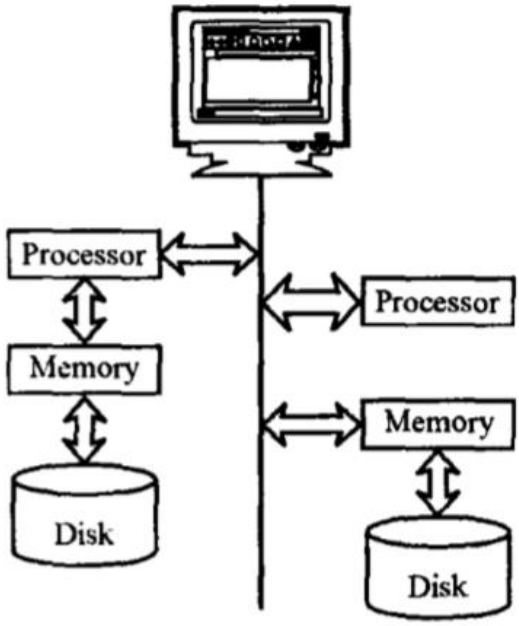
Arsitektur sistem database paralel terdiri dari beberapa CPU dan disk penyimpanan data secara paralel. Oleh karena itu, mereka meningkatkan kecepatan pemrosesan dan input/output. Sistem Database paralel digunakan dalam aplikasi yang harus menanyakan Database yang sangat besar atau yang harus memproses transaksi dalam jumlah yang sangat besar per detik. Beberapa arsitektur yang berbeda dapat digunakan untuk sistem database paralel, yaitu sebagai berikut:

- **Memori Bersama:** Semua prosesor berbagi memori yang sama.
- **Disk penyimpanan data bersama:** Semua prosesor berbagi satu set disk yang sama. Sistem disk bersama kadang-kadang disebut cluster.
- **Sumber Daya Independen:** Prosesor tidak berbagi memori atau disk umum.
- **Hirarki:** Model ini adalah hibrida dari tiga arsitektur sebelumnya.

Gambar menggambarkan arsitektur yang berbeda dari sistem database paralel. Dalam disk penyimpanan data bersama, semua prosesor berbagi disk (atau kumpulan disk) yang sama, seperti yang ditunjukkan pada Gambar. (a). Dalam arsitektur memori bersama, semua prosesor berbagi memori yang sama, seperti yang ditunjukkan pada Gambar. (b). Dalam arsitektur sumber daya independen, prosesor tidak berbagi memori umum maupun disk umum. Mereka memiliki sumber daya independen mereka sendiri seperti yang ditunjukkan pada Gambar. (c). Arsitektur hierarkis adalah hibrida dari ketiga arsitektur sebelumnya, seperti yang ditunjukkan pada Gambar. (d).

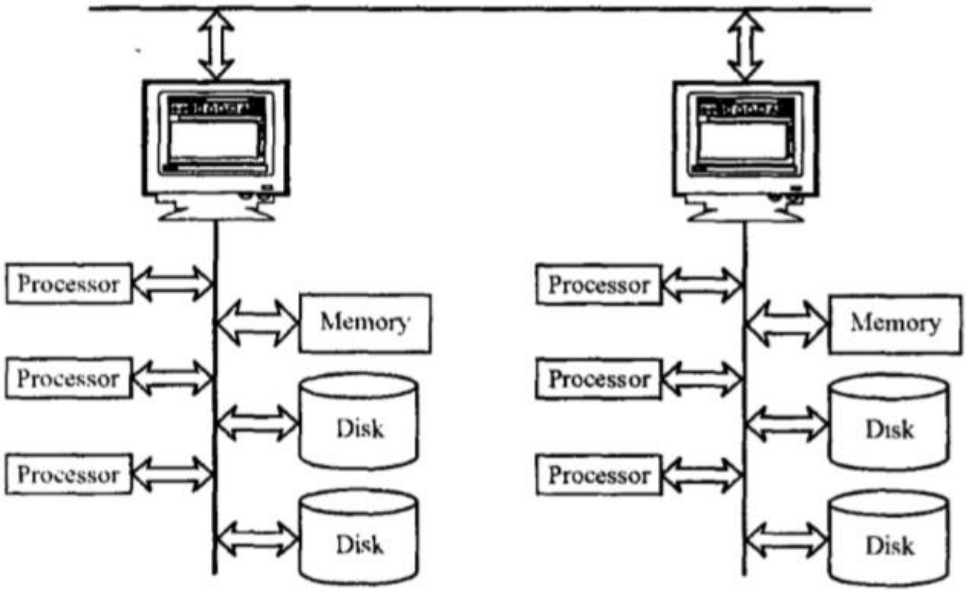


Gambar 1.11 Database Paralel (a) (b)



(c) Independent resource

Gambar 1.12 Database Paralel (c)



(d) Hierarchical

Gambar 1.13 Database Paralel (d)

Keuntungan Sistem Database Paralel

- Sistem Database paralel sangat berguna untuk aplikasi yang harus meminta Database yang sangat besar atau yang harus memproses transaksi dalam jumlah yang sangat besar per detik.
- Teknik ini digunakan untuk mempercepat proses transaksi pada sistem data-server.

- Dalam sistem database paralel, throughput (yaitu, jumlah tugas yang dapat diselesaikan dalam interval waktu tertentu) dan waktu respons (yaitu, jumlah waktu yang diperlukan untuk menyelesaikan satu tugas dari waktu diajukan) sangat tinggi.

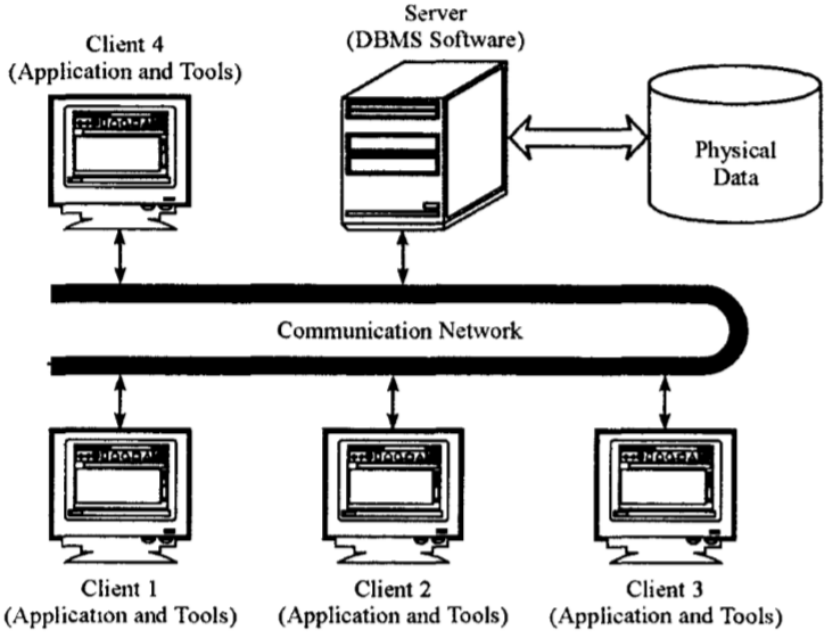
Kekurangan Sistem Database Paralel

- Dalam sistem database paralel, ada biaya startup yang terkait dengan memulai satu proses dan waktu startup dapat menutupi waktu pemrosesan yang sebenarnya, mempengaruhi percepatan secara merugikan.
- Karena proses yang dijalankan dalam sistem paralel sering mengakses sumber daya bersama, perlambatan dapat terjadi akibat interferensi dari setiap proses baru karena proses tersebut bersaing dengan proses yang ada untuk sumber daya yang dimiliki bersama, seperti disk penyimpanan data bersama, bus sistem, dan sebagainya.

Sistem Database Client/Server

Arsitektur Client/Server dari sistem database memiliki dua komponen logis yaitu client dan server. Client umumnya komputer pribadi atau workstation sedangkan server adalah *workstation* besar. Aplikasi dan alat DBMS berjalan pada satu atau lebih platform klien, sedangkan perangkat lunak DBMS berada di server. Komputer server disebut backend dan komputer klien disebut frontend. Komputer server dan klien ini terhubung melalui jaringan komputer. Aplikasi dan alat bertindak sebagai klien DBMS, membuat permintaan untuk layanannya.

Arsitektur client / server adalah bagian dari arsitektur sistem terbuka di mana semua perangkat keras komputasi, sistem operasi, protokol jaringan, dan perangkat lunak lainnya saling berhubungan sebagai jaringan dan bekerja bersama untuk mencapai tujuan pengguna. Ini sangat cocok untuk pemrosesan transaksi online dan aplikasi pendukung keputusan, yang cenderung menghasilkan sejumlah transaksi yang relatif singkat dan membutuhkan tingkat konkurensi yang tinggi.



Gambar 1.14 Arsitektur Client/Server

Keuntungan Data Base Client

- Lingkungan Client/Server memfasilitasi pekerjaan yang lebih produktif oleh pengguna dan memanfaatkan data yang ada dengan lebih baik.
- Sistem Database Client/Server lebih fleksibel dibandingkan dengan sistem terpusat.
- Sebuah database tunggal (di server) dapat dibagi di beberapa sistem klien (aplikasi) yang berbeda.
- Arsitektur Client/Server memberikan kinerja DBMS yang lebih baik.
- Sistem Client/Server memiliki platform yang lebih murah untuk mendukung aplikasi yang sebelumnya hanya berjalan pada komputer mainframe.

Keuntungan Data Base Client

- Biaya tenaga kerja atau pemrograman tinggi di lingkungan Client/Server, terutama pada fase awal.
- Ada kekurangan alat manajemen untuk pemantauan kinerja dan penyetelan dan kontrol keamanan, untuk DBMS, klien dan sistem operasi dan lingkungan jaringan.

1.12 BAHASA DATABASE

Bahasa definisi data digunakan untuk menentukan skema database. misalnya, Contoh DOL adalah membuat, mengubah, dan menjatuhkan tabel. Seperti Buat tabel siswa : (Nomor guling (10), Nama char (15), Jenis kelamin char (2), Alamat varchar (15));
Eksekusi pernyataan DOL di atas membuat tabel siswa. Ini memperbarui satu set tabel khusus yang disebut 'Kamus Data'. Kamus data berisi meta data artinya data tentang data. Skema (desain) sebuah tabel adalah contoh dari meta data.
Contoh :

(i) **CREATE** : Untuk membuat objek dalam database.
(ii) **ALTER**: Mengubah struktur database.
(iii) **DROP**: Menghapus objek dari data.
(iv) **TRUNCATE**: Hapus semua catatan dari tabel, termasuk semua ruang yang dialokasikan untuk catatan dihapus
(v) **COMMENT**: Menambahkan komentar ke kamus data.

Data-Manipulation Language (DMl)

Data Manipulation Berarti :

- Pengambilan data dari database.
- Penghapusan data dari database
- Penyisipan data baru ke dalam database
- Modifikasi data dalam database.

DML adalah bahasa yang memungkinkan pengguna untuk mengakses atau memodifikasi data dari database. Artinya DML digunakan untuk mengakses atau memanipulasi data dari database. DML pada dasarnya adalah dua jenis:

(i) DML Prosedural
(ii) DML non prosedural

DML Prosedural : DML prosedural mengharuskan pengguna untuk menentukan Data apa yang dibutuhkan dan bagaimana cara mendapatkan data tersebut. misalnya, PL/SOL. DML Non Prosedural : DML non prosedural mengharuskan pengguna untuk menentukan data apa yang dibutuhkan tanpa menentukan cara mendapatkan data tersebut. misalnya, SQL

Contoh: Contoh DML (NDML)

(i) Pilih Gulungan, Nama, Alamat dari Siswa Dimana Gulungan = 3;

(ii) Pilih * dari siswa:

Misalnya,

(i) **INSERT**: Menyisipkan data ke dalam tabel

(ii) **UPDATE**: Memperbarui data yang ada dalam tabel

(iii) **DELETE**: Menghapus semua record dari tabel, ruang untuk record tetap ada.

(iv) **LOCK TABLE**: Kontrol konkurensi.

Data Control Language (DCl)

Ini adalah komponen pernyataan SQL yang mengontrol akses ke data dan ke database. Sebagai contoh,

(i) **COMMIT**: Simpan pekerjaan yang sudah selesai.

(ii) **ROLL-BACK:** Kembalikan database ke aslinya sejak komit terakhir.

(iii) **SAVE POINT**: Identifikasi titik dalam transaksi yang nantinya dapat Anda putar kembali

(iv) **GRANTT/REVOKE:** Memberikan atau mengambil kembali izin ke atau dari pengguna oracle.

(v) **SET TRANSACTION**: Ubah atau ambil kembali izin ke atau dari pengguna oracle.

Data Query Language (DQL)

Ini adalah komponen pernyataan SQL yang memungkinkan mendapatkan data dari database dan memaksakan pemesanan padanya. SQL:

- DDL (membuat tabel)
- DML (memanipulasi tabel)

Catatan: DDL dan DML bukan dua bahasa yang berbeda, ini adalah bagian dari SQL.

Query : Sebuah query adalah pernyataan yang meminta pengambilan informasi.

1.13 INTERFACE DBMS

Antarmuka yang mudah digunakan yang disediakan oleh DBMS dapat mencakup hal-hal berikut:

- Antarmuka Berbasis Menu untuk Penjelajahan: Mereka sering digunakan dalam antarmuka penjelajahan, yang memungkinkan pengguna untuk melihat isi database dengan cara yang eksploratif dan tidak terstruktur.
- Antarmuka Berbasis Formulir: Antarmuka berbasis formulir menampilkan formulir untuk setiap pengguna.
- Antarmuka Pengguna Grafis: Antarmuka grafis menampilkan skema kepada pengguna dalam bentuk diagram.

- Antarmuka Bahasa Alami: Ini memiliki "skema" sendiri yang mirip dengan skema konseptual Database.
- Antarmuka untuk DBA
- Antarmuka untuk Pengguna

1.14 DATABASE USER DAN ADMINISTRATOR

Orang-orang yang bekerja dengan database dapat dikategorikan sebagai administrator database dan pengguna database.

Database Administrator (DBA)

Seseorang yang memiliki kendali pusat baik data maupun program yang mengakses data tersebut melalui sistem disebut administrator Database.

Fungsi DBA

Berikut ini adalah fungsi-fungsi DBA.

1. Mendefinisikan Skema Konseptual: DBA membuat skema konseptual (menggunakan bahasa definisi data) yang sesuai dengan desain database tingkat abstrak yang dibuat oleh administrator data. DBA membuat skema database asli dan struktur database.
2. **Mendefinisikan Skema Fisik:** DBA membuat skema fisik (menggunakan DDL) yang sesuai dengan desain database tingkat abstrak yang dibuat oleh administrator data.
3. **Skema dan Modifikasi Organisasi Fisik:** DBA melakukan perubahan atau modifikasi deskripsi database atau hubungannya dengan organisasi fisik database untuk mencerminkan perubahan kebutuhan organisasi atau untuk mengubah organisasi fisik untuk meningkatkan kinerja
4. **Pemberian Otorisasi untuk Akses Data :** DBA memberikan berbagai jenis otorisasi untuk menggunakan database kepada penggunanya. Ini mengatur penggunaan bagian-bagian tertentu dari database oleh berbagai pengguna. Informasi otorisasi disimpan dalam struktur sistem khusus yang dikonsultasikan oleh sistem Database setiap kali seseorang mencoba mengakses data dalam sistem. DBA membantu pengguna dengan definisi masalah dan resolusinya.
5. **Struktur Penyimpanan dan Metode Akses Definisi:** DBA memutuskan bagaimana data akan direpresentasikan dalam database yang disimpan, proses yang disebut desain database fisik. Database administrator mendefinisikan struktur penyimpanan database menggunakan bahasa definisi data dan metode akses data dari database.
6. **Pemeliharaan Rutin:** DBA memelihara backup database secara berkala, baik ke hard disk, compact disk atau ke server jarak jauh, untuk mencegah hilangnya data jika terjadi bencana. Ini memastikan bahwa ruang penyimpanan kosong yang cukup tersedia untuk operasi normal dan meningkatkan ruang disk sesuai kebutuhan. DBA juga bertanggung jawab untuk memperbaiki kerusakan database karena misue atau kegagalan perangkat lunak dan perangkat keras. DBA mendefinisikan dan menerapkan mekanisme kontrol kerusakan yang sesuai yang melibatkan pembongkaran Database secara berkala ke perangkat penyimpanan cadangan dan memuat ulang Database dari dump terbaru kapan pun diperlukan.

7. **Ketersediaan, Pencadangan, dan Pemulihan:** Tugas terpenting DBA adalah ketersediaan, pencadangan, dan pemulihan data. Berdasarkan nilai yang ditempatkan pada data elektronik, database harus dilindungi dari segala bentuk kegagalan seperti perangkat keras, perangkat lunak dan manusia. DBA mempertahankan informasi tentang kebutuhan organisasi agar berhasil. Availability artinya data harus tersedia bagi semua orang yang membutuhkannya pada saat mereka membutuhkannya. Jika data tidak tersedia, bisnis berhenti berfungsi. Karena tidak semua kegagalan dapat diprediksi, DBA perlu menerapkan prosedur pemulihan yang akan mengurangi waktu henti yang terkait dengan kegagalan.
8. **Performance Monitoring and Thning:** DBA harus memastikan database cepat dan responsif. Database respons lambat biasanya menunjukkan kinerja sistem yang buruk, ada sesuatu yang salah di suatu tempat. DBA memantau keadaan database untuk kinerja opsional dan log kesalahan atau log peristiwa juga dipantau untuk kesalahan database. Database yang disetel dengan buruk membuat frustrasi untuk digunakan - mereka cenderung menambah lebih banyak tekanan daripada nilai. Pemantauan sangat penting untuk mengakses status database dan menyesuaikannya.
9. **Implementasi dan Desain Database:** Tugas penting DBA adalah merancang Database untuk kinerja, skalabilitas, fleksibilitas, dan keandalan yang maksimal. Sebuah database yang dirancang dan diimplementasikan dengan baik membenarkan investasi database. DBA bertanggung jawab untuk menginstal DBMS baru dan memutakhirkan DBMS yang ada. DBA harus fasih dengan masalah instalasi dan peningkatan, yaitu masalah, persyaratan, dll.
10. **Kontrol migrasi program**, perubahan Database, referensi perubahan Database, dan perubahan menu melalui siklus hidup pengembangan.
11. **Membuat dan memelihara semua database:** yang diperlukan untuk pengembangan, pengujian, dan penggunaan produksi.
12. **DBA** memberlakukan dan memelihara batasan untuk memastikan integritas database.

Keterampilan DBA

DBA harus memiliki keterampilan berikut:

1. Pengetahuan yang baik tentang sistem operasi.
2. Pengetahuan yang baik tentang desain Database fisik.
3. Kemampuan untuk melakukan baik database dan juga pemantauan kinerja sistem operasi dan penyesuaian yang diperlukan.
4. Mampu memberikan arahan database strategis bagi organisasi.
5. Pengetahuan yang sangat baik tentang cadangan Database dan skenario pemulihan.
6. Keterampilan yang baik dalam semua alat database.
7. Pengetahuan yang baik tentang manajemen keamanan Database.
8. Pengetahuan yang baik tentang bagaimana database memperoleh dan mengelola sumber daya.
9. Pengetahuan yang baik tentang aplikasi di situs Anda.

10. Pengalaman dan pengetahuan dalam memigrasikan kode, perubahan Database, data, dan menu melalui berbagai tahap siklus hidup pengembangan.
11. Pengetahuan yang baik tentang cara database menegakkan integritas data.
12. Pengetahuan yang baik tentang penyetelan kinerja Database dan kode program.
13. DBA harus memiliki keterampilan komunikasi yang baik dengan manajemen, tim pengembangan, vendor, administrator sistem, dan penyedia layanan terkait lainnya.

User Database

Ada empat jenis pengguna sistem Database yang berbeda.

1. **Pengguna Naif:** Pengguna naif adalah pengguna tidak canggih yang berinteraksi dengan sistem dengan menjalankan salah satu program aplikasi yang telah ditulis sebelumnya. misalnya, Teller Bank yang perlu mentransfer Rp. 500 dari rekening A ke rekening B menjalankan program yang disebut transfer. Program ini meminta teller untuk jumlah uang yang akan ditransfer. Seorang pengguna ATM termasuk dalam kategori ini.
2. **Pemrogram Aplikasi :** Pemrogram aplikasi adalah profesional komputer yang menulis program aplikasi. Pemrogram aplikasi dapat memilih dari banyak alat untuk mengembangkan antarmuka pengguna. Contoh: Alat Rapid Application Development (RAD) adalah alat yang memungkinkan pemrogram aplikasi untuk membuat formulir dan laporan tanpa menulis program.
3. **Pengguna Canggih:** Pengguna canggih berinteraksi dengan sistem tanpa menulis program. Sebagai gantinya, mereka membentuk permintaan mereka dalam bahasa kueri database. Mereka mengirimkan setiap kueri tersebut ke prosesor kueri, yang fungsinya untuk memecah pernyataan DML menjadi instruksi yang dipahami oleh manajer penyimpanan. Analis yang mengirimkan kueri untuk mengeksplorasi data dalam database termasuk dalam kategori ini. (4) Pengguna khusus: Pengguna khusus adalah pengguna canggih yang menulis aplikasi database khusus yang tidak sesuai dengan kerangka pemrosesan data tradisional. • Di antara aplikasi tersebut adalah sistem desain Computer Aided, basis pengetahuan dan sistem Pakar.

1.15 STRUKTUR DATABASE KESELURUHAN

Sistem Database dibagi menjadi modul-modul yang menangani masing-masing tanggung jawab sistem secara keseluruhan. Beberapa fungsi sistem database mungkin disediakan oleh sistem operasi komputer. Komponen fungsional dari sistem database dapat dibagi menjadi:

- Manajer penyimpanan
- Pemroses kueri

Manajer Penyimpanan

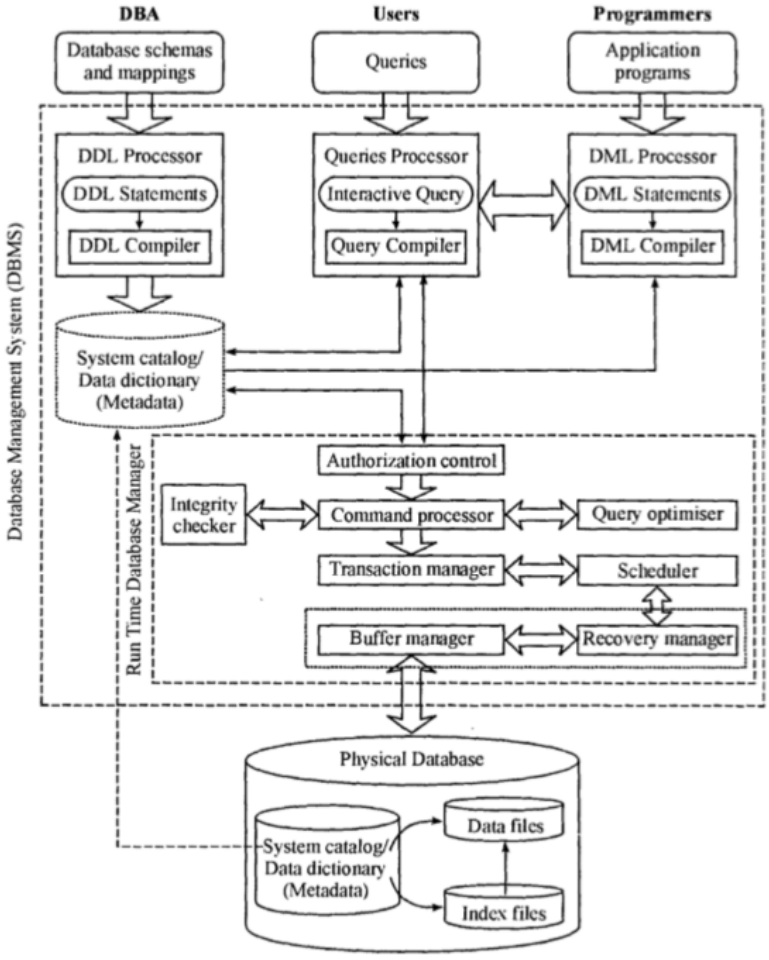
Manajer penyimpanan adalah modul program yang menyediakan antarmuka antara data tingkat rendah yang disimpan dalam database dan program aplikasi dan kueri yang dikirimkan ke sistem.

- Manajer penyimpanan bertanggung jawab atas interaksi dengan manajer pengisian.

- Manajer penyimpanan penting karena Database membutuhkan ruang penyimpanan yang besar.

Komponen berikut busur ini termasuk dalam manajer penyimpanan.

- **Manajer Otorisasi dan Integritas:** Ini menguji kepuasan batasan integritas dan memeriksa otoritas pengguna untuk mengakses data.
- **Manajer Transaksi:** Ini memastikan bahwa database tetap dalam keadaan konsisten meskipun ada kegagalan sistem dan eksekusi transaksi saat ini tanpa konflik.
- **File Manager:** Ini mengelola alokasi ruang pada penyimpanan disk dan struktur data yang digunakan untuk mewakili informasi yang disimpan pada disk.
- **Buffer Manager:** Bertanggung jawab untuk mengambil data dari penyimpanan disk ke memori utama dan memutuskan data apa yang akan diambil di memori utama.



Gambar 1.15 Struktur DBMS

Berikut ini adalah struktur data yang diperlukan sebagai bagian dari implementasi sistem fisik.

(i) File Data : Ini menyimpan database itu sendiri.

(ii) Kamus Data: Ini menyimpan metadata tentang struktur database, khususnya, skema database.

(iii) Indeks: Ini menyediakan akses cepat ke item data yang memiliki nilai tertentu

(iv) Data Statistik : Ini menyimpan informasi statistik tentang data dalam database

Pemroses Kueri Berikut adalah komponen-komponen berikut:

(i) Kompilator DML: Ini menerjemahkan pernyataan DML ke dalam bahasa kueri ke dalam instruksi tingkat rendah yang dipahami oleh mesin evaluasi kueri.

(ii) Precompiler DML Tertanam: Ini mengubah pernyataan DML yang disematkan dalam program aplikasi ke panggilan prosedur normal dalam bahasa host.
   a. Precompiler berinteraksi dengan compiler DML untuk menghasilkan kode yang sesuai.

(iii) DDL Interpreter : Ini menafsirkan pernyataan DDL dan mencatat definisi dalam kamus data.

(iv) Mesin Evaluasi Kueri: yang mengeksekusi instruksi tingkat rendah yang dihasilkan oleh kompiler DML.

1.16 *FOURTH-GENERATION LANGUAGE* (4GL)

4GL adalah bahasa pemrograman yang ringkas, efisien dan non-prosedural yang digunakan untuk meningkatkan produktivitas DBMS. Dalam 4GL, pengguna menentukan apa yang harus dilakukan dan bukan bagaimana melakukannya. 4GL bergantung pada alat 4GL tingkat yang lebih tinggi, yang digunakan oleh pengguna untuk menentukan parameter guna menghasilkan program aplikasi. 4GL memiliki komponen-komponen berikut di dalamnya:

- Bahasa kueri
- Pembuat laporan
- Spreadsheet
- Bahasa Database
- Generator aplikasi untuk operasi deflne seperti menyisipkan, mengambil dan memperbarui data dari database untuk membangun aplikasi.
- Bahasa tingkat tinggi untuk menghasilkan program aplikasi.

Structured Query Language (SQL) dan query by example (QBE) adalah contoh dari 4GL.

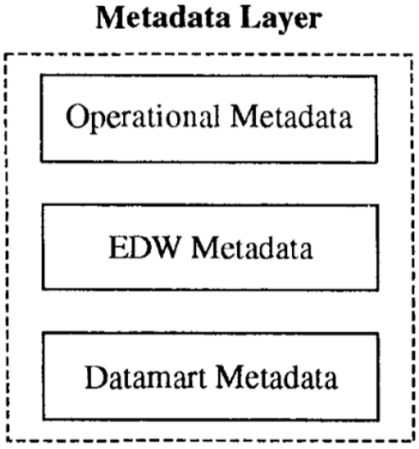
1.17 METADATA

Sebuah metadata (juga disebut kamus data) adalah data tentang data. Ini juga merupakan katalog sistem callt'{d, yang merupakan sifat database yang menggambarkan diri sendiri yang menyediakan independensi program-data. Katalog sistem mengintegrasikan matadata. Matadata menggambarkan struktur database, kendala, aplikasi, otorisasi, ukuran tipe data dan sebagainya. Matadata tersedia untuk administrator database, desainer, dan pengguna resmi sebagai dokumentasi sistem online.

Sebuah metadata (juga disebut kamus data) adalah data tentang data. Ini juga merupakan katalog sistem callt'{d, yang merupakan sifat database yang menggambarkan diri sendiri yang menyediakan independensi program-data. Katalog sistem mengintegrasikan matadata. Matadata menggambarkan struktur database, kendala, aplikasi, otorisasi, ukuran tipe data dan sebagainya. Matadata tersedia untuk administrator database, desainer, dan pengguna resmi sebagai dokumentasi sistem online.

Jenis Metadata

- **Metadata Operasi:** Ini menggambarkan data dalam berbagai sistem operasi yang memberi makan Data warehouse perusahaan. Metadata operasional biasanya ada dalam sejumlah format yang berbeda dan sayangnya seringkali berkualitas buruk.
- **Enterprise data warehouse (Metadata EDW):** Jenis metadata ini berasal dari model data perusahaan. Meta data EDW menggambarkan lapisan data yang direkonsiliasi serta aturan untuk mengubah data operasional menjadi data yang direkonsiliasi.
- **Metadata Data mart:** Mereka menggambarkan lapisan data turunan dan aturan untuk mengubah data yang direkonsiliasi menjadi data turunan.



Gambar 1.16 Jenis Metada

1.18 KONSEP MODEL ER

Diagram ER dapat mengekspresikan keseluruhan struktur logis dari database secara grafis. Ini adalah elemen-elemen berikut yang terdiri dari model E-R.

(i) Himpunan entitas
(ii) Himpunan relasi
(iii) Atribut

Entitas

Entitas adalah "benda" atau objek di dunia nyata yang dapat dibedakan dari objek lain. misalnya, Setiap orang adalah suatu entitas dan rekening bank dapat dianggap sebagai entitas. **Himpunan Entitas** : Himpunan entitas adalah sekumpulan entitas dengan tipe yang sama yang memiliki properti atau atribut yang sama. misalnya, Himpunan" dari semua orang yang menjadi pelanggan di bank tertentu. Itu adalah pelanggan adalah himpunan entitas

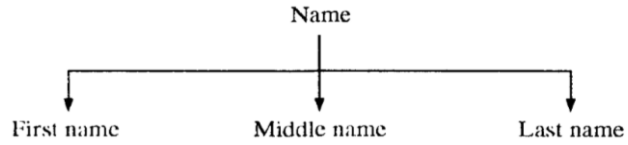
Atribut

Setiap entitas diwakili oleh satu set properti yang disebut atribut. **Atribut Kunci** : Tipe entitas biasanya memiliki atribut yang nilainya berbeda untuk setiap entitas individu dalam koleksi. Atribut seperti itu disebut atribut kunci dan nilainya dapat digunakan untuk mengidentifikasi setiap entitas secara unik.

Jenis Atribut

Atribut Sederhana : Atribut yang tidak dapat dibagi menjadi sub bagian disebut atribut sederhana. misalnya, nomor Roll adalah contoh atribut sederhana.

Atribut Komposit : Atribut yang dapat dibagi menjadi pendukung disebut atribut komposit. misalnya, Tanggal Lahir adalah contoh atribut komposit, karena dapat dibagi menjadi "Hari lahir", "Bulan lahir" dan "Tahun lahir". OR Name juga merupakan contoh dari atribut komposit.

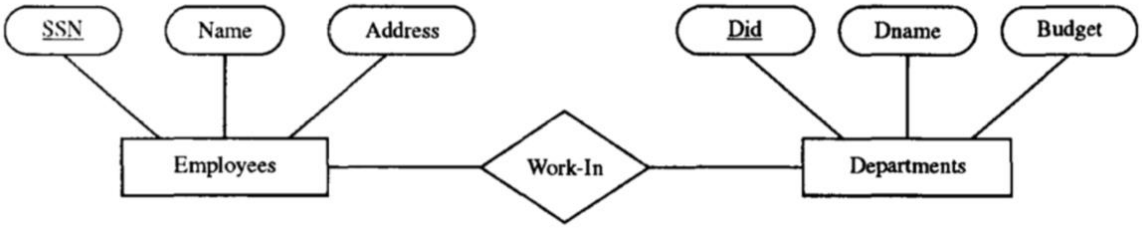


Gambar 1.17 Contoh Atribut Komposit

- **Atribut Bernilai Tunggal :** Atribut yang dapat mengasumsikan satu dan hanya satu nilai disebut "Atribut bernilai tunggal". Misalnya: ''Usia seseorang adalah contoh Atribusi bernilai tunggal. Atribut Bernilai Banyak: Atribut yang dapat mengasumsikan sekumpulan nilai disebut "Atribut Bernilai Banyak". misalnya, Entitas Karyawan yang ditetapkan dengan atribut nomor telepon adalah contoh atribut multinilai karena seorang karyawan mungkin memiliki nol, satu atau banyak nomor telepon.
- **Atribut Null:** Nilai nol digunakan ketika entitas tidak memiliki nilai untuk atribut. misalnya, Atribut gelar perguruan tinggi hanya berlaku untuk orang dengan gelar sarjana.
- **Derived Attributes :** Nilai dari tipe atribut ini dapat diturunkan dari atribut atau entitas lain yang terkait. misalnya, Dalam sistem penggajian, HRA dan PE diturunkan dari atribut gaji.

1.19 RELATION DAN RELATION SET

- Relasi adalah hubungan antara dua entitas atau lebih.
- Himpunan relasi adalah himpunan relasi yang bertipe sama dengan nilai yang sama.
- Relasi matematis pada n ≥ 2 himpunan entitas. Jika E_1, E_2, ... E_n adalah himpunan entitas maka himpunan relasi adalah himpunan bagian dari $\{(e_1, e_2, \ldots, e_n)/e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$. di mana $(e_1, e_2, e_3, \ldots, e_n)$ adalah himpunan hubungan.



Gambar 1.18 Work-In Relation Set

1.20 KENDALA

Skema perusahaan ER dapat menentukan batasan tertentu yang harus dikonfirmasi oleh isi database. Kendala:

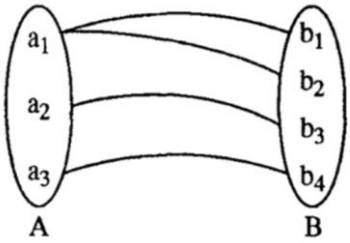
- Kardinalitas pemetaan
- Batasan partisipasi

Pemetaan Kardinalitas

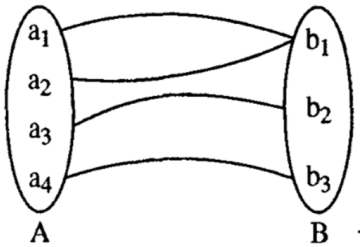
Kardinalitas pemetaan atau rasio kardinalitas, menyatakan jumlah entitas yang entitas lain dapat dikaitkan melalui hubungan st:1l: kardinalitas pemetaan paling berguna dalam menggambarkan himpunan hubungan biner. Untuk himpunan relasi biner R antara himpunan entitas A dan B, kardinalitas pemetaan harus salah satu dari berikut ini :

1. *One to One:* Entitas di A diasosiasikan dengan tepat satu entitas di B. Dan satu entitas di B diasosiasikan dengan tepat satu entitas di A.

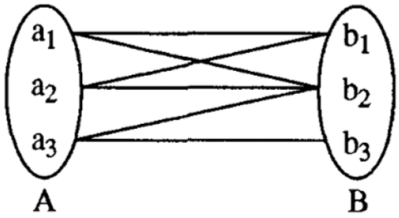
2. Satu ke banyak: Sebuah entitas di A diasosiasikan dengan sejumlah entitas di B. Sebuah entitas di B dapat diasosiasikan dengan paling banyak satu entitas di A.



3. Banyak ke Satu: Sebuah entitas di A dikaitkan dengan paling banyak satu entitas di B. Sebuah entitas di B, dapat dikaitkan dengan sejumlah entitas di A.



4. Banyak ke Banyak: Entitas di A dikaitkan dengan sejumlah entitas di B, dan entitas di B dikaitkan dengan sejumlah entitas di A.



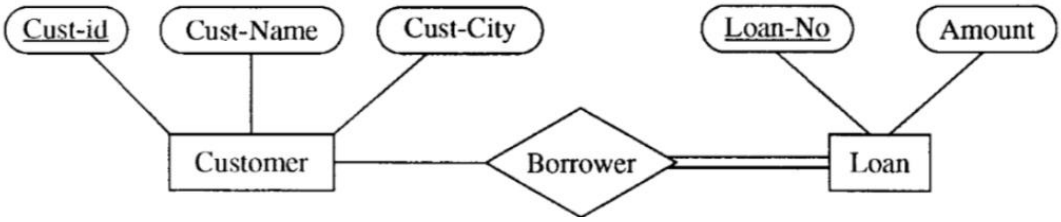
Batasan Partisipasi

Batasan partisipasi menentukan apakah keberadaan himpunan entitas bergantung pada keterkaitannya dengan entitas lain

Batasan partisipasi:

- Total
- Sebagian

1. **Total Participate :** Partisipasi himpunan entitas E dalam himpunan relasi R dikatakan total jika setiap entitas pada E berpartisipasi dalam setidaknya satu hubungan dalam R. Contoh : Kita mengharapkan setiap entitas pinjaman berelasi dengan setidaknya satu pelanggan melalui hubungan peminjam. Oleh karena itu, partisipasi pinjaman dalam himpunan hubungan peminjam adalah total.



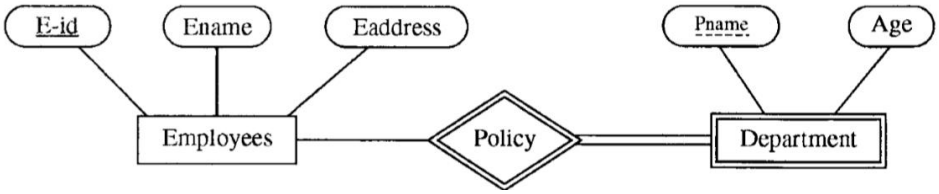
Gambar 1.19 Total Participal

2. **Partiall Participate :** Jika hanya beberapa entitas di E yang berpartisipasi dalam relasi di R. Partisipasi himpunan entitas E dalam relasi R dikatakan partisipasi parsial. Contoh: Seseorang dapat menjadi nasabah bank baik ia memiliki pinjaman di bank atau tidak. Oleh karena itu partisipasi nasabah dalam hubungan peminjam diatur secara parsial.

1.21 KETERGANTUNGAN EKSISTENSI

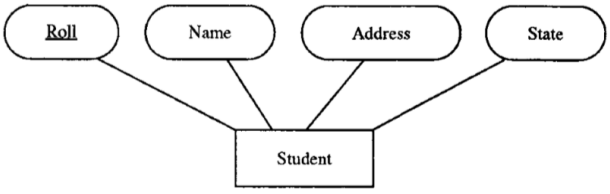
Jika keberadaan suatu entitas x bergantung pada keberadaan entitas lain y, maka x dikatakan keberadaan bergantung pada y.

1. **Tipe Entitas Lemah :** Tipe entitas yang tidak memiliki atribut kunci sendiri disebut entitas lemah.



Gambar 1.20 Tipe Entitas Lemah

2. **Tipe entitas kuat** : Tipe entitas yang memiliki atribut kunci disebut tipe entitas kuat. misalnya.



Gambar 1.21 Tipe Entitas Kuat

- Secara operasional y dihilangkan maka entitas y dikatakan sebagai entitas yang didominasi dan x dikatakan sebagai entitas subordinat.

- Tipe entitas yang lemah selalu memiliki batasan partisipasi total sehubungan dengan hubungan pengidentifikasiannya karena entitas yang lemah tidak dapat diidentifikasi tanpa entitas pemilik
- Kami dapat mewakili entitas lemah yang diatur oleh persegi panjang ganda
- Kami menggarisbawahi diskriminator dari himpunan entitas lemah dengan garis putus-putus.

1.22 KEY (KUNCI)

Key adalah bidang yang secara unik mengidentifikasi catatan, tabel, atau data. Dalam model data relasional ada banyak 'key'. Beberapa 'key' tersebut adalah:

- *Primary Key*
- *Foreign Key*
- *Unique Key*
- *Super Key*
- *Candidate Key*

Peimary Key/Primary Key: Primary Key adalah satu atau lebih kolom dalam tabel yang digunakan untuk mengidentifikasi secara unik setiap baris dalam tabel. Contoh :

Tabel 1.3 Untuk Mengidentifikasi Secara Unik Setiap Baris Dalam Tabel

Mahasiswa

Roll_No	Nama	Alamat
1 2 3	Agus Ade Josh	Semarang Jepara Yogyakarta

Roll_No. adalah Primary Key. **Catatan**: Primary Key tidak boleh berisi nilai Null.

Unique Key/Unique Key: Unique Key adalah satu atau lebih kolom dalam tabel yang digunakan untuk mengidentifikasi secara unik setiap baris dalam tabel. Itu dapat berisi nilai NULL. Mahasiswa :

Tabel 1.4 Contoh Unique Key

Roll_No	Nama	Alamat
1 - 3	Agus Ade Josh	Semarang Jepara Yogyakarta

Roll_No. adalah 'key' yang unik.

DitT b/w primary key dan Unique key

- Unique Key mungkin berisi nilai NULL tetapi Primary Key tidak pernah dapat berisi nilai NULL.

Super Key : Super key adalah sekumpulan satu atau lebih atribut yang, diambil secara kolektif, memungkinkan kita untuk mengidentifikasi secara unik sebuah tuple dalam relasi. misalnya, {Roll-No}, {Roll-No, Name}, {Roll-No, Address}, {Roll-No, Name, Address} semua set ini adalah Super Key.

Candidate Key: Jika skema relasional memiliki lebih dari satu 'key', masing-masing disebut Candidate Key.

Semua 'key' yang memenuhi kondisi primary key dapat menjadi Candidate Key. misalnya.,

Tabel 1.5 Contoh Candidat Key

Mahasiswa

Roll_No	Nama	No_HP	Kota
112	Agus	0812300000	Semarang
113	Ade	0878000000	Jepara
114	Josh	0856000000	Yogyakarta
115	Widya	0813260000	Solo

{Roll-No} dan {Phone-No} adalah dua Candidate Key. Karena kita dapat mempertimbangkan salah satu dari ini sebagai Primary Key.

Foreign Key : Foreign Key mewakili hubungan antar tabel.

- Foreign Key adalah kolom yang nilainya diturunkan dari Primary Key dari beberapa tabel lain. ATAU
- Foreign Key adalah 'key' yang merupakan Primary Key dalam satu tabel dan digunakan untuk berhubungan dengan beberapa atribut di tabel lain dengan entri data yang sama. misalnya:

Mahasiswa (Atas) dan Nilai (Bawah) :

Tabel 1.6 Di sini Roll_No tabel tanda adalah Foreign Key

Roll_ No	Nama	Mata Kuliah
1	Agus	Sistem Operasi
2	Ade	Statistika
3	Josh	Desain Web

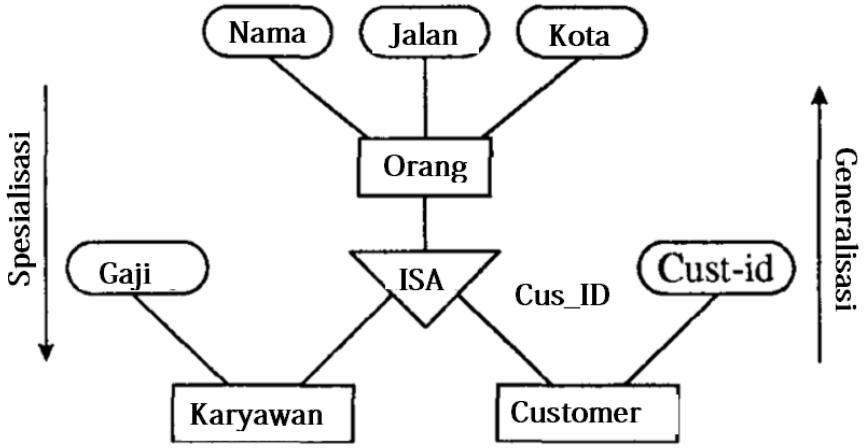
Roll_ No	Semester	Nilai
1	VII	80
2	V	75
3	III	70

1.23 ASOSIASI

Asosiasi menghubungkan dua atau lebih tipe entitas independen. Sebagai contoh: Teacherrreachers by adalah hubungan asosiatif biner dasar yang menghubungkan dua entitas tipe guru dan mata kuliah. Sedangkan menulis/ditulis oleh adalah hubungan ternary yang menghubungkan mahasiswa, mata kuliah dan Laporan.

1.24 SPESIALISASI

Spesialisasi adalah hasil mengambil subset dari himpunan entitas tingkat yang lebih tinggi untuk membentuk himpunan entitas tingkat yang lebih rendah. Kita dapat mengatakan: Sebuah set entitas tingkat yang lebih rendah mewarisi semua atribut dan partisipasi hubungan dari set entitas tingkat yang lebih tinggi yang terhubung.



Gambar 1.22 Spesialisasi mengikuti pendekatan "Top-down".

1.25 GENERALISASI

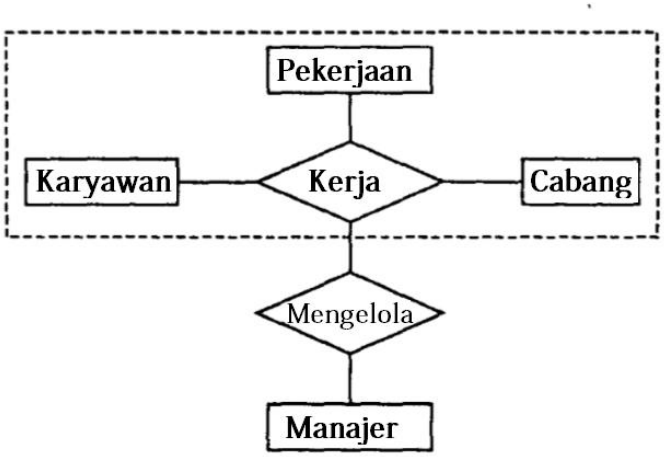
Generalisasi adalah hasil dari penggabungan dua atau lebih himpunan entitas yang terpisah (tingkat yang lebih rendah) untuk menghasilkan himpunan entitas tingkat yang lebih tinggi. Atribut set entitas tingkat yang lebih tinggi diwarisi oleh set entitas tingkat yang lebih rendah.

- Generalisasi mengikuti pendekatan "Bottom-up".
- Orang adalah superclass (entitas tingkat yang lebih tinggi) dan pelanggan dan karyawan adalah entitas tingkat yang lebih rendah.

1.26 PENGUMPULAN

Agregasi adalah abstraksi di mana hubungan diperlakukan sebagai entitas tingkat yang lebih tinggi.

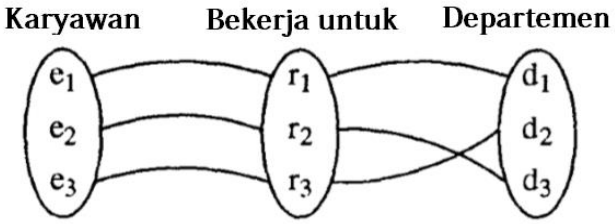
- Agregasi memungkinkan kita untuk menunjukkan bahwa kumpulan hubungan berpartisipasi dalam kumpulan hubungan lain, misalnya, Kumpulan hubungan bekerja, menghubungkan kumpulan entitas karyawan, cabang, dan pekerjaan sebagai kumpulan entitas tingkat yang lebih tinggi yang disebut pekerjaan.



Gambar 1.23 Agregasi

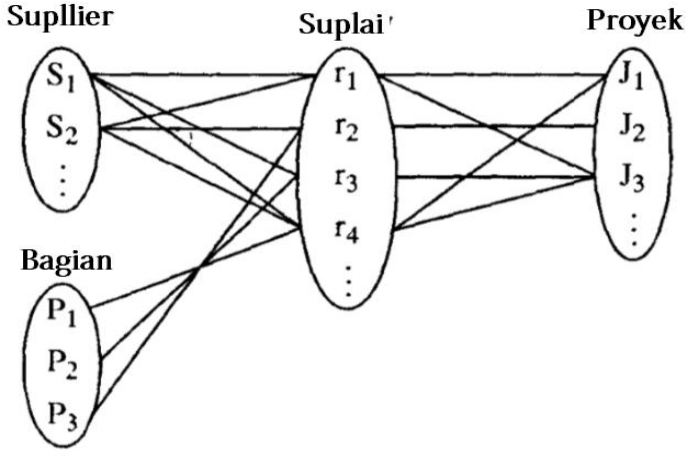
1.27 HUBUNGAN TINGKAT TINGGI

- Derajat tipe hubungan adalah jumlah tipe entitas parkir.
- Jenis hubungan derajat dua disebut biner. Salah satu dari derajat tiga disebut ternary. misalnya.,



Gambar 1.24 Biner

Hubungan kerja-untuk adalah derajat dua.



Gambar 1.25 Ternary

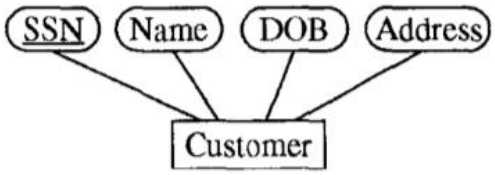
Hubungan penawaran adalah derajat tiga atau hubungan terner adalah penawaran.

1.28 PENGURANGAN DIAGRAM ER KE TABEL

- Sebuah database yang sesuai dengan skema database ER dapat diwakili oleh kumpulan tabel.
- Untuk setiap kumpulan entitas dan untuk setiap kumpulan hubungan dalam database, ada tabel unik yang diberi nama kumpulan entitas atau kumpulan hubungan yang sesuai.
- Setiap tabel memiliki beberapa kolom, yang masing-masing memiliki nama yang unik. Kami dapat mewakili dalam skema E-R dengan tabel.

Representasi Tabular dari Himpunan Entitas Kuat

Misalkan E adalah himpunan entitas kuat dengan atribut deskriptif ab a2, a3, ... , an. Kami mewakili entitas ini dengan tabel yang disebut E dengan n kolom berbeda, yang masing-masing sesuai dengan salah satu atribut E. Setiap baris dalam tabel ini sesuai dengan satu entitas dari himpunan entitas E.



Gambar 1.26 Contoh Himpunan Entitas E

Pelanggan

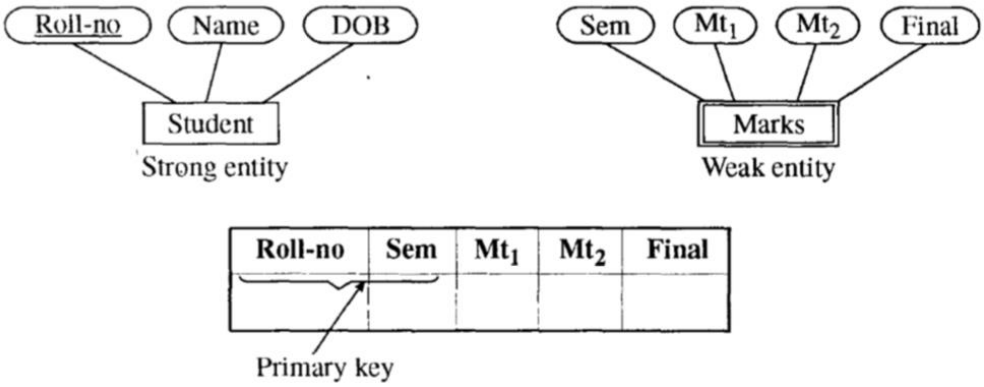
Tabel 1.8 Contoh Tabel

SSN	Name	DOB	Address

Representasi Tabular dari Himpunan Entitas Lemah

Misalkan A adalah himpunan entitas lemah dengan atribut a_l, a_2, ... , a_m. Misalkan B adalah himpunan entitas kuat yang bergantung pada A. Misalkan kunci utama B terdiri dari atribut bl , b2, ... , bn. Kami mewakili himpunan entitas A dengan tabel yang disebut A dengan satu kolom untuk setiap atribut himpunan

$\{a_1, a_2, ... , a_n\} \cup \{b_l, b_2, ... , b_n\}$



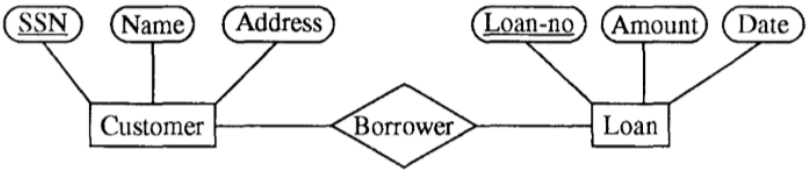
Gambar 1.27 Representasi Tabular Dari Himpunan Entitas Lemah

Representasi tabular dari Himpunan Hubungan

Misalkan R adalah himpunan relasi, misalkan di> a2, ... , am adalah himpunan atribut yang dibentuk oleh gabungan kunci-kunci utama dari masing-masing himpunan entitas yang berpartisipasi dalam R dan biarkan atribut deskriptif dari R menjadi bl, bz , ... , bn . Kami mewakili hubungan yang ditetapkan oleh tabel yang disebut R dengan satu kolom untuk setiap atribut dari himpunan.

$\{a_1, a_2, ..., a_n\} \cup \{b_1, b_2, ...Ub_n\}$

Misalnya:



SSN	Name	DOB	Address

SSN	Loan

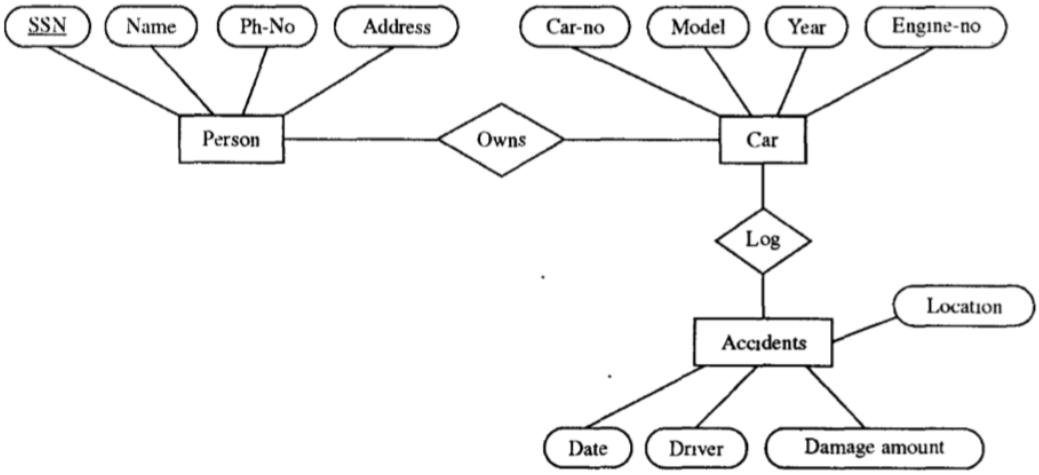
Loan-no	Amount	Date

Gambar 1.28 Representasi Tabular Dari Himpunan Hubungan

1.29 PENYELESAIAN MASALAH

Pertanyaan 1 : Buatlah model E-R untuk perusahaan asuransi mobil yang pelanggannya masing-masing memiliki satu atau lebih mobil. Setiap mobil telah dikaitkan dengan itu nol untuk sejumlah kecelakaan yang tercatat. (UPTU 2003, 06)

Jawaban :



Gambar 1.29 Model ER

Pertanyaan 2 : Jelaskan kunci berikut

- Beralternatif
- Secondary Key

Jawaban :

- Alternatifnya: Jika terdapat banyak kunci kandidat, salah satunya digunakan sebagai kunci utama, maka semua kunci kandidat lainnya dikenal sebagai kunci alternatif.
- Kunci sekunder: Kunci sekunder adalah atribut atau kombinasi atribut yang mungkin bukan kunci kandidat tetapi mereka mengklasifikasikan himpunan entitas pada karakteristik tertentu.

Pertanyaan 3 : Diskusikan model Extended ER:

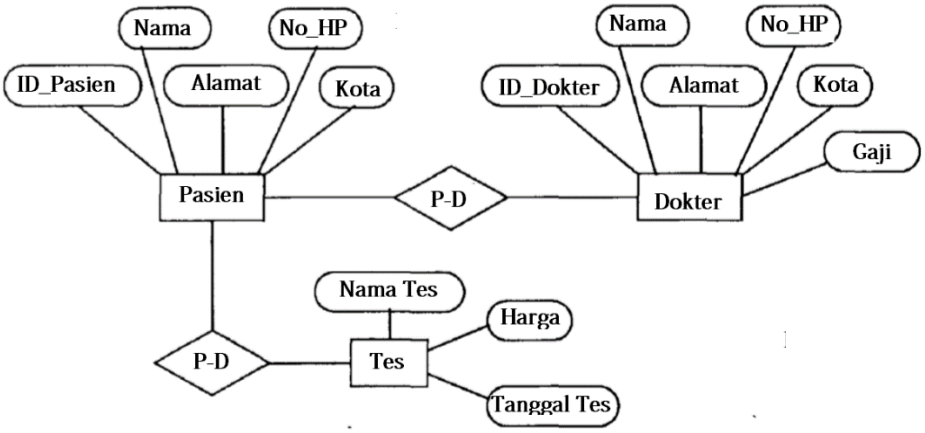
Jawaban :

Iklan. Model ER yang Diperpanjang: Sejak akhir 1970-an, aplikasi baru teknologi Database telah muncul, ini terutama Database untuk desain dan manufaktur teknik, telekomunikasi, gambar dan grafik, multimffdia, penambangan data, pergudangan data, dll. Jenis Database ini memiliki lebih banyak persyaratan kompleks daripada aplikasi tradisional. Untuk merepresentasikan persyaratan ini seakurat mungkin, perancang aplikasi database harus menggunakan konsep pemodelan data semantik tambahan.

Model ER dapat ditingkatkan untuk memasukkan konsep-konsep ini, yang mengarah ke model Enhanced ER. Menggunakan konsep hubungan kelas/subkelas dan pewarisan tipe ke dalam Model E-R dapat melakukan ini. Model Enhance E-R mencakup semua konsep pemodelan model E-R dan konsep subclass, super-class, spesialisasi dan generalisasi.

Pertanyaan 4: Buat diagram E-R untuk rumah sakit dengan sekumpulan pasien dan sekumpulan dokter medis. Kaitkan dengan setiap pasien, log dari berbagai tes dan pemeriksaan yang dilakukan. (UPTU 2005-06)

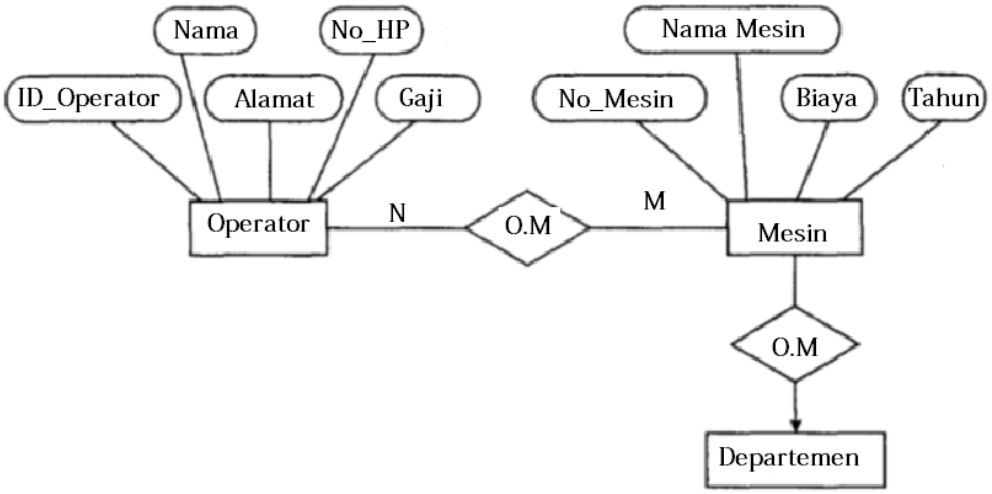
Jawaban :



Gambar 1.30 Diagram E-R

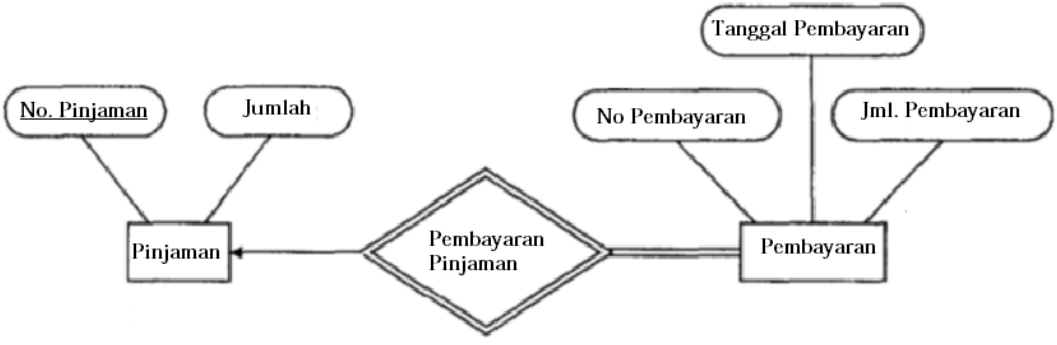
Pertanyaan 5 : Gambarlah diagram E-R untuk seorang operator yang dapat bekerja pada banyak mesin dan setiap mesin memiliki banyak operator. Setiap mesin milik satu departemen tetapi departemen dapat memiliki banyak mesin. (UPTU 2005-06)

Jawaban :



Gambar 1.31 Diagram E-R

Pertanyaan 6 : Jelaskan diagram E-R berikut.



Gambar 1.32 Diagram E-R

Jawaban :

Penjelasan diagram E-R:

(i) Kotak bergaris ganda (pembayaran) menunjukkan kumpulan entitas yang lemah. Berlian garis ganda (hubungan pembayaran pinjaman) menunjukkan hubungan pengidentifikasian yang sesuai.

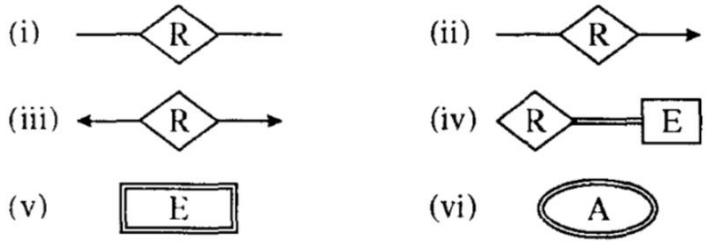
(ii) Pembayaran adalah entitas lemah yang bergantung pada pinjaman, entitas kuat, melalui hubungan yang mengatur pembayaran pinjaman.

(iii) Garis ganda menunjukkan partisipasi total.

(iv) Panah dari pembayaran pinjaman ke pinjaman menunjukkan bahwa setiap pembayaran adalah untuk satu pinjaman.

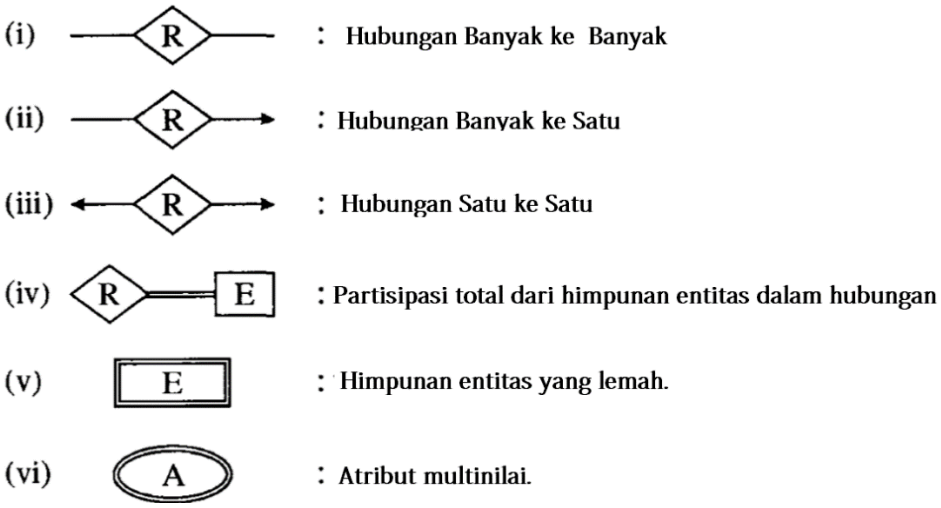
(v) Loan-No adalah kunci utama dari pinjaman entitas. Sedangkan payment-No adalah kunci parsial dari pembayaran entitas.

Pertanyaan 7 : Jelaskan simbol-simbol diagram E-R berikut. (UPTU 2004)



Gambar 1.33 Simbol Diagram E-R

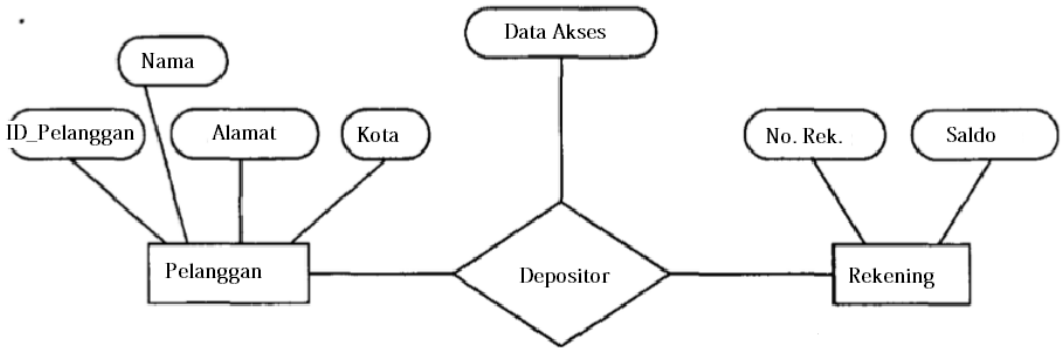
Jawaban :



Gambar 1.34 Pnjelasan Simbol Diagram E-R

Pertanyaan 8 : Gambarlah diagram E-R untuk pelanggan dan rekening dengan atribut yang dilampirkan ke rangkaian hubungan.

Jawaban :

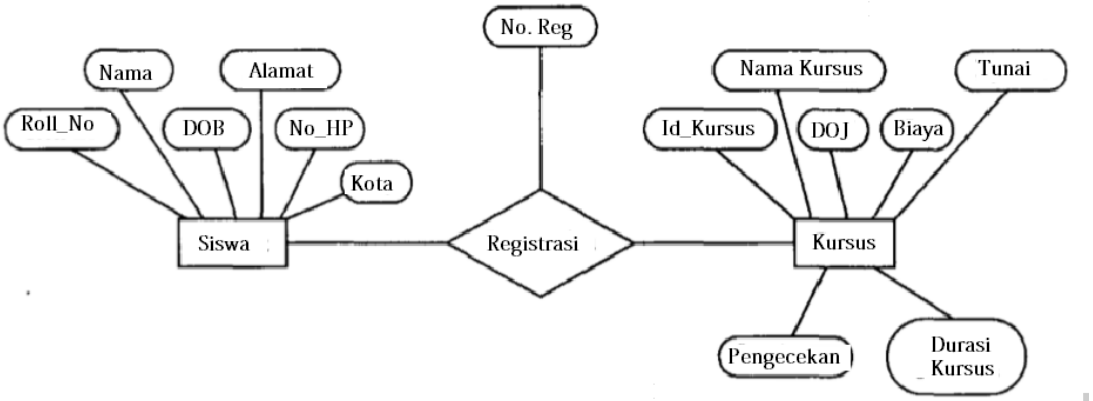


Gambar 1.35 Diagram E-R

Pertanyaan 9: Gambarlah diagram E-R dari proses pendaftaran mahasiswa pada mata kuliah tertentu. (UPTU 2005-06)

Jawaban :

Diagram E-R dari proses pendaftaran seorang mahasiswa pada mata kuliah tertentu.



Gambar 1.36 Diagram E-R

Pertanyaan 10: Perbedaan antara sistem database dan sistem file.

Jawaban :

Tabel 1.8 Perbedaan Antara Sistem Database Dan Sistem File

Sistem File	Sistem Database
Redundansi data : Duplikasi data dalam file yang berbeda. Berbagai salinan data yang sama mungkin memiliki informasi yang berbeda.	**Redundansi data:** Kontrol data terpusat menghindari duplikasi data yang tidak perlu menggunakan ketergantungan fungsional dan normalisasi.
Isolasi data : Karena data tersebar di berbagai file dan file mungkin dalam format yang berbeda, sulit untuk menulis program aplikasi baru untuk mengambil data yang sesuai.	**Berbagi data :** Database memungkinkan berbagi data di bawah kendalinya oleh sejumlah program aplikasi atau pengguna.

Sulit mengakses data : Pengambilan data rumit karena setiap kali ada permintaan baru, programmer harus menulis kode/aplikasi yang diperlukan.	**Mudah untuk mengakses data :** Karena berbagi data dan kemandirian data, maka mudah untuk mengakses data.
Keamanan data yang tidak memuaskan: Setiap pengguna sistem harus diizinkan untuk melihat hanya bagian data yang relevan dengan departemennya.	**Keamanan data:** Dalam sistem database, prosedur akses yang tepat diikuti, termasuk skema otentikasi yang tepat dan pemeriksaan tambahan sebelum mengizinkan akses data.
Risiko integritas data: Nilai data harus memenuhi batasan integritas tertentu. Tapi tugas ini menjadi kompleks ketika kendala melibatkan beberapa item data dari file yang berbeda.	**Integritas data:** Kontrol terpusat memastikan bahwa pemeriksaan yang memadai dimasukkan ke dalam database untuk menyediakan integritas.
Akses bersamaan: Banyak sistem memungkinkan pengguna untuk memanipulasi data secara bersamaan. Tetapi manipulasi semacam itu dapat menghasilkan data yang konsisten karena data berada di lokasi yang berbeda dan format yang berbeda.	**Independensi data:** Independensi data memungkinkan perubahan pada satu tingkat database tanpa mempengaruhi tingkat lainnya. Oleh karena itu konsistensi data" tetap terjaga.

Pertanyaan 11: Jelaskan Aturan Codd untuk RDBMS.

Jawaban :

Aturan Codd: Ini adalah aturan berikut.

1. **Aturan Informasi:** Semua data harus disajikan dalam bentuk tabel.
2. **Aturan Akses Pasti :** Semua data harus dapat diakses tanpa ambiguitas. Ini dapat dicapai melalui kombinasi nama tabel, kunci utama, dan nama kolom.
3. **'Freatment of Null Values yang sistematis :** Bidang harus dibiarkan kosong. Ini melibatkan dukungan nilai nol, yang berbeda dari string kosong atau angka dengan nilai nol.
4. **Katalog On-Line Dinamis berdasarkan Model Relasional :** Database relasional harus menyediakan akses ke strukturnya melalui alat yang sama yang digunakan untuk mengakses data.
5. **Aturan Subbahasa Data:** Database harus mendukung setidaknya satu bahasa yang didefinisikan dengan jelas yang mencakup fungsionalitas untuk definisi data, manipulasi data, integritas data, dan kontrol transaksi Database.
6. **Lihat Memperbarui Aturan:** Data dapat disajikan dalam kombinasi logis yang berbeda yang disebut tampilan. Setiap tampilan harus mendukung berbagai manipulasi data yang sama yang memiliki akses langsung ke tabel yang tersedia.
7. **Sisipkan, Perbarui, dan Hapus Tingkat Tinggi:** Data dapat diambil dari database relasional dalam kumpulan data yang dibuat dari beberapa baris dan/atau beberapa

tabel. Aturan ini menyatakan bahwa operasi penyisipan, pembaruan, dan penghapusan harus didukung untuk setiap set yang dapat diambil.

8. **Independensi Data Fisik:** Akses ke database harus tetap konsisten setiap kali perubahan begitu kuat atau akses ke data diubah.
9. **Independensi Data Logis :** Bagaimana data dilihat tidak boleh diubah ketika struktur logis dari Database berubah. Aturan ini sangat sulit untuk dipenuhi.
10. **Independensi Integritas:** Bahasa database (seperti SQL) harus mendukung batasan pada input pengguna yang m ... mempertahankan integritas database. Minimal, semua database mempertahankan dua kendala melalui SQL.
    a. Tidak ada komponen kunci utama yang dapat memiliki nilai null.
    b. Jika kunci asing didefinisikan dalam satu tabel, dan nilai di dalamnya harus ada sebagai kunci utama di tabel lain.
11. **Distribusi Independen:** Seorang pengguna harus sama sekali tidak menyadari apakah database didistribusikan atau tidak.
12. **Aturan Non Subversion:** Seharusnya tidak ada cara untuk mengubah struktur database selain melalui bahasa database baris ganda (seperti SQL). Kebanyakan database saat ini mendukung alat administratif yang memungkinkan beberapa manipulasi langsung dari struktur data.

Pertanyaan 12: Apa perbedaan antara DBMS dan RDBMS

Jawaban :

Tabel 1.9 Perbedaan Antara DBMS dan RDBMS

DBMS	RDBMS
Dalam DBMS hubungan antara tabel teo atau file dipelihara secara terprogram.	Dalam RDBMS hubungan antara dua tabel atau file dapat ditentukan pada saat pembuatan tabel.
DBMS tidak mendukung arsitektur Client/Server.	Sebagian besar RDBMS mendukung arsitektur Client/Server.
DBMS tidak mendukung database terdistribusi.	Sebagian besar RDBMS mendukung database terdistribusi.
Setiap tabel diberi ekstensi dalam DBMS.	Banyak tabel dikelompokkan dalam satu database di RDBMS.
DBMS memungkinkan hanya satu orang untuk mengakses database pada waktu tertentu seperti MS-Access, FoxPro.	RDBMS memungkinkan banyak pengguna mengakses database secara simultan seperti Oracle dan SQL-Server.
Dalam DBMS ada kekurangan keamanan data.	Di RDBMS ada beberapa tingkat keamanan. (i) Masuk pada level O/S. (ii) Tingkat komando. (iii) Tingkat objek.
DBMS dapat memenuhi kurang dari 7 sampai 8 aturan codd.	RDBMS memenuhi lebih dari 7 hingga 8 aturan codd.
Contoh DBMS adalah MS-Access, FoxPro dll.	Contoh RDBMS adalah Oracle, SQL-Server.

Pertanyaan 13: Apa kelebihan Object Relational Database Management System (ORDBMS)?

Jawaban : Kelebihan ORDBMS : Berikut ini adalah :

1. byek-obyek tersebut dapat disimpan dalam database

2. Bahasa DBMS dapat diintegrasikan dengan bahasa pemrograman berorientasi objek. Bahasa bahkan mungkin persis sama dengan yang digunakan dalam aplikasi, yang tidak memaksa programmer untuk memiliki dua representasi dari objeknya.

Pertanyaan 14 : Apa kerugian dari DBMS Hirarki?

Jawaban : Kekurangan HDBMS

(i) Tautan dalam model hierarkis bersifat searah, yaitu, kita dapat berpindah dari induk ke anak saja.

(ii) Model Hirarki tidak mendukung hubungan banyak ke banyak.

(iii) Dalam model ini jika : kita membutuhkan informasi dari tingkat yang lebih rendah maka kita harus mengikuti hierarki sistem yang lengkap.

Pertanyaan 15: Apa itu Data?

Jawaban: Data adalah kumpulan informasi mentah.

Pertanyaan 16 : Apa itu Informasi?

Jawaban: Informasi adalah kumpulan data yang diproses

Pertanyaan 17: Apa itu Database?

Jawaban: Database adalah kumpulan item data yang saling terkait yang dapat diproses oleh satu atau lebih sistem aplikasi.

Pertanyaan 18: Apa itu DBMS?

Jawaban: Sistem Manajemen Database adalah kumpulan data yang saling terkait dan serangkaian program untuk mengakses data tersebut. DBMS adalah sistem perangkat lunak tujuan umum yang memfasilitasi proses mendefinisikan membangun dan memanipulasi database untuk berbagai aplikasi.

Pertanyaan 19: Apa kerugian dari Sistem Berorientasi file?

Jawaban: Sistem berorientasi file yang khas didukung oleh sistem operasi konvensional. Catatan permanen disimpan dalam berbagai file dan sejumlah program aplikasi yang berbeda ditulis untuk mengekstrak catatan dari dan menambahkan catatan ke file yang sesuai. Berikut ini adalah kelemahan dari sistem berorientasi file:

(i) **Redundansi dan Inkonsistensi Data:** Karena file dan program aplikasi dibuat oleh pemrogram yang berbeda dalam jangka waktu yang lama, file tersebut cenderung memiliki format yang berbeda dan program dapat ditulis dalam beberapa bahasa pemrograman. Selain itu, informasi yang sama dapat diduplikasi di beberapa tempat. Redundansi ini menyebabkan penyimpanan dan biaya akses yang lebih tinggi. Selain itu, ini dapat menyebabkan inkonsistensi data, yaitu, berbagai salinan data yang sama mungkin tidak lagi cocok.

(ii) **Kesulitan dalam mengakses data** : Lingkungan pemrosesan file konvensional tidak memungkinkan data yang diperlukan diambil dengan cara yang nyaman dan efisien. Sistem pengambilan data yang lebih baik harus dikembangkan untuk penggunaan umum.

(iii) **Isolasi data :** Karena data tersebar di berbagai file, dan file mungkin dalam format yang berbeda, sulit untuk menulis program aplikasi baru untuk mengambil data yang sesuai.

(iv) **Anomali akses serentak :** Untuk meningkatkan kinerja sistem secara keseluruhan dan memperoleh waktu respons yang lebih cepat, banyak sistem memungkinkan banyak pengguna untuk memperbarui data secara bersamaan. Dalam lingkungan seperti itu, interaksi pembaruan bersamaan dapat menghasilkan data yang tidak konsisten.

(v) **Masalah keamanan:** Tidak setiap pengguna sistem Database harus dapat mengakses semua data. Misalnya, dalam sistem perbankan, personel penggajian hanya memerlukan bagian database yang memiliki informasi tentang berbagai pegawai bank. Mereka tidak memerlukan akses ke informasi tentang rekening pelanggan. Sulit untuk menegakkan batasan keamanan seperti itu.

(vi) **Masalah integritas :** Nilai data yang disimpan dalam database harus memenuhi jenis kendala konsistensi tertentu. Misalnya, saldo rekening bank mungkin tidak pernah jatuh di bawah jumlah yang ditentukan. Kendala ini diterapkan dalam sistem dengan menambahkan kode yang sesuai di berbagai program aplikasi. Ketika batasan baru ditambahkan, sulit untuk mengubah program untuk menegakkannya. Masalahnya diperparah ketika kendala melibatkan beberapa item data untuk file yang berbeda.

(vii) **Masalah atomisitas:** Sistem komputer seperti perangkat mekanis atau listrik lainnya dapat mengalami kegagalan. Dalam banyak aplikasi, sangat penting untuk memastikan bahwa sekali kegagalan telah terjadi dan telah terdeteksi, data dikembalikan ke keadaan konsisten yang ada sebelum kegagalan.

Pertanyaan 20: Apa kelebihan DBMS dibandingkan Sistem Berorientasi File?

Menjawab : Berikut ini adalah kelebihan DBMS dibandingkan sistem berorientasi file.

1. **Redundansi Data:** Kesulitan utama adalah bahwa banyak aplikasi menggunakan file data khusus mereka sendiri. Dengan demikian, beberapa item data umum untuk beberapa aplikasi. Di bank, misalnya, nama pelanggan yang sama mungkin muncul dalam file rekening giro, file rekening tabungan, dan file pinjaman angsuran. Selain itu, meskipun selalu nama pelanggan, bidang terkait sering kali memiliki nama yang berbeda di berbagai file akun. Dengan demikian, CNAME pada berkas rekening giro menjadi SNAME pada berkas rekening tabungan dan INAME pada berkas pinjaman angsuran. Bidang yang sama juga memiliki panjang yang berbeda di berbagai file. Misalnya, CNAME dapat berisi hingga 20 karakter, tetapi SNAME dan INAME mungkin dibatasi hingga 15 karakter. Redundansi ini meningkatkan biaya overhead pemeliharaan dan penyimpanan. Redundansi data juga meningkatkan risiko inkonsistensi di antara berbagai versi data umum. Misalkan nama pelanggan diubah. Bidang nama mungkin akan segera diperbarui dalam file rekening giro, diperbarui minggu depan di file rekening tabungan dan salah diperbarui dalam file pinjaman angsuran. Seiring waktu, perbedaan tersebut dapat menyebabkan penurunan serius dalam kualitas informasi yang terkandung dalam file data. Sistem Database dapat menghilangkan redundansi data, karena semua aplikasi berbagi kumpulan data yang sama. Informasi penting seperti nama pelanggan akan muncul hanya sekali dalam

database. Dengan demikian, kita dapat memasukkan nama atau mengubah sekali dan mengetahui bahwa aplikasi akan mengakses data yang konsisten.

2. **Kontrol Data Buruk :** Dalam sistem file, tidak ada kontrol terpusat di tingkat elemen data. Sangat umum untuk elemen data yang sama memiliki banyak nama, tergantung pada file yang dimilikinya. Pada tingkat yang lebih mendasar, selalu ada kemungkinan bahwa berbagai departemen perusahaan akan tidak konsisten dalam terminologi mereka.
3. **Kemampuan Manipulasi Data yang Tidak Memadai : File** seqential yang diindeks memungkinkan aplikasi untuk mengakses catatan tertentu dengan kunci seperti produk 10. Misalnya, jika kita mengetahui ID Produk untuk tabel, maka akan mudah untuk mengakses catatan dalam tabel. Misalkan kita menginginkan satu set record. Tidak mungkin untuk mendapatkan satu set catatan menggunakan sistem file karena mereka tidak dapat menyediakan koneksi yang kuat antara data dalam file yang berbeda. Sistem database secara khusus dikembangkan untuk membuat keterkaitan data dalam file yang berbeda.
4. **Upaya Pemrograman Berlebihan :** Program aplikasi baru sering kali membutuhkan kumpulan definisi file yang sama sekali baru. Meskipun file yang ada mungkin berisi beberapa data yang diperlukan, aplikasi seringkali membutuhkan sejumlah item data lainnya. Akibatnya, programmer harus mengkode ulang definisi item data yang dibutuhkan dari file yang ada serta definisi semua item data baru. Jadi, dalam sistem berorientasi file, ada saling ketergantungan yang berat antara program dan data. Database menyediakan pemisahan antara program dan data, sehingga program dapat agak independen dari detail definisi data. Dengan menyediakan akses ke kumpulan data bersama dan dengan mendukung bahasa manipulasi data yang kuat, sistem Database menghilangkan sejumlah besar pemrograman awal dan pemeliharaan.

Pertanyaan 21: Apa itu Instance dan Skema?

Jawaban : Instance : Kumpulan informasi yang disimpan dalam database pada saat tertentu disebut Instance database. **Skema**: Desain keseluruhan database disebut skema database.

Pertanyaan 22: Apa itu Independensi Data?

Jawaban : Independensi Data: Kemampuan untuk memodifikasi definisi skema dalam satu tingkat tanpa mempengaruhi definisi skema di tingkat berikutnya disebut Independensi Data. Ada dua tingkat independensi data:

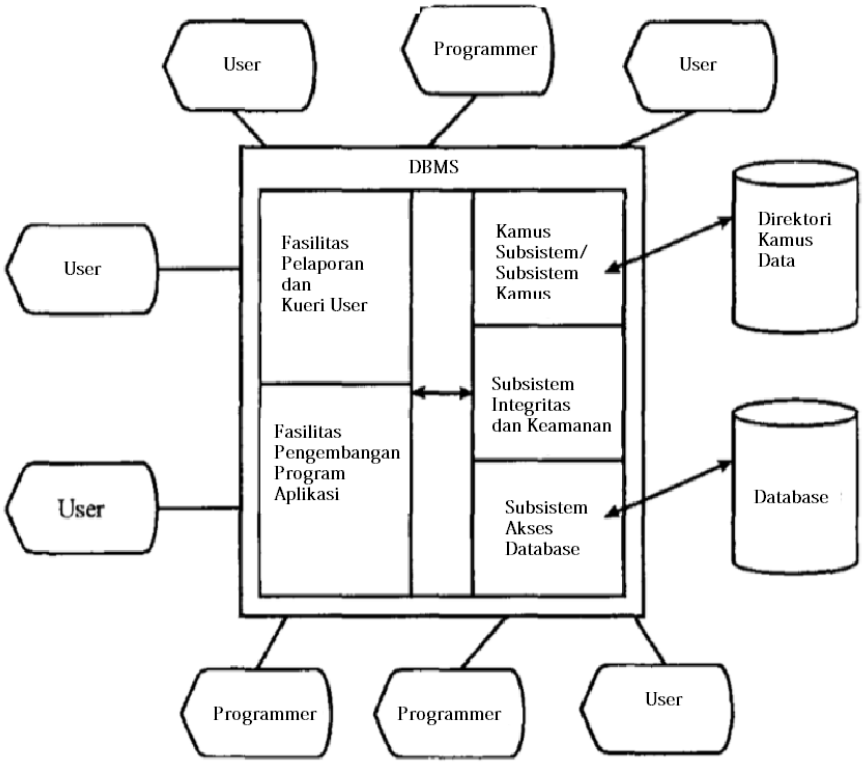
(i) **Independensi Data Fisik :** Kemampuan untuk memodifikasi skema fisik tanpa menyebabkan program aplikasi ditulis ulang.

(ii) **Independensi Data Logis:** Kemampuan untuk memodifikasi skema konseptual tanpa menyebabkan program aplikasi ditulis ulang. Independensi Data Logis lebih sulit dicapai daripada independensi data fisik karena program aplikasi sangat bergantung pada struktur logika data yang mereka akses.

Pertanyaan 23: Apa itu model data?

Jawaban : Model Data : Sebuah metode konseptual penataan data disebut Data Model. Pengembangan sistem berdasarkan tiga

Pertanyaan 24: Jelaskan komponen-komponen Sistem Database.



Gambar 1.37 Komponen Sistem Database

Jawaban : Sebuah sistem database yang lengkap dalam suatu organisasi terdiri dari empat komponen.

(i) **Perangkat Keras (Hardware):** Perangkat keras adalah seperangkat perangkat fisik tempat database berada. Ini terdiri dari satu atau lebih komputer, disk drive, terminal CRT, printer, driver tape, kabel penghubung, dll. Komputer yang digunakan untuk memproses data dalam database dapat berupa mainframe, komputer mini atau komputer pribadi. Mainframe dan komputer mini secara tradisional telah digunakan secara stand-alone untuk mendukung banyak pengguna mengakses database umum. Komputer pribadi sering digunakan dengan database yang berdiri sendiri dikendalikan dan diakses oleh satu pengguna. Driver disk adalah mekanisme penyimpanan utama untuk database. Komputer desktop, terminal CRT, dan printer digunakan untuk memasukkan dan mengambil informasi dari database. Keberhasilan sistem database sangat bergantung pada kemajuan teknologi perangkat keras. Memori utama dan penyimpanan disk dalam jumlah yang sangat besar diperlukan untuk memelihara dan mengontrol sejumlah besar data yang disimpan dalam database.

(ii) **Perangkat Lunak :** Sistem Database mencakup dua jenis perangkat lunak:

a. Perangkat lunak manajemen Database tujuan umum yang biasanya disebut sistem manajemen Database (DBMS).
b. Perangkat lunak aplikasi yang menggunakan fasilitas DBMS untuk memanipulasi database untuk mencapai fungsi bisnis tertentu.

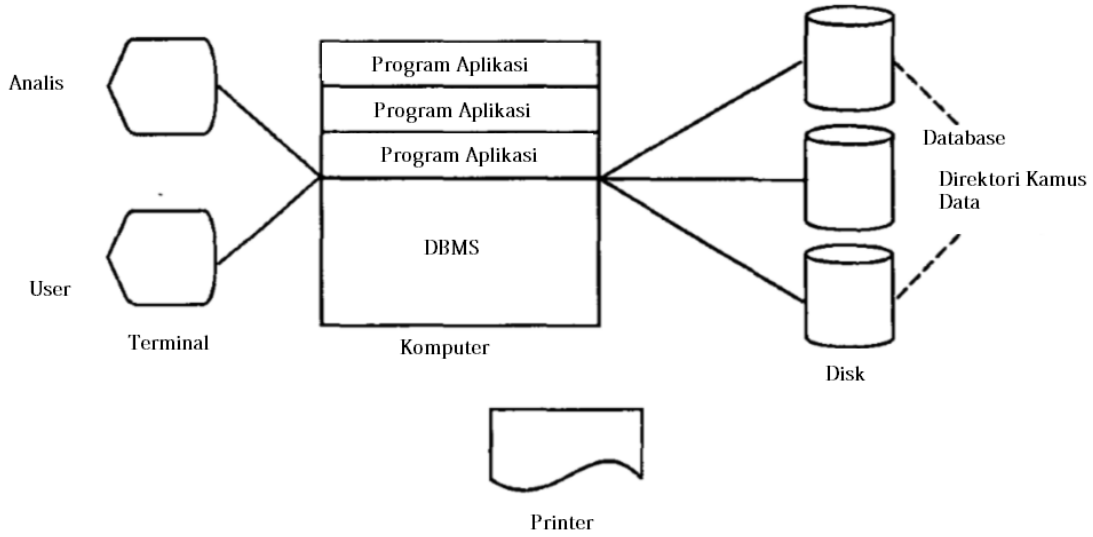
Perangkat lunak aplikasi umumnya ditulis oleh pemrogram untuk memecahkan masalah perusahaan tertentu. Ini dapat ditulis dalam bahasa seperti COBOL atau C atau mungkin ditulis dalam bahasa yang disediakan oleh DBMS seperti SQL. Perangkat lunak aplikasi menggunakan fasilitas DBMS untuk mengakses dan memanipulasi data dalam database yang menyediakan laporan atau dokumen yang diperlukan untuk kebutuhan informasi dan pemrosesan perusahaan. DBMS adalah perangkat lunak sistem yang mirip dengan sistem operasi. Ini menyediakan sejumlah layanan kepada pengguna akhir dan pemrogram. DBMS biasanya menyediakan sebagian besar layanan berikut.

1. Definisi data pusat dan fasilitas kontrol data yang dikenal sebagai kamus/direktori data atau katalog.
2. Mekanisme keamanan dan integritas data. 3. Akses data serentak untuk banyak pengguna.
3. Kemampuan manipulasi dan pelaporan kueri data berorientasi pengguna.
4. Kemampuan pengembangan sistem aplikasi berorientasi pemrogram

(iii) **Data**: Tidak ada sistem Database tanpa data. Data dapat dikumpulkan dan dimasukkan ke dalam database sesuai dengan struktur yang ditentukan.

(iv) **People**: Dua tipe orang yang berbeda yang berhubungan dengan database. Mereka adalah:

a. Pengguna: Eksekutif, Manajer, Staf, Staf Klerikal.
b. Praktisi: Database Administrator, Programmer.



Gambar 1.38 Sistem Database

Pertanyaan 25: Apa itu Kamus Data?

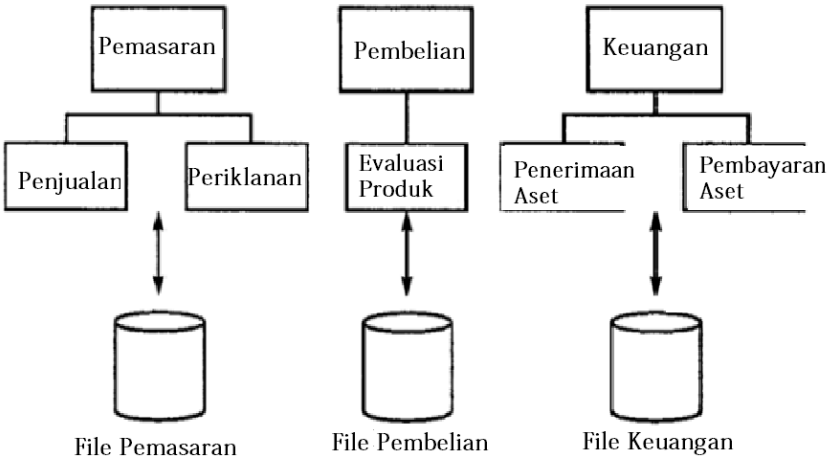
Jawaban: Sebuah kamus data / subsistem direktori melacak definisi semua item data dalam database. Ini termasuk item data tingkat dasar (bidang), struktur data tingkat grup dan record dan tabel relasional. Itu melacak hubungan yang ada antara berbagai struktur data. Itu memelihara indeks yang digunakan untuk mengakses data dengan cepat. Itu juga melacak definisi format layar dan laporan yang mungkin digunakan oleh berbagai program aplikasi.

Pertanyaan 26: Menjelaskan Berbagi Data.

Jawaban: Data tanpa berbagi:

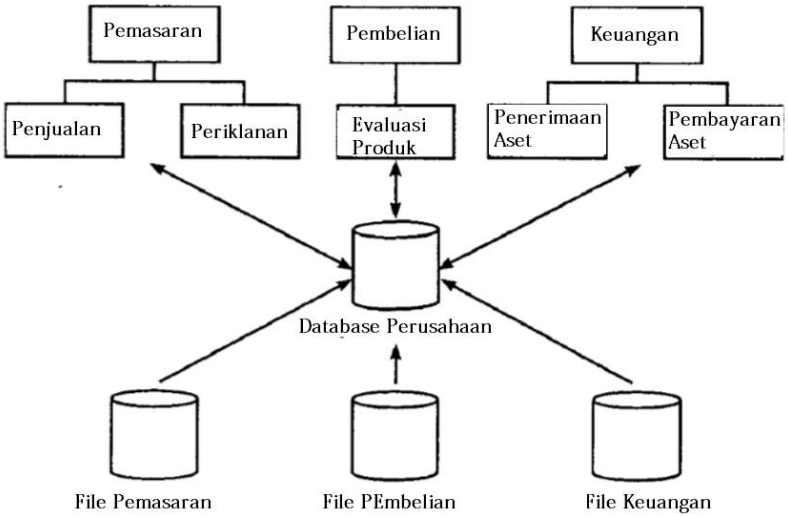
Perbedaan paling signifikan antara sistem berbasis file dan sistem Database adalah bahwa data dibagikan. Ada tiga jenis berbagi data:

(i) **Berbagi antar Unit Fungsional:** Berbagi data menunjukkan bahwa orang-orang di area fungsional yang berbeda menggunakan kumpulan data yang sama, masing-masing aplikasi mereka sendiri. Tanpa berbagi data, grup pemasaran mungkin memiliki file data mereka, grup pembelian milik mereka, grup akuntansi milik mereka dan seterusnya. Setiap kelompok hanya mendapat manfaat dari datanya sendiri. Data gabungan lebih berharga daripada jumlah data dalam file terpisah. Setiap grup tidak hanya terus memiliki akses ke datanya sendiri tetapi, dalam batas kendali yang wajar, mereka juga memiliki akses ke data lain. Konsep menggabungkan data untuk penggunaan umum disebut integrasi data.



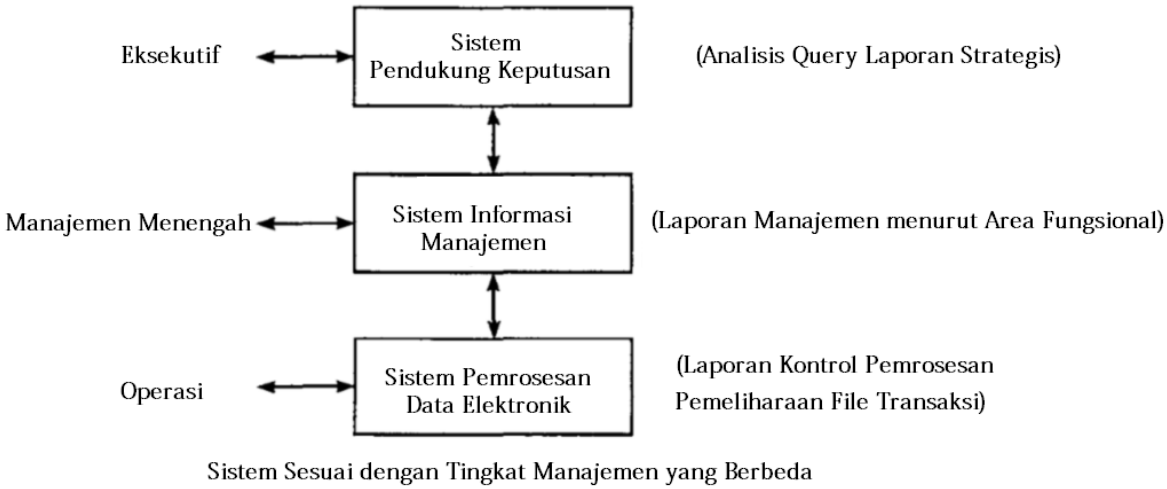
Gambar 1.39 Integrasi Data

(ii) **Berbagi data antara Tingkat Pengguna yang Berbeda:** Tingkat pengguna yang berbeda perlu berbagi data. Tiga tingkat yang berbeda dari pengguna biasanya dibedakan: operasi, manajemen menengah dan eksekutif. Tingkat ini sesuai dengan tiga jenis sistem bisnis otomatis yang telah berkembang selama tiga dekade terakhir:



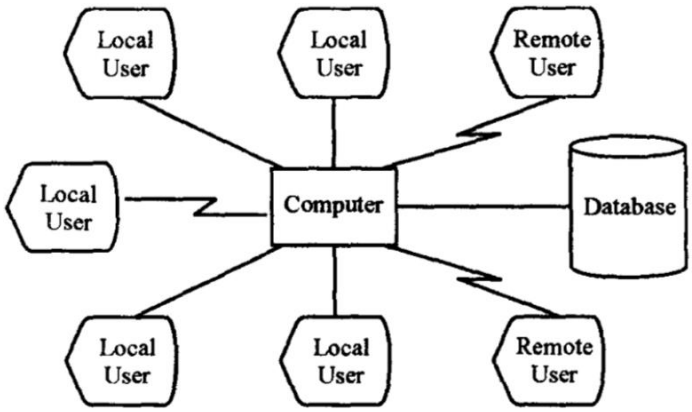
Gambar 1.40 Tingkat Tiga Jenis Sistem Bisnis Otomatis

a. *Electronic Data Processing/Pemrosesan Data Elektronik (EDP):* EDP pertama kali diterapkan ke tingkat operasional organisasi yang lebih rendah untuk mengotomatisasi dokumen. Karakteristik dasarnya meliputi:
    - Fokus pada data, penyimpanan, pemrosesan, dan aliran di tingkat operasional.
    - Pemrosesan transaksi yang efisien.
    - Ringkasan laporan untuk manajemen.
b. *Management Information System /Sistem Informasi Manajemen (SIM):* Pendekatan SIM meningkatkan fokus pada kegiatan sistem informasi dengan penekanan tambahan pada integrasi dan perencanaan fungsi sistem informasi. Ini termasuk :
    - Fokus informasi yang ditujukan untuk manajer menengah.
    - Integrasi pekerjaan EDP berdasarkan fungsi bisnis seperti MIS produk, MIS pemasaran, MIS personel, dll.
    - Penyelidikan dan pembuatan laporan biasanya dengan database.
c. *Sistem Pendukung Keputusan*/*Decision Support System*: DSS terfokus masih lebih tinggi dalam organisasi dengan penekanan pada karakteristik sebagai berikut:
    - Fokus keputusan, ditujukan untuk manajer puncak dan pengambil keputusan eksekutif.
    - Penekanan pada fleksibilitas, kemampuan beradaptasi dan respon cepat.
    - Dukungan untuk gaya pengambilan keputusan personel manajer individu.



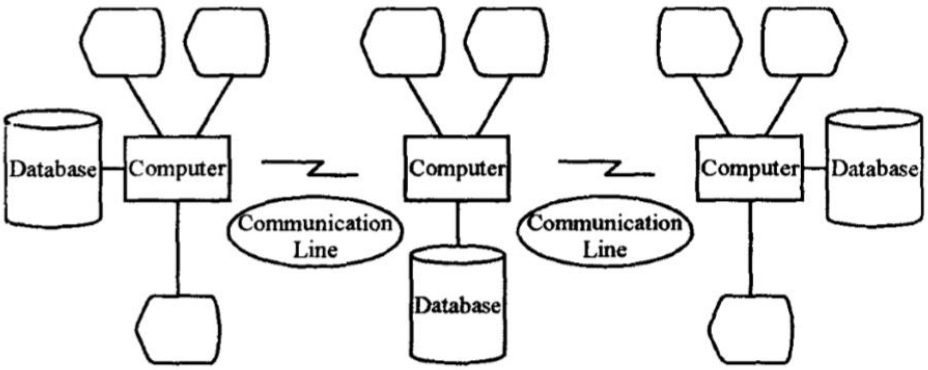
Gambar 1.41 Sistem Sesuai Dengan Tingkat Manajemen yang Berbeda

(iii) **Berbagi data antar Lokasi yang Berbeda** : Perusahaan dengan beberapa lokasi memiliki data penting yang tersebar di wilayah geografis yang luas. Berbagi data ini adalah masalah yang signifikan. Database **terpusat** secara fisik terbatas pada satu lokasi, dikendalikan oleh satu komputer. Sebagian besar fungsi untuk database dibuat dan diselesaikan dengan lebih mudah jika database terpusat. Artinya, lebih mudah untuk memperbarui, mencadangkan, membuat kueri, dan mengontrol akses ke database jika kita tahu persis di mana letaknya dan y, perangkat lunak yang mengontrolnya.



Gambar 1.42 Struktur Database Distribusi

Sistem Database terdistribusi terdiri dari beberapa sistem Database yang berjalan di situs lokal yang dihubungkan oleh jalur komunikasi. Sebuah query atau update kemudian tidak lagi menjadi proses tunggal yang dikendalikan oleh satu modul perangkat lunak, tetapi satu set proses bekerja sama yang berjalan di beberapa situs yang dikendalikan oleh modul perangkat lunak independen. Agar sistem database terdistribusi berfungsi secara efisien, teknologi komunikasi yang memadai harus tersedia dan DBMS dalam sistem harus dapat berkomunikasi saat berinteraksi dengan fasilitas komunikasi.



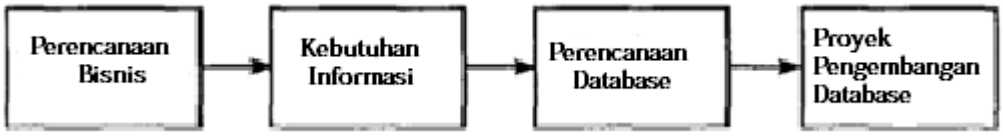
Gambar 1.43 Struktur Database Distribusi

Pertanyaan 27: Jelaskan Perencanaan Database Strategis.

Jawaban: Perencanaan Database adalah upaya strategis perusahaan untuk menentukan kebutuhan informasi dalam jangka waktu yang lama. Sebuah proyek perencanaan database yang sukses akan mendahului proyek operasional untuk merancang dan mengimplementasikan database baru untuk memenuhi kebutuhan informasi organisasi.

Kebutuhan Perencanaan Database: Perencanaan database memiliki keuntungan yang signifikan:

- Ini mengungkapkan pemahaman manajemen saat ini tentang sumber daya informasi.
- Ini mengidentifikasi dan membenarkan kebutuhan sumber daya.
- Ini mengidentifikasi peluang untuk manajemen sumber daya yang efektif termasuk kolaborasi antar departemen atau divisi dalam organisasi.
- Ini menentukan rencana aksi untuk mencapai tujuan.
- Hal ini dapat memberikan rangsangan yang kuat dan rasa arah kepada karyawan di semua tingkatan, memfokuskan upaya mereka, meningkatkan produktivitas mereka dan membuat mereka merasa bahwa mereka adalah bagian asli dari perusahaan.



Gambar 1.44 Perencanaan Database

Proyek Perencanaan Database: Perencanaan Database Strategis diprakarsai oleh manajemen senior. Mereka mengalokasikan sumber daya dan mengidentifikasi personel untuk berpartisipasi dalam proyek. Dengan komisi mereka dari manajemen, anggota tim memiliki sumber daya yang dibutuhkan untuk melaksanakan proyek yang sukses. Tim proyek harus memiliki pengalaman yang luas dalam sistem informasi dan area fungsional lainnya dari perusahaan. Sekelompok empat anggota penuh waktu, dua dari sistem informasi dan dua kenal dengan sebagian besar area lain dari perusahaan. Semua anggota tim harus menjadi karyawan yang terampil dan dihormati, karena pekerjaan mereka akan berdampak besar pada organisasi selama bertahun-tahun. Jika mereka tidak ahli dalam metodologi untuk melaksanakan penelitian, konsultan luar harus dipekerjakan sebagai penasihat untuk

melatih tim dalam metodologi yang sesuai. Pemimpin tim proyek harus menjadi konsultan tetapi karyawan tetap dan mungkin kepala administrasi database.

Selama proyek, tim berinteraksi dengan manajer senior dari semua area pengguna utama. Pengguna akhir senior mengidentifikasi proses utama, aktivitas, dan entitas yang digunakan dalam pemrosesan informasi manual atau otomatis. Tim proyek mensintesis data ini ke dalam model informasi perusahaan yang disertakan sebagai bagian dari rencana Database yang komprehensif. Sebuah laporan yang mencakup setidaknya lima berikutnya harus disampaikan kepada manajemen senior. Laporan ini akan mencakup analisis hal-hal berikut: • Kebutuhan informasi bidang fungsional.

- Kebutuhan informasi dari tingkat manajemen yang berbeda.
- Kebutuhan informasi lokasi geografis.
- Model kebutuhan informasi ini.
- Volume data yang diantisipasi berdasarkan proyek lokasi geografis untuk periode yang diteliti.
- Perkiraan awal biaya yang terkait dengan peningkatan sistem.
- Rekomendasi untuk pengembangan rinci database baru atau yang disempurnakan dengan jadwal.

Pertanyaan 28: Jelaskan fungsi DBA.

Jawaban: Database Administrator adalah orang yang bertanggung jawab untuk mengontrol dan melindungi data. DBA harus mengoordinasikan desain database, memandu pengembangan dan penerapan prosedur keamanan data, melindungi integritas nilai data, dan memastikan kinerja sistem memuaskan. Dalam sebuah organisasi kecil, satu orang melaksanakan semua tanggung jawab ini. Seringkali, fungsi ini diberikan kepada sekelompok orang. Hal ini kemungkinan besar terjadi dalam organisasi besar di mana tanggung jawab DBA dibagi di antara beberapa orang yang dikelola oleh seorang administrator kepala.

Fungsi DBA meliputi:

(i) **Desain Database:** Rancangan Database Konseptual terutama terdiri dari pendefinisian elemen data yang akan dimasukkan ke dalam Database, hubungan yang ada di antara mereka dan batasan nilai yang berlaku. Batasan nilai adalah aturan yang menentukan nilai yang diizinkan untuk item data tertentu. Desain Database Fisik menentukan struktur fisik Database dan mencakup keputusan seperti metode akses apa yang akan digunakan untuk mengambil data dan indeks apa yang akan dibangun untuk meningkatkan kinerja sistem.

(ii) **Pelatihan Pengguna:** DBA bertanggung jawab untuk mendidik pengguna dalam struktur database dan aksesnya melalui DBMS. Hal ini dapat dilakukan dalam sesi pelatihan formal dengan berinteraksi dengan pengguna untuk membuat tampilan database, melalui manual pengguna dan memo berkala dan melalui pusat informasi perusahaan. Pusat informasi adalah area di mana pengguna diberikan fasilitas untuk melakukan komputasi mereka sendiri.

(iii) **Keamanan dan Integritas Database:** Konsep menggabungkan data organisasi ke dalam satu kumpulan umum yang dapat diakses oleh semua memiliki kelebihan dan kekurangan. Keuntungan yang jelas dari berbagi data diimbangi oleh

kerugian bahwa data dapat disalahgunakan atau dirusak oleh pengguna yang tidak memiliki tanggung jawab dan wewenang asli atas data tersebut. DBA menyediakan prosedur dan kontrol untuk mencegah penyalahgunaan data. Akses Database pada akhirnya dikendalikan oleh mekanisme kata sandi, di mana pengguna yang mencoba mengakses memberikan kata sandi yang memvalidasi sistem. Sistem mengizinkan pengguna yang divalidasi hanya hak akses yang dicatat dalam kamus data. DBA bertanggung jawab untuk menetapkan kata sandi dan mengontrol hak istimewa. Integritas data mengacu pada masalah menjaga akurasi dan konsistensi nilai data. Mekanisme keamanan seperti kata sandi dan tampilan data melindungi integritas data.

(iv) **Kinerja Sistem Database:** Suatu sistem Database yang diakses secara bersamaan oleh banyak pengguna terkadang merespons dengan sangat lambat karena masalah fisik yang terkait dengan pengguna yang bersaing untuk sumber daya yang sama bukanlah masalah sepele. Dengan demikian, staf DBA harus mencakup personel yang terampil secara teknis yang dapat mendiagnosis dan memecahkan masalah waktu respons sistem. Solusi masalah mungkin akuisisi perangkat keras, penataan ulang fisik data pada disk konstruksi indeks untuk akses cepat ke data volume tinggi atau penulisan perangkat lunak khusus untuk meningkatkan waktu akses. DBA dapat memutuskan untuk mempertahankan salinan data yang berlebihan untuk meningkatkan kinerja sistem. Redundansi seperti itu harus dikendalikan; namun masalah inkonsistensi data akan dihindari.

Pertanyaan 29: Apa Risiko dan Biaya database?

Jawaban: Sistem Database memiliki kelemahan. Berikut ini adalah Risiko dan Biaya dari database:

(i) **Konflik Organisasi:** Pengumpulan data dalam database umum mungkin tidak layak secara politis di beberapa organisasi. Kelompok pengguna tertentu mungkin tidak mau melepaskan kendali atas data mereka sejauh yang diperlukan untuk mengintegrasikan data. Selain itu, risiko yang terlibat dalam berbagi data misalnya, bahwa satu grup dapat merusak data grup lain - dan potensi masalah sistem yang dapat membatasi akses grup ke datanya sendiri dapat dianggap lebih merepotkan daripada menguntungkan. Masalah orang seperti itu dapat mencegah implementasi sistem database yang efektif.

(ii) **Kegagalan Proyek Pengembangan:** Karena berbagai alasan, proyek untuk mengembangkan sistem Database mungkin gagal. Terkadang manajemen tidak sepenuhnya yakin akan nilai sistem Database sejak awal. Proyek Database yang tampaknya memakan waktu terlalu lama dapat dihentikan. Sebuah proyek yang cakupannya terlalu besar mungkin hampir mustahil untuk diselesaikan dalam waktu yang wajar. Sekali lagi, manajemen dan pengguna menjadi kecewa dan proyek gagal. Selama proyek berlangsung, personel kunci mungkin tiba-tiba meninggalkan perusahaan. Jika personel pengganti tidak dapat ditemukan, maka proyek mungkin tidak akan berhasil diselesaikan.

(iii) **Kegagalan Sistem:** Ketika sistem mati, semua pengguna yang terlibat langsung dalam mengakses harus menunggu sampai sistem berfungsi kembali. Ini mungkin membutuhkan penantian yang lama. Selain itu, jika sistem atau perangkat lunak aplikasi gagal, mungkin ada kerusakan permanen pada database. Oleh karena itu, sangat penting untuk mengevaluasi dengan cermat semua perangkat lunak yang akan memiliki efek langsung pada Database untuk memastikan bahwa itu sebebas mungkin dari kesalahan. Jika organisasi tidak menggunakan database, organisasi tidak terkena risiko ini, karena data dan perangkat lunaknya didistribusikan.

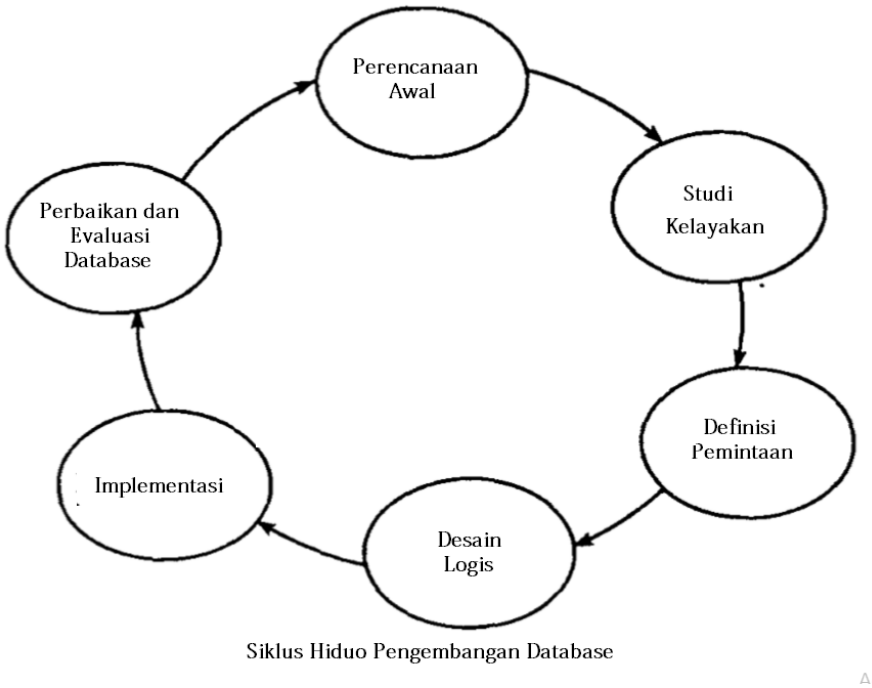
(iv) **Biaya Overhead:** Pendekatan database mungkin memerlukan investasi baik dalam perangkat keras maupun perangkat lunak. Perangkat keras untuk menjalankan DBMS besar harus efisien dan umumnya akan membutuhkan lebih banyak memori utama dan penyimpanan disk daripada sistem berbasis file yang lebih sederhana. Tape drive untuk membuat cadangan database dengan cepat juga diperlukan. Selain itu, DBMS itu sendiri mungkin cukup mahal. DBMS mungkin juga memerlukan peningkatan biaya operasi" karena memerlukan waktu eksekusi yang lebih lama. Misalnya, sistem aplikasi yang menggunakan DBMS biasanya akan mengeksekusi lebih lambat daripada sistem yang tidak menggunakan DBMS.

(v) **Kebutuhan Personil yang canggih:** Administrasi database fungsi membutuhkan personel terampil yang mampu mengoordinasikan kebutuhan kelompok pengguna yang berbeda, merancang tampilan, mengintegrasikan pandangan tersebut ke dalam satu skema, menetapkan prosedur pemulihan data, dan menyempurnakan struktur fisik database untuk memenuhi kriteria kinerja yang dapat diterima. Ada risiko terlibat dalam identifikasi personel untuk DBA, karena jika tidak ada orang yang memiliki keterampilan yang diperlukan dapat ditemukan, fungsi DBA mungkin tidak dapat dilakukan dengan benar. Hal ini dapat mengakibatkan masalah yang signifikan dan bahkan dapat mengakibatkan kegagalan implementasi database

Pertanyaan 30: Sebutkan dan jelaskan tahapan-tahapan DDLC yang berbeda

Jawaban: DDLC (Database Development Life Cycle): Merupakan proses untuk merancang, mengimplementasikan dan memelihara sistem Database. Ini terdiri dari enam tahap:

1. Desain awal
2. Desain kelayakan
3. Definisi persyaratan
4. Desain konseptual
5. Implementasi
6. Evaluasi dan pemeliharaan database.



Gambar 1.45 Siklus Hiduo Pengembangan Database

Perencanaan awal: Ini adalah sistem Database khusus yang terjadi selama proyek perencanaan Database strategis. Setelah proyek implementasi Database dimulai, model informasi umum yang dihasilkan selama perencanaan Database ditinjau dan ditingkatkan jika diperlukan. Selama proses ini, perusahaan mengumpulkan informasi untuk menjawab pertanyaan-pertanyaan berikut:

1. Berapa banyak program aplikasi yang digunakan, dan fungsi apa yang mereka lakukan?
2. File apa yang terkait dengan masing-masing aplikasi ini?
3. Aplikasi dan file baru apa yang sedang dikembangkan?

Informasi ini dapat digunakan untuk membangun hubungan antara aplikasi saat ini dan untuk mengidentifikasi penggunaan informasi aplikasi. Hal ini juga membantu untuk mengidentifikasi kebutuhan sistem masa depan dan untuk menilai manfaat ekonomi dari sistem database.

Studi Kelayakan: Studi kelayakan melibatkan penyusunan laporan tentang isu-isu berikut:

1. *Kelayakan teknologi:* Apakah teknologi yang sesuai tersedia untuk mendukung pengembangan database?
2. *Kelayakan operasional :* Apakah perusahaan memiliki personel, anggaran dan keahlian internal untuk membuat sistem database berhasil?
3. *Kelayakan ekonomi:* Dapatkah manfaat diidentifikasi? Apakah sistem yang diinginkan akan menguntungkan dari segi biaya? Dapatkah biaya dan manfaat diukur?

Definisi Persyaratan: Ini melibatkan pendefinisian ruang lingkup database yang mengidentifikasi manajemen dan persyaratan informasi area fungsional dan menetapkan persyaratan perangkat keras/lunak. Persyaratan informasi ditentukan dari tanggapan kuesioner, wawancara dengan manajer dan pengguna klerikal serta laporan dan formulir yang saat ini digunakan.

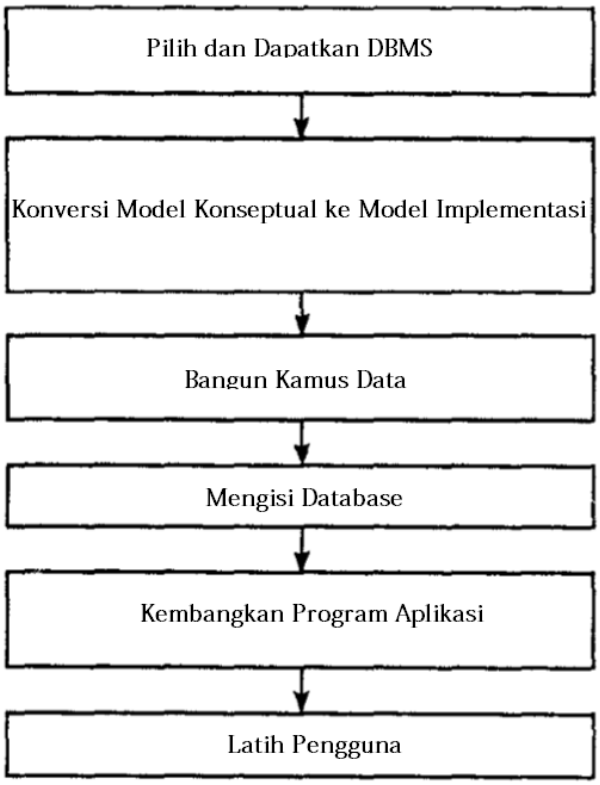
Desain Konseptual: Tahap desain konseptual menciptakan skema konseptual untuk database. Spesifikasi dikembangkan ke titik di mana implementasi dapat dimulai. Selama tahap ini, model tampilan pengguna yang terperinci dibuat dan diintegrasikan ke dalam model data konseptual yang merekam semua elemen data perusahaan untuk dipelihara dalam database.

Implementasi: Selama implementasi Database, DBMS dipilih dan diperoleh. Kemudian model konseptual rinci diubah menjadi model implementasi DBMS, kamus data dibangun, database terisi, program aplikasi dikembangkan dan pengguna dilatih.

Evaluasi dan Pemeliharaan Database: Evaluasi melibatkan mewawancarai pengguna untuk menentukan apakah ada kebutuhan data yang tidak terpenuhi. Perubahan dilakukan sesuai kebutuhan. Seiring waktu sistem dipertahankan melalui pengenalan perangkat tambahan dan penambahan program baru dan elemen data sebagai kebutuhan bisnis berubah dan berkembang.

Pertanyaan 31: Jelaskan Arsitektur Database Tiga Tingkat.

Jawaban: Ini adalah struktur database standar yang terdiri dari Tiga-Level.



Gambar 1.46 Struktur Database Standar

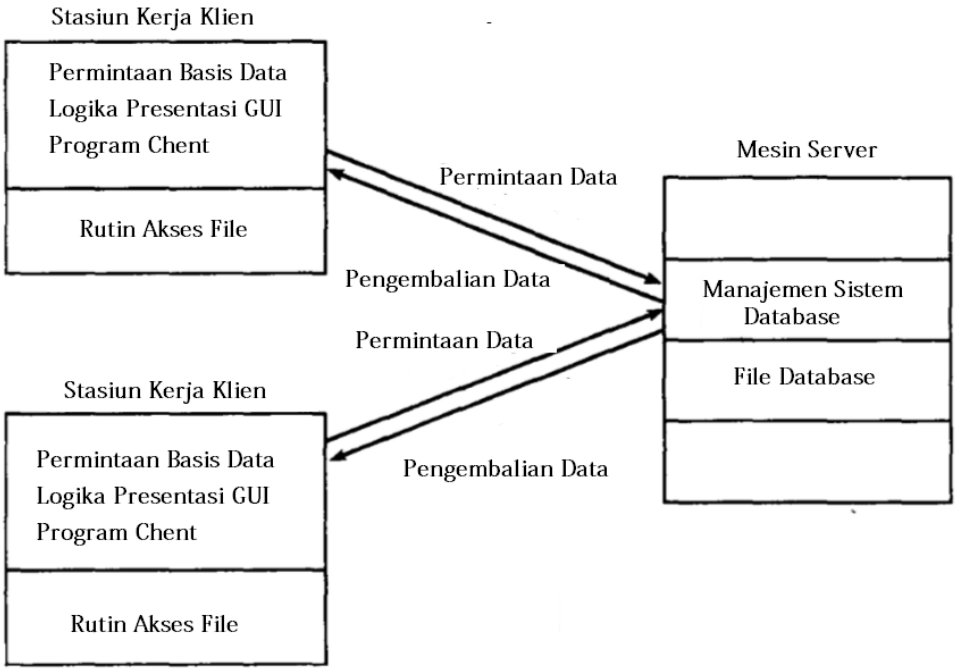
(i) **Tingkat Konseptual:** Ini adalah tingkat di mana desain database konseptual dilakukan. Desain Database konseptual melibatkan analisis kebutuhan informasi pengguna dan definisi item data yang diperlukan untuk memenuhinya. Hasil desain konseptual adalah skema konseptual, deskripsi tunggal dan logis dari semua elemen data dan hubungannya.

(ii) **Tingkat Eksternal:** Ini terdiri dari pandangan pengguna dari database. Setiap grup pengguna yang ditentukan akan memiliki tampilan databasenya sendiri. Masing-masing tampilan ini memberikan deskripsi berorientasi pengguna dari elemen data dan hubungan yang terdiri dari tampilan tersebut. Hal ini dapat langsung diturunkan dari skema konseptual.

(iii) **Tingkat internal:** Tingkat internal menyediakan tampilan fisik dari database-disk drive, alamat fisik, indeks, pointer dan sebagainya. Level ini merupakan tanggung jawab perancang Database fisik, yang memutuskan perangkat fisik mana yang akan berisi data, metode akses apa yang akan digunakan untuk mengambil dan memperbarui data, dan tindakan apa yang akan diambil untuk mempertahankan atau meningkatkan kinerja datahase.

Pertanyaan 32: Jelaskan Arsitektur Two-Tier dan Three-Tier dari DBMS.

Jawaban:

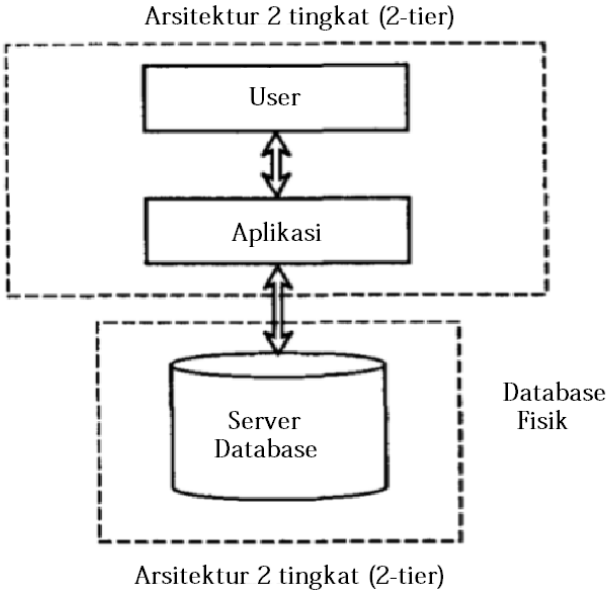


Gambar 1.47 Arsitektur Two-Tier dan Three-Tier dari DBMS

1. **Arsitektur client/server Two-Tier untuk DBMS**: Dalam arsitektur dua tingkat, aplikasi dipartisi menjadi komponen yang berada di mesin klien, yang membangkitkan fungsionalitas sistem Database pada mesin server melalui pernyataan bahasa kueri. Standar antarmuka program aplikasi digunakan untuk interaksi antara klien dan server. Arsitektur dua tingkat Client/Server memiliki dua komponen penting:
    a. PC klien dan
    b. Server database

2. **Pertimbangan Tingkat**
    a. Program klien mengakses database secara langsung:
        - Memerlukan perubahan kode untuk port ke database yang berbeda.
        - Volume lalu lintas yang tinggi karena pengiriman data.

b. Program klien mengeksekusi logika aplikasi:
    - Dibatasi oleh kemampuan pemrosesan workstation klien (memori, CPU).
    - Membutuhkan kode aplikasi untuk didistribusikan ke setiap workstation klien.
    - SQL menyediakan bahasa standar untuk RDBMS. Ini menciptakan titik pemisah logis antara klien dan server. Oleh karena itu, fungsi kueri dan transaksi tetap berada di sisi server. Dalam arsitektur seperti itu, server sering disebut server kueri atau server transaksi, karena menyediakan dua hal ini:



Gambar 1.48 Arsitektur 2 Tingkat (2-tier)

Keuntungan Arsitektur 2 Tingkat

1. Struktur sederhana
2. Mudah diatur dan dipelihara
3. Performa yang memadai untuk lingkungan dengan volume rendah hingga sedang.
4. Logika bisnis dan database secara fisik dekat, yang memberikan kinerja lebih tinggi.
5. Ini adalah kompatibilitas yang sederhana dan mulus dengan sistem yang ada.

Kekurangan:

1. Aturan aplikasi yang kompleks sulit untuk diterapkan dalam database, server membutuhkan lebih banyak kode untuk klien.
2. Aturan aplikasi yang kompleks sulit diterapkan di klien dan memiliki kinerja yang buruk.
3. Perubahan logika bisnis tidak secara otomatis diberlakukan oleh server-hanges memerlukan perangkat lunak sisi klien baru untuk didistribusikan dan diinstal.
4. Tidak portabel untuk platform server database lainnya.

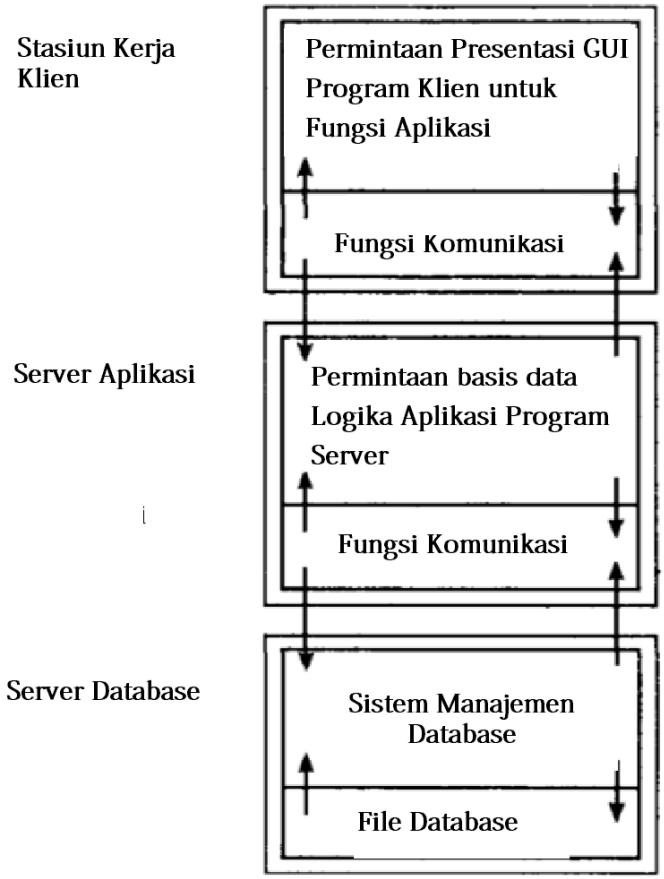
5. Performa yang tidak memadai untuk lingkungan volume sedang hingga tinggi, karena server database diperlukan untuk menjalankan logika bisnis. Ini memperlambat operasi database di server database.

Arsitektur Client/Server 3-Tier untuk DBMS

Arsitektur tiga tingkat, yang menambahkan lapisan perantara antara klien dan server database. Lapisan perantara atau tingkat menengah ini disebut server aplikasi. Server ini berperan sebagai perantara dengan menyimpan aturan bisnis (prosedur atau kendala) yang digunakan untuk mengakses data dari server database. Itu juga dapat meningkatkan keamanan database dengan memeriksa kredensial klien sebelum meneruskan permintaan ke server database. Klien berisi antarmuka GUI dan beberapa aturan bisnis khusus aplikasi tambahan. Server perantara menerima permintaan dari klien, memproses permintaan dan mengirimkan perintah database ke server database, dan kemudian bertindak sebagai saluran untuk meneruskan data yang diproses dari server database ke klien, di mana dapat diproses lebih lanjut dan disaring dan disajikan ke pengguna dalam format GUI. Dengan demikian, antarmuka pengguna, aturan aplikasi, dan akses data bertindak sebagai tiga tingkatan.

Arsitektur server klien 3-tier memiliki tiga komponen penting:

- PC klien
- Server Aplikasi
- Server Database



Gambar 1.49 Arsitektur server klien 3-tier

3. Pertimbangan Arsitektur Tier
    a. Program klien hanya berisi logika presentasi:
        - Lebih sedikit sumber daya yang dibutuhkan untuk workstation klien.
        - Tidak ada modifikasi klien jika lokasi database berubah.
        - Lebih sedikit kode untuk didistribusikan ke cleint workstation.
    b. Satu server menangani banyak permintaan klien :
        - Tersedia lebih banyak sumber daya untuk program server.
        - Mengurangi lalu lintas data di jaringan.

Kelebihan:

1. Aturan aplikasi yang kompleks mudah diimplementasikan di server aplikasi.
2. Logika bisnis diturunkan dari server database dan klien, yang meningkatkan kinerja.
3. Perubahan logika bisnis secara otomatis diberlakukan oleh server-perubahan hanya memerlukan perangkat lunak server aplikasi baru yang akan diinstal.
4. Logika server aplikasi portabel ke platform server database lain berdasarkan perangkat lunak aplikasi.
5. Performa superior untuk lingkungan volume sedang hingga tinggi.
6. Teknologi enkripsi dan dekripsi membuatnya lebih aman untuk mentransfer data sensitif dari server ke klien dalam bentuk terenkripsi, di mana ia akan didekripsi.
7. Berbagai teknologi untuk kompresi data membantu dalam mentransfer sejumlah besar data dari server ke klien.

Kekurangan :

1. Struktur lebih kompleks.
2. Lebih sulit untuk diatur dan dipelihara.
3. Masalah keamanan jaringan.
4. Pemisahan fisik server aplikasi yang mencakup fungsi logika bisnis dan server Database yang berisi Database mungkin sedikit memengaruhi kinerja.

1.28 REVIEW PERTANYAAN

1. Apa itu data?
2. Apa yang dimaksud dengan informasi?
3. Apa perbedaan antara data dan informasi?
4. Apa itu Database dan sistem Database?
5. Mengapa kita membutuhkan database?
6. Apa itu katalog sistem?
7. Apa itu sistem manajemen Database?
8. Apa itu kamus data? Jelaskan fungsinya dengan diagram yang rapi.
9. Menguraikan keuntungan menerapkan sistem manajemen Database dalam suatu organisasi.
10. Apa perbedaan antara bahasa definisi data dan bahasa manipulasi data?
11. Siapa itu DBA? Apa tanggung jawab DBA?
12. Bandingkan sistem berorientasi file dan sistem database.
13. Mendiskusikan kelebihan dan kekurangan DBMS.

14. Jelaskan perbedaan antara skema eksternal, internal dan konseptual.
15. Diskusikan karakteristik utama dari pendekatan database dan bagaimana perbedaannya dari sistem file tradisional.
16. Apa saja jenis pengguna akhir Database yang berbeda? Diskusikan kegiatan utama masing-masing.
17. Definisikan istilah berikut: model data, skema Database, status Database, skema internal, skema konseptual, skema eksternal, independensi data, DDL, DML.
18. Diskusikan kategori utama model data.
19. Menjelaskan arsitektur tiga skema. Mengapa kita membutuhkan pemetaan antar level skema? Bagaimana bahasa definisi skema yang berbeda mendukung arsitektur ini?
20. Apa perbedaan antara DML prosedural dan non-prosedural?
21. Diskusikan berbagai jenis antarmuka yang ramah pengguna dan jenis pengguna yang biasanya menggunakan masing-masing.
22. Apa perbedaan antara independensi data logis dan independensi data fisik?

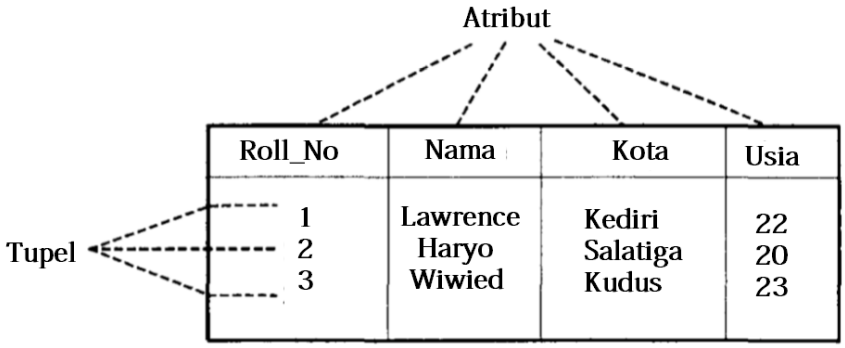
BAB 2
KONSEP MODEL DATA RELASIONAL

2.1 KONSEP MODEL DATA RELASIONAL

Model relasional menggunakan kumpulan tabel untuk mewakili data dan hubungan di antara data tersebut.

- Tabel adalah kumpulan baris dan kolom. Setiap kolom memiliki nama yang unik.
- Setiap baris dalam tabel mewakili kumpulan nilai data terkait.
- Dalam model relasional, baris disebut tuple, kolom disebut atribut dan tabel disebut relasi.

Tabel 2.1 Konsep Model Data Relasional



2.2 KENDALA INTEGRITAS

Sebagian besar aplikasi database memiliki batasan integritas tertentu yang harus dimiliki untuk data.

- Jenis batasan integritas yang paling sederhana melibatkan penentuan tipe data untuk setiap item data.
- Batasan integritas dapat diklasifikasikan sebagai :
    - Batasan integritas :
        - Integritas entitas
        - Referensi

Integritas Entitas

Batasan integritas entitas menyatakan bahwa tidak ada nilai kunci utama yang dapat berupa Null. Ini karena nilai kunci utama digunakan untuk mengidentifikasi tupel individu dalam suatu relasi, memiliki nilai Null untuk kunci utama menyiratkan bahwa kita tidak dapat mengidentifikasi beberapa tupel. misalnya, Jika dua atau lebih tupel memiliki Null untuk kunci utama mereka, maka kami mungkin tidak dapat membedakannya.

Integritas referensial

Kami ingin memastikan bahwa nilai yang muncul dalam satu relasi untuk satu set atribut tertentu juga muncul untuk satu set atribut tertentu di yang lain. Kondisi ini disebut "Integritas Referensial".

- Batasan ini menetapkan hubungan antara catatan di seluruh tabel master dan detail. Hubungan ini memastikan:
    - Catatan tidak dapat dimasukkan ke dalam tabel detail jika catatan terkait dalam tabel master tidak ada.
    - Catatan tabel master tidak dapat dihapus jika catatan yang sesuai dalam tabel detail ada.

2.3 KENDALA DOMAIN

Ini adalah kumpulan nilai yang dapat kita ekstrak nilainya untuk set dengan mengambil beberapa kondisi. Kita telah melihat bahwa domain nilai yang mungkin harus dikaitkan dengan setiap atribut. Kami melihat jumlah tipe domain standar seperti tipe Integer, tipe karakter dan tipe tanggal/waktu" yang didefinisikan dalam SQL.

- Batasan domain adalah bentuk paling dasar dari batasan integritas. Mereka diuji dengan mudah oleh sistem setiap kali item data baru dimasukkan dimasukkan ke dalam database. Beberapa atribut mungkin memiliki domain yang sama. Misalnya, atribut nama-pelanggan dan nama-karyawan mungkin memiliki domain yang sama. Himpunan semua nama orang.
- Batasan domain tidak hanya memungkinkan kita nilai pengujian yang dimasukkan ke dalam database, tetapi juga mengizinkan kita untuk menguji kueri untuk memastikan bahwa perbandingan tersebut masuk akal, misalnya, klausa create domain dapat digunakan untuk mendefinisikan domain baru.

    Buat domain dolar numerik (12, 2) :
    Buat domain pound numerik (12, 2) :

Upaya untuk menetapkan nilai jenis dolar ke variabel bertipe pound akan menghasilkan kesalahan sintaks, sementara keduanya dari jenis numerik yang sama .

2.4 ALJABAR RELASIONAL

Aljabar relasional adalah bahasa query prosedural. Ini terdiri dari satu set operasi yang mengambil satu atau dua relasi sebagai input dan menghasilkan relasi baru sebagai hasilnya. Operasi fungsional dalam aljabar relasional adalah:

- Pilih
- Proyek
- Persatuan
- Tetapkan perbedaan
- produk kartesius
- Ganti nama
- Persimpangan
- Divisi
- Bergabung
- Gabung secara Alami

Select Operation

Pilih tupel yang memenuhi predikat yang diberikan. Notasi operasi pemilihan adalah :

σp (R)

dimana

R = relasi masukan

P = predikat yang akan dievaluasi

misal, Pilih rekening yang memiliki SBI.

$$\sigma_{nama\ cabang\ =\ SBI}(Rekening)$$

Catatan: Kita dapat menggunakan predikat berikut.

(i) = (sama dengan)

(ii) < (kurang dari)

(iii) > (lebih besar dari)

(iv) ≠ (tidak sama dengan)

(v) ≤ (kurang dari atau sama dengan)

(vi) ≥ (lebih besar dari sama dengan)

(viii) ∧ (dan)

mis., Pilih pinjaman yang diambil dari SBI dan masing-masing jumlahnya lebih besar dari 50.000.

Jawaban

$$\sigma_{nama\_cabang\ =\ SBI\ 1\ \wedge\ jumlah\ >\ 50.000}(Pinjaman)$$

Operasi-Proyek Operasi proyek adalah operasi unier yang mengembalikan relasi argumennya, dengan atribut-atribut tertentu ditinggalkan. Misalkan, kita ingin mencantumkan semua nomor pinjaman dan jumlah pinjaman tetapi tidak peduli dengan nama cabang. Operasi proyek memungkinkan kita untuk menghasilkan hubungan ini. Dilambangkan dengan π (Pi). misalnya, Temukan nama semua pelanggan yang tinggal di Indonesia.

$$\pi_{nama\_pelanggan}(\sigma_{kota\_customer\ =\ "Semarang"}(Customer))$$

misalnya, Untuk menemukan nomor pinjaman dan jumlah semua pinjaman

$$\pi_{jumlah\_pinjaman}(Pinjaman)$$

Operasi Serikat

Penggabungan dua set menggabungkan semua data yang muncul salah satu atau kedua relasi. Untuk operasi gabungan r ∪ s, relasi rand s harus memiliki jumlah atribut yang sama. misalnya, Untuk menemukan nama semua nasabah bank yang memiliki rekening atau pinjaman atau keduanya. Biarkan dua hubungan busur:

Tabel 2.2 Penggabungan dua set menggabungkan semua data

Depositor :

Nama_Customer	No. Rek
Topo	A–101
Santi	A–102
Andik	A–103
Teguh	A–104
Martinus	A–105

Peminjam

Nama_Customer	No. Pinjaman
Dwi	L–17
Sindu	L–18
Fitro	L–19
Jarot	L–20
Wilu	L–21

Kueri memberikan hasil sebagai:

$$\pi_{nama\_Customer}(Peminjam) \cup \pi_{nama\_Customer}(Depositor)$$

Tabel 2.3 Menampilkan Nama Costumer

Nama_Customer
Topo
Santi
Andik
Teguh
Martinus
Dwi
Sindu
Fitro
Jarot
Wilu

Operasi Set-Difference

Operasi set-difference, memungkinkan kita untuk menemukan tupel yang berada dalam satu relasi tetapi tidak berada di relasi lain. Ekspresi r-s menghasilkan sebuah relasi yang berisi tupel-tupel tersebut di dalam *r*, tetapi tidak di dalam s, misalnya : Temukan semua pelanggan bank yang memiliki rekening tetapi tidak memiliki pinjaman. (Penyimpan) (Peminjam)

Jawabannya adalah :

$\pi_{Nama\_Customer}{}^{(Deposit)} - \pi_{Nama\_Customer}{}^{(Peminjam)}$

Tabel 2.4 Tabel Pelanggan

Nama_Customer
Topo
Santi
Andik
Teguh
Martinus

Operasi Produk Cartesian

Operasi produk kartesius, memungkinkan kita untuk menggabungkan informasi dari dua relasi. Produk kartesius dari relasi r_1 dan 2_1 sama dengan r_1 x r_2

Contoh : r_1 x r_2

Tabel 2.5 Produk kartesius dari relasi r_1 dan 2_1

Roll_No	Name
1	Abigail
2	Nico
3	Rista
4	Azeda

Matkul
B. Inggris 3
Nirmana

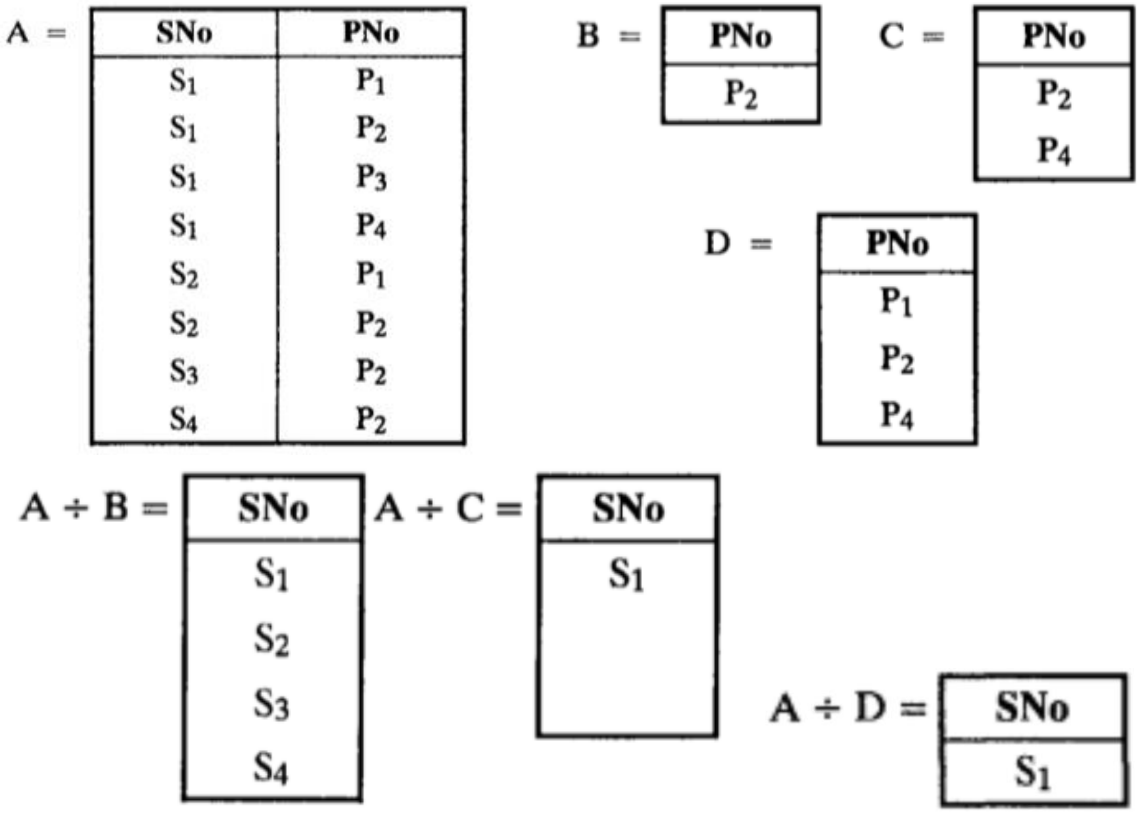
r_1 x r_2 Hasil kueri adalah

Tabel 2.6 Hasil kueri

Roll_No	Nama	Matkul
1	Abigail	B. Inggris 3
2	Nico	B. Inggris 3
3	Rista	B. Inggris 3
4	Azeda	B. Inggris 3
1	Abigail	Nirmana
2	Nico	Nirmana
3	Rista	Nirmana
4	Azeda	Nirmana

Operasi Divisi

Simbol dari operator ini adalah + dan digunakan untuk memilih "untuk semua". Operator divisi dapat melamar sebagai



Gambar 2.1 Operasi Divisi

Ganti Nama-Operasi

Dalam aljabar relasional, A rename adalah operasi unary yang ditulis sebagai

Pa/b (R)

dimana

- a dan b adalah nama atribut
- R adalah sebuah relasi.

Hasilnya identik dengan R kecuali bahwa bidang b di semua tupel diubah namanya menjadi bidang. misalnya, Pertimbangkan relasi Karyawan dan versi yang diganti namanya :

Karyawan:

Tabel 2.7 Sebelum Diubah

Nama	ID_Karyawan
Abigail Nico	3415 2241

ρNama_Karyawan/Nama *(Karyawan)*

Tabel 2.8 Setelah Dibubah

Nama_Karyawan	ID_Karyawan
Abigail Nico	3415 2241

Join

Operator join memungkinkan penggabungan dua relasi untuk membentuk satu relasi baru. Misalnya:

Tabel 2.9 Sebelum Penggabungan

Tabel Karyawan

ID_Kar	Nama
100	Abigail
101	Nico
102	Rista
103	Azeda

Tabel Gaji

ID_Gaji	Gaji
100	15000
101	25000
102	20000
103	15000

Hasil Gaji Gabungan karyawan

Tabel 2.10 Setelah Penggabungan

ID_Kar	Nama	ID_Gaji	Gaji
100	Abigail	100	15000
101	Nico	101	25000
102	Rista	102	20000
103	Azeda	103	15000

Natural Join: Gabung Alami adalah operator diadik yang ditulis sebagai Rand S, di mana Rand S adalah relasi. Hasil dari natural join adalah himpunan semua kombinasi tupel di Rand S yang sama pada nama atribut umumnya.

Tabel 2.11 Penggabungan tabel Karyawan dan Departemen (Natural Join)

Karyawan

Nama	ID_Kar	Nama_Dept
Edy	100	Keuangan
Dani	101	Penjualan
Yayuk	102	Keuangan
Rusito	103	Penjualan

Departemen

Nama_Dept	Manajer
Finance	Setiyo
Sales	Sumar
Production	Alex

Karyawan Departemen

Nama	ID_Kar	Nama_Dept	Manajer
Edy	100	Keuangan	Setiyo
Dani	101	Penjualan	Sumar
Yayuk	102	Keuangan	Setiyo
Rusito	103	Penjualan	Sumar

Semi Join: Semi join mirip dengan natural join dan ditulis sebagai R 0 S, di mana Rand S adalah relasi. Hasil dari semi join hanya himpunan semua tupel di R yang ada tupel di S yang sama pada nama atribut umum mereka.

Tabel 2.12 Semi Join Tabel Karyawan dan Departemen

Nama	ID_Kar	Nama_Dept
Edy	100	Keuangan
Dani	101	Penjualan
Yayuk	102	Keuangan
Rusito	103	Produksi

Nama_Dept	Manajer
Penjualan	Setiyo
Produksi	Alex

Nama	ID_Kar	Nama_Dept
Dani	101	Penjualan
Rusito	103	Keuangan

Anti Join: Anti join, ditulis sebagai R I> S, di mana Rand S adalah relasi, mirip dengan natural join, tetapi hasil dari anti join hanya tupel di R yang tidak ada tupel di S yang adalah sama pada nama atribut umum mereka.

Tabel 2.13 Tabel Karyawan Dan Dept Dan Anti Join.

Nama	ID_Kar	Nama_Dept
Edy	100	Keuangan
Dani	101	Penjualan
Yayuk	102	Keuangan
Rusito	103	Produksi

Nama_Dept	Manajer
Penjualan	Setiyo
Produksi	Sumar

Nama	ID_Kar	Nama_Dept
Edy	100	Keuangan
Yayuk	102	Keuangan

Outer Join: Gabungan luar penuh ditulis sebagai R = X = S dimana R dan S adalah relasi. Hasil dari full outer join adalah himpunan semua kombinasi tupel di R dan S yang sama pada nama atribut umumnya, selain tupel S yang memiliki tupel yang cocok di R dan tupel di R yang tidak memiliki tupel yang cocok di S dalam nama atribut umum mereka. Contoh. " Pertimbangkan tabel Employee dan Dept dan Gabung luar penuh mereka:

Tabel 2.14 Outer Join Tabel Employee Dan Departemen

Nama	ID_Kar	Nama_Dept
Edy	100	Keuangan
Dani	101	Penjualan
Yayuk	102	Keuangan
Rusito	103	Produksi
Lawrence	104	Eksekutif

Nama_Dept	Manajer
Penjualan	Setiyo
Produksi	Alex

Karyawan=*X*=Departemen

Nama	ID_Kar	Nama_Dept	Manajer
Edy	100	Keuangan	ω
Dani	101	Penjualan	Setiyo
Yayuk	102	Keuangan	ω
Rusito	103	Produksi	Setiyo
Lawrence	104	Eksekutif	ω
ω	ω	Produksi	Alex

ω=Nilai sama dengan 0

Proyeksi

Proyeksi adalah operasi unier yang ditulis sebagai $^{\pi}a_1, a_z, \ldots, an\ (R)$, di mana $a_1, a_2, \ldots, a_n$ adalah himpunan nama atribut. Hasil proyeksi tersebut didefinisikan sebagai himpunan yang diperoleh ketika semua tupel di R dibatasi pada himpunan $(a_1, a_2, \ldots, a_n)$. Contoh, Tabel Pegawai(E)

Tabel 2.15 Tabel Pegawai

ID	Nama	Gaji ('000)
1	Eko	15.000
5	Solikhan	30.000
7	Tyo	25.000

Seleksi : Contoh seleksi

Tabel 2.16 Contoh Seleksi

SQL	Hasil		Relasi Aljabar
Pilih Gaji dari *E*	Gaji		$^{\pi}$Gaji (*E*)
	15.000		
	30.000		
	25.000		
Pilih ID, Gaji dari *E*	ID	Gaji	$^{\pi}$ID, Gaji (*E*)
	1	15.000	
	5	30.000	
	7	25.000	

SQL	Hasil			Relasi Aljabar
Pilih * dari E dimana Gaji < 30.000	ID	Nama	Gaji	(E) $^{\sigma}$Gaji < 30.000
	1	Eko	15.000	
	7	Tyo	25.000	
Pilih * dari E dimana Gaji < 30.000 dan ID < 7	ID	Nama	Gaji	(E) $^{\sigma}$Gaji < 30.000 Dan ID <7
	1	Eko	15.000	

2.5 PERHITUNGAN RELASIONAL

Kalkulus relasional adalah bahasa non prosedural yang mewakili kekuatan dasar yang diperlukan dalam bahasa kueri relasional. Kalkulus relasional digunakan dalam desain bahasa kueri komersial seperti SQL, QBL. Sebuah query dalam kalkulus relasional tupel dinyatakan sebagai:

$\{t/P(t)\}$

yaitu, itu adalah himpunan semua tupel t sedemikian rupa sehingga predikat P benar untuk t. Rumus kalkulus-relasional-tupel *dibuat* dari atom-atom. Sebuah atom memiliki salah satu bentuk berikut:

- S ∈ r, di mana S adalah variabel tupel dan r adalah relasi.
- S [x] ⊙ u [y], di mana e adalah operator pembanding dan S, u adalah variabel tupel, x adalah atribut di mana S didefinisikan dan y adalah atribut di mana u didefinisikan.
- S [x] ⊙C, di mana S adalah variabel tupel, x adalah atribut di mana S adalah defme, ⊙adalah operator pembanding dan C adalah kendala dalam domain atribut x.

Kami membangun rumus dari atom dengan menggunakan aturan berikut:

- Atom adalah rumus.
- Jika P_1 adalah rumus, maka $\rightarrow P_1$ dan (P_1)
- Jika P_1 dan P_2 adalah rumus, maka $P_1 \vee P_2$, $P_1 \wedge P_2$ dan $P_1 \Rightarrow P_2$
- Jika $P_1(S)$ adalah rumus yang mengandung variabel tupel bebas S, dan r adalah relasi, maka

$\exists S \in r\ (P_1(S))$ *dan* $\forall S \in (P_1(S))$

'Dalam kalkulus relasional tupel, kesetaraan ini mencakup tiga aturan berikut:

(1) $P_1 \wedge P_2$ setara dengan $\rightarrow(\rightarrow P_1) \vee \rightarrow(P_2)$

(2) $\forall t \in r\ (P_1(t))$ *adalah wkuivalen untuk* $\rightarrow \exists t \in r\ (\rightarrow P_1(t))$

(3) $P_1 \Rightarrow P_2$ sama dengan $\rightarrow P_1 \vee P_2$.

Misalnya: (1) Temukan jumlah pinjama n untuk setiap pinjaman dengan jumlah yang lebih besar dari RP. 1200

{t/∃ S ∈ pinjaman (t [jumlah-pinjaman] = S [Jumlah-pinjaman] ∧ S [jumlah] > 1200)}

(4) Temukan semua pelanggan yang memiliki pinjaman, rekening atau keduanya di bank.

{ t/∃ S ∈ peminjam (t [nama-pelanggan] = S [nama-pelanggan])

∨ /∃ u ∈ depositor (t [nama-pelanggan] = u [nama-pelanggan}

(5) Temukan semua pelanggan yang memiliki rekening di bank tetapi tidak tidak memiliki pinjaman dari bank

{t|∃ u ∈ depositor (t [nama-pelanggan] = u [nama-pelanggan

∧→∃ s ∈peminjam (t [nama-pelanggan] = s [nama-pelanggan])

(6) Temukan nama cabang, nomor pinjaman, dan jumlah pinjaman lebih dari Rp. 12000.

t|t ∈ pinjaman ∧ t [jumlah] > 120002

2.6 PERHITUNGAN DOMAIN RELASIONAL

- Kalkulus relasional domain adalah bahasa non prosedural yang mewakili kekuatan dasar yang diperlukan dalam bahasa kueri relasional.

- Kalkulus relasional domain menggunakan variabel domain yang mengambil nilai dari domain atribut.

Definisi Formal: Dalam domain kalkulus relasional berbentuk :

{(x$_l$, x$_2$, ... ,x$_n$) | P (x$_2$, x$_2$, ... ,x$_n$)}

dimana xl, x$_2$, ... ,x$_n$ adalah variabel domain P merupakan rumus yang terdiri dari atom.

Sebuah atom dalam kalkulus relasional domain memiliki salah satu bentuk berikut :

- < x$_1$, x$_2$, x$_3$, ...x$_n$ > ∈ r, di mana r adalah relasi pada n atribut dan x$_1$, x$_2$, ... ,x$_n$ adalah variabel domain.
- x ⊙ y, di mana variabel domain x dan yare dan ⊙adalah operator pembanding.
- x ⊙ c, di mana x adalah variabel domain, ⊙ adalah operator perbandingan dan c adalah kendala dalam domain dari atribut yang x adalah variabel domain.

Kami membangun rumus dari atom dengan menggunakan aturan berikut:

- Atom adalah rumus. Jika PI adalah rumus, maka demikian pula, ⇁P$_1$ dan P$_2$.
- Jika P$_1$ dan P$_2$ adalah rumus, maka demikian pula P$_1$ ∨ P$_2$, P$_1$ ∧ P$_2$, dan P$_1$ ⇒P$_2$.
- Jika P$_1$ (x) adalah rumus dalam x, di mana x adalah variabel domain, maka

∃ x (P$_1$((x)) dan ∀x (P$_1$(x))

Contoh:

(1) Temukan nomor pinjaman, nama cabang, dan jumlah pinjaman lebih dari Rp. 12.000.
{< l, b, a> | <l, b, a> ∈ pinjaman ∧ a > 12000}

(2) Temukan semua nomor pinjaman untuk pinjaman dengan jumlah lebih besar dari Rp. 12000
{< l > | ∃ b, a (l, b, a> ∈ loan ∧ a > 12000)}

(3) Temukan nama semua nasabah yang memiliki pinjaman dari cabang SBI dan temukan jumlah pinjaman
{< c, a> | ∃ l < c, l > ∈ peminjam ∧ ∃ b, a <l, b, a> ∈ pinjaman ∧ b = "SBI")}

(4) Cari nama semua nasabah yang memiliki pinjaman, rekening, atau keduanya di cabang SBI.

{<c>| *∃ l<c, n> ∈ pelanggan ∧ ∀x, y, z (<x, y, z> ∈ cabang ∧ y = "Indonesia")}*
⇒ ∃ a, b ∈depositor ∧ ∃ b, n> (<a, b, n) ∈ pinjaman ∧ b = "SBI")}
V 3 a «c, a> E depositor A 3 b, n « a, b, n> E jumlah A b = "SBI"))))}

(5) Cari nama semua nasabah yang memiliki rekening di semua cabang yang berlokasi di Indonesia.
{< c, a> | ∃ l < c, l > ∈ peminjam ∧ ∃ b, a <l, b, a> ∈ pinjaman ∧ b = "SBI")}
⇒∃ a, b (<a, x, b> ∈ rekening ∧<c, a>∈ depositor

2.7 PENGENALAN SQL

Structured Ouery Language (SQL) adalah bahasa yang menyediakan antarmuka ke sistem Database relasional. Dalam penggunaan umum SQL juga mencakup:

- DML (Bahasa Manipulasi Data) untuk INSERT, UPDATE, DELETE
- DDL (Data Definition Language) digunakan untuk membuat dan memodifikasi tabel dan struktur database lainnya.

Fitur SQL:

1. SQL dapat digunakan oleh berbagai pengguna, termasuk mereka yang memiliki sedikit atau tanpa pengalaman pemrograman.
2. Ini adalah bahasa non prosedural
3. Ini mengurangi jumlah waktu yang dibutuhkan untuk membuat dan memelihara sistem.
4. Ini adalah Bahasa Seperti Bahasa Inggris.

Komponen SQL: Ada beberapa komponen SQL berikut ini.

1. DDL

Ini adalah satu set perintah SQL yang digunakan untuk membuat, memodifikasi dan menghapus struktur Database tetapi bukan data. Misalnya "

a. **CREATE**: Untuk membuat objek dalam database.
b. **ALTER**: Mengubah struktur database.
c. **DROP**: Menghapus objek dari database.
d. **TRUNCATE** : Menghapus semua record dari tabel , termasuk semua ruang yang dialokasikan untuk catatan dihapus
e. **COMMENT**: Tambahkan komentar ke kamus data.

2. DML (Data Manipulation Language)

Ini adalah area SQL yang memungkinkan perubahan data dalam database. Sebagai Contoh:

a. **INSERT** : Menyisipkan data ke dalam tabel
b. **UPDATE**: Memperbarui data yang ada di dalam tabel.
c. **DELETE** : Menghapus semua record dari tabel, ruang untuk record tetap ada.
d. **LOCK TABLE** : Kontrol konkurensi.

3. DCL (Data Control Language)

Ini adalah komponen pernyataan SQL yang mengontrol akses ke data dan ke database.*Sebagai Contoh:*

a. **COMMIT**: Simpan pekerjaan yang telah selesai
b. **SAVE POINT:** Identifikasi titik dalam transaksi yang nantinya dapat Anda putar kembali.
c. **ROLL BACK:** Mengembalikan database ke aslinya sejak COMMIT terakhir.
d. **HIBAH/REVOKE:** Memberikan atau mengambil kembali izin ke atau dari pengguna oracle.
e. **SET TRANSAKSI:** Ubah opsi transaksi seperti segmen rollback yang akan digunakan.

4. DQL (Bahasa Kueri Data)

Ini adalah komponen pernyataan SQL yang memungkinkan mendapatkan data dari database dan memaksakan pemesanan di atasnya.

Misalnya:

SELECT: Mengambil data dari database.

Jenis Data Tipe data datang dalam beberapa bentuk dan ukuran, memungkinkan pemrogram untuk membuat tabel yang sesuai dengan ruang lingkup proyek.

- **CHAR (Size):** Tipe data ini digunakan untuk menyimpan nilai karakter string dengan panjang tetap. Jumlah maksimum karakter yang dapat ditampung oleh tipe data ini adalah 255 karakter. misalnya, Dalam kasus 'Nama Char (15)" maka data yang disimpan dalam variabel Nama hanya memiliki panjang 15 karakter.
- **VARCHAR (Size)NARCHAR2 (size) :** Tipe data ini digunakan untuk menyimpan data alfanumerik dengan panjang variabel. Jenis maksimum yang dapat ditampung adalah 2000 karakter.
- **NUMBER (P, S) :** Tipe data NUMBER digunakan untuk menyimpan angka. Presisi (P), menentukan panjang maksimum data, sedangkan skala, (S), menentukan jumlah tempat di sebelah kanan desimal. Presisi maksimum (P), adalah 38 digit.
- **DATE**: Tipe data ini digunakan untuk merepresentasikan tanggal dan waktu. Format standar adalah DD-MM-YY seperti pada 26-Juni-07. Tanggal waktu menyimpan tanggal dalam format 24 jam. Secara default, waktu dalam bidang tanggal adalah 12 : 00 : 00 AM, jika tidak ada porsi waktu yang ditentukan.
- **LONG**: Tipe data LONG digunakan untuk menyimpan string karakter dengan panjang variabel yang berisi hingga 2 GB. Data LONG dapat digunakan untuk menyimpan array data biner dalam format ASCII.
- **RAW/LONG RAW:** Tipe data RAW/LONG RAW digunakan untuk menyimpan data biner, seperti gambar atau gambar digital. Tipe data RAW dapat memiliki panjang maksimum 255 byte. Tipe data LONG RAW dapat berisi hingga 2 GB.
- **ROWID (rowid) :** Format rowidnya adalah :

  BBBBBBB . RRRR. FFFFF

  dimana BBBBBBB adalah blok dalam file database;

  RRRR adalah baris dalam blok;

  FFFFF adalah file Database.
- **Boolean**: Valid di PL/SQL tetapi tipe data ini tidak ada di oracle 8i atau oracle 9i.

Jenis Perintah SQL

1. Perintah Buat tabel:

Sintaks: CREATE TABLE nama tabel
(nama kolom tipe data (ukuran), nama kolom tipe data (ukuran),
tipe data nama kolom (ukuran));

Contoh : Membuat Tabel Karyawan.

```
CREATE TABLE Employee (E-ID number (6),
ENAME char (15),
ADDRESS varchar (15),
CITY char (15),
STATE char (15),
PIN CODE number (6));
```

Output : Tabel Dibuat

2. Buat tabel Siswa

```
CREATE TABLE Siswa (Nomor Roll-No (15),
```

Name char (15),
Address varchar (15),
Sex char (2),
City char (15),
Phone number (15),
State char (15));

Output : Tabel Dibuat

3. Penyisipan Data Ke Tabel

Setelah tabel dibuat, hal yang paling alami untuk dilakukan adalah memuat tabel ini dengan data yang akan dimanipulasi nanti.

Sintaks: INSERT INTO nama tabel
(Nama kolom 1, Nama kolom 2 ...)
VALUES (ekspresi, ekspresi);
Contoh:
INSERT INTO Employee (E-ID, ENAME, ADDRESS, CITY, STATE, PINCODE)
NILAI (115, 'DANI', 'C-6 Jl. Majapahit', 'Semarang', 'Indonesia');

Metode lain: INSERT INTO Employee VALUES (& E-ID, '& NAMA', '& ALAMAT', '& KOTA', '& NEGARA');

Catatan:

- Karakter atau ekspresi varchar harus diapit oleh tanda kutip tunggal (').
- Dalam pernyataan insert ke dalam SQL, kolom dan nilai memiliki hubungan satu ke satu.

4. Pilih Perintah

Setelah data dimasukkan ke dalam tabel, operasi paling logis berikutnya adalah melihat apa yang telah dimasukkan. Ini dicapai dengan SELECT SQL Verb.

a. Lihat data tabel global sintaksnya adalah :
SELECT * FROM table name;
misalnya,
SELECT * FROM Karyawan;

b. Ambil ID, nama, kota karyawan
PILIH E-ID, ENAME, CITY FROM Karyawan;
Kolom Terpilih dan Baris Terpilih:
WHERE Clause
Syntax: SELECT * FROM table name
WHERE kondisi pencarian;
misalnya,
a. SELECT * FROM Employee WHERE E-ID > 112;
b. SELECT Roll-No, Name FROM Student WHERE Roll No. < = 150;

5. Penghapusan Duplikat dari Pernyataan Pilihan

Sebuah tabel dapat berisi baris duplikat. Kita dapat menghilangkan menggunakan pernyataan pilih.

Sintaks: SELECT DISTINCT Nama kolom 1,

Nama kolom 2 FROM nama tabel;

Sintaks: SELECT DISTINCT * FROM Nama tabel

Contoh: (1) Pilih hanya baris unik dari tabel student:

SELECT DISTINCT * FROM Student;

6. Mengurutkan Data dalam Tabel

Oracle memungkinkan data dari tabel untuk dilihat dalam urutan yang diurutkan. Baris yang diambil dari tabel akan diurutkan dalam urutan menaik atau menurun tergantung pada kondisi yang ditentukan dalam pernyataan pilih.

Sintaks: SELECT * FROM nama tabel ORDER BY Nama kolom 1, Nama kolom 2 [Urutkan urutan]; .

Contoh : Ambil semua baris dari siswa dan tampilkan data ini diurutkan berdasarkan nilai yang terdapat pada field Roll-No. dalam urutan menaik;

SELECT * FROM Siswa ORDER BY Roll-No;

Catatan: Mesin Oracle mengurutkan dalam urutan menaik secara default.

Contoh: Untuk melihat data dalam urutan menurun kata desc.

SELECT * FROM Siswa ORDER BY Roll-No desc;

7. Membuat Tabel dari Tabel

Sintaks: CREATE TABLE Nama tabel

[(Nama kolom, Nama kolom)]

AS SELECT nama kolom, nama kolom FROM Nama tabel;

Contoh: Buat tabel siswa 1 dari siswa.

CREATE TABLE Siswa 1

(Nomor-SRoll, SName, Alamat, Jenis Kelamin, Kota, No Telp, Negara)

AS SELECT No Rol, Nama, Alamat, Jenis Kelamin, Kota, No Telp, Negara FROM Mahasiswa;

8. Memasukkan Data Ke Tabel dari Tabel Lain

Sintaks: INSERT INTO Nama tabel

SELECT nama kolom, nama kolom, FROM nama tabel;

Contoh : Masukkan ke tabel siswa 1 dari tabel siswa;

INSERT INTO student 1 SELECT Roll-No, Name, Address, Sex, City, Ph-No, State FROM Student;

9. Penyisipan Kumpulan Data Ke Tabel dari Tabel Lain

Sintaks : INSERT INTO nama tabel SELECT nama kolom, nama kolom FROM nama tabel

WHERE Column = Expression;

Contoh: Menyisipkan record ke tabel student 1 dari tabel student dimana field Roll-No berisi nilai '115';

INSERT INTO Student 1 PILIH Roll-No, Name, Address, Sex, City, Ph-No, State FROM Student

WHERE Roll-No = '115';

10. Hapus Operasi

Perintah DELETE menghapus baris dari tabel yang memenuhi kondisi yang disediakan oleh klausa WHERE-nya, dan mengembalikan jumlah record yang dihapus. Verb DELETE dalam SQL digunakan untuk menghapus baris dari tabel. Untuk menghapus

- Semua baris dari tabel.

OR

- Sekumpulan baris pilihan dari sebuah tabel.

11. Penghapusan Semua Baris:

Sintaks: DELETE FROM nama tabel;

Contoh: (1) Hapus semua baris dari tabel student

DELETE FROM Student;

12. Penghapusan Baris Tertentu

Sintaks: DELETE FROM table name WHERE kondisi pencarian;

Contoh : Hapus baris siswa dari tabel dimana Roll-No > 115.

DELETE FROM siswa dimana Roll-No > 115;

13. Perbarui Perintah

Memperbarui Isi Table: Perintah UPDATE digunakan untuk mengubah atau memodifikasi nilai data dalam tabel. Untuk memperbarui:

- Semua baris dari tabel.

OR

- Sekumpulan baris pilihan dari sebuah tabel.

14. Memperbarui Semua Baris:

Sintaks : UPDATE nama tabel

SET nama kolom = ekspresi,

Nama kolom = ekspresi;

Contoh: Berikan setiap karyawan bonus 10%. Perbarui nilai yang ada di kolom gaji bersih.

UPDATE SET Pegawai Netsal = Gaji Bersih + Gaji Pokok * 0.10;

15. Memperbarui Catatan Kondisional:

Sintaks : UPDATE nama tabel

SET nama kolom = ekspresi,

nama kolom = ekspresi

WHERE nama kolom = ekspresi;

Contoh : Perbarui tabel perubahan siswa, isi field name menjadi ‘Joseph Teguh’ dan isi field Address menjadi ‘Semarang’ untuk record yang diidentifikasi oleh field Roll-No yang berisi nilai 115;

UPDATE siswa

nama SET = .’Joseph Teguh’,

Alamat = ‘Semarang’

WHERE Roll-No = 115;

16. Menambahkan Kolom Baru :

Sintaks: ALTER TABLE nama tabel

ADD (Tipe data nama kolom baru/ukuran)

Tipe data nama kolom baru (ukuran ...);

Contoh: Tambahkan field Fax yang merupakan field yang dapat menampung nomor hingga panjang 15 digit dan Mobile-No, yaitu field yang dapat menampung nomor hingga panjang 10 digit.

ALTER TABLE student

ADD (Nomor Handphone-No (10), Nomor Fax (15));

17. Memodifikasi Kolom yang Ada :

Sintaks: ALTER TABLE nama tabel

MODIFY (nama kolom, tipe data baru (Ukuran baru))

Contoh: Memodifikasi field fax dari tabel siswa untuk sekarang memegang nilai karakter maksimum 25. ALTER TABLE student

MODIFY (Fax Varchar (25));

Batasan ALTER TABLE: Menggunakan klausa ALTER TABLE tugas-tugas berikut tidak dapat dilakukan:

- Mengubah nama tabel.
- Mengubah nama tabel. nama kolom
- Jatuhkan kolom
- Kurangi ukuran kolom jika ada data tabel.

Mengganti Nama

Untuk mengganti nama tabel, sintaksnya adalah:

Sintaks: RENAME nama tabel lama menjadi nama tabel baru

Contoh: Ubah nama tabel Employee menjadi Employee 1;

RENAME Karyawan MENJADI Karyawan 1;

Menghancurkan Tabel

Sintaks: nama tabel DROP TABLE;

Contoh: Hancurkan tabel Employee dan semua data yang ada di dalamnya; DROP TABLE Karyawan;

Perintah DESCRIBE : Untuk mencari informasi tentang kolom yang didefenisikan pada tabel gunakan sintaks berikut;

Sintaks: DESCRIBE nama tabel;

Perintah ini menampilkan nama kolom, tipe data dan atribut khusus yang terhubung ke tabel.

Contoh: Menampilkan kolom dan atributnya dari tabel siswa. DESCRIBE Mahasiswa;

Operator Logika

Berikut adalah operator logika yang digunakan dalam SQL.

- **Operator AND:** Mesin oracle akan memproses semua baris dalam tabel dan menampilkan hasilnya hanya jika semua kondisi yang ditentukan menggunakan operator AND terpenuhi.

Contoh: Ambil isi kolom product-no, profit-percent, sell-price dari tabel product-master dimana nilai yang terdapat pada field profit persen di antara 10 dan 20.

SELECT Produk-tidak, persen keuntungan, harga jual
FROM Master-Produk
WHERE profit-persen> = 10 DAN profit-persen < = 20;

- **Operator OR:** Mesin oracle akan memproses semua baris dalam tabel dan menampilkan hasilnya hanya jika salah satu kondisi yang ditentukan menggunakan operator OR terpenuhi.

  Contoh: Ambil semua bidang tabel siswa di mana bidang Roll-No memiliki nilai 115 ATAU 200;

  SELECT Roll-No, Nama, Alamat, Jenis Kelamin, Kota, Ph-No,
  Nyatakan FROM Student WHERE (Roll-No = 115 OR Roll-No = 200);

- Operator NOT: Mesin oracle akan memproses semua baris dalam tabel dan menampilkan hasilnya hanya jika tidak ada kondisi yang ditentukan menggunakan operator NOT yang terpenuhi.

  Contoh : Ambil informasi siswa tertentu untuk klien, yang TIDAK di 'Semarang'· ATAU 'Semarang';

  SELECT Roll-No, Nama, Alamat, Kota, Negara Bagian
  FROM Mahasiswa WHERE NOT
  (Kota = 'Jakarta' OR Kota = 'Semarang');

Pencarian Rentang

Operator BETWEEN:

Contoh: Ambil Roll-No, Name, Address, Ph-No, State dari tabel student dimana nilai yang terdapat dalam field Roll-No adalah antara 100 dan 200 keduanya inklusif.

SELECT Roll-No, Name, Address, Ph-No, State FROM Student
WHERE Roll-No BETWEEN 100 AND 200;

Pencocokan Pola

Penggunaan predikat LIKE

Predikat LIKE memungkinkan untuk membandingkan satu nilai string dengan nilai string lain, yang tidak identik.

Untuk tipe data karakter :
Tanda persen (%) cocok dengan string apa pun.
Garis bawah (J cocok dengan karakter tunggal apa pun.

Contoh:

(1) Ambil semua informasi tentang siswa yang namanya dimulai dengan huruf 'vi' dari tabel siswa.

SELECT * FROM Siswa
WHERE Nama LIKE 'vi %';

(2) Ambil semua informasi tentang siswa di mana karakter kedua dari nama adalah 'V' atau 'S'

SELECT * FROM Siswa
WHERE Nama LIKE '_ V%' OR

Nama LIKE '_S%';

Predikat IN: Jika nilai perlu dibandingkan dengan daftar nilai, maka predikat IN digunakan.

Contoh: Ambil Roll-No, Name, Address, City, Ph-No dari tabel student dimana namanya adalah Dani atau Lawrence atau Dani atau Joni.

SELECT Roll-No, Nama, Alamat, Kota, No Telp
FROM Siswa
WHERE Nama IN ('Dani', 'Lawrence', 'Dani', 'Joni');

Predikat NOT IN: Predikat NOT IN merupakan kebalikan dari predikat IN. Ini akan memilih semua baris yang nilainya tidak cocok dengan semua nilai dalam daftar.

Contoh:

SELECT Roll-No, Name, Address, City, Ph-No FROM Student
WHERE Nama NOT IN ('Dani', 'Lawrence', 'Agus', 'Joni');

Batasan UNIQUE KEY Didefinisikan di Tingkat Tabel

Sintaks: UNIQE (nama kolom);

Contoh: Buat tabel siswa sedemikian rupa sehingga batasan kunci unik pada kolom Roll-No digambarkan sebagai batasan level tabel.

CREATE TABEL Siswa
(Roll-No number (5), Name char (15), Sex char (2),
Ph-No number (10), Address varchar (15), City char (10), State char (15), UNIQUE (Roll-No));

Batasan PRIMARY KEY Didefinisikan di Tingkat Kolom

Sintaks: PRIMARY KEY (kolom);

Contoh: Buat tabel siswa sedemikian rupa sehingga batasan kunci utama pada kolom Roll-No digambarkan sebagai batasan tingkat tabel.

CREATE TABLE Siswa
(Roll-No number (5), Name char (15), Sex char (2),
Ph-No number (10), Alamat varchar (15), City char (10),
State char (15), PRIMARY KEY (Roll-No));

Batasan FOREIGN KEY Didefinisikan di Tingkat Tabel

Sintaks: FOREIGN KEY (nama kolom [nama kolom))
REFRENCE nama tabel [nama kolom [, nama kolom));

Contoh: Buat tabel pesanan penjualan dengan kunci utama sebagai detlorder-no dan product-no dan foreign key di tingkat tabel sebagai detloder-no referensi kolom order-no di tabel pesanan penjualan.

CREATEA TABLE pesanan penjualan
(detlorder-tanpa varchar (6), produk tanpa varchar (6), jumlah pesanan (7), nomor tarif produk (8, 2),
PRIMARY KEY (detlorder-no, product-no),
Foreign key (detlorder-tidak)
REFERENCE pesanan penjualan);

Fungsi Agregat

Fungsi agregat adalah fungsi yang mengambil kumpulan nilai sebagai input dan mengembalikan nilai tunggal. SQL menawarkan lima fungsi agregat suit-in.

- Rata-rata/Average : AVG
- Minimal/Minumim : MIN
- MAKSIMUM/MAXIMUM : MAX
- Jumlah/Total : SUM
- Hitung/Count : COUNT

Sintaks: COUNT ([DISTINCT/ALL] expr)

Rcmengembalikan jumlah baris di mana 'expr' bukan NULL.

Contoh: SELECT COUNT (No-Produk) "Jumlah produk".

DARI master.produk;

Output: Jumlah produk = 10

AVG

Sintaks: AVG ([DISTINCT/ALL]n)

Contoh: SELECT AVG (harga jual) "Rata-rata"

FROM master.produk;

Rata-rata

Output : (205.137)

MIN

Sintaks : MIN ([DISTINCT\ALL] expr)

Contoh : SELECT MIN (Batal jatuh tempo) "Saldo Minimum"

FROM klien-master;

Output : 'Jumlah Saldo Terutang

30.000

POWER

Syntax: POWER (m, n)

Rcturns 'm' dinaikkan ke 'n' pangkat 'n' harus bilangan bulat, jika tidak kesalahan akan dikembalikan.

Contoh : SELECT POWER (3,2) "RESULT =" FROM math;

Output : RESULT = 9

ABS

Sintaks : ABS (n)

Mengembalikan nilai absolut 'n'

Contoh: SELECT ABS (-10) "Absolute =" FROM math;

Output : Absolut = 10

LOWER

Syntax: LOWER (char)

Mengembalikan char, dengan semua huruf dalam huruf kecil

Example: SELECT LOWER (‘AGUS WIBOWO’) "Lower Case" FROM math;

Output : Lower Case

Agus Wibowo

UPPER

Syntax : UPPER (char)

Mengembalikan karakter dengan semua huruf dipaksa menjadi huruf besar

Example : SELECT UPPER ('Agus Wibowo') "Result" FROM math;

Output: Result

AGUS WIBOWO

INITCAP

Syntax : INITCAP (char)

Mengembalikan string dengan huruf pertama dalam huruf besar

Example: SELECT INITCAP ('AGUS WIBOWO') "Result" FROM math;

Output: Result

Agus Wibowo

SQRT:

Syntax: SORT (n)

Mengembalikan akar kuadrat dari 'n'. Jika n < 0, NULL. SORT mengembalikan hasil nyata.

Example: SELECT SORT (49) "Square root ="

FROM Math;

Output : Square Root = 7

LENGTH:

Syntax: LENGTH (char)

Mengembalikan panjang char.

Contoh :'SELECT LENGTH ('DANI') "Length"

FROM Match;

Output: Length

5

LTRIM

Syntax: LTRIM (char [, set])

- Menghapus karakter dari kiri char dengan karakter awal dihapus hingga karakter pertama tidak di set.

  Contoh: SELECT LTRIM ('DANI', 'V')

  "Result" FROM Math;

  Output: Result

  DANI

RTRIM

Syntax: RTRIM (char, [set])

- Mengembalikan char, dengan karakter terakhir dihapus setelah karakter terakhir tidak ada di set.
- Contoh: SELECT RTRIM ('DANI X, W)

  "Result" FROM Math;

Output: Result

DANI

2.8 SUBQUERI

Subquery adalah bentuk pernyataan SQL yang muncul di dalam pernyataan SQL lain. Itu juga diubah sebagai kueri bersarang. Pernyataan yang mengandung subquery disebut pernyataan induk. Pernyataan induk menggunakan baris yang dikembalikan oleh subquery. Itu dapat digunakan dengan perintah berikut:

- Untuk menyisipkan record dalam tabel target.
- Untuk membuat tabel dan menyisipkan record pada tabel yang dibuat.
- Untuk memperbarui catatan dalam tabel target.
- Untuk membuat tampilan.
- Untuk memberikan nilai untuk kondisi di WHERE, HAVING, IN dll. digunakan dengan pernyataan SELECT, UPDATE, dan DELETE

Contoh: Ambil semua pesanan yang dilakukan oleh klien bernama 'AGUS WIBOWO' dari tabel pesanan penjualan.

Tabel 2.18 Pesanan penjualan

Order_No	Client_No	Order_Date
A1901	B004	12 April 2021
A1902	B002	13 April 2021
A1903	B007	03 Juni 2021
A1904	B005	20 Mei 2021
A1905	B007	12 Juli 2021

Tabel 2.19 Tabel Client-Master

Client_No	Name	Bal Due
B001	Agus	400
B002	Joni	300
B003	Setiyo	200
B004	Lawren	100
B005	Topo	0
B006	Solikhan	0
B007	DANI SASMOKO	0

```
SELECT * FROM Penjualan-Pesanan (Sales-Order)
        WHERE client-no = (SELECT client-no
FROM client-master
        WHERE Name = 'AGUS WIBOWO);
```

Tabel 2.20 Output Sintaks Diatas

Order_No	Client_No	Order_Date
A1903	B007	03 Juni 2021
A1905	B007	12 Juli 2021

Join

Join dengan Beberapa Tabel (Equi Joins) : Terkadang kita perlu memperlakukan beberapa tabel seolah-olah mereka adalah satu kesatuan. Kemudian satu kalimat SQL dapat memanipulasi data dari semua tabel untuk mencapai ini, kita harus menggabungkan tabel. Tabel digabungkan pada kolom yang memiliki tipe data dan lebar data yang sama dalam tabel.

Contoh: Ambil nomor pesanan, nama klien dan tanggal pesanannya dari tabel master klien dan pesanan penjualan. Tanggal pemesanan harus ditampilkan dalam format 'DD/MM/YY' dan diurutkan dalam urutan menaik.

Tabel 2.21 Pesanan penjualan

Order_No	Client_No	Order_Date
A1901	B006	12 April 2021
A1902	B002	13 April 2021
A1903	B001	03 Juni 2021
A1904	B005	20 Mei 2021
A1905	B004	12 Juli 2021
A1906	B001	

Tabel 2.22 Client-Master

Client_No	Name	Bal Due
B001	Agus	400
B002	Joni	300
B003	Setiyo	200
B004	Lawren	100
B005	Topo	0
B006	Solikhan	0
B007	Dani	0

```
SELECT order-no, to-char (order-date 'DD/MM/YY;) "Order Date"
FROM sales-order, client-master
WHERE client-master. client-no = sales-order. client-no
ORDER BY to-char (order-date, 'DD/MM/YY);
```

Tabel 2.23 Tabel Output

Order_No	Name	Order_Date
A1903		
A1906		
A1901		
A1904		
A1905		
A1902		

Klausa Union

Union menggabungkan output dari dua atau lebih kueri ke dalam satu set baris dan kolom. Contoh: Ambil nama semua klien dan salesman di kota 'Indonesia' dari tabel client-master dan salesman-master.

Tabel 2.24 Client-Master

Client_No	Name	City
A0001	Agus	Ungaran
A0002	Joni	Gunung Pati
A0003	Setiyo	Kendal
A0004	Lawrence	Kediri
A0005	Sulartopo	Semarang
A0006	M. Solikhan	Kudus
A0007	Dani Sasmoko	Semarang

Ambil Nama Semua Klien Dan Salesman Di Kota 'Indonesia' Dari Tabel Client-Master Dan Salesman-Master

Tabel 2.25 salesman-master

Salesman_No	Name	City
B0001	Azeda	Jepara
B0002	Rista	Surabaya
B0003	Adhi	Solo
B0004	Yogga	Semarang

Batasan penggunaan serikat pekerja adalah sebagai berikut:

- Jumlah kolom di semua kueri harus sama.
- Tipe data kolom di setiap kueri harus sama.
- Union tidak dapat digunakan dalam subquery.
- Fungsi agregat tidak dapat digunakan dengan klausa gabungan.

Klausa Intersect

Output dalam klausa berpotongan hanya akan menyertakan baris yang diambil oleh kedua quene. *Contoh*: Ambil nama salesman di 'Indonesia' yang usahanya telah menghasilkan setidaknya satu transaksi penjualan.

Tabel 2.26 Salesman-master

Salesman_No	Name	City
A0001	Azeda	Jepara
A0002	Rista	Surabaya
A0003	Yogga	Semarang
A0004	Nicho	Semarang

Tabel Salesman-Master Mengambil Nama Salesman Di 'Indonesia' Yang Usahanya Telah Menghasilkan Setidaknya Satu Transaksi Penjualan

Tabel 2.27 Sales-order

Order_No	Order_Date	Salesman_No
B0001	12 April 2021	A0001
B0002	14 April 2021	A0003
B0003	03 Juni 2021	A0001
B0004	05 Juni 2021	A0004
B0005	02 Juli 2021	A0003
B0006	12 Juli 2021	A0002

SELECT Salesman_No, Name FROM salesman-master WHERE City = 'Indonesia'
INTERSECT
SELECT salesman-master.Salesman-No, Name FROM salesman-master, sales-order
WHERE salesman-master.Salesman-No = sales-order.Salesman-No;

Tabel 2.28 Output

Salesman_No	Name
A0001	Azeda
A0003	Rista

Note: For the first query.

SELECT salesman No, Name
FROM salesman-master,
WHERE City = 'Indonesia'

Untuk pertanyaan kedua

SELECT salesman-master.Salesnian No, nama
FROM salesman-master, sales-order
WHERE salesman-master.Nomor Penjual = pesanan penjualan.Nomor Penjual;

Tabel 2.29 Tabel pertanyaan ke Dua

Salesman_No	Name
A0001	Azeda
A0003	Yogga
A0001	Azeda
A0004	Nicho
A0003	Yogga
A0002	Rista

Klausa Minus

Klausa Minus menampilkan baris yang dihasilkan oleh kueri pertama, setelah memfilter baris yang diambil oleh kueri kedua. Contoh: Ambil semua nomor produk dari item yang tidak bergerak dari tabel master produk.

Tabel 2.30 Pesanan Penjualan

Order_No	Product_No
A0001	B0001
A0001	B0004
A0001	B0006
A0002	B0002
A0002	B0005
A0003	B0003
A0004	B0001
A0005	B0006
A0005	B0004
A0006	B0006

Tabel 2.31 Produk-master

Product_No	Description
B0001	1.44 Drive
B0002	128 MB RAM
B0003	Keyboard
B0004	Mouse
B0005	Monitors
B0006	HDD
B0007	CD Drive
B0008	128 MB RAM
B0009	Monitors

SELECT Product_No FROM product-master
MINUS
SELECT Product No FROM sales-order

Output: Product No
B0007
B0008
B0009

Note: Untuk kueri pertama
SELECT Product No FROM Product-master

Tabel 2.32 Tabel SELECT Product No FROM Product-master (output)

A :

Product_No
B0001
B0002
B0003
B0004
B0005
B0006
B0007
B0008
B0009

Untuk Kueri kedua
SELECT No Produk FROM sales-order

Tabel 2.33 Tabel SELECT No Produk FROM sales-order (output)

B :

Product_No
A0001
A0004
A0006
A0002
A0005
A0003
A0001
A0006
A0004
A0006

Sekarang

Tabel 2.34 Hasil

A − B =

Product_No
A0007
A0008
A0009

View

Fakta menarik tentang view adalah bahwa view hanya disimpan sebagai definisi dalam katalog sistem Oracle. Ketika referensi dibuat untuk tampilan; definisinya dipindai, tabel dasar dibuka dan tampilan dibuat di atas tabel dasar. Oleh karena itu vjew tidak menyimpan data sama sekali, sampai panggilan khusus ke tampilan dibuat. Ini mengurangi qata yang berlebihan. Pada HDD untuk sebagian besar. Saat tampilan digunakan untuk memanipulasi data tabel, tabel dasar yang mendasarinya akan sama sekali tidak terlihat. Ini akan memberikan tingkat keamanan data yang dibutuhkan. Alasan mengapa tampilan dibuat adalah:

- Ketika keamanan data diperlukan.
- Ketika redundansi data harus dijaga seminimal mungkin dengan tetap menjaga keamanan data.
- Tampilan dapat memberikan independensi data logis.
- Tampilan menyediakan kemampuan "makro".

Pembuatan View:

Sintaks:

```
CREATE VIEW View name As
SELECT column name, column name
FROM table name WHERE column name = expr. List;
GROUP BY kriteria pengelompokan  HAVING predikat
```

Contoh :

(i) Buat tampilan tabel salesman-master untuk departemen penjualan.

```
CREATE VIEW VW-sales AS
SELECT*FROM Salesman-master;
```

(ii) Buat tampilan pada tabel klien-master untuk Departemen Administrasi

```
CREATE VIEW VW-clientadmin AS
SELECT name, address, city, pin code State FROM client-master;
```

Mengganti nama kolom View : Kolom View dapat mengambil nama yang berbeda dari kolom tabel, jika diperlukan. Contoh: CREATE VIEW VW-clientadmin SEBAGAI PILIH nama nama 1, alamat alamat 1, kota, kode pin PIN, status FROM client-master;

2.9 INDEKS

Mengindeks tabel adalah 'strategi akses', yaitu cara untuk mengurutkan dan mencari catatan dalam tabel. Indeks sangat penting untuk meningkatkan kecepatan catatan dapat ditemukan dan diambil dari tabel.

- Pengindeksan melibatkan pembentukan matriks dua dimensi yang sepenuhnya independen dari tabel di mana indeks sedang dibuat.
- Kolom, yang akan menampung data yang diurutkan, diekstraksi dari tabel tempat indeks dibuat.
- Bidang alamat yang mengidentifikasi lokasi catatan dalam database oracle. Bidang alamat ini disebut Rowid.

- Ketika data dimasukkan ke dalam tabel, mesin oracle memasukkan nilai data ke dalam indeks. Untuk setiap nilai data yang disimpan dalam indeks, mesin oracle memasukkan nilai rowid yang unik. Rowid ini menunjukkan dengan tepat di mana record disimpan dalam tabel.

Oleh karena itu setelah nilai data indeks yang sesuai telah ditemukan, mesin oracle menempatkan catatan terkait dalam tabel menggunakan rowid yang ditemukan dalam tabel.
Bidang Alamat dalam Indeks: Bidang alamat indeks disebut ROWID.

Format ROWID yang digunakan oleh Oracle sebagai berikut:
BBBBBBB. RRRR. FFFF
di mana FFFF adalah nomor unik yang diberikan oleh mesin oracle untuk setiap file data.
Misalnya, database dapat berupa kumpulan file data sebagai berikut:

Tabel 2.35 Kumpulan File Data

Data File Name	Data File No	Size of Data File
Student-Ora	1	50 MB
Staff-Ora	2	10 MB
Temporcl-Ora	3	30 MB
Sysorcl-Ora	4	40 MB

BBBBBBB : Setiap file data dibagi lagi menjadi 'Blok Data' dan setiap blok diberi nomor unik. Nomor unik yang ditetapkan untuk blok data pertama dalam file data O. Dengan demikian nomor blok dapat digunakan untuk mengidentifikasi blok data tempat record disimpan. BBBBBBB adalah nomor blok tempat record disimpan.
RRRR : Setiap blok dapat menyimpan satu atau lebih record. Jadi setiap record dalam blok data diberi nomor record yang unik. Nomor record unik yang ditetapkan untuk urutan pertama di setiap blok data adalah O. Dengan demikian, nomor record dapat digunakan untuk mengidentifikasi record yang disimpan dalam sebuah blok. RRRR adalah nomor rekor unik.
Pembuatan Indeks : Indeks dapat dibuat pada satu atau lebih kolom. Berdasarkan jumlah kolom yang termasuk dalam indeks. Sebuah indeks dapat berupa:

- Indeks Sederhana
- Indeks Komposit

Indeks Sederhana: Indeks yang dibuat pada satu kolom tabel disebut indeks sederhana.
Sintaks: CREATE INDEX nama indeks
ON nama tabel (Nama kolom)
Contoh: membuat indeks sederhana pada kolom Roll-No pada tabel siswa.

```
CREATE INDEX IdX-Roll-No
    ON Student (Roll-No);
```

Composite Index: Indeks yang dibuat pada lebih dari satu kolom disebut indeks komposit.
Syntax: CREATE INDEX Nama indeks
ON nama tabel (column name, column name);

Contoh : Buat indeks komposit pada tabel pesanan penjualan pada kolom no pesanan dan no produk.

```
CREATE INDEX idx-sales-order
        ON sales-order (order-no, product-no)
```

2.10 ROW NUM DALAM PERNYATAAN SQL

Untuk setiap baris yang dikembalikan oleh kueri, kolom pseudo ROW NUM mengembalikan angka yang menunjukkan urutan di mana mesin oracle memilih baris dari tabel atau kumpulan baris yang digabungkan. Baris pertama yang dipilih memiliki NUM BARIS 1; Yang kedua memiliki 2 dan seterusnya.

Limitasi : ROW NUM dapat digunakan untuk membatasi jumlah baris yang diambil.

Contoh : Ambil 5 baris pertama dengan menggunakan ROW NUM.

Nama tabel : Siswa

Tabel 2.36 Tabel Siswa

Roll_No	Name
001	Ibnu
002	Kenny
003	Novita
004	Dewi
005	Yesinta
006	Abigail
007	Erwin
008	Nia
	Fay

```
SELECT ROW NUM, Roll-No, Name
FROM student
WHERE ROW NUM < 6;
```

Output:

Tabel 2.37 Output dari Sintaks diatas

RowNum	Roll_No	Name
1	001	Ibnu
2	002	Kenny
3	003	Novita
4	004	Dewi
5	005	Yesinta

2.11 URUTAN (SEQUENCES)

Oracle menyediakan objek yang disebut urutan yang dapat menghasilkan nilai numerik. Nilai yang dihasilkan dapat memiliki maksimal 38 digit. Suatu barisan dapat didefinisikan menjadi

- Menghasilkan angka dalam menaik atau menurun.
- Berikan interval antar angka.
- Caching nomor urut dalam memori.

Membuat Urutan: Informasi minimum yang diperlukan untuk menghasilkan angka menggunakan urutan adalah.

- Nomor awal
- Jumlah maksimum yang dapat dihasilkan oleh urutan.
- Nilai kenaikan untuk menghasilkan nomor berikutnya

Sintaks :

CREATE SEQUENCE Sequence name
[INCREMENT BY integer value
START WITH integer value
MAX VALUE integer value/
/NON MAX VALUE
MIN VALUe integer value/NON MIN VALUE
CYCLE/NO CYCLE
CACHE integer value/NO
CACHE ORDER/NO ORDER]

2.12 KURSOR

Oracle Engine menggunakan area kerja untuk pemrosesan internal untuk mengeksekusi pernyataan SQL. Area kerja ini disebut kursor. Data yang disimpan dalam kursor disebut Kumpulan Data aktif'. Ukuran kursor dalam memori adalah ukuran yang diperlukan untuk menampung jumlah baris dalam Kumpulan Data Aktif.

Tabel 2.38 Ukuran Kursor

SERVER RAM			
Active Data Set			
11	Abigail	Eng.	20000
12	Erwin	Eng.	20000
13	Nia	Analyst	15000
14	Fay	Manager	15000

Conten Kursor

Saat pengguna menjalankan pernyataan pilih sebagai:

SELECT EMP No, EName, Job, Salary
FROM Employee
WHERE Dept No = 20

Kumpulan data yang dihasilkan di kursor dibuka di server dan akan ditampilkan seperti yang ditunjukkan di atas. Ketika kursor dimuat dengan beberapa baris melalui kueri, mesin oracle membuka dan mempertahankan penunjuk baris. Bergantung pada permintaan pengguna untuk melihat data, penunjuk baris akan dipindahkan dalam Kumpulan Data Aktif kursor.

Jenis Kursor: Kursor diklasifikasikan tergantung pada keadaan di mana mereka dibuka. Ini adalah jenis berikut:

- Kursor Tersirat
- Kursor Eksplisit
- **Kursor Implisit:** Jika mesin oracle untuk pemrosesan internalnya telah membuka kursor, mereka dikenal sebagai kursor implisit. Itu adalah kursor yang membuka mesin oracle untuk pemrosesan internalnya disebut kursor implisit.
- **Kursor Eksplisit:** Seorang pengguna juga dapat membuka kursor untuk memproses data sesuai kebutuhan. Kursor yang ditentukan pengguna seperti itu dikenal sebagai 'Kursor Eksplisit'.

Atribut Kursor: Baik kursor Implisit maupun Eksplisit memiliki empat atribut.

(i) %**ISOPEN** : Mengembalikan TRUE jika kursor terbuka, FALSE sebaliknya.

(ii) %**FOUND**: Mengembalikan TRUE jika record berhasil diambil, FALSE sebaliknya.

(iii) %**NOT FOUND**: Mengembalikan TRUE jika record tidak berhasil diambil FALSE sebaliknya.

(iv) %**ROW COUNT**: Mengembalikan jumlah record yang diproses dari kursor.

Kekurangan Kursor Implisit: Kursor implisit memiliki kelemahan sebagai berikut :

- Ini kurang efisien daripada kursor eksplisit.
- Ini lebih rentan terhadap kesalahan data.
- Ini memberi Anda lebih sedikit kontrol terprogram.

Deklarasi Kursor: Kursor didefinisikan di bagian deklaratif dari blok PL/SQL. Ini dilakukan dengan memberi nama kursor dan memetakannya ke kueri. Ketika kursor dideklarasikan, mesin oracle diberitahu bahwa kursor dari nama tersebut perlu dibuka. Deklarasi itu hanya sebuah isyarat. Tidak ada alokasi memori pada saat ini.

CURSOR cursor-name [([Paramater [, Parameter ... 1)]
[RETURN return-specification]
IS SELECT-Statement.
Where Cursor-name: Nama Kursor
return-specification : Opsional dari RETURN klause untuk kursor .

Where **Cursor-name**: Nama Cursor return-specification: Klausa RETURN opsional untuk kursor.

SELECT-Statement: Setiap pernyataan SQL SELECT yang valid. Tiga perintah yang digunakan untuk mengontrol kursor selanjutnya adalah buka, ambil, dan tutup.

Fetch: Pernyataan Ambil memindahkan data yang disimpan dalam Kumpulan Data Aktif ke dalam variabel memori. Pernyataan fetch ditempatkan di dalam loop ... Konstruksi loop akhir, yang menyebabkan data diambil ke dalam variabel memori dan diproses hingga semua baris dalam Kumpulan Data Aktif diproses.

Sintaks: CURSOR nama kursor IS SELECT Pernyataan;

Opening Cursor: Membuka kursor akan mengeksekusi kueri dan membuat set aktif yang berisi semua baris, yang memenuhi kriteria pencarian kueri. Pernyataan terbuka mengambil catatan dari tabel database dan menempatkan catatan di kursor. Kursor dibuka di memori server.

Sintaks: OPEN nama kursor;

Closing a Cursor: Pernyataan tutup menonaktifkan kursor dan set aktif menjadi tidak terdefinisi. Ini akan melepaskan memori yang ditempati oleh kursor dan kumpulan datanya baik di klien maupun di server.
Sintaks: CLOSE Nama Kursor.

2.13 TRIGGER DATABASE

Trigger terdiri dari kode PL/SQL, yang mendefinisikan beberapa tindakan yang harus dilakukan database ketika beberapa peristiwa terkait database terjadi. Oracle Engine memungkinkan kita untuk mendefinisikan prosedur yang dijalankan secara implisit ketika pernyataan insert, update, atau delete dikeluarkan terhadap tabel terkait. Jenis prosedur ini disebut Trigger Database.

Penggunaan Triger Database, berikut ini

- Trigger dapat mengizinkan pernyataan DML terhadap tabel hanya jika dikeluarkan, selama jam kerja reguler.
- Trigger juga dapat digunakan untuk menjaga sifat audit dari tabel, bersama dengan operasi yang dilakukan dan waktu operasi dilakukan.
- Terapkan otorisasi keamanan yang kompleks.

Bagian dasar Trigger:

- Trigger memiliki tiga bagian dasar
- Peristiwa atau pernyataan yang memicu
- Pembatasan Trigger
- Tindakan Trigger

Memicu Peristiwa atau Pernyataan: Ini adalah pernyataan SQL yang menyebabkan Trigger dipecat. Itu bisa INSERT, UPDATE atau DELETE Statement untuk tabel tertentu.

Pembatasan Trigger: Pembatasan Trigger menentukan ekspresi Boolean yang harus TRUE agar Trigger diaktifkan. Ini adalah opsi yang tersedia untuk Trigger yang diaktifkan untuk setiap baris. Pembatasan Trigger ditentukan menggunakan klausa WHEN.

Tindakan Trigger: Tindakan Trigger adalah kode PL/SQL yang akan dieksekusi ketika pernyataan Trigger ditemukan dan pembatasan Trigger apa pun dievaluasi ke TRUE.

Jenis Trigger: Ini adalah jenis berikut:

Trigger Baris: Trigger baris dipicu setiap kali baris dalam tabel dipengaruhi oleh pernyataan Trigger. *Contoh*: Jika pernyataan UPDATE memperbarui beberapa baris tabel, Trigger baris diaktifkan sekali untuk setiap baris yang terpengaruh oleh pernyataan UPDATE. Trigger baris harus digunakan saat beberapa pemrosesan diperlukan setiap kali pernyataan Trigger memengaruhi satu baris dalam tabel.

Trigger Pernyataan: Trigger pernyataan dipicu sekali atas nama pernyataan Trigger, terlepas dari jumlah baris yang dipengaruhi oleh pernyataan Trigger. Trigger pernyataan harus digunakan ketika pernyataan Trigger memengaruhi baris dalam tabel tetapi pemrosesan yang diperlukan sepenuhnya independen dari jumlah baris yang terpengaruh

BEFORE TRIGGER: Trigger menjalankan tindakan Trigger sebelum pernyataan Trigger. Jenis Trigger ini biasanya digunakan dalam situasi berikut:

(i) BEFORE Trigger digunakan ketika tindakan Trigger harus menentukan apakah pernyataan Trigger harus diselesaikan atau tidak.

(ii) BEFORE trigger digunakan untuk menurunkan nilai kolom tertentu sebelum menyelesaikan pernyataan INSERT atau UPDATE yang memicu.

AFTER Triggers : AFTER triggers menjalankan aksi trigger setelah pernyataan triggering dieksekusi. Jenis Trigger ini biasanya digunakan dalam situasi berikut:

(i) AFTER trigger digunakan saat Anda ingin pernyataan triggering selesai sebelum mengeksekusi trigger action.

(ii) Jika Trigger BEFORE sudah ada, Trigger AFTER dapat melakukan tindakan yang berbeda pada pernyataan Trigger yang sama.

Catatan: Ketika sebuah trigger di-ftred, sebuah pernyataan SQL di dalam blok kode PL/SQL trigger juga dapat menggunakan trigger yang sama atau beberapa trigger lainnya. Ini disebut 'Trigger berjenjang'.

Database Triggers versus Prosedur

Ada sangat sedikit perbedaan antara trigger dan prosedur Database

- Trigger tidak menerima parameter sedangkan prosedur bisa.
- Trigger dieksekusi secara implisit oleh mesin oracle itu sendiri setelah modifikasi tabel terkait atau datanya. Untuk mengeksekusi suatu prosedur, itu harus secara eksplisit disebut pengguna.

Perbedaan antara prosedur dan fungsi :

Sebuah fungsi harus mengembalikan hanya satu nilai kembali ke pemanggil. Sementara prosedur tidak pernah dapat mengembalikan nilai ke pemanggil.

2.14 PAKET ORACLE

Paket adalah objek oracle, yang menyimpan objek lain di dalamnya. Objek-objek yang biasa disimpan dalam sebuah paket adalah prosedur, fungsi, variabel, konstanta, kursor, dan pengecualian. Ini adalah cara membuat kode generik, enkapsulasi, dan dapat digunakan kembali. Komponen Paket Oracle : Sebuah paket biasanya memiliki dua komponen :

- Sebuah spesifikasi.
- Sebuah tubuh

Spesifikasi: Spesifikasi paket mendeklarasikan tipe, konstanta variabel memori. Pengecualian, kursor, dan sub program yang tersedia untuk digunakan. **Tubuh:** Sebuah paket tubuh penuh dermes kursor, fungsi, dan prosedur dan dengan demikian mengimplementasikan spesifikasi.

Keuntungan dari Paket : Ini adalah keuntungan berikut:

(i) Paket memungkinkan pengorganisasian aplikasi komersial menjadi modul yang efisien.

(ii) Paket memungkinkan pemberian hak istimewa secara efisien.

(iii) Variabel publik dan kursor paket tetap ada selama sesi berlangsung. Oleh karena itu, semua kursor dan prosedur yang dijalankan di lingkungan ini dapat membagikannya.

(iv) Paket memungkinkan prosedur dan fungsi kelebihan beban bila diperlukan.

(v) Paket meningkatkan kinerja dengan memuat beberapa objek ke dalam memori sekaligus.

(vi) Paket mempromosikan penggunaan kembali kode melalui penggunaan perpustakaan yang berisi prosedur dan fungsi tersimpan, sehingga mengurangi pengkodean yang berlebihan.

2.17 ASSERTION/ASSERSI

Asersi adalah predikat yang menyatakan suatu kondisi yang kita harapkan selalu dipenuhi oleh database. Kendala domain dan batasan integritas referensial adalah bentuk khusus dari pernyataan. Mereka mudah diuji dan diterapkan ke berbagai aplikasi database. Sebuah pernyataan dalam SQL mengambil bentuk

CREATE ASSERTION < assertion-name>

Check < predicate>

Dua contoh kendala tersebut adalah:

- Jumlah semua jumlah pinjaman untuk setiap cabang harus kurang dari jumlah semua saldo rekening di cabang.
- Setiap pinjaman memiliki setidaknya satu pelanggan yang memiliki rekening dengan saldo minimal Rp. 1000.000.

Ketika sebuah pernyataan dibuat, sistem menguji validitasnya. Jika asersi valid, maka modifikasi apa pun di masa mendatang pada Database hanya diperbolehkan jika tidak menyebabkan asersi tersebut dilanggar. Oleh karena itu, pernyataan harus digunakan dengan sangat hati-hati.

2.18 PENYELESAIAN MASALAH

Pertanyaan 1 : Apa fitur dari PL/SQL?

Jawaban : Fitur PL/SQL : Ini adalah sebagai berikut

(1) PL/SQL menerima entri pernyataan ad hoc

(2) Menerima input SQL dari file.

(3) Ini menyediakan editor baris untuk memodifikasi pernyataan SQL.

(4) Ini mengontrol pengaturan lingkungan.

(5) Ini memformat hasil kueri ke dalam laporan dasar.

(6) Mengakses database lokal dan menghapus.

Pertanyaan 2 : Apa perbedaan antara SQL dan SQL * PLUS

Jawaban :

Tabel 2.39 Perbedaan antara SQL dan SQL * PLUS

SQL	**SQL*Plus**
SQL adalah bahasa untuk berkomunikasi dengan server oracle dan database lain untuk mengakses data.	SQL*Plus mengenali pernyataan SQL yang dikirim ke server

SQL didasarkan pada SQL standar ANSI.	SQL * Plus adalah antarmuka kepemilikan Oracle untuk mengeksekusi pernyataan SQL.
SQL memanipulasi data dan definisi tabel dalam database	SQL *Plus tidak mengizinkan manipulasi nilai dalam database.
SQL dimasukkan ke dalam buffer SQL pada satu atau lebih baris	SQL * Plus dimasukkan satu baris pada satu waktu, tidak disimpan dalam buffer SQL.
Itu tidak bisa disingkat.	Itu bisa disingkat.
Ia menggunakan karakter terminasi untuk menjalankan perintah dengan segera	Itu tidak memerlukan karakter penghentian; mengeksekusi perintah dengan segera.
Ini menggunakan fungsi untuk melakukan beberapa pemformatan	Ini menggunakan perintah untuk memformat data.

Pertanyaan 3 : Tulislah penegasan untuk pernyataan berikut. (UPTU 2005.06) *"Setiap pinjaman memiliki setidaknya satu pelanggan yang memiliki rekening dengan saldo minimum Rp. 1000 di sistem perbankan.*

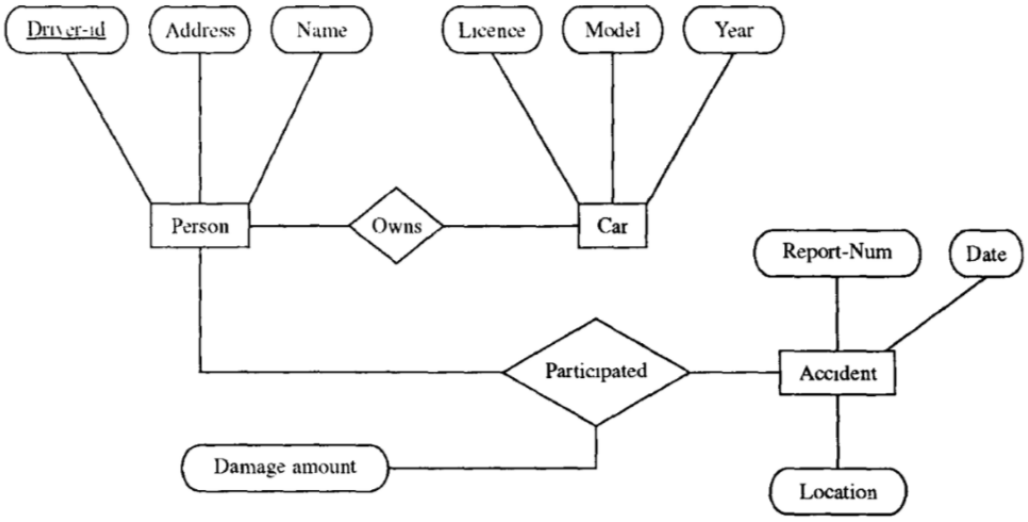
Jawaban : Buat pernyataan cek batasan saldo (tidak ada (pilih * dari pinjaman jika tidak ada (pilih * dari peminjam, depositor, rekening)

di mana loan.loan-number = peminjam.loan-number
dan peminjam. nama-pelanggan = depositor.nama-pelanggan
dan penyimpan. nomor-rekening = rekening.nomor-rekening
dan rekening.saldo > = 1000));

Pertanyaan 4: Tulislah penegasan untuk pernyataan berikut. *"Jumlah semua jumlah pinjaman untuk setiap cabang harus kurang dari jumlah semua saldo rekening di cabang."*

Jawaban : Buat pernyataan sum-constraint check (tidak ada (pilih jumlah (jumlah) dari pinjaman dimana loan.branch-name = branch.branch-name)
> = (pilih jumlah (saldo) dari rekening dimana
rekening.nama-cabang = cabang.nama-cabang)));

***Pertanyatan 5*:** Merancang Database relasional yang sesuai dengan diagram E-R yang diberikan



Gambar 2.2 Diagram E-R

Jawaban : Skema Database relasional adalah sebagai

PERSON (driver_id, Nama, alamat)

CAR (Lisensi, model, tahun)

ACCIDENT (Report-num, date, location)

OWNS (driver Jd, licence)

ParticIpated

PARTICIPATED (driver_id, report-num, licence, damage-amount)

EMPLOYEE (person-name, address, city)

COMPANY (company-name, city)

Pertanyaan 6*:* Perhatikan database relasional berikut ini:

CUSTOMER (nama customer, Jalan, Kota pelanggan)

BRANCH (nama cabang, Aset, cabang kota)

DEPOSIT (nama cabang, No rekening, nama customer, saldo)

Give SQL DDL Definisi database ini.

Jawaban:

CREATE TABLE CUSTOMER

(Customer-name char (20), street varchar (10), customer city char (15));

CREATE TABLE BRANCH

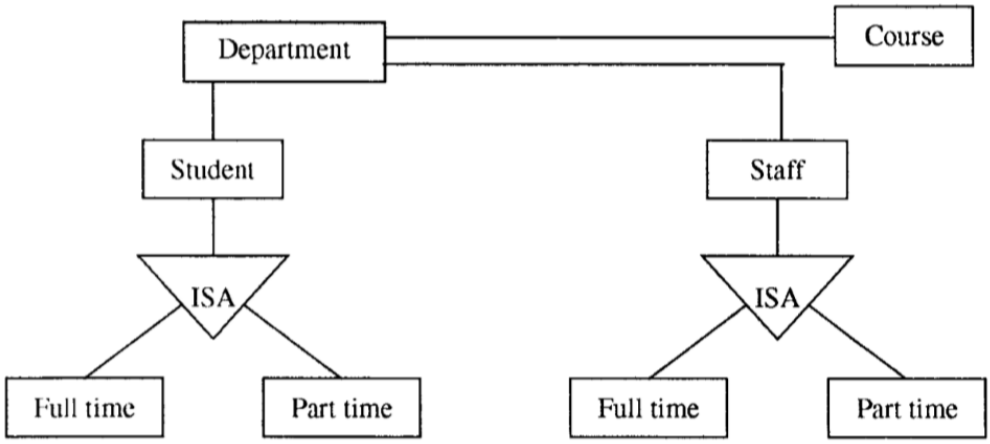
(Branch-name char (15), Assets varchar (20), Branch-city char (15));

CREATE TABLE DEPOSIT (Branch-Name char (15), Account-number number (10),

Customer-name char (15), Balance number (10));

Pertanyaan 7**:** Sebuah universitas memiliki banyak jurusan. Setiap departemen mungkin memiliki siswa penuh waktu dan paruh waktu. Setiap departemen dapat menawarkan beberapa kursus untuk siswanya sendiri. Setiap departemen memiliki anggota staf yang mungkin penuh waktu dan paruh waktu. Merancang generalisasi, hierarki spesialisasi untuk universitas.

Jawaban :



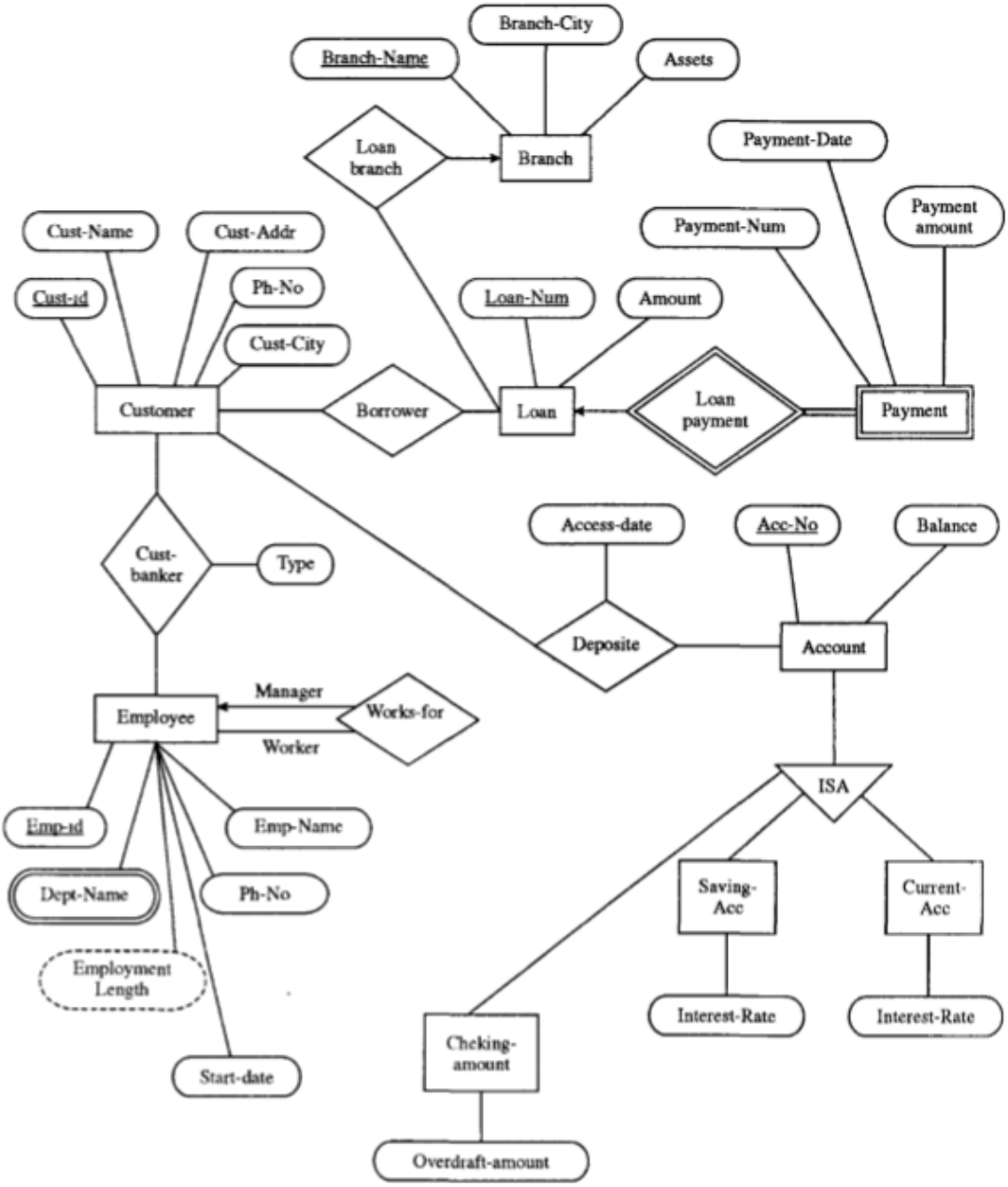
Gambar 2.3 Rancangan Generalisasi, Hierarki Spesialisasi Untuk Universitas

Pertanyaan 8**:** Perhatikan skema berikut untuk database PROJECT
Proyek (No-Proyek, Nama-Proyek, Manajer-Proyek)
Karyawan (No-Karyawan, Nama-Karyawan)
Ditugaskan ke (No-Proyek, No-Karyawan)
Menulis pernyataan SQL-DDL untuk database PROJECT
Pernyatasn SQL harus secara jelas menunjukkan kunci utama dan kunci asing.
Jawaban : CREATE TABLE PROJECT
(Project-no, number (8) PRIMARY KEY, Project-name
varchar2 (20), Project-manager char (15));•

- CREATE TABLE EMPLOYEE (Employee-no number (6) PRIMARY KEY, Employee-name char (30));
- CREATE TABLE ASSIGNED TO
  (Project-no number (8), Employee-no number (6),
  PRIMARY KEY (Project-no, Employee-no),
  FOREIGN KEY (Project-no)
  REFERENCES (Project-1));

Pertanyaan 9: Gambar diagram untuk perusahaan perbankan.

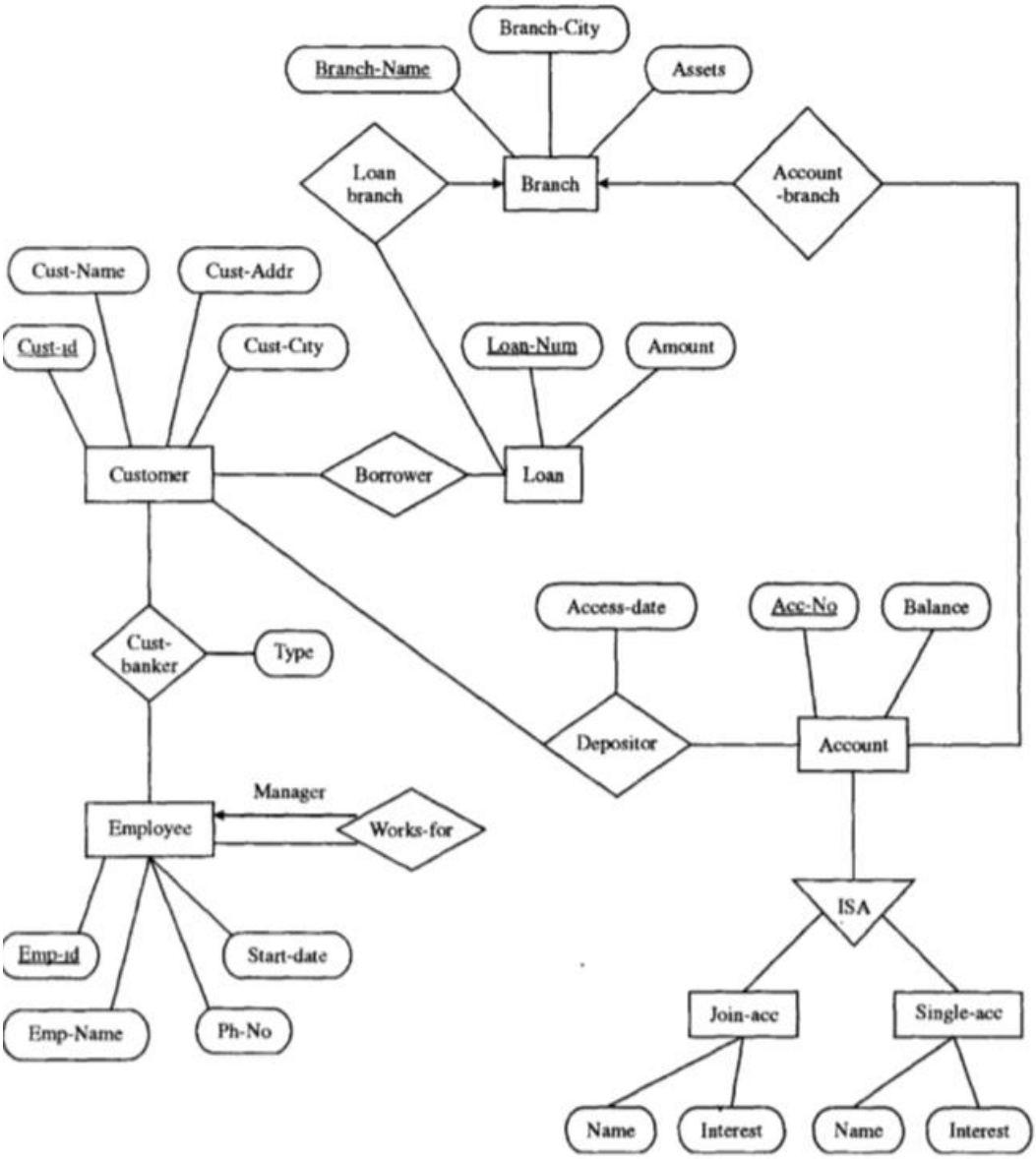
Jawaban :



Gambar 2.4 Gambar Diagram Untuk Perusahaan Perbankan

Pertanyaan 10:

(a) Pertimbangkan serangkaian persyaratan berikut untuk database bank: bank besar memiliki beberapa cabang di tempat yang berbeda. Setiap cabang menyimpan rincian rekening pelanggan. Pelanggan dapat membuka bergabung serta rekening tunggal. Bank juga memberikan pinjaman kepada nasabah untuk berbagai tujuan. Bank menyimpan catatan setiap transaksi oleh pelanggan ke rekeningnya. Semua cabang memiliki karyawan dan beberapa karyawan adalah manajer". Gambarlah diagram E-R yang menangkap informasi ini.



Gambar 2.5 Informasi Untuk Menggambar Diagram E-R

(b) Transformasikan diagram ER ini ke skema Database relasional

Jawaban :

Customer (customer-id, customer-name, customer-address, customer-city)
Loan (loan-num, amount) Branch (branch-name, branch-city, assets)
Employee (emp-id, emp-name, ph-no, start-date)
Account (acc-num, balance) Borrower (customer-id, loan-num)
Loan-branch (branch-name, loan-num)
Depositor (customer-id, acc-num)
Joint-ace (name, interest)
Single-ace (name, interest)

Contoh Aljabar Relasional

Pertanyaan 11: Pilih tupel karyawan siapa

a. DEPT_NO adalah 10

b. GAJI lebih besar dari 80.000

Jawaban :

a. $^{\sigma}$DEPT NO = 10 (Karyawan

b. $^{\sigma}$GAJI > 80.000 (Karyawan)

Pertanyaan 12: Pilih tupel untuk semua karyawan di KARYAWAN yang bekerja di DEPT_NO 10 dan mendapatkan gaji tahunan lebih dari IDR 80.000 atau bekerja di DEPT_NO 12 dan mendapatkan gaji tahunan lebih dari IDR 90.000.

Jawaban: $^{\sigma}$ (DEPT-NO = 10 DAN GAJI > 80.000) ATAU (Karyawan)

(DEPT-NO = 12 DAN GAJI > 90.000)

***Pertanyaan 13*:** Cantumkan Nomor Induk Kepegawaian (EMP_ID), Nama, (EMP_NAME) dan gaji (GAJI).

Jawaban : $^{\pi}$EMP_ID, EMP_NAME, GAJI (Karyawan)

***Pertanyaan 14*:** Mendapatkan kembali nama (EMP_NAME) dan gaji (GAJI) dari semua karyawan yang ada di relasi KARYAWAN yang bekerja di DEPT NO 10.

Jawaban : $^{\pi}$EMP _NAME, GAJI ($^{\sigma}$ DEPT= 10) (Karyawan)

OR EMP-DEPT-lO ← (O($^{\sigma}$ DEPT-NO-IO) (Karyawan)

RESULTS ← ($^{\pi}$EMP-NAME, SALARY (EMP-DEPT -10))

Pertanyaan 15: Mendapatkan nomor identifikasi karyawan semua karyawan baik yang bekerja di DEPT-NO 10 maupun yang langsung membawahi karyawan yang bekerja di DEPT-NO = 10.

Jawaban : EMP-DEPT ← ($^{\sigma}$ DEPT-NO=lO (Employee))

RESULTl ← $^{\pi}$EMP-ID (Employee)

RESULT2 (EMP-ID) ← $^{\pi}$EMP-SUPERV (EMP-DEP-lO)

FINAL RESULT ← RESULTl U RESULT2

***Pertanyaan 16*:** Ambil untuk setiap karyawan wanita (EMP-SEX = 'F') daftar nama tanggungan (EMP-DEPENDENT).

Jawaban : FEMALE-EMP ← ($^{\sigma}$ EMP-SEX='F' (Employee))

ALL-EMP ← $^{\pi}$EMP-ID, EMP-NAME (FEMALE_EMP)

DEPENDENTS ← ALL-EMPxEMP-DEPENDENT

ACTUAL-DEP ← ($^{\sigma}$EMP-ID=FEPT-ID (DEPENDENTS))

FINAL-RESULT ← $^{\pi}$EMP-NAME, DEPENDENT-NAME (ACTUAL_DEP)

***Pertanyaan 17*:** Dapatkan nama manajer masing-masing departemen (DEPT).

Jawaban : DEPT-MANAGER ← DEPT MANAGER-ID=EMP-ID (Employee)

FINAL-RESULT ←$^{\pi}$DEPT-NAME,EMP-NAME (DEPT_MANAGER)

***Pertanyaan 18*:** Cari tahu nama dan alamat semua karyawan yang bekerja di departemen 'komputer'.

Jawaban : COMP-DEP ←$^{\sigma}$DNAME='COMPUTER' (Department)

COMP-EMPS ← (COMP-DEP DNUMBER = DNDEMPLOYEE)

RESULT ← $^{\pi}$NAME, ADDRESS (COMP _ EMPS)

***Pertanyaan 19*:** Untuk setiap proyek yang berlokasi di 'Indonesia', sebutkan nomor proyek, nomor departemen pengontrol dan nama belakang manajer departemen, alamat dan data kelahiran.

Jawaban : INDONESIA-PROJS ← ᵅPLOCATION='INDONESIA' (Project)

CONTR-DEPT ← (INDONESIA-PROJS DNUM=DNUMBER (Department))

PROJ-DEPT-MGR ← (CONTRO-DEPT MGRSSN=SSN (Employee))

RESULT ← C ᵀPNUMBER, DNUM, LNAME, ADDRESS, BDATE (PROJ _DEPT_ MGR)

***Pertanyaan 20*:** Temukan nama karyawan yang mengerjakan semua proyek yang dikendalikan oleh nomor departemen

Jawaban : DEPT-PROJS (PNO) ←ᵀPNUMBER (O'DNUM=10 (Project))

EMP-PROJ (SSN, PNO) ←ᵀESSN PNO (Work-On)

RESULT-EMP-SSN ← EMP-PROJ + DEPT-PROJS

RESULT ←ᵀNAME (RESULT-MP-SSNS*EMPLOYEE)

Pertanyaan 21: Menampilkan nama karyawan yang tidak memiliki tanggungan.

Jawaban : ALL-EMPS ←ᵀSSN (Employee)

EMP-WITH-DEPS (SSN) ←ᵀESSN (Dependent)

EMP-WITHOUT-DEPS ← (ALL-EMPS-EMP-WITH-DEPS)

RESULT ←ᵀNAME (EMPS-WITHOUT-DEPS*EMPLOYEE)

***Pertanyaan 22*:** Sebutkan nama pengurus yang memiliki minimal satu tanggungan.

Jawaban : MGRS (SSN) ←ᵀMGRSSN (Department)

EMPS-WITH-DEPS (SSN) ←ᵀESSN (Dependent)

MGRS-WITH-DEPS ← (MGRS ∩ EMPS-WITH-DEPS)

RESULT ←ᵀNAME (MGRS-WITH-DEPS*EMPLOYEE)

***Pertanyaan 23*:** Jelaskan secara singkat istilah-istilah berikut : Atribut, domain, entitas, relasi,. himpunan entitas, himpunan relasi, relasi satu ke banyak, relasi banyak ke banyak, batasan partisipasi, batasan tumpang tindih, himpunan entitas lemah, kesepakatan, dan indikator peran.

Jawaban : Penjelasan istilah :

- ***Atribut*** - properti atau deskripsi suatu entitas. Entitas karyawan departemen mainan dapat memiliki atribut yang menjelaskan nama karyawan, gaji, dan masa kerja.
- ***Domain*** - satu set nilai yang mungkin untuk sebuah atribut.
- ***Entity*** - sebuah objek di dunia nyata yang dapat dibedakan dari objek lain seperti mainan naga hijau.
- ***Relasi*** - asosiasi antara dua atau lebih entitas.
- ***Kumpulan entitas*** - kumpulan entitas serupa seperti semua mainan di departemen mainan.
- ***Himpunan hubungan*** - kumpulan hubungan serupa.
- ***hubungan satu ke banyak*** - batasan utama yang menunjukkan bahwa satu entitas dapat dikaitkan dengan banyak entitas lain. Contoh hubungan satu ke banyak adalah ketika seorang karyawan hanya dapat bekerja untuk satu departemen, dan sebuah departemen dapat memiliki banyak karyawan.
- ***Hubungan banyak ke banyak*** - batasan utama yang menunjukkan bahwa banyak dari satu entitas dapat dikaitkan dengan banyak entitas lain. Contoh hubungan banyak ke

banyak adalah karyawan dan hobi mereka: seseorang dapat memiliki banyak hobi yang berbeda, dan banyak orang dapat memiliki hobi yang sama.

- ***Batasan partisipasi*** **-** batasan partisipasi menentukan apakah hubungan harus melibatkan entitas tertentu. Contohnya adalah jika setiap entitas departemen memiliki entitas manajer. Batasan partisipasi dapat bersifat total atau sebagian. Batasan partisipasi total mengatakan bahwa setiap departemen memiliki manajer. Batasan partisipasi parsial mengatakan bahwa setiap karyawan tidak harus menjadi manajer.
- ***Batasan tumpang tindih*** - dalam hierarki ISA, batasan tumpang tindih menentukan apakah dua subkelas dapat berisi entitas yang sama atau tidak.
- ***Batasan penutup*** - dalam hierarki ISA, batasan penutup menentukan di mana entitas dalam subkelas secara kolektif mencakup semua entitas dalam superkelas. Misalnya, dengan entitas Karyawan yang ditetapkan dengan subkelas HourlyEmployee dan SalaryEmployee, apakah setiap entitas Karyawan harus berada dalam HourlyEmployee atau SalaryEmployee?
- ***Kumpulan entitas yang lemah*** - entitas yang tidak dapat diidentifikasi secara unik tanpa mempertimbangkan beberapa atribut kunci utama dari entitas pemilik pengidentifikasi lainnya. Contohnya termasuk informasi Dependent untuk karyawan untuk tujuan asuransi.
- ***Agregasi*** - fitur dari model hubungan entitas yang memungkinkan suatu kumpulan hubungan untuk berpartisipasi dalam kumpulan hubungan lain. Ini ditunjukkan pada diagram ER dengan menggambar kotak putus-putus di sekitar agregasi.
- ***Indikator peran*** - Jika satu set entitas memainkan lebih dari satu peran, ·indikator peran menjelaskan tujuan yang berbeda dalam hubungan. Contohnya adalah entitas Karyawan tunggal yang ditetapkan dengan relasi Reports-To yang menghubungkan supervisor dan bawahan.

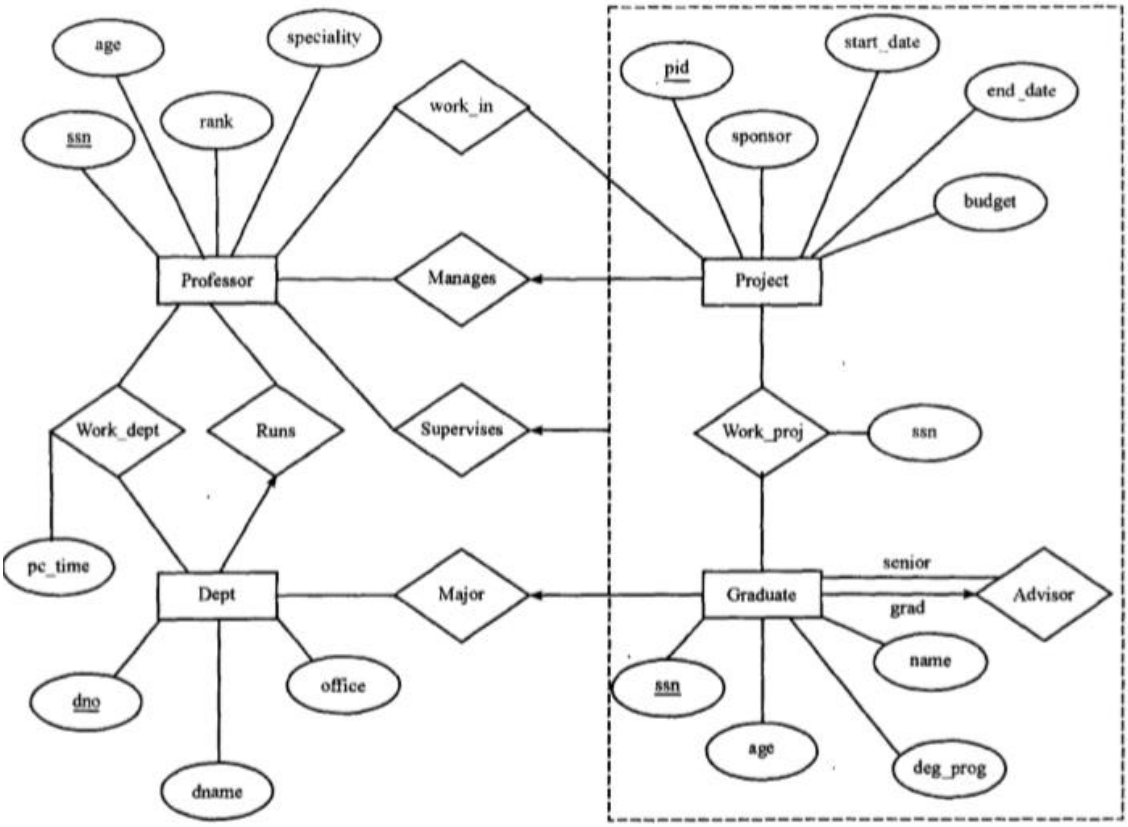
Pertanyaan 24*:* Simak informasi tentang database universitas berikut ini:

Jawaban:

- Profesor memiliki SSN, nama, usia, pangkat, dan spesialisasi penelitian.
- Proyek memiliki nomor proyek, nama sponsor (misalnya, NSF), tanggal mulai, tanggal berakhir, dan anggaran.
- Mahasiswa pascasarjana memiliki SSN, nama, usia, dan program gelar (misalnya, MS atau Ph. D.).
- Setiap proyek dikelola oleh satu profesor (dikenal sebagai peneliti utama proyek).
- Setiap proyek dikerjakan oleh satu atau lebih profesor (dikenal sebagai rekan penyelidik proyek).
- Profesor dapat mengelola dan/atau mengerjakan banyak proyek.
- Setiap proyek dikerjakan oleh satu atau lebih mahasiswa pascasarjana (dikenal sebagai asisten peneliti proyek).
- Ketika mahasiswa pascasarjana bekerja pada sebuah proyek, seorang profesor harus mengawasi pekerjaan mereka pada proyek tersebut. Mahasiswa pascasarjana dapat

mengerjakan beberapa proyek, dalam hal ini mereka akan memiliki supervisor (yang berpotensi berbeda) untuk masing-masing proyek.

- Departemen memiliki nomor departemen, nama departemen, dan kantor utama.
- Departemen memiliki seorang profesor (dikenal sebagai ketua) yang menjalankan departemen.
- Profesor bekerja di departemen nada atau lebih, dan untuk setiap departemen tempat mereka bekerja, persentase waktu dikaitkan dengan pekerjaan mereka.
- Mahasiswa pascasarjana memiliki satu departemen utama di mana mereka mengerjakan gelar mereka.
- Setiap mahasiswa pascasarjana memiliki mahasiswa pascasarjana lain yang lebih senior (dikenal sebagai penasihat mahasiswa) yang menasihatinya tentang kursus apa yang harus diambil.



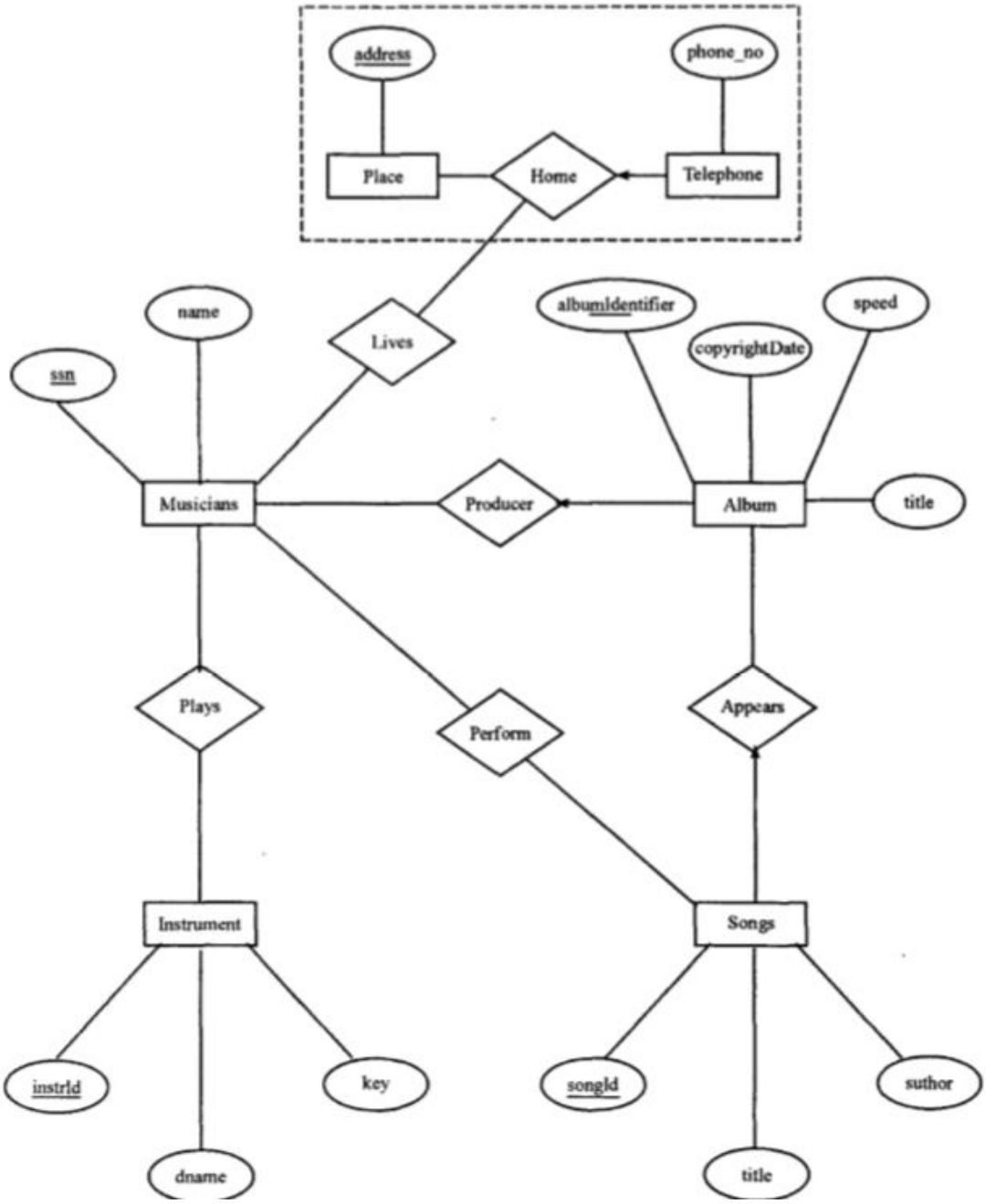
Gambar 2.6 Diagram ER

Rancang dan gambar diagram ER yang menangkap informasi tentang universitas. Gunakan hanya model ER dasar di sini; yaitu, entitas, hubungan, dan atribut. Pastikan untuk menunjukkan batasan kunci dan partisipasi.

Pertanyaan 25: Notown Records memutuskan untuk menyimpan informasi tentang musisi yang tampil di albumnya (serta data perusahaan lainnya) di database. Perusahaan telah dengan bijak memilih untuk mempekerjakan Anda sebagai perancang Database (dengan biaya konsultasi biasa sebesar Rp. 250000/hari).

- Setiap musisi yang merekam di Notown memiliki SSN, nama, alamat, dan nomor telepon. Musisi yang dibayar rendah sering berbagi alamat yang sama, dan tidak ada alamat yang memiliki lebih dari satu telepon.
- Setiap instrumen yang digunakan dalam lagu yang direkam di Notown memiliki nomor identifikasi unik, nama (misalnya, gitar, synthesizer, seruling) dan kunci musik (misalnya, C, B-flat, E-flat).
- Setiap album yang direkam pada label Notown memiliki nomor identifikasi unik, judul, data hak cipta, format (misalnya, CD atau MC), dan pengidentifikasi album.
- Setiap lagu yang direkam di Notown memiliki judul dan pengarang.
- Setiap musisi dapat memainkan beberapa instrumen, dan instrumen tertentu dapat dimainkan oleh beberapa musisi.
- Setiap album memiliki sejumlah lagu di dalamnya, tetapi tidak ada lagu yang dapat muncul di lebih dari satu album.
- Setiap lagu dibawakan oleh satu atau lebih musisi, dan seorang musisi dapat membawakan sejumlah lagu.
- Setiap album memiliki tepat satu musisi yang bertindak sebagai produsernya. Seorang musisi dapat menghasilkan beberapa alumbs, tentu saja.

Jawaban : Rancang skema konseptual untuk Notown dan -gambar diagram ER untuk skema Anda. Informasi sebelumnya menjelaskan situasi yang harus dimodelkan oleh database Nottown. Pastikan untuk menunjukkan semua batasan kunci dan kardinalitas dan asumsi apa pun yang Anda buat. Identifikasi kendala apa pun yang tidak dapat Anda tangkap dalam diagram ER dan jelaskan secara singkat mengapa Anda tidak dapat mengungkapkannya.



Gambar 2.7 *Diagram ER*

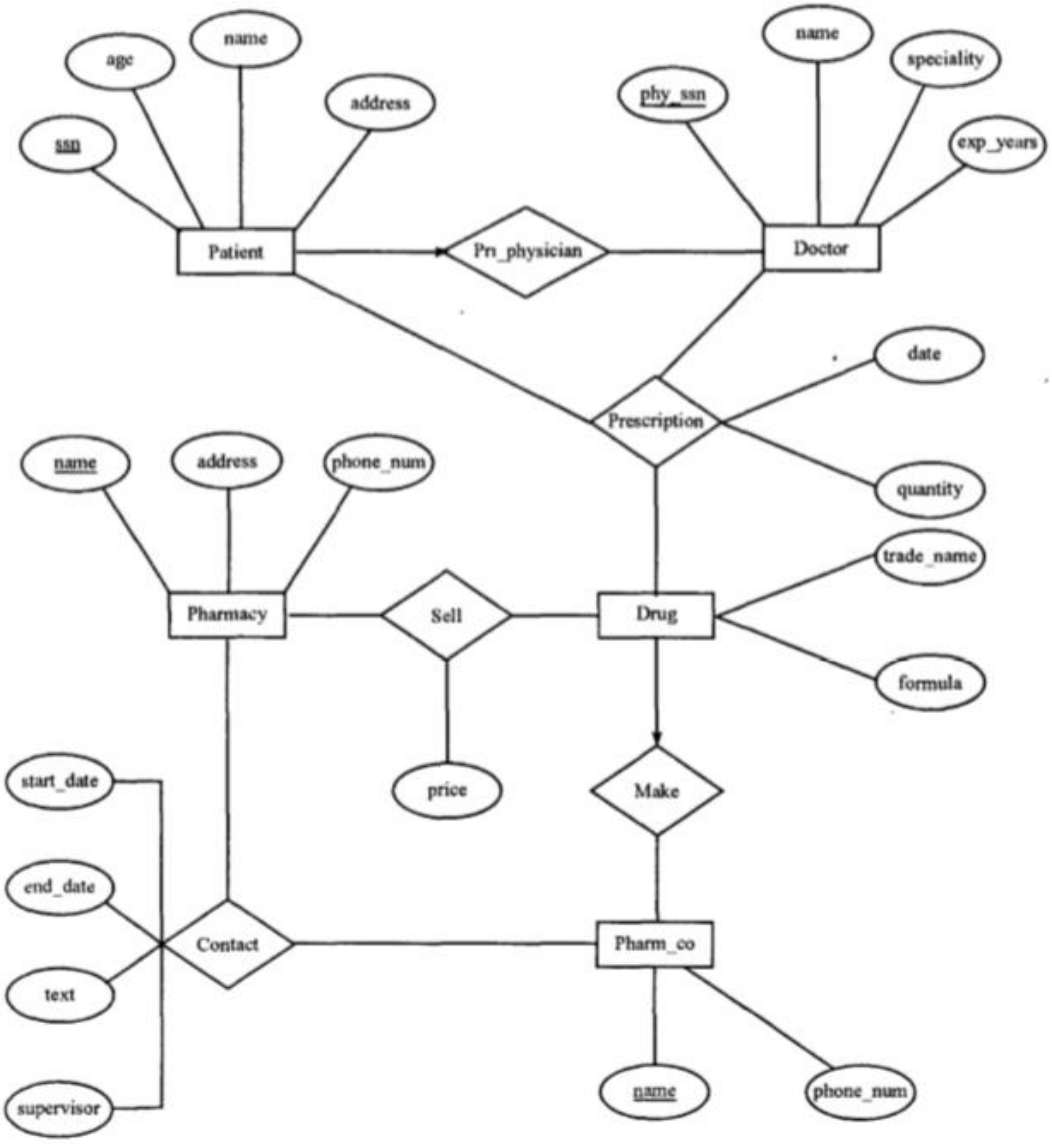
Pertanyaan 26 :

- Rantai apotek resep-R-X telah menawarkan untuk memberi Anda persediaan obat seumur hidup gratis jika Anda merancang Databasenya. Mengingat meningkatnya biaya perawatan kesehatan, Anda setuju. Berikut informasi yang Anda kumpulkan:
- Pasien diidentifikasi oleh SSN, dan nama, alamat, dan usia mereka harus dicatat.
- Dokter diidentifikasi oleh SSN. Untuk setiap dokter, nama, spesialisasi, dan pengalaman bertahun-tahun harus dicatat.
- Setiap perusahaan farmasi diidentifikasi dengan nama dan memiliki nomor telepon.
- Untuk setiap obat, nama dagang dan formula harus dicatat. Setiap obat dijual oleh perusahaan farmasi tertentu, dan nama dagang mengidentifikasi obat secara unik

dari antara produk perusahaan itu. Jika perusahaan farmasi dihapus, Anda tidak perlu lagi melacak produknya.

- Setiap apotek memiliki nama, alamat, dan nomor telepon.
- Setiap pasien memiliki dokter primer. Setiap dokter memiliki setidaknya satu pasien.
- Setiap apotek menjual beberapa obat dan memiliki harga masing-masing. Obat dapat dijual di apotek server, dan harganya dapat bervariasi dari satu apotek ke apotek lainnya.
- Dokter meresepkan obat untuk pasien. Seorang dokter dapat meresepkan satu atau lebih obat untuk beberapa pasien, dan seorang pasien dapat memperoleh resep dari beberapa dokter. Setiap resep memiliki tanggal dan jumlah yang terkait dengannya. Anda dapat berasumsi bahwa, jika seorang dokter meresepkan obat yang sama untuk pasien yang sama lebih dari satu kali, hanya resep terakhir yang perlu disimpan.
- Perusahaan farmasi memiliki kontrak jangka panjang dengan apotek. Sebuah perusahaan farmasi dapat membuat kontrak dengan beberapa apotek, dan sebuah apotek dapat membuat kontrak dengan beberapa perusahaan farmasi. Untuk setiap kontrak, Anda harus menyimpan tanggal mulai, tanggal akhir, dan teks kontrak.
- Apotek menunjuk seorang supervisor untuk setiap kontrak. Harus selalu ada supervisor untuk setiap kontrak, tetapi supervisor kontrak dapat berubah selama masa kontrak.

1. Gambarlah diagram ER yang menangkap informasi sebelumnya. Identifikasi kendala yang tidak ditangkap oleh diagram ER.
2. Bagaimana desain Anda akan berubah jika setiap obat harus dijual dengan harga tetap oleh semua apotek?
3. Bagaimana desain Anda akan berubah jika persyaratan desain berubah sebagai berikut : jika seorang dokter meresepkan obat yang sama untuk pasien yang sama lebih dari satu kali, beberapa resep tersebut mungkin harus disimpan.



Gambar 2.8 *Diagram DR*

Pertanyaan 27: Misalkan kita memiliki hubungan ternary R antara himpunan entitas A, Band C sehingga A memiliki batasan kunci dan partisipasi total dan B memiliki batasan kunci; ini adalah satu-satunya kendala. A memiliki atribut a_1 dan a_2, dengan a1 sebagai kuncinya; Band C serupa. R tidak memiliki atribut skrip d. Tulis pernyataan SQL yang membuat tabel yang sesuai dengan informasi ini sehingga dapat menangkap sebanyak mungkin kendala. Jika Anda tidak dapat menangkap beberapa kendala, jelaskan mengapa ?

Jawaban : Pernyataan SQL berikut membuat relasi yang sesuai

```
CREATE TABLE A          (a1 CHAR (10),
                        a2 CHAR(IO),
                        b1 CHAR(lO),
                        c1 CHAR(lO),
                        PRIMARY KEY (al),
                        UNIOUE (bl),
                        FOREIGN KEY (bl) REFERENCES B,
```

```
                        FOREIGN KEY (cl) REFERENCES C);
CREATE TABLE B          ( bl CHAR(lO),
                        b2 CAHR(lO),
                        PRIMARY KEY (b1));
CREATE TABLE C          ( bl CHAR(lO),
                        c2 CHAR(lO),
                        PRIMARY KEY (cl));
```

Pernyataan SQL pertama melipat hubungan R ke dalam tabel A dan dengan demikian menjamin batasan partisipasi.

***Pertanyaan 28**:* Pertimbangkan database universitas dari Quest 24 dan diagram ER yang Anda rancang. Tulis pernyataan SQL untuk membuat relasi yang sesuai dan tangkap sebanyak mungkin kendala. Jika Anda tidak dapat menangkap beberapa kendala, jelaskan mengapa?

Jawaban: Pernyataan SQL berikut membuat hubungan yang sesuai.

1. CREATE TABLE Professors (prof_ssn CHAR(lO),
   name CHAR(64),
   age NUMBER(5),
   rank NUMBER(5),
   speciality CHAR (64) ,
   PRIMARY KEY (prof_ssn));
2. CREATE TABLE Depts (dno NUMBER(5),
   dname CHAR(64),
   office CHAR(lO),
   PRIMARY KEY (dno));
3. CREATE TABLE Runs (dno NUMBER(5),
   prof_ssn CHAR(lO),
   PRIMARY KEY (dno, prof_ssn),
   FOREIGN KEY (prof_ssn) REFERENCES
   Professors,
   FOREIGN KEY (dno) REFERENCES Depts);
4. CREATE TABLE Work_Dept(dno NUMBER(5),
   prof_ssn CHAR(lO),
   pc_time NUMBER(6),
   PRIMARY KEY (dno, prof_ssn),
   FOREIGN KEY (prof_ssn) REFERENCES
   Professors,
   FOREIGN KEY (dno) REFERENCES Depts);

   Perhatikan bahwa kita perlu memeriksa batasan atau pernyataan dalam SQL untuk menegakkan aturan bahwa Profesor bekerja di setidaknya satu departemen.
5. CREATE TABLE Project (pid NUMBER(5),
   sponsor CHAR (32),
   start_date DATE,

budget NUMBER(8, 4),
PRIMARY KEY (pid))

6. CREATE TABLE Graduates (grad_ssn CHAR(lO),
   age NUMBER(5),
   name CHAR(64),
   deg_prog CHAR(32),
   major NUMBER(lO),
   PRIMARY KEY (grad_ssn),
   FOREIGN KEY (major) REFERENCES Depts);

   Perhatikan bahwa tabel Jurusan tidak diperlukan karena setiap Lulusan hanya memiliki satu jurusan sehingga ini dapat menjadi atribut dalam tabel Lulusan.

7. CREATE TABLE Advisor (senior_ssn CHAR(lO),
   grad_ssn CHAR(lO),
   PRIMARY KEY (senior_ssn, grad_ssn),
   FOREIGN KEY (senior_ssn)
   REFERENCES Graduates (grad_ssn),
   FOREIGN KEY (grad_ssn) REFERENCES Graduates);
8. CREATE TABLE Manages (pid INTEGER,
   Prof_ssn CHAR(lO),
   PRIMARY KEY (pid, prof_ssn),
   FOREIGN KEY (prof_ssn) REFERENCES Professors,
   FOREIGN KEY (pid) REFERENCES Projects);
9. CREATE TABLE Work_In (pid INTEGER,
   Prof_ssn CHAR(lO),
   PRIMARY KEY (pid, prof_ssn),
   FOREIGN KEY (prof_ssn) REFERENCES Professors,
   FOREIGN KEY (pid) REFERENCES Profjects);

   Mengamati bahwa kami tidak dapat menerapkan batasan partisipasi untuk Proyek di tabel Work_In tanpa batasan pemeriksaan atau pernyataan di SQL.

10. CREATE TABLE Supervises (
    Prof_ssn CHAR(lO),
    grad_ssn CHAR(lO),
    pid INTEGER, PRIMARY KEY (protssn, grad_ssn, pid),
    FOREIGN KEY (prof_ssn) REFERENCES Proffessors,
    FOREIGN KEY (grad_ ssn) REFERENCES Graduates,
    FOREIGN KEY (pid) REFERENCES Projects):

Perhatikan bahwa kita tidak memerlukan tabel eksplisit untuk relasi Work_Proj karena setiap kali seorang Lulusan mengerjakan suatu Proyek, dia harus memiliki seorang Supervisor.

Pertanyaan 29: Pertimbangkan database Notown dari pertanyaan 25. Anda telah memutuskan untuk merekomendasikan agar Notown menggunakan sistem database relasional untuk menyimpan data perusahaan. Perlihatkan pernyataan SQL untuk membuat relasi yang sesuai dengan kumpulan entitas dan kumpulan hubungan dalam desain Anda.

Identifikasi kendala apa pun dalam diagram ER yang tidak dapat Anda tangkap dalam pernyataan SQL dan jelaskan secara singkat mengapa Anda tidak dapat mengungkapkannya.
Jawaban : Pernyataan SQL berikut menciptakan hubungan yang sesuai

1. CREATE TABLE Musicians (ssn CHAR(lO),
   name CHAR(30),
   PRIMARY KEY (ssn));
2. CREATE TABLE Instruments (instrId CHAR(lO)
   dname CHAR(30),
   key CHAR(S),
   PRIMARY KEY (instrId));
3. CREATE TABLE Plays (ssn CHAR(lO),
   instrId NUMBER(S),
   PRIMARY KEY (ssn, instrId),
   FOREIGN KEY (ssn) REFERENCES Musicians,
   FOREIGN KEY (instrId) REFERENCES Instruments);
4. CREATE TABLE Songs_Appears (songId NUMBER(8)
   author CHAR(30),
   title CHAR (30) ,
   albumIdentifier NUMBER(lO) NOT NULL,
   PRIMARY KEY (songId),
   FOREIGN KEY (albumIdentifier)
   References Album_Producer);
5. CREATE TABLE Telephone_Home (Phone CHAR(ll),
   address VARCHAR(30),
   PRIMARY KEY (phone),
   FOREIGN KEY (address) REFERENCES Place);
6. CREATE TABLE Lives (ssn CHAR(10),
   phone NUMBER(ll),
   address VARCHAR(30),
   PRIMARY KEY (ssn, address),
   FOREIGN KEY (phone, address)
   References Telephone_Home,
   FOREIGN KEY (ssn) REFERENCES Musicians);
7. CREATE TABLE Place (address VARCHAR(30);
8. CREATE TABLE Perform (songId NUMBER(8),
   ssn CHAR(lO),
   PRIMARY KEY (ssn, songId),
   FOREIGN KEY (songId) REFERENCES Songs,
   FOREIGN KEY (ssn) REFERENCES Musicians);
9. CREATE TABLE Album_producer (albumIdentifier NUMBER(8),
   ssn CHAR(lO),
   copyrightDate DATE,

speed NUMBER(5),
title CHAR(30),
PRIMARY KEY (albumIdentifier),
FOREIGN KEY (ssn) references Musicians);

Pertanyaan 30: Pertimbangkan diagram ER yang Anda rancang untuk rantai apotek Prescriptions-R-X di Quest 26. Tentukan relasi yang sesuai dengan himpunan entitas dan himpunan hubungan dalam desain Anda menggunakan SQL

Jawaban : Pernyataan untuk membuat tabel yang sesuai dengan himpunan entitas Doctor, Pharmacy, dan Pharm_ co disederhanakan dan dihilangkan. Tabel lain yang diperlukan dapat dibuat sebagai berikut:

1. CREATE TABLE Pri_Phy_Patient (ssn CHAR(ll),
   name CHAR(20),
   age NUMBER(5),
   address VARCHAR(20),
   phy_ssn CHAR(ll),
   PRIMARY KEY (ssn),
   FOREIGN KEY (phy_ssn) REFERENCES Doctor);
2. CREATE TABLE prescription (ssn CHAR(ll),
   phy_ssn CHAR(ll),
   date CHAR(ll),
   quantity NUMBER(5),
   trage_ name CHAR (20),
   pharm_ id CHAR(ll),
   PRIMARY KEY (ssn, phy_ssn),
   FOREIGN KEY (phy_ssn) REFERENCES Doctor,
   FOREIGN KEY (trade_name, pharm_id)
   References Make_Drug)
3. CREATE TABLE Make_Drug (trade_name CHAR (20) ,
   pharm_id CHAR(ll),
   PRIMARY KEY (trade_name, pharm_id),
   FOREIGN KEY (trade_name) REFERENCES Drug,
   FOREIGN KEY (pharm_id) REFERENCES Pharm_co);
4. CREATE TABLE Sell (price NUMBER(8),
   name CHAR(lO),
   trade_mime CHAR(lO),
   PRIMARY KEY (name, trade_name),
   FOREIGN KEY (name) REFERENCES Pharmacy,
   FOREIGN KEY (trade_name) REFERENCES Drugs);
5. CREATE TABLE Contract (name CHAR(20),
   pharm_ id CHAR(l1),
   start_date CHAR(ll),
   end_date CHAR(ll),

```
text CHAR(10000),
supervisor CHAR(20),
PRIMARY KEY (name, pharm_ id),
FOREIGN KEY (name) REFERENCES
Drug);
Pharmacy, FOREIGN KEY (pharm Jd) REFERENCES Pharm_co) Q
```

Pertanyaan 31: Jawablah secara singkat pertanyaan-pertanyaan berikut berdasarkan skema ini:

Emp(eid : integer, ename : string, age: integer, salary: real)
works (eid : integer, did: integer, pet_time: integer)
Dept (did: integer, budget: real, managerial: integer)

1. Misalkan Anda memiliki tampilan SeniorEmp yang didefinisikan sebagai berikut:

```
CREATE VIEW SeniorEmp (sname, sage, salary)
AS SELECT E.ename, E.age, E.salary
FROM Emp E
WHERE E.age > 50;
```

   Jelaskan apa yang akan dilakukan sistem untuk memproses kueri berikut:

```
SELECT S.snamc
FROM SeniroEmp S
WHERE S.salary > 100,000;
```

2. Berikan contoh tampilan pada Emp yang dapat diperbarui secara otomatis dengan memperbarui Emp.
3. Berikan contoh tampilan di Emp yang tidak mungkin diperbarui (secara otomatis) dan jelaskan mengapa contoh Anda menyajikan masalah pembaruan yang terjadi.

Jawaban : Jawaban untuk setiap pertanyaan diberikan di bawah ini.

1. Sistem akan melakukan hal berikut:

```
SELECT S.name
FROM (SELECT E.ename AS name, E.age, E.salary
        FROM Emp E
            WHERE E.age > 50) AS S
WHERE S.salary > 100000;
```

2. Tampilan berikut di Emp dapat diperbarui secara otomatis dengan memperbarui Emp:

```
CREATE VIEW SeniorEmp (eid, name, age, salary)
        AS SELECT E.eid, E.ename, E.age, E.salary
            FROM Emp E
            WHERE E.age > 50;
```

3. Tampilan berikut tidak dapat diperbarui secara otomatis karena tidak jelas catatan karyawan mana yang akan terpengaruh oleh pembaruan yang diberikan:

```
CREATE VIEW AvgSalaryBy Age (age, avgSalary)
        AS SELECT E.eid, AVG (E.salary)
            FROM EmpE
            GROUP BY E.age;
```

***Pertanyaan 32*:** Jelaskan pernyataan yang dapat menyusun operator aljabar relasional. U'hy adalah kemampuan untuk menulis operator penting?

Jawaban: Setiap operator dalam aljabar relasional menerima satu atau mOl I 'instance sebagai argumen dan hasilnya selalu merupakan instance relasi. Jadi, argumen dari satu operator bisa menjadi hasil dari operator lain. Ini penting karena, hal ini memudahkan untuk menulis kueri kompleks hanya dengan menyusun operator aljabar relasional.

***Pertanyaan 33*:** Perhatikan skema berikut :

Suppliers (sid: integer, sname : string, address: string)
Parts (pid : integer, pname : string, color: string)
Catalog (sid: integer, pid : integer, cost: real)

Bidang kunci digarisbawahi, dan domain setiap bidang dicantumkan setelah nama bidang. Oleh karena itu sid adalah kunci untuk Pemasok, pid adalah kunci untuk Suku Cadang, dan sid dan pid bersama-sama membentuk kunci untuk Katalog. Relasi Katalog mencantumkan harga yang dibebankan untuk suku cadang oleh pemasok. Tulis pertanyaan berikut dalam aljabar relasional, kalkulus relasional tupel, dan kalkulus relasional domain:

1. Temukan nama pemasok yang memasok beberapa bagian merah.
2. Temukan sisi pemasok yang memasok beberapa bagian merah atau hijau.

Jawaban : Pada jawaban di bawah RA mengacu pada Aljabar Relasional, TRC mengacu pada Kalkulus Relasional Tuple dan DRC mengacu pada Kalkulus Relasional Domain.

1. RA

   πnsname (πsid ((πp1dσcolor = 'red' Parts) $\ltimes$ Catalog) $\ltimes$Suppliers)

   a. TRC,

   {T | ∃T1 ∈ Suppliers (∃ X ∈ Parts (X.color = 'red' ^ ∃ Y ∈ Catalog (Y.pid = X .pid ^ Y.sid = T1.sid)) ^ T.sname = T1sname)}

   b. DRC,

   {<Y>|(X, Y, Z) ∈ Supplier ^ ∃P, Q, R((P, Q, R) ∈ Part ^ R = ‘red’ ^ ∃I, J, K((I, J, K) ∈ Catalog ^ J=P^ I=X)))}

2. RA : πs1d (πpid ((σcolor = 'red' ∨ color=’green’ Parts) $\ltimes$ Catalog)

   a. TRC,

   {T | ∃T1 ∈ Catalog (∃ X ∈ Parts (X.color = 'red' ∨ X.color=’green’ ^ $^{X.p1}$ ∃ Y ∈ Catalog (Y.pid = Id =T1.pid) $^{T.sid=T1sid))}$

   b. DRC,

   {(X) | (X, Y, Z) ∈ Catalog ^ ∃A, B, C ((A, B, C) ∈ Parts ^ (C = 'red' ∨ C = 'green') ^ A = Y)}

   c. SQL

```
SELECT C.sid
FROM Catalog C, Parts P
WHERE (P.color = 'red' OR P.Color = 'green')
AND P.pid = C.pid;
```

Pertanyaan 34: Pertimbangkan skema Supplier-Parts-Catalog dari pertanyaan sebelumnya. Nyatakan apa yang dihitung kueri pada gambar berikut:

1. $\pi_{\text{sname}}(\pi_{\text{sid}}((\sigma_{\text{color}=\text{'red'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog)) \rtimes \text{'}Suppliers)$

2. $\pi_{\text{sname}}(\pi_{\text{sid}}((\sigma_{\text{color}=\text{'red'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \bowtie Suppliers))$

3. $(\pi_{\text{sname}}((\sigma_{\text{color}=\text{'red'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \bowtie Suppliers)) \cap$
$(\pi_{\text{sname}}((\sigma_{\text{color}=\text{'green'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \rtimes Suppliers))$

4. $(\pi_{\text{sid}}((\sigma_{\text{color}=\text{'red'}} Parts) \bowtie (\sigma_{\text{cost}<100} Catalog) \bowtie Suppliers)) \cap$
$(\pi_{\text{sid}}((\sigma_{\text{color}=\text{'green'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \rtimes Suppliers))$

5. $\pi_{\text{sname}}((\pi_{\text{sid,sname}}((\sigma_{\text{color}=\text{'red'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \bowtie Supplirs)) \cap$
$(\pi_{\text{sid,sname}}((\sigma_{\text{color}=\text{'green'}} Parts) \rtimes (\sigma_{\text{cost}<100} Catalog) \rtimes Suppliers)))$

Jawaban : Pernyataan-pernyataan tersebut dapat diartikan sebagai:

1. Temukan nama-nama Pemasok dari pemasok yang memasok bagian merah yang harganya kurang dari 100 ribu rupiah.
2. Pernyataan Aljabar Relasional ini tidak mengembalikan apapun karena urutan operator proyeksi. Setelah, sid diproyeksikan, itu adalah satu-satunya bidang di set. Oleh karena itu, memproyeksikan pada nama tidak akan mengembalikan apa pun.
3. Temukan nama Pemasok pemasok yang memasok bagian merah yang harganya kurang dari 100 ribu rupiah dan bagian hijau yang harganya kurang dari 100 ribu rupiah.
4. Temukan sisi Pemasok dari pemasok yang memasok bagian merah yang harganya kurang dari 100 ribu rupiah dan bagian hijau yang harganya kurang dari 100 ribu rupiah.
5. Temukan nama Pemasok pemasok yang memasok bagian merah yang harganya kurang dari 100 ribu rupiah dan bagian hijau yang harganya kurang dari 100 ribu rupiah.

   Siswa (snum : integer, sname : string, major: string, level: string, age: integer)
   Kelas (name : string, meets_at: string, room: string, fid : integer)
   Enrolled (mum: integer, cname : string)
   Fakultas (fid : integer,fname : string, deptid : integer)

Arti dari hubungan ini adalah langsung; misalnya, Terdaftar memiliki satu catatan per pasangan kelas siswa sehingga siswa terdaftar di kelas. Tulis query berikut dalam SQL. Tidak ada duplikat yang harus dicetak di salah satu jawaban.

1. Temukan nama semua Junior (level = JR) yang terdaftar di kelas yang diajar oleh I. Teach.
2. Temukan usia siswa tertua yang mengambil jurusan Sejarah atau terdaftar dalam kursus yang diajarkan. 1.
3. Temukan nama semua kelas yang bertemu di ruang Rl28 atau memiliki lima atau lebih siswa yang terdaftar.
4. Temukan nama semua siswa yang terdaftar di dua kelas yang bertemu pada waktu yang sama.

5. Temukan nama-nama dosen yang mengajar di setiap ruangan di mana beberapa kelas diajarkan.
6. Temukan nama-nama anggota fakultas yang pendaftaran gabungan mata kuliah yang mereka ajarkan kurang dari lima.
7. Untuk setiap level, cetak level dan rata-rata usia siswa untuk level tersebut.
8. Untuk semua level kecuali JR, cetak level dan rata-rata usia siswa untuk level tersebut.
9. Untuk setiap dosen yang mengajar kelas hanya di ruang R128, tulis nama dosen dan jumlah kelas yang dia ajar. 10. Cari nama siswa yang terdaftar dalam jumlah maksimum kelas.
10. Temukan nama siswa yang tidak terdaftar di kelas mana pun.
11. Untuk setiap nilai usia yang muncul di Siswa, cari nilai level yang paling sering muncul.' Misalnya, jika siswa tingkat FR berusia 18 tahun lebih banyak daripada siswa SR, JR, atau SO berusia 18 tahun, Anda harus mencetak pasangan (18, FR).

1. SELECT DISTINCT S.Sname
   FROM Studnet S, Class C, Enrolled E, Faculty F
   WHERE S.snum = E.snum AND E.cname = C.name AND C.fid = F.fid AND
   F.fname = 'I.Teach' AND S.level = 'JR';
2. SELECT MAX (S.age)
   FROM Student S
   WHERE (S.major = 'History')
   OR S.snum IN (SELECT E.snum
   FROM Class C, Enrolled E, Faculty F
   WHERE E.cname = C.name AND C.fid = F.fid
   AND F.fname = 'I.Teach');
3. SELECT C.name
   FROM Class C WHERE C.room = 'R128'
   OR C.name IN (SELECT E.cname
   FROM Enrolled E
   GROUP BY E.cname
   HAVING COUNT (*) > = 5);
4. SELECT DISTINCT S.sname
   FROM Studnet S
   WHERE S.snum IN (SELECT E1.snum
   FROM Enrolled El, Enrolled E2, Class C1, Class C2
   WHERE E1.snum = E2.snum AND E1.cname < > E2.cname
   AND El.cname = C1.name AND E2.cname = C2.name
   AND C1.meets_at = C2.meets_at);
5. SELECT DISTINCT Efname
   FROM Faculty F
   WHERE NOT EXISTS ((SELECT *
   FROM Class C)
   EXCEPT

```
                (SELECT C1.room
                FROM Class Cl
                WHERE C1.fid = Efid));
6. SELECT DISTINCT Efname
   FROM Faculty F
   WHERE 5 > (SELECT COUNT (E.snum)
                FROM Class C, Enrolled E
                WHERE C.name = E.cname·
                AND C.fid = Efid);
7. SELECT S.level, AVG(S.age)
   FROM Student S
   GROUP BY S.level;
8.  S.level, AVG(S.age)
   FROM Student S
   WHERE S.level < > 'JR'
   GROUP BY S.level;
9. SELECT F.fname, COUNT(*) AS CourseCount
   FROM Faculty F, Class C
   WHERE Efid = C.fid
   GROUP BY Efid, Efname
   HAVING EVERY (C.room = 'Rl28');
10. SELECT DISTINCT S.sname
   FROM Student S
   WHERE S.snum IN (SELECT E.snum
   FROM Enrolled E
   GROUP BY E.snum
                HAVING COUNT (*) > = ALL (SELECT COUNT (*)
                                FROM Enrolled E2
                                GROUP BY E2.snum))
11. SELECT DISTINCT S.sname
   FROM Studnet S
   WHERE S.snum NOT IN (SELECT E.snum FROM Enrolled E)
12. SELECT S.age, S.level
   FROM Student S
   GROUP BY S.age, S.level,
   HAVING S.level IN (SELECT Sl.level
                FROM Student S1
                WHERE S1.age = S.age
                GROUP BY Sl.level, S1.age
                HAVING COUNT (*) > = ALL (SELECT COUNT (*)
                                FROM Student S2
                                WHERE S1.age = S2.age
```

GROUP BY S2.level, S2.age))

***Pertanyaan 35**:* Hubungan berikut melacak informasi penerbangan maskapai :

1. Temukan nama-nama pesawat sedemikian rupa sehingga semua pilot yang disertifikasi untuk mengoperasikannya memiliki gaji lebih dari Rp.80.000.
2. Untuk setiap penerbang yang disertifikasi untuk lebih dari tiga pesawat, tentukan lebaran dan jarak jelajah maksimum pesawat yang disertifikasi.
3. Temukan nama-nama pilot yang gajinya kurang dari harga rute termurah dari Bali ke Lombok.
4. Untuk semua pesawat dengan daya jelajah lebih dari 1000 mil, temukan nama pesawat dan gaji rata-rata semua pilot yang disertifikasi untuk pesawat ini.
5. Temukan nama-nama pilot yang disertifikasi untuk beberapa pesawat Being.
6. Temukan bantuan semua pesawat yang dapat digunakan pada rute dari Los Angles ke Chicago.
7. Identifikasi rute yang dapat diujicobakan oleh setiap pilot yang berpenghasilan lebih dari Rp.100.000.
8. Cetak foto pilot yang dapat mengoperasikan pesawat dengan daya jelajah lebih dari 30.000 meter tetapi tidak bersertifikat pada pesawat Garuda mana pun.
9. Seorang pelanggan ingin melakukan perjalanan dari Siliwangi ke Majapahit dengan tidak lebih dari dua kali pergantian penerbangan. Cantumkan pilihan waktu keberangkatan dari Siliwangi jika pelanggan ingin tiba di Majapahit sebelum jam 6 sore.
10. Hitung selisih antara gaji rata-rata seorang pilot dan gaji rata-rata semua karyawan (termasuk pilot).
11. Cetak nama dan gaji setiap nonpilot yang gajinya lebih dari gaji rata-rata pilot.
12. Cetak nama karyawan yang bersertifikat hanya pada pesawat dengan daya jelajah lebih dari 10.000 km.
13. Cetak nama karyawan yang disertifikasi hanya pada pesawat dengan daya jelajah lebih dari 1000 km, tetapi setidaknya pada dua pesawat tersebut.
14. Cetak nama karyawan yang bersertifikat hanya pada pesawat terbang dengan daya jelajah lebih dari 1000 meter dan yang bersertifikat pada beberapa pesawat Garuda.

Jawaban: Jawabannya ada di bawah ini :

1.
```
SELECT DISTINCT Aaname
FROM Aircraft A
WHERE AAid IN (SELECT C.aid
                FROM Certified C, Employees E
                WHERE C.eid = E.eid AND
                NOT EXISTS (SELECT *
                            FROM Employees E1
                            WHERE El.eid = E.eid AND El.salary < 80000));
```
2.
```
SELECT C.eid, MAX (Acruisingrange)
FROM Certified C, Aircraft A
WHERE C.aid = Aaid G
```

```
ROUP BY C.eid
HAVING COUNT (*) > 3;
```

3.
```
SELECT DISTINCT E.ename
FROM Employees E
WHERE E.salary < (SELECT MIN (Eprice)
FROM Flights F
WHERE F.from = 'Bali' AND Eto = 'Lombok')
```

4. Observe that aid is the key for Aircraft, but the question asks for aircraft names; we deal with this complication by using an intermediate relation Temp :
```
SELECT Temp.name, Temp.AvgSalary
FROM (SELECT A.aid, A.aname AS namt;, AVG (E. salary) AS AvgSalary
FROM Aircraft A, Certified C, Employees E
WHERE A.aid = C.aid AND
        C.eid = E.eid AND Acruisingrange > 1000
GROUP BY~.aid, Aaname) As Temp;
```

5.
```
SELECT DISTINCT E.ename
FROM Employees E, Certified C, Aircraft A
WHERE E.eid = C.eid AND
        C.aid = Aaid AND
        Aaname LIKE 'Garuda%';
```

6.
```
SELECT a.aid
FROM Aircraft A
WHERE Acruisingrange > (SELECT MIN (F.distance)
        FROM Flights F
        WHERE F.from = 'Bali' AND F.to = 'Chicago');
```

7.
```
SELECT DISTINCT F.from, F.to
FROM Flights F
WHERE NOT EXISTS (SELECT*
        FROM Employees E
        WHERE E.salarY > 100000
        AND
        NOT EXISTS (SELECT*
                FROM Aircraft A, Certified C
                WHERE Acruisingrange > F.distance
                AND E.eid = C.eid
                AND Aaid = C.aid))
```

8.
```
SELECT DISTINCT E.ename
FROM Employees E
WHERE E.eid IN «SELECT C.eid
    FROM Certified C
    WHERE EXISTS (SELECT Aaid
            FROM Aircraft A
```

```
                    WHERE Aaid = C.aid
                    AND Acruisingrange > 3000)
        AND
        NOT EXISTS (SELECT A1.aid
                        FROM Aircraft A1
                        WHERE A1.aid = C.aid
                        AND A1.aname LIKE 'Garuda%'))
```

9.
```
SELECT Edeparts
FROM Flights F
WHERE Efino IN «SELECT FO.fino
            FROM Flights FO
            WHERE FO.from = 'Siliwangi'
                AND FO.to = 'Majapahit' AND FO.arrives < '18:00')
            UNION (SELECT FO.fino
            FROM Flights FO, Flights F1, Figiths F2
            WHERE FO.from = 'Siliwangi'
                AND FO.to = Fl.from
                AND F1.to = F2.from
                AND F2.to = 'Majapahit'
                AND FO.to < > 'Majapahit'
                AND F1.to < > 'Majapahit'
                AND F1.departs > FO.arrives
                AND F2.departs > F1.arrives
                AND F2.arrives < '18:00'));
```

10.
```
SELECT Temp1.avg - Temp2.avg
FROM (SELECT AVG (E.salary) AS avg
            FROM Employees E
            WHERE E.eid IN (SELECT DISTINCT C.eid
                        FROM Certified C)) AS Temp1,
            (SELECT AVG (E1.salary) AS avg
            FROM Employees E1) AS Temp2;
```

11.
```
SELECT E.ename, E.salary
FROM Employees E
WHERE E.eid NOT IN (SELECT DISTINCT C.eid
                    FROM Certified C)
        AND E.salary > (SELECT AVG E1.salary)
                        FROM Employees E1
                        WHERE E1.eid IN
                            (SELECT DISTINCT C1.eid
                            FROM Certified C1))
```

12.
```
SELECT E.ename
FROM Employees E, Certified C, Aircraft A
```

WHERE C.aid = A.aid ANI)" E.eid = C.eid
GROUP BY E.eid, E.ename
HAVING EVERY (A.cruisingrange > 1000)

13. SELECT E.ename
    FROM Employees E, Certified C, Aircraft A
    WHERE C.aid = A.aid AND E.eid = C.eid
    GROUP BY E.eid, E.ename
    HAVING EVERY (A.crusingrange > 1000) AND COUNT (*) > 1
14. SELECT E. ename
    FROM Employees E, Certified C, Aircraft A
    WHERE C.aid = A.aid AND E.eid = C.eid
    GROUP BY E.eid, E.ename
    HAVING EVERY (A.crusingrange > 1000) AND ANY (A. Name = 'Garuda');

2.17 REVIEW PERTANYAAN

1. Dalam konteks model relasional, diskusikan masing-masing konsep berikut.
    a. Relasi
    b. Atribut
    c. Thple
    d. Kardinalitas
    e. Domain
2. Diskusikan berbagai jenis kunci yang digunakan dalam model relasional.
3. Apakah yang Anda maksud: aljabar relasional Derme semua operator aljabar relasional.
4. Apakah yang Anda maksud: struktur model relasional sistem Database Jelaskan pentingnya domain dan kunci dalam model relasional.
5. Apa itu aljabar relasional? Apa kegunaannya? Daftar operator relasional.
6. Apakah yang Anda maksud: kalkulus relasional Apa saja jenis-jenis kalkulus relasional?
7. Apa perbedaan antara JOIN & OUTER JOIN dperator?
8. Jelaskan operasi SELECT. Apa yang dicapainya?
9. Jelaskan operasi PROJECT. Apa yang dicapainya?
10. Jelaskan operasi JOIN. Apa yang dicapainya.
11. Apa itu bahasa kueri? Apa kelebihannya?
12. Apa itu bahasa kueri terstruktur? Apa kelebihan dan kekurangannya?
13. Jelaskan sintaks SQL untuk mengeksekusi query.
14. Apa itu operator SQL? Jelaskan itu.
15. Tulislah catatan singkat berikut ini :
    a. Data Manipulation Language (DML).
    b. Bahasa Definisi Data (DDL).
    c. Pernyataan Pengendalian Transaksi (TCS).
    d. Bahasa Kontrol Data (DCL).
16. Bagaimana kita membuat label? Tampilan dan indeks menggunakan perintah SOL?

17. Jelaskan secara singkat hal-hal berikut:
    a. Kursor Database
    b. Pemicu Database
    c. Paket Database
18. Menjelaskan urutan.
19. Apa ekspresi aman dalam kalkulus relasional tupel? Jelaskan dengan contoh.
20. Jelaskan pemicu dan penegasan dengan contoh yang sesuai. Tulis pernyataan untuk pernyataan berikut: "Setiap pinjaman memiliki setidaknya satu pelanggan yang memiliki rekening dengan saldo minimum Rp. 1000 di sistem perbankan?"
21. Apa itu pemicu? Jelaskan keuntungan dari pemicu dengan contoh yang sesuai.
22. Tulis catatan singkat tentang subquery, fungsi agregat, gabungan, indeks, urutan.

BAB 3
DESAIN DAN NORMALISASI DATABASE

3.1 DESAIN DATABASE

Desain keseluruhan database disebut skema database. Sistem database memiliki beberapa skema, dipartisi sesuai dengan tingkat abstraksi.

- Skema fisik
- Skema logis
- Lihat skema

Skema relasional menghadapi beberapa masalah yang tidak diinginkan.

1. **Redundansi**: Tujuan dari sistem Database adalah untuk mengurangi redundansi, artinya data hanya disimpan sekali. Menyimpan data/informasi berkali-kali menyebabkan pemborosan ruang penyimpanan dan peningkatan ukuran total data yang disimpan.

Tabel 3.1 Contoh Database

Name	Course	Phone_No	Major	Prof.	Grade
Dani	160	2374539	Comp Sci	Dwi	A
Joni	170	4277390	Physics	Sri	B
Dani	165	2374539	Comp Sci	Andreas	B
Agus	456	3885183	Mathematics	Irdha	A
Lawren	491	8237293	Chemistry	Juli	C
Lawren	356	8237293	Chemistry	David	A
Dani	168	2374539	Comp Sci	Yoyo	In prof.

Pembaruan ke database dengan redundansi seperti itu berpotensi menjadi tidak konsisten. Pada tabel di atas mayor dan no telepon. siswa disimpan berkali-kali dalam database. misalnya, Mayor dan no telepon. Dani disimpan berkali-kali dalam database. Demikianlah contoh redundansi data dalam database.

2. **Perbarui Anomali:** Beberapa salinan dari fakta yang sama dapat menyebabkan anomali atau inkonsistensi pembaruan. Ketika pembaruan dilakukan dan hanya beberapa dari beberapa salinan yang diperbarui. Jadi, perubahan nomor telepon. dari 'Dani' harus dibuat untuk konsistensi, di semua tupel yang berkaitan dengan 'Dani' siswa. Jika satu-tiga tupel dalam tabel tidak berubah untuk mencerminkan telepon baru-tidak. dari 'Vi jay', akan ada inkonsistensi dalam data. misalnya, Dalam tabel yang diberikan, telepon-no. Dani di ketiga baris adalah 2374539 jika kita memperbarui telepon-no. dari Dani dan dua baris. Maka database akan menjadi tidak konsisten.

Nama	Phone_No
Dani	2374539
Dani	3278435
Dani	3278435

3. **Penyisipan Anomali :** Jika ini adalah satu-satunya hubungan dalam database yang menunjukkan hubungan antara anggota fakultas dan mata kuliah yang dia ajarkan, fakta bahwa seorang profesor yang diberikan mengajar di mata kuliah tertentu tidak dapat dimasukkan dalam database kecuali seorang siswa terdaftar dalam kursus.
4. **Anomali Penghapusan :** Jika satu-satunya mahasiswa yang terdaftar dalam mata kuliah tertentu yang menghentikan mata kuliah tersebut, informasi tentang profesor mana yang menawarkan mata kuliah tersebut akan hilang jika ini adalah satu-satunya relasi dalam database yang menunjukkan hubungan antara anggota fakultas dan mata kuliah yang dia pilih. dia mengajari. Contoh

Tabel 3.2 Contoh Tabel

Roll-No.	Nama	Kursus	Biaya
10	Dani	DBA	150.000
11	Lawrence	VB. Net	50.000
12	Agus	VC++	80.000
13	Joni	Java	70.000

Di sini, kita tidak dapat menghapus atribut tertentu Roll-no. = 11, karena setelah ini kami tidak dapat mengakses mata kuliah, nama dan biaya siswa tersebut karena kehilangan seluruh informasi

3.2 DEKOMPOSISI

Dekomposisi skema relasiR = {A_1,A_2, .. A_n} adalah penggantiannya dengan sekumpulan skema relasi {R_1, R_2, ... ,R_m}

Seperti $R_1 \leq R, 1 \leq i \leq m$

Dan $R_1 \cup R_2 \cup R_3 \ldots R_m = R$

Sebuah skema relasi R dapat didekomposisi menjadi kumpulan skema relasi {R_1, R_2, R_3, ... , R_m} untuk menghilangkan beberapa anomali yang terdapat pada relasi asli R. Permasalahan dalam skema relasi siswa dapat diselesaikan, jika kita dekomposisi dengan skema relasi berikut : seperti :

INFO Mahasiswa (Nama, No. Telp. Jurusan)

Transkrip (Nama, Kursus, Nilai)

Guru (Profesor Kursus)

Dekomposisi ini buruk karena alasan berikut:

(i) Anomali redundansi dan pembaruan, karena data untuk atribut no hp dan jurusan diulang.

(ii) Kehilangan informasi, karena kita kehilangan fakta bahwa seorang siswa memiliki nilai tertentu pada mata kuliah tertentu.

3.3 HUBUNGAN UNIVERSAL

Mari kita pertimbangkan masalah merancang database. Desain seperti itu akan diperlukan untuk mewakili sejumlah himpunan entitas yang terbatas. Setiap himpunan

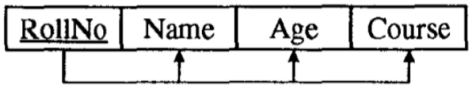
entitas akan diwakili oleh sejumlah atributnya. Jika kita menyebut himpunan semua atribut sebagai skema universal U maka relasi R (U) disebut relasi universal. Relasi universal adalah relasi tunggal yang terdiri dari semua atribut dalam database.

3.4 DEPENDENSI FUNGSIONAL

Sebuah ketergantungan fungsional ada ketika nilai satu hal sepenuhnya ditentukan oleh yang lain. Misalnya " Mengingat relasi EMP (No Emp, Nama Emp, Gaji), atribut EmpName secara fungsional bergantung pada atribut Emp No.
Jika kita tahu Emp No, kita juga tahu Nama Emp. Ini diwakili oleh
No Karyawan → Nama Karyawan
Ketergantungan fungsional adalah batasan antara dua set atribut dalam suatu relasi dari database.

Jenis Ketergantungan Fungsional:

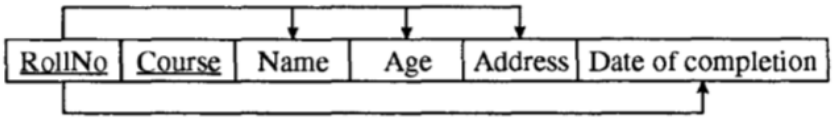
1. **D Trivial :** Ketergantungan fungsional dari bentuk X → Y sepele jika Y ⊆ X.
2. **Ketergantungan Fungsi Penuh:** A FD X→ Y adalah ketergantungan fungsional penuh jika penghapusan atribut A dari X berarti ketergantungan tidak berlaku lagi. Yaitu,



Gambar 3.1 D Trivial

Contoh Ful/ FD :
untuk setiap atribut A ∈ X, (X - {A}) tidak secara fungsional menentukan Y.
(X - {A}) → Y disebut ketergantungan fungsional penuh.

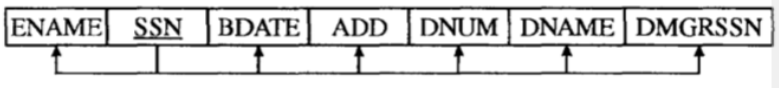
3. **FD Parsial** : A FD X → Y adalah ketergantungan parsial jika beberapa atribut A ∈ X dapat dihilangkan dari X dan ketergantungan tersebut masih berlaku.



Gambar 3.2 FD Parsial

Yaitu jika untuk beberapa A ∈ X, (X - {A}) → Y, maka disebut ketergantungan parsial

4. **Ketergantungan Iransitif:** Ketergantungan fungsional X → Y dalam skema relasi R adalah ketergantungan transitif jika ada himpunan atribut Z yang bukan merupakan kunci kandidat atau subset dari sembarang kunci R dan keduanya X → .Z dan Z → Y tahan.



Gambar 3.3 Ketergantungan Iransitif

Misalnya:
Ketergantungan SSN → DMGRSSN bersifat transitif melalui SSN → DNUM dan DNUM → DMGRSSN

5. **Ketergantungan Mulvalued:** Misalkan R adalah skema relasi dan misalkan $\alpha \subseteq R$ dan $\beta \subseteq R$. Ketergantungan multinilai $\alpha \rightarrow \beta$ bertahan pada R jika ada relasi hukum r (R), untuk semua pasangan tupel t_1 dan t_2 dalam r s.t. $t_1[\alpha] = t_2[\alpha]$, terdapat tupel t_3 dan t_4 di r s.t.
   $t_1[\alpha] = t_2[\alpha] = t_3[\alpha] = t_4[\alpha]$
   $t_3[\beta] = t_1[\beta]$
   $t_3[R-\beta] = t_2[R-\beta]$
   $t_4[\beta] = t_2[\beta]$
   $t_4[R-\beta] = t_1[R-\beta]$
   Contoh: Tabel dengan skema (nama, alamat, mobil)

Tabel 3.3 Tabel Dengan Skema (Nama, Alamat, Mobil)

Name	Address	Car
Agus	Semarang	Toyota
Agus	Jakarta	Honda
Agus	Semarang	Honda
Agus	Jakarta	Toyota

   (Nama, Alamat, mobil) dimana
   name→ address
   name→car

3.5 ATRIBUT UTAMA

Atribut skema relasi R disebut atribut prima dari R jika atribut tersebut merupakan anggota dari beberapa kunci kandidat R.

Untuk Ex. : Work-on

SSN	PNUMBER	HOURS

Gambar 3.4 Atribut

Baik SSN maupun PNUMBER adalah atribut utama dari work-on

Non Prime Attribute

Atribut skema relasi R disebut atribut non prime jika bukan merupakan anggota dari setiap candidate key. misalnya, Jam bukan atribut utama dari pekerjaan.

3.6 AKSIOM ARMSTRONG

Ada aturan berikut yang secara logis menyiratkan ketergantungan fungsional. Misalkan X, Y, Z menunjukkan set atribut di atas skema relasional. Maka aturannya adalah:

1. **Refleksivitas:** Jika X adalah himpunan atribut dan $Y \subseteq X$
   maka
   X→Y memegang
2. **Aturan Augmentasi:** Jika X → Y berlaku dan Z adalah himpunan atribut, maka

ZX → ZY memegang

3. **Aturan Transitivitas :** Jika X Y dan Y Z maka
   X → Z tahan
   Aturan-aturan ini disebut aksioma Armstrong.
   Ada beberapa aturan tambahan yaitu:
4. **Aturan gabungan:** Jika X →Y berlaku dan X→ Z berlaku
   maka
   X → YZ memegang
5. **Aturan dekomposisi:** jika X →YZ berlaku,
   maka X→Y berlaku dan X →Z berlaku
6. **Aturan transitivitas semu:** Jika X Y berlaku dan ZY W berlaku
   maka
   XZ →W memegang

Contoh, Skema A R = (A, B, C, G, H, l) dan Ketergantungan fungsional adalah {A →B,A →C, CG →H, CG →I, B →H} kita dapat membuat daftar beberapa anggota p^+

(i) Sejak A →Band B →H Menerapkan transitivitas
Kita mendapatkan A → H memegang

(ii) ∵CG →I, dan CG →H Menerapkan aturan serikat
Kita mendapatkan CG →HI memegang

(iii) ∵ A →C dan CG →I Menerapkan transitivitas semu
Kita mendapatkan AG →saya pegang

3.7 PENUTUPAN SET DEPENDENSI FUNGSIONAL

Himpunan dependensi fungsional yang secara logis tersirat oleh F disebut penutupan F dan ditulis sebaga F+.

Algoritma : Algoritma untuk menghitung penutupan X.

Biarkan X^+ ke semua atribut di X.

Tentukan X^+, penutupan X di bawah F.

X^+:=X;

ulangi

X^+ lama: =X^+;

untuk setiap ketergantungan fungsional Y →Z di F do

jika Y~X+ maka

X+: =x+ UZ; sampai (x+ = lamaX+);

Contoh, Misalkan R = {A, B, C, D, E, F} dan himpunan FD.

A →BC, E →CF, B →E, CD →EF, F→D

Hitung penutupan himpunan atribut {A, B} di bawah himpunan FD yang diberikan.

Solusi " Misalkan X = {A, B}

Sekarang inisialisasi X^+

∵ X^+:=X

∴ X^+ = {A,B}

Untuk A →BC, .: A ⊆ X^+ maka

$X^+ = \{A,B\} \cup \{B, C\}$

$X^+ = \{A,B,C\}$

Untuk B →E, .: B ⊆ X+ maka

$X^+ = X^+ \cup \{E\}$

$= \{A,B, C\} \cup \{E\}$

$X^+ = \{A,B, C,E\}$

Untuk E →CF, .: E $\subseteq X^+$

∴ $X^+ = \{A,B, C,E\} \cup \{C,F\}$

$= \{A,B, C,E,F\}$

For F →D, .: F $\subseteq X^+$, then

$X^+ = \{A,B, C,E,F\} \cup \{D\}$

$= \{A,B, C,D,E,F\}$

∴ $X^+ = \{A,B, C,E,F\}$

Contoh, Misalkan X = BCD dan F = {A →BC, CD →E, E →C, D →AEH, ABH →BD, DH →BC}

Hitung penutupan X^+ dari X di bawah F.

∵ R = {A, B, C, D, E, H}

Solusi: Kita meninisialkan X^+ ke X.

∵ $X^+ = X$

∵ $X^+ = \{B, C, D\}$

Untuk CD→E ∵D⊆X^+

∵ $X^+ = X^+ \cup \{E\} = \{B, C, D\} \cup \{E\}$: = {B, C, D, E}

Untuk D→AEH, ∵D⊆ X^+

∵ $X^+ = X^+ \cup \{A, E, H\}$

$X^+ = \{A, B, C, D, E, H\}$

3.8 COVER NON REDUNDANT

Algoritma : Input : Satu set FDS F

Output : Cover F . yang tidak berlebihan

G: = F; (Inisialisasi G ke F)
untuk setiap FD X →Y dalam G
melakukan
jika X →Y E $\{F - (X \rightarrow Y)\}^+$
maka F: = {F - (X →Y)};
G: = F; (G adalah penutup F yang tidak berlebihan)
akhir;

Contoh: Jika F = {A →BC, CD →E, E →C, D →AEH, ABH →BD, DH →BC}. Kemudian temukan penutup yang tidak berlebihan untuk F.

Solusi: Kami menemukan bahwa (A)+ di bawah {F - (A →BC)}

Misal X =A·: R = {A,B,C,D,E,H}

∴ X^+:=X

∵ $X^+ = \{A\}$

Untuk semua FDS, (A)+ = {A} ~ {A,B, C,D,E,H}

Jadi A →BC tidak berlebihan.
Sekarang untuk CD →E, kita temukan $(CD)^+$ di bawah
{F - (CD →E)}
Sekarang $X^+ = \{C,D\}$
Untuk D →AEH, ∵D⊆ X^+
∴ $X^+ = X^+ \cup \{A,E,H\}$
$= \{C,D\} \cup \{A,E,H\}$
$X^+ = \{A, C,D,E,H\}$
Untuk A →BC, .: A ⊆ X^+
∴ $X^+ = X^+ \cup \{B, \}$
$= \{A,B, C,D,E,H\} = R = \{A,B, C,D,E,H\}$
Jadi CD →E berlebihan sehingga dihapus. Sekarang untuk DH →BC, kita temukan $(DH)^+$ di bawah {F - (DH →BC)}
∴ $X^+ := \{D, H\}$
D →AEH, ∵D⊆ X^+
∴ $X^+ = X^+ \cup \{A,E,H\}$
$X^+ = \{A, ,D,E,H\}$
A →BC, .: A ⊆ X^+
∴ $X^+ = X^+ \cup \{B, C \}$
$= \{A,B, C, D, E, H\} = R = \{A,B, C,D,E,H\}$
Jadi DH →BC redundant dihilangkan saja. Tidak ada FDS yang tersisa dari F yang dimodifikasi. Dengan demikian, penutup yang tidak berlebihan untuk
F adalah {A →BC, E →C,D →AEH,ABH →BD}

3.9 COVER KANONIK ATAU SET MINIMAL FD'S

Sebuah penutup minimal dari satu set dependensi fungsional E adalah satu set dependensi fungsional F yang memenuhi properti bahwa setiap ketergantungan di E adalah dalam penutupan F^+ dari F. Satu set dependensi fungsional F menjadi minimal jika memenuhi kondisi berikut.

(i) Setiap ketergantungan dalam F memiliki satu atribut untuk ruas kanannya.
(ii) Kami tidak dapat mengganti ketergantungan X → A di F dengan ketergantungan Y → A, di mana Y adalah subset yang tepat dari X dan masih memiliki satu set ketergantungan yang setara dengan F.
(iii) Kami tidak dapat menghapus ketergantungan dari F dan masih memiliki satu set dependensi yang setara dengan F.

Kita dapat menganggap satu set dependensi minimal sebagai satu set dependensi dalam bentuk kanonik dan tanpa redundansi. Sampul kanonik terkadang disebut minimal. *Contoh*: Jika F = {A ∵ BC, CD → E, E → C, D → AEH, ABH → BD, DH → BC} Temukan sampul kanonik. Solusi: Pertama-tama temukan penutup yang tidak berlebihan
∵ Penutup nonredundan untuk F adalah
{A → BC, E → C, D → AEH, ABH → BD}

Karena CD → E dan DH → BC adalah FDS yang berlebihan. Sekarang FD ABH → BD dapat didekomposisi menjadi FDS

ABH →B dan ABH → D

Demikian pula A → Diurai menjadi

A → B dan A → *C*

Karena A → B ada di F, kita dapat mereduksi dekomposisi ABH → *B dan* ABH →D menjadi

AH →*B dan* AH -+ D

Sekarang kita juga perhatikan bahwa AH →B adalah redundan karena A →B sudah di F. Sekarang kita dekomposisi FD D →AEH menjadi FDS

D → A,D → E,D → H

Jadi canonical over adalah

{A → B,A → *C*, E → *C*, D → A, D → E, D → H, AH → D}

3.10 NORMALISASI

Tujuan dari desain database relasional adalah untuk menghasilkan satu set skema relasi yang memungkinkan kita untuk menyimpan informasi tanpa data yang berlebihan (berulang). Hal ini juga memungkinkan kita untuk mengambil informasi dengan mudah dan lebih efisien. Untuk ini kami menggunakan bentuk pendekatan normal sebagai seperangkat aturan. Aturan dan peraturan ini dikenal sebagai Normalisasi. Normalisasi Database adalah desain data dan proses organisasi yang diterapkan pada struktur data berdasarkan dependensi fungsional dan kunci utama yang membantu membangun Database relasional.
Normalisasi Membantu:

- Meminimalkan redundansi data.
- Meminimalkan penyisipan, penghapusan dan pembaruan anomali.
- Mengurangi penundaan input dan output
- Mengurangi penggunaan memori.
- Mendukung satu versi kebenaran yang konsisten.
- Ini adalah metode terbaik industri tabel atau desain entitas.

Kegunaan: Normalisasi database adalah alat yang berguna untuk analisis kebutuhan dan proses pemodelan data pengembangan perangkat lunak. Jadi Normalisasi adalah proses untuk mengurangi semua masalah yang tidak diinginkan dengan menggunakan dependensi fungsional dan kunci. Di sini kita menggunakan bentuk normal berikut:

- First Normal Form (lNF)
- Second Normal Form (2NF)
- Third Normal form (3NF)
- Fourth Normal Form (4NF)
- Fifth Normal Form (5NF)
- Sixth Normal Form (6NF)

First Normal Form (1NF)

Skema relasi R dikatakan dalam Bentuk Normal Pertama (INF) jika nilai-nilai dalam domain setiap atribut relasi adalah atomik. Untuk Contoh. : INFO Kursus

Fact–Dept	Professor	Course Preferences	
		Course	Course–Dept
Comp–Sci	Dani	353	Comp Sci
		370	Comp Sci
		310	Physics
	Lawrence	353	Comp Sci
		320	Comp Sci
		370	Comp Sci
Chemistry	Agus	456	Chemistry
		410	Mathematics
		370	Comp Sci

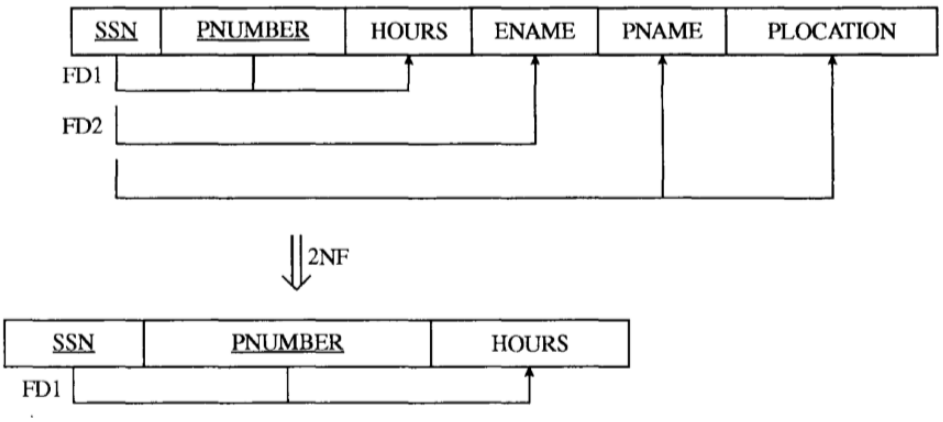
In 1NF : Course INFO

Professor	Course	Fact–Dept	Course
Dani	353	Comp Sci	Comp Sci
Dani	370	Comp Sci	Comp Sci
Dani	310	Comp Sci	Physics
Lawrence	353	Comp Sci	Comp Sci
Lawrence	320	Comp Sci	Comp Sci
Lawrence	370	Comp Sci	Comp Sci
Agus	456	Chemistry	Chemistry
Agus	410	Chemistry	Mathematics
Agus	370	Chemistry	Comp Sci

Gambar 3.5 Info Khursus (Normalisasi Bentuk Pertama)

Second Normal Form (2NF)

- 2NF adalah bentuk normal dalam normalisasi database. Ini mensyaratkan bahwa semua elemen data dalam tabel secara fungsional bergantung pada kunci utama tabel.
- Jika data clement hanya bergantung pada bagian dari kunci utama, maka mereka diurai ke tabel terpisah.
- Jika tabel memiliki satu bidang sebagai kunci utama, maka secara otomatis dalam 2NF.



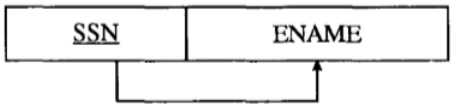
Gambar 3.6 Normalisasi Bentuk Ke 2

Sebuah tabel berada dalam 2NF jika dan hanya jika

(i) Ada di INF

(ii) Setiap atribut non primary key secara fungsional bergantung penuh pada primary key.

Contoh 1: EMP-PROJ

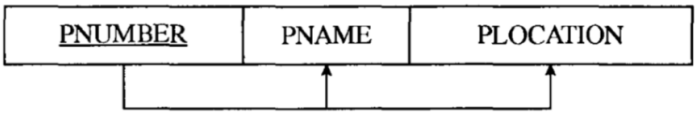


Gambar 3.7 EMP-PROJ

.: {SNN, PNUMBERj adalah kunci utama dan Jam adalah atribut bukan kunci.

(SSN, PNUMBER) → HOURS

SSN → ENAME

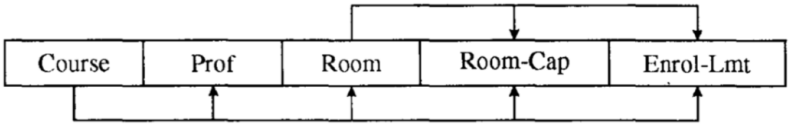


Gambar 3.8 atribut bukan kunci

.: SSN adalah kunci utama dan NAMA adalah atribut bukan kunci

Karena atribut bukan kunci ENAME adalah FDS penuh pada atribut kunci utama SSN. Jadi, itu dalam 2NF.

PNUMBER → {PNAME, PLOCATION}



Gambar 3.9 Atribut Bukan kunci ENAME

Contoh 2:

GURU :

Tabel 3.4 Tabel Guru

Course	Prof	Room	Room–Cap	Enrol–Unt
353	Dani	A532	45	40
351	Dani	C320	100	60
355	Joni	H940	50	45
456	Joni	B278	50	45
459	Agus	D110	300	200

Hubungan GURU dalam Bentuk Normalisasi Kedua

Course	Prof	Enrol–Unt
353	Dani	40
351	Dani	60
355	Joni	45
456	Joni	45
459	Agus	200

(a)

Course	Room
353	A532
351	C320
355	H940
456	B278
459	D110

(b)

Room	Room–Cap
A532	45
C320	100
H940	50
B278	50
D110	300

(c)

Gambar 3.10 Normalisasi Ke 2 Tabel Guru

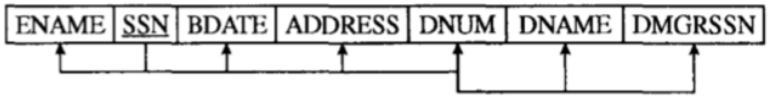
Third Normal Form (3NF)

3NF adalah bentuk normal yang digunakan dalam normalisasi database untuk memeriksa apakah semua atribut bukan kunci dari suatu relasi hanya bergantung pada kunci kandidat dari relasi tersebut. Artinya semua atribut bukan kunci saling bebas atau dengan kata lain atribut bukan kunci tidak dapat bergantung secara transitif pada atribut bukan kunci lainnya.

Skema relasi R berada dalam 3NF jika setiap atribut non prima dari R memenuhi kedua hal berikut.

- Bergantung secara fungsional penuh pada setiap kunci R.
- Tidak bergantung secara transitif pada setiap kunci R.

ATAU kita dapat mengatakan: Skema relasi R berada dalam 3NF jika, setiap kali ketergantungan fungsional yang tidak sepele



Gambar 3.11 Skema relasi R berada dalam 3NF

x →A tertahan di R.

Lainnya

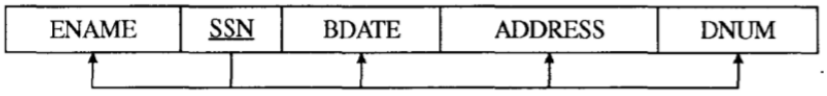
(a) X adalah kunci super R. OR

(b) A adalah atribut prima dari R.

Contoh 1: EMP-DEP

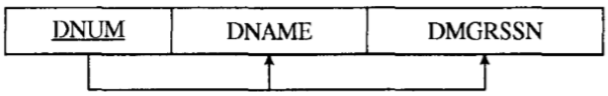
Ketergantungan SSN →DMGRSSN bersifat transitif melalui FDS

SSN →DNUM dan DNUM →DMGRSSN



Gambar 3.12 *EMP-DEP*

Jadi EMP-DEP tidak dalam 3NF karena ketergantungan transitif DMGRSSN pada SSN Via DNUM.



Gambar 3.15 EMP-DEP tidak dalam 3NF

Kita dapat menormalkan EMP-DEP dengan mendekomposisi menjadi dua kata 3NF ED_I dan ED_2 masing-masing. Jadi dalam 3NF :

(i) ED!

(ii) ED2

Contoh : Hubungan siswa

Tabel 3.5 Contoh Tabel Hubungan Siswa

Roll_No	Name	Dept	Year	Hostel_Name
1784	Rohman	Physics	1	Horizon
1648	Krisna	Chemistry	1	Louis
1768	Agus	Maths	2	Novotel
1848	Maya	Botany	2	Louis
1682	Sam	Geology	3	Novotel
1485	Lia	Zoology	4	Horizon

Di sini ketergantungan Roll-No →HOSTAL NAME transitif melalui

ROLL-NO →TAHUN DAN TAHUN →NAMA HOSTAL

Jadi Relasi siswa bukan 3NF. Jadi kita dapat menormalkan Relasi siswa dengan dekomposisi menjadi dua masing-masing 3NF, STUD1 DAN STUD2.

(i) hubungan STUDI

Tabel 3.6 Tabel Studi

Roll-No	Name	Dept	Year
1784	Rajesh	Physics	1
1648	Azeda	Chemistry	1
1768	Kenny	Maths	2
1848	Ibnu	Botany	2
1682	Lisa	Geology	3
1485	Novita	Zoology	4

(ii) hubungan STUD2

Tabel 3.7 Tabel Stud2

Year	Hostel_Name
1	Louis
2	Novotel
3	Star Hotel
4	Ibis Budget

Boyce-Codd Normal Form (BCNF)

BCNF adalah bentuk normal yang digunakan dalam normalisasi database. Ini adalah versi 3NF yang sedikit lebih kuat. Sebuah tabel berada dalam BCNF jika dan hanya jika :

(a) Berada dalam 3NF dan

(b) Untuk setiap dependensi fungsional nontrivialnya X →Y, X adalah super key.

ATAU Suatu skema relasi R berada dalam BCNF jika setiap kali ketergantungan fungsional nontrivial X →A berlaku di R maka

(1) X adalah kunci super dari R.

Tabel 3.8 Contoh Tabel Mahasiswa

Stud_Id	SName	Subject	Grade
10	Dani	Computer	A
10	Dani	Physics	B
10	Dani	Maths	B
20	Agus	Computer	A
20	Agus	Physics	A
20	Agus	Maths	C

Ada dua kunci kandidat (Stud_Id, Subject) dan (SName, Subject)

Pada relasi di atas terdapat FDS berikut :

SName, Subyek .... Grade

Stud _ Id, Subject .... Grade

Stud Id .... SName

Sekarang BCNF menguraikan R menjadi R_I dan R_2.

Tabel 3.9 BCNF menguraikan R menjadi R_I dan R_2

R_1

Stud_Id	Subject	Grade
10	Computer	A
10	Physics	B
10	Maths	B
20	Computer	A
20	Physics	A
20	Maths	C

R_2

Stud_Id	SName
10	Dani
20	Agus

Tabel 3.10 Normalisasikan relasi professor seperti pada BCNF.

Prof Code	Department	HOD	Percent Time
P_1	Physics	Krisna	50
P_1	Maths	Laura	50
P_2	Chemistry	Lidya	25
P_2	Physics	Mino	75
P_3	Maths	Joko	100
P_4	Maths	Dimas	30
P_4	Physics	Santoso	70

Kode Prof, Departemen →Persen waktu

Departemen → HOD

Relasi PROFESSOR masing-masing didekomposisi menjadi dua relasi PROF 1 dan PROF 2

Tabel 3.11 Tabel PROF1

Professor Code	Department	Percent Time
P_1	Physics	50
P_1	Maths	50
P_2	Chemistry	25
P_2	Physics	75
P_3	Maths	100
P_4	Maths	30
P_4	Physics	70

Tabel 3.12 PROF2

Department	HOD
Physics	Dina
Maths	Lutfi
Chemistry	Naufal

Catatan: Setiap relasi dalam BCNF juga berada dalam 3NF, tetapi sebuah relasi adalah 3NF belum tentu berada dalam BCNF misalnya, relasi di atas juga dalam BCNF dan 3NF.

Tabel 3.13 TEACH

Student	Course	Instructor
Febi	Database	Rusito
Dion	Database	Widya
Lusi	OS	Edy
Ina	Computer	Budi
Ryan	OS	Andre
Ratih	Database	Novi

Ketergantungan ditampilkan sebagai

{STUDENT, COURSE} →INSTRUCTOR

INSTRUCTOR → COURSE Tapi TEACH Relation ada di 3NF

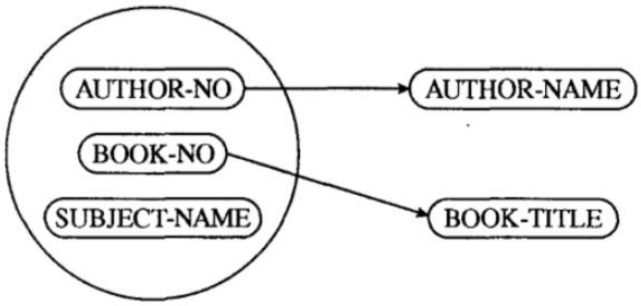
tapi tidak di BCNE

Fourth Normal Form (4NF)

- Tipe entitas berada dalam 4~F jika BCNF dan ada dependensi non multinilai antara tipe atributnya.
- Setiap entitas adalah BCNF yang ditransformasikan dalam 4NF

(i) Mengarahkan semua dependensi multinilai.

(ii) Mengurai tipe entitas.



Gambar 3.16 Mengurai tipe entitas.

Tabel 3.14 Contoh Tabel dari Diagram diatas

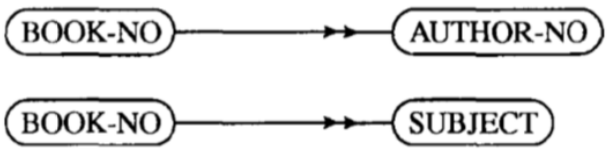
Author_No	Book_No	Subject	Book_Title	Author_Name
A_1	B_1	Comp Sci	Methods	Dani
A_1	B_1	Maths	Methods	Dani
A_2	B_1	Comp Sci	Methods	Agus
A_2	B_1	Maths	Methods	Agus
A_1	B_2	Maths	Calculus	Joni

AUTHOR (Author-No, Author-Name)

BOOK (Book-No, Book-title)

AUTHOR-BOOK-SUBJECT (Author-No, Book-No, SUbject)

- Contoh model bahwa "setiap AUTHOR dikaitkan dengan semua SUBJECT di mana Buku diklasifikasikan.
- Atribut SUBJECT berisi nilai-nilai yang berlebihan. Jika SUBJECT dihapus dari baris 1 dan 2, nilainya dapat disimpulkan dari baris 3 dan 4.



Gambar 3.17 AUTHOR-BOOK-SUBJECT

Ketergantungan multinilai:

Author_No	Book_No	Subject
A_1	B_1	Comp Sci
A_1	B_1	Maths
A_2	B_1	Comp Sci
A_2	B_1	Maths
A_1	B_2	Maths

Sekarang NF ke-4 akan menjadi

Author_No	Book_No
A_1	B_1
A_2	B_1
A_1	B_2

Book_No	Subject
B_1	Comp Sci
B_1	Maths
B_2	Maths

Gambar 3.18 Normalisasi ke 4 dari tabel diatas

Contoh 2 : Hubungan Fakultas (Formulir BCNF)
Hubungan di 4NF

Faculty	Subject	Institute
Dani	DBMS	IILM
Dani	Data Structure	IILM
Dani	C++	IILM
Dani	DBMS	GIMT
Dani	Data Structure	GIMT
Agus	Data Structure	GIMT
Agus	Java	IILM

4NF

Faculty Subject Relation

Faculty	Subject
Dani	DBMS
Dani	C++
Dani	Data Structure
Agus	Data Structure
Agus	Java

Faculty Institute Relation

Faculty	Institute
Dani	IILM
Dani	GIMT
Agus	GIMT
Agus	IILM

Relation in 4NF

Gambar 3.19 Hubungan di 4NF

Catatan: 4NF menghilangkan hubungan MVD. Jadi tidak ada tabel yang dapat memiliki lebih dari satu hubungan banyak-ke-satu atau banyak-ke-banyak yang tidak berhubungan langsung.

Fifth Normal Form (5NF)

Sebuah tabel dikatakan berada dalam 5NF jika dan hanya jika berada dalam 4NF dan setiap ketergantungan Join di dalamnya diimplikasikan oleh kunci kandidat. Perhatikan contoh berikut :

Tabel 3.15 Psikiater-ke-Insurer-to-Condition

Psychiatrist	Insurer	Condition
Dr. Dani	Healthco	Anxiety
Dr. Dani	Healthco	Depression
Dr. Lawren	Friendly Care	Dementia
Dr. Lawren	Friendly Care	Anxiety
Dr. Lawren	Friendly Care	Depression
Dr. Lawren	Friendly Care	Mood Disorder
Dr. Agus	Friendly Care	Schizopherenia
Dr. Agus	Healthco	Anxiety
Dr. Agus	Healthco	Dementia
Dr. Agus	Victorian Life	Conversion Disorder

Untuk membagi relasi menjadi tiga bagian

Tabel 3.16 Psikiater-untuk-Kondisi

Psychiatrist	Condition
Dr. Dani	Anxiety
Dr. Dani	Depression
Dr. Lawren	Dementia
Dr. Lawren	Anxiety
Dr. Lawren	Depression
Dr. Lawren	Mood Disorder
Dr. Agus	Schizopherenia
Dr. Agus	Anxiety
Dr. Agus	Dementia
Dr. Agus	Conversion Disorder

Tabel 3.17 Psikiater-ke-Penanggung

Psychiatrist	Insurer
Dr. Dani	Healthco
Dr. Lawren	Friendly Care
Dr. Agus	Friendly Care
Dr. Agus	Healthco
Dr. Agus	Victorian Life

Tabel 3.18 Penanggung-ke-Kondisi

Insurer	Condition
Prudential Life Ins.	Anxiety
Prudential Life Ins.	Depression
Prudential Life Ins.	Dementia
Allianz Ins.	Dementia
Allianz Ins.	Anxiety
Allianz Ins.	Depression
Allianz Ins.	Mood Disorder
Allianz Ins.	Schizopherenia
Lippo Ins.	Conversion Disorder

Bentuk Normal Keenam

'Bentuk normal ini, pada 2005 baru saja diusulkan. Sayangnya, 6NF hanya didefinisikan ketika memperluas modal relasional untuk memperhitungkan dimensi temporal, sayangnya, sebagian besar teknologi SQL saat ini pada 2005 tidak memperhitungkan pekerjaan ini, dan sebagian besar ekstensi temporal ke SQL tidak relasional.

Bentuk Normal Domain/Kunci

Bentuk Normal Domain/Kunci adalah bentuk normal yang digunakan dalam normalisasi Database yang mensyaratkan bahwa Database tidak mengandung batasan selain batasan domain dan batasan kunci. Batasan domain menentukan nilai yang diizinkan untuk atribut tertentu, sedangkan batasan kunci menentukan atribut yang secara unik mengidentifikasi baris dalam tabel tertentu. NF domain/Key adalah Cawan Suci dari desain Database relasional, dicapai ketika setiap batasan pada relasi adalah konsekuensi logis dari definisi kunci dan domain, dan menerapkan batasan dan kondisi kunci dan domain menyebabkan semua batasan harus dipenuhi. Oleh karena itu menghindari semua anomali non-temporal Ini harus lebih mudah untuk membangun database dalam bentuk normal domain/kunci daripada mengkonversi database yang lebih kecil yang mungkin berisi banyak anomali. Namun, berhasil membangun Database bentuk normal domainlkey tetap menjadi tugas yang sulit, bahkan untuk pemrogram Database yang berpengalaman. Jadi, sementara bentuk Normal domain/kunci menghilangkan masalah yang ditemukan di sebagian besar Database, bentuk normal cenderung menjadi yang paling mahal untuk dicapai.

Kesimpulan Normalisasi Database

Normalisasi Data adalah teknik yang memastikan beberapa properti dasar:

- Tidak ada tupel duplikat.
- Tidak Ada Hubungan Bersarang.

Normalisasi Data sering digunakan sebagai satu-satunya teknik untuk tampilan implementasi desain database. Pendekatan yang lebih tepat adalah melengkapi pemodelan konseptual dengan Normalisasi data.

3.11 DEKOMPOSISI LOSSLES-JOIN

- Biarkan R menjadi skema relasi.

- Misalkan F adalah himpunan ketergantungan fungsional dari R.
- Biarkan RI dan R2 dari dekomposisi R.

Dekomposisinya adalah dekomposisi Gabungan-kehilangan dari R jika setidaknya salah satu dari FDS berikut berada dalam F^+.

(1) $R_I \cap R_2 \rightarrow R_1$

(2) $R_I \cap \rightarrow R_2 \rightarrow R_2$

Mengapa ini benar? Sederhananya, ini memastikan bahwa atribut yang terlibat dalam natural join ($R_I \cap R_2$) adalah kunci kandidat untuk setidaknya satu dari dua relasi. Ini memastikan bahwa kita tidak akan pernah mendapatkan situasi di mana tupel palsu dihasilkan, karena untuk nilai apa pun pada atribut gabungan akan ada tupel unik di salah satu relasi.

Misalnya :

$R = \{A,B, C,D\}$

$F = \{A \rightarrow B, B \rightarrow C\}$

Misalkan dekomposisi $R_1 = \{A,B\}$, $R_2 = \{B, C\}$ dan $R_3 = \{A,D\}$. Cari dekomposisinya lossless atau lossy.

Jawaban (1) Pertimbangkan R_1 dan R_3

$R_1 \cap R_3 = \{A,B\} \cap \{A,D\} = \{A\}$

Karena $A \rightarrow B$ dan A adalah kunci di RI.

$R_1 \cap R_3 \rightarrow R_1 = \{A,B\}$

Mari kita gabungkan RI dan R_3 dan bentuk R_4.

$R_4 = \{A,B\} \cup \{A,D\} = \{A,B,D\}$

Penguraian (A, B, D) menjadi R_I dan R_3 adalah lossless-Join.

(2) Selanjutnya pertimbangkan R_4 dan R_2

Sekarang $R_4 \cap R_2 = \{A, B, D\} \cap \{B, C\} = \{B\}$

Karena $B \rightarrow C$ dan B adalah kunci di R_2

$\therefore$ $R_4 \cap R_2 \rightarrow R_{2 =\{B,} C\}$

$\because$ $B \rightarrow C$

Penguraian (A, B, C, D) menjadi R_2 dan R_4 adalah lossless-Join.

3.13 PENYELESAIAN MASALAH

Pertanyaan 1: Pertimbangkan skema R = (V, W, X, Y, Z) misalkan FD berikut ini berlaku:

F = {Z →V, W→ Y,XY → Z, V → WX}

Nyatakan apakah dekomposisi skema R berikut ini merupakan dekomposisi join loss-less. Justifikasi jawaban Anda.

Jawaban : Jawab. Untuk dekomposisi skema relasi R menjadi Rl dan Rz menjadi lossy atau lossless, salah satu dari kondisi berikut berlaku.

(1) $R_1 \cap R_2 \rightarrow R_1$

(2) $R_1 \cap R_2 \rightarrow R_2$

(i) Mempertimbangkan dekomposisi pertama

R_I = (V, W,X)

R_2 = (V, Y, Z)

$R_1 \cap R_2$ = {V}

Karena V → WX dan V adalah kunci dalam R_1

∴ $R_1 \cap R_2 \rightarrow R_1$ karena V →VWX= R_1

Dengan demikian kita dapat mengatakan dekomposisi

R_1 = (V, W,X)

R_2 = (V, Y, Z) adalah dekomposisi lossless.

(ii) Sekarang pertimbangkan dekomposisi kedua

R_I = (V, W,X)

R_2 = (X, Y,Z)

$R_1 \cap R_2$ = {X}

Jadi juga

X →VWX [∵$R_1 \cap R_2 \rightarrow R_{1]}$

Atau

X → XYZ [$R_1 \cap R_2 \rightarrow R_2$]

Tapi menggunakan set FDS yang diberikan, kita tidak bisa.

Lainnya X →VWX

Atau X → XYZ

Jadi dekomposisi

R_I = (V, W,X)

R = (X, Y, Z) dekomposisi loss

Pertanyaan 2 : Pertimbangkan skema R = (A, B, C, D, E) misalkan mengikuti FDS terus,'

E→*A*

CD→E

A→BC

B→D

Nyatakan, apakah dekomposisi Rare lossless join decomposition berikut atau tidak, justify

1. {(A, B, C), (A, D, E)}
2. {(A,B, C), (C,D,E)}

Jawaban :

1. {(A,B, C), (A,D,E)}

   Letakkan α dalam tabel di mana atribut ada dalam relasi dan letakkan β di mana mereka tidak ada dalam relasi. Sekarang kita mendapatkan tabel berikut:

Tabel 3.19 Tabel Atribut R1 dan R2

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_1	β_1
R_2	α_A	β_2	β_2	α_D	α_E

Letakkan α ada dalam relasi dan letakkan β tidak ada dalam relasi. Setelah melihat tabel tersebut kita mengetahui bahwa kolom A memiliki kesamaan a dan A → BC Jadi kita dapat menempatkan α pada Baris R_2 kolom Band C. Setelah itu kita mendapatkan tabel berikut.

Tabel 3.20 Tabel Hasil

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_1	β_1
R_2	α_A	α_B	α_C	α_D	α_E

Karena Baris R_2 memiliki semua α. Jadi dekomposisinya lossless.

2. {(A,B, C), (C,D,E)}

R_1 = (A,B, C)

R_2 = (C,D,E)

- Masukkan α di tabel di mana atribut ada dalam relasi.
- Letakkan β di mana mereka tidak ada dalam hubungannya.

Sekarang kita mendapatkan tabel berikut:

Tabel 3.21 Tabel FD

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_1	β_1
R_2	β_2	β_2	α_C	α_D	α_E

Setelah melihat tabel, kami mengetahui bahwa kolom C memiliki kesamaan α. Dalam FD yang diberikan, C tidak menyiratkan nilai apa pun, jadi kami tidak dapat memperbarui tabel. Jadi dekomposisi

R_1 = (A,B, C)

R_2 = (C,D,E) adalah loss.

***Pertanyaan 3*:** Diketahui R = (A, B, C, D, E) dengan FD.

F = {AB → CD,A → E, C → D}.

Temukan, jika dekomposisi R menjadi

R_I = (A, B, C),

R_2 = (B, C, D)

dan R_3 = (C, D, E) lossy atau tidak.

Jawaban: Diketahui R = (A, B, C, D, E)

Dekomposisi R menjadi tiga relasi

R_1 = (A,B,C)

R_2 = (B, C, D)

dan R_3 = (C,D,E) F = {AB → CD,A → E, C → D}

- Letakkan α dalam tabel di mana atribut ada dalam relasi dan
- Letakkan β di mana atribut tersebut tidak ada dalam relasi.

Jadi kita mendapatkan tabel.

Tabel 3.22 Tabel Dekomposisi Rumus Diatas

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_1	β_2
R_2	β_2	α_B	α_C	α_D	β_2
R_3	β_3	β_3	α_C	α_D	α_E

Setelah melihat tabel tersebut kita mengetahui bahwa, kolom C memiliki kesamaan α dan FD C → D. Jadi kita dapat menempatkan α pada Baris R_1 kolom D. Setelah itu kita mendapatkan tabel baru.

Tabel 3.23 Tabel Hasil Dekomposisi Rumus Diatas

	A	B	C	D	E
R_1	α_A	α_B	α_C	α_D	β_1
R_2	β_2	α_B	α_C	α_D	β_2
R_3	β_3	β_3	α_C	α_D	α_E

Karena setiap baris dalam tabel ini tidak berisi semua α. Dengan demikian dekomposisi ini bersifat lossy.

Pertanyaan 4: Diketahui R (A, B, C, D) dengan FDs

F = {A → B,A → C, C → D}.

Cari apakah penguraian R menjadi R_I (A, B, C) dan R_2 (C, D) lossy atau tidak.

R_I = (A, B, C)

R_2 = (C,D)

- Masukkan α tabel di mana atribut ada dalam Relasi dan
- Letakkan β di mana atribut tersebut tidak ada dalam Relasi.

Sekarang kita mendapatkan tabel berikut.

Tabel 3.24 Tabel Hasil Dekomposisi Rumus Diatas

	A	B	C	D
R_1	α_A	α_B	α_C	β_1
R_2	β_2	β_2	α_C	α_D

Setelah melihat tabel, kami mengetahui bahwa kolom C memiliki kesamaan α dan FD C → D. Jadi kita bisa meletakkan α di Baris R_I kolom D. Jadi setelah itu kita dapatkan tabelnya.

Tabel 3.25 kesamaan α dan FD C → D.

	A	B	C	D
R_1	α_A	α_B	α_C	α_D
R_2	β_2	β_2	α_C	α_D

Karena Baris R_1 memiliki semua α. Jadi dekomposisi R_1 (A, B, C) dan R_2 (C, D) adalah lossless.

Pertanyaan 5 : Diketahui *R (A, B, C, D, E)* dengan *FD*

F = {AB → CD,A → E, C→ D},

Dekomposisi pada R ke R_1 (A,B, C), R_2 (B, C,D) dan R_3 (C,D,E) adalah lossless atau lossy.

Jawaban : *R (A, B, C, D, E)*

Dekomposisi R menjadi tiga relasi

R1 = (A, B, C)

Rz = (B, C, D)

R3 = (C, D, E)

FD F = {AB → CD, → E, C → D}

- Letakkan α pada tabel dimana atribut ada dalam relasi dan
- Letakkan β dimana atribut tersebut tidak ada dalam relasi.

Kami mendapatkan tabel berikut.

Tabel 3.25 Tabel Hasil F = {AB → CD, → E, C → D}

	A	B	C	D	E
R_1	α_A	α_B	α_C	β_1	β_1
R_2	β_2	α_B	α_C	α_D	β_2
R_3	β_3	β_3	α_C	α_D	α_E

Setelah melihat tabel tersebut kita mengetahui bahwa kolom C mempunyai persamaan α dan FD C→ D. Sehingga kita dapat menempatkan α pada baris R1 kolom D. Setelah itu kita mendapatkan tabelnya.

Tabel 3.25 Tabel Hasil Akhir

	A	B	C	D	E
R_1	α_A	α_B	α_C	α_D	β_1
R_2	β_2	α_B	α_C	α_D	β_2
R_3	β_3	β_3	α_C	α_D	α_E

Tidak ada perubahan lebih lanjut yang dimungkinkan dan versi final tabel sama dengan tabel di atas. Akhirnya kami tidak menemukan baris dalam tabel dengan semua sebagai. Oleh karena itu dekomposisinya lossy.

Pertanyaan 6 : Pertimbangkan skema relasional

R (A, B, C, D, E, F, G, H) dengan FD

F = {AB → C,BC→ D,E → F, G → F,H→ A,FG → H}

Apakah penguraian R menjadi R_l (A, B, C, D), R_2 (A, B, C, E, F) dan R_3 (A, D, F, G, H) lossless?

Jawaban : Diketahui R (A, B, C, D, E, F, G, H)

Dekomposisi R ke R_1, R_2, R_3, S.T.

R_2 = (A, B, C, E, F),

R_1= (A, B, C, D)

R_3 = (A,D,F, G,H)

FD F = {AB → C,BC→ D,E → F, G → F,H→ A,FG → H}

- Masukkan α tabel di mana atribut ada dalam Relasi, dan
- Letakkan β di mana atribut tersebut tidak ada dalam relasi.

Kami mendapatkan tabel berikut.

Tabel 3.26 Perhitungan {AB → C,BC→ D,E → F, G → F,H→ A,FG → H}

	A	B	C	D	E	F	G	H
R_1	α_A	α_B	α_C	α_D	β_1	β_1	β_1	β_1
R_2	α_A	α_B	α_C	β_2	α_E	α_F	β_2	β_2
R_3	α_A	β_3	β_3	α_D	β_3	α_F	α_G	α_H

Setelah melihat tabel, kami mengetahui bahwa kolomA memiliki kesamaan α. Tetapi dalam FD A yang diberikan tidak menyiratkan nilai apa pun.

Jadi kami tidak dapat memperbarui tabel. Karena setiap baris dalam tabel ini tidak berisi semua sebagai. Oleh karena itu, dekomposisi ini bersifat lossy.

Pertanyaan 7 : Diberikan skema relasional R (A, B, C, D, E) dan diberikan himpunan FD berikut yang didefinisikan pada R.

F= {A → BC, CD→ E, AC→ E, B→ D, E→ AB}

(a) Tentukan dekomposisi lossless-join dari R.

(b) Tentukan dekomposisi R yang bukan lossless-join.

(c) Tentukan jika R dalam 3NF w.r. ke F jika tidak melanggar 3NR

Jawaban : (a) Satu dari banyak kemungkinan dekomposisi lossless-join

Misal R_1 = (A,D,E), R_2 (A,B, C)

Kedua dekomposisi R_I dan R_2 dari R adalah dekomposisi lossless-join. Hal ini karena

$R_I \cap R_2$= {A,D,E}, {A, B,C} = {A}

Sejak A → BC dan A adalah kunci di R_2.

∴ $R_1 \cap R_2$ = {A,B, C} = R_2·

(b) Dekomposisi R adalah R_1 dan R_2 :

Misalkan R_1 = {A,B, C,D} dan R_2 = {B,E}

Dekomposisi ini merupakan dekomposisi lossy karena

$R_1 \cap R_2$ = {A,B,C,D} ∩ {B,E} = {B}

·: B→ D

Oleh karena itu, B→ R_1 atau R_2, yaitu {BD→ R_1 atau R_2}

(c) Kunci kandidat dari R adalah A

karena A → *BC*

∴ A^+ = {A,B, C}

∴ B→ D∵B⊆A^+

∴ A^+ =A^+ ∪ {D}

A^+ = {A,B, C,D}

AC→ E ∴ AC⊆A^+

∴ A^+=A^+ ∪ E = {A,B, C,D,E} = R.

Jadi A adalah kunci kandidat.

Demikian pula CD, AC, dan E juga merupakan kunci kandidat. Oleh karena itu, kunci kandidat Rare A,AC,CD danE. Dengan demikian tidak ada ketergantungan yang melanggar R Oleh karena itu dalam 3NF.

Pertanyaan 8: Untuk setiap skema relasi berikut dan himpunan FD.

(1) R adalah (A, B, C, D) dengan FD.

F = {A → B,B → C, C→ D,D → A}

(2) R adalah (A, B, C, D) dengan FDs

F = {B → C, B → D}

(i) Identifikasi kunci kandidat untuk R.

(ii) Tunjukkan pelanggaran BCNF dan dekomposisi jika perlu.

(iii) Tunjukkan pelanggaran 3NF dan dekomposisi jika perlu.

Jawaban :

(1) Diketahui *R (A, B, C, D)* dengan *FD*

F (A → B,B → C, C → D,D → A)

(i) Untuk *A → B, ∵A_+ = {A,B}*

∴B → C, .: B ⊆A^+

A^+ =A^+ ∪{C}

A^+ = {A,B,C}

C→ D, ∵ C⊆A+

∴A^+ = {A,B, C,D} = R.

∴A→ ABCD

Demikian pula,

B → ABCD

C→ ABCD

D→ ABCD

Jadi kunci kandidatnya adalah A, B, C, D.

(ii) Karena semua {A, B, C, D} adalah kunci kandidat. Dengan demikian tidak ada pelanggaran BCNF. Oleh karena itu tidak diperlukan dekomposisi.

(iii) Karena A,B,C,D semuanya adalah kunci kandidat. Dengan demikian tidak ada pelanggaran 3NF Oleh karena itu tidak diperlukan dekomposisi.

(2) Diketahui R (A, B, C, D) dengan FD

F = {B → C, B → D}

(i) B^+ = {B,C,D}

Karena penutupan B (B^+) tidak mencakup semua atribut R

yaitu, B^+ = {B,C,D} ≠FR = {A,B,C,D}

Oleh karena itu tidak ada kunci kandidat.

(ii) B → C, B → D keduanya melanggar BCNF. karena tidak ada kunci kandidat. Oleh karena itu perlu dilakukan dekomposisi. Karena AB → ABCD. Jadi AB adalah kunci kandidat. Oleh karena itu satu kemungkinan dekomposisi BCNF adalah

{(A,B), (B, C,D)}

(iii) B → C, B → D keduanya melanggar 3NF. Karena B bukan kunci super dan CD bukan bagian dari kunci kandidat. Salah satu kemungkinan dekomposisi 3NF adalah

{(A,B), (B, C,D)} atau {(A,B), (B, C), (B,D)}

Karena AB→ ABCD. Jadi AB adalah kunci kandidat. Q

Pertanyaan 9 : Apakah skema ini dalam 3NF ?

(a) R = {city,street,zip}

F = {kota, jalan → kode pos, kode pos → kota}

(b) R = (A,B, C)

F = {A → B,B → C}

(c) R = (A,B, C,D)

F = {B → C, B → D}

(d) R = (A,B, C,D)

F = {AB → C, C → D, D → A}

(e) R = (A,B, C,D)

F = {AB → C, BC → D, CD → A,AD -+ B}

Jawaban :

(a) R = {kota, jalan, kode pos}

F = {kota, jalan →kode pos, kode pos → kota}

Untuk kota, jalan → kode pos

∴ (kota, jalan)$^+$= (kota, jalan,kode pos)

= *R*

Jadi, {kota, jalam} adalah kuncinya. Demikian pula, {kdoe pos, jalan} juga merupakan kuncinya. Oleh karena itu, {kota, jalan} dan {kode pos, jalan} adalah kuncinya. LHS kota, jalan → zip adalah kuncinya.

Jadi tidak apa-apa. RHS zip → city adalah bagian dari kunci jadi tidak apa-apa. Oleh karena itu, skema dalam 3NF.

(b) R = (A,B, C)

F = {A → B,B → C}

Satu-satunya kunci adalah A. LHS dari B → C bukan kunci super. RHS bukan bagian dari kunci. Oleh karena itu, skema tidak dalam 3NF.

(c) Diketahui *R = (A,B,C,D)*

F = {B → C, B → D}

Kuncinya hanya AB. LHS B → C bukan kunci super. RHS bukan bagian dari kunci. Oleh karena itu, skema tidak dalam 3NF.

(d) Diketahui *R = (A, B, C, D)*

F = {AB → C, C → , D → A}

Kuncinya adalah AB, BC dan BD.

LHS dari AB → C adalah kunci dan RHS dari C → D adalah bagian dari kunci. RHS dari D → A adalah bagian dari kunci. Oleh karena itu, skema dalam 3NF.

(e) R = (A,B, C,D)

F = {AB → C, BC → D, CD → A,AD → B}

Pertanyaan 10 : Berikan dekomposisi lossless-join, dependensi-melestarikan menjadi 3NF dari skema R.

R = (A,B, C,D,E)

F= {A → BC,CD→ E,B→ D,E→ A}

Jawab : Skema R = (A, B, C, D, E) sudah menjadi 3NF karena kunci kandidat Rare (A, BC, CD, E). Kami juga dapat membuat skema dari algoritma.

R = {(A,B, C), (C,D,E), (B,D), (E,A)}

skema (A, B, C) berisi kunci kandidat.

∴ R adalah dependensi join lossless 3NF yang mempertahankan ketergantungan.

Pertanyaan 11 : Berikan dekomposisi pelestarian lossless-join ke dalam BCNF skema R pertanyaan di atas.

Jawaban: Kami. ketahuilah bahwa FD B → D tidak sepele karena D ⊆ B, dan LHS bukan superkey. Dengan algoritma kita menurunkan relasi {(A, B, C, E), (B, D)} dalam BCNF.

Pertanyaan 12 : Diketahui R = (A, B, C, D)

$F = \{A \rightarrow B, B \rightarrow C\}$

Terapkan dekomposisi BCNF.

Jawaban:

(a) Menggunakan A → B Terlebih Dahulu

$R_I = (A, B), R_2 = (A, C, D)$

Oleh A → C (yang ada di F^+) dekomposisi R_2

$R_3 = (A, C), R_4 = (A, D)$

Skema relasi yang dihasilkan adalah Rh R3 dan R4.

∴ **Hasil**: (A,B),.(A, C), (A,D)

Hasilnya bukan pelestarian ketergantungan.

(b) Menggunakan B → C Pertama

$R_I = (B, C), R_2 = (A, B, C)$

Dekomposisi R_2

$R_3 = (A, B),$

$R_4 = (A, D)$

Hasil: BC,AB,AD

Skema relasi yang dihasilkan adalah R_1, R_3, R_4.

Hasil: (B, C), (A, B), (A, D)

Hasilnya adalah pelestarian ketergantungan.

Pertanyaan 13: Hitung penutupan himpunan F berikut dari dependensi fungsional untuk skema relasi R = (A,B, C,D,E).

A→ BC

CD→ E

B→ D

E→ A

Buat daftar kunci kandidat untuk R.

Jawaban: Dimulai dengan A → BC

Kita dapat menyimpulkan A → B dan A → C

∵ A → Band B → D, lalu A → D (Transitivitas) ...(i)

∵ A → BC, B → D, lalu A → CD. ...(ii)

∵ A → CD dan CD → E, lalu A → E (Transitivitas) ...(iii)

∵ A → A yang kita miliki (Refleksif) ...(iv)

Dari langkah di atas (i), (ii), (iii), (iv) mengambil serikat. Kita mendapatkan A→ ABCDE. Jadi A adalah kunci kandidat.

∵ E → A dan A → ABCDE, maka

E → ABCDE (Transitivitas)

Jadi E adalah kunci kandidat.

Untuk FD CD→ E

∵ CD → E dan E → ABCDE, maka

CD → ABCDE (Transitivitas)

Jadi CD juga merupakan kunci kandidat

∵ B → D dan BC → CD, maka

BC → ABCDE (augmentasi dan transitivitas).

Jadi BC adalah kunci kandidat. Oleh karena itu kunci kandidat adalah

(A, BC, CD, E)

Oleh karena itu, setiap FD dengan A, E, BC atau CD pada LHS dari panah berada di F+, tidak peduli dengan atribut lain yang muncul di FD. Izinkan untuk merepresentasikan himpunan atribut dalam R, maka F+ adalah BD →B, BD →D, C → C, D → D BD → BD, B → D, B → B, B → BD dan semua FD dari formulir

$A \blacksquare\!\!\rightarrow \alpha, BC \blacksquare\!\!\rightarrow \alpha, CD \blacksquare\!\!\rightarrow \alpha, E \blacksquare\!\!\rightarrow \alpha$

di mana a adalah himpunan bagian dari (A, B, C, D, E).

Pertanyaan 14 : Pertimbangkan relasi R = (A,B,C,D,E) M adalah himpunan dependensi fungsional multinilai.

M = (A ⇉BC, B ⇉CD,E ⇉AD)

Berikan kerugian kurang bergabung dekomposisi skema R menjadi 4NF.

Jawaban: Diketahui A ⇉ BC ...(i)

∵ A ⇉ BC dan B ⇉ CD [∵ A ⇉ B C (penguraian) A ⇉ B dan A ⇉ CJ]

∴ A ⇉ CD ...(ii) ∵ B ⇉ *CD*

dengan penyatuan (i) dan (ii) A ⇉ BCD [∵ A ⇉ C dan B ⇉ *CD* ∴ A ⇉ CD]

Diketahui R (A, B, C, D, E) Biarkan dekomposisi R menjadi R_1 dan R_2

R_I = (A, B, C, D) (A adalah atribut kunci)

R_2 = (A, D, E) (E adalah atribut kunci)

Untuk dekomposisi lossless-join baik

$R_I \cap R_2 \rightarrow R_I$

Atau $R_I \cap R_2 \rightarrow R_2$

∴ $R_I \cap R_2 = (A, B, C, D) \cap (A, D, E) = \{A, D\}$

Sejak A → BCD.

∴ $R_I \cap R_2 = (A,B, C,D) \cap (A,D,E) \rightrightarrows (A,B, C,D) = R_I$

Oleh karena itu, kita dapat mengatakan bahwa skema R dapat didekomposisi menjadi (A, B, C, D) dan (A, E, D), yang merupakan dekomposisi lossless-join dan berada dalam 4NF.

Pertanyaan 15: Diketahui R = {A, B, C, D, E} dan himpunan M dari dependensi multinilai.

M = {A ⇉ BC,B ⇉ CD,E ⇉ AD}

Daftar semua MUD non-sepele di M^+.

Jawaban:

M^+ = {A ⇉ BCD

E ⇉ BCD

Untuk A ⇉ BCD

Memberikan A ⇉ BC (didekomposisikan)

A ⇉ B

A ⇉ C

Sejak B ⇉ *CD*

A ⇉ B dan B ⇉ CD

A ⇉ CD

Dengan gabungan (i) dan (ii) A⇉ BCD

Sejak	$E \rightrightarrows AD$
Menerapkan dekomposisi	
	$E \rightrightarrows A$ dan $E \rightrightarrows D$
Karena	$E \rightrightarrows A$ dan $A \rightrightarrows BCD$
∴	$E \rightrightarrows BCD$

Pertanyaan 16 : Jelaskan bagaimana FD dapat digunakan untuk menunjukkan hal berikut,'

- Himpunan hubungan satu-ke-satu ada antara akun kumpulan entitas dan pelanggan.
- Sebuah set hubungan banyak ke satu ada antara entitas set akun dan pelanggan.

Jawaban: Misalkan PK (r) menyatakan atribut kunci utama dari relasi r.

- FD PK (akun) → PK (pelanggan) dan PK (pelanggan) → PK (akun) menunjukkan hubungan satu-ke-satu, Karena dua tupel dengan nilai yang sama untuk akun harus memiliki nilai yang sama untuk pelanggan dan setiap dua tupel yang menyetujui pelanggan harus memiliki nilai yang sama untuk akun.
- FD PK (akun) → PK (pelanggan) menunjukkan hubungan banyak ke satu, karena setiap nilai akun yang diulang akan memiliki nilai pelanggan yang sama tetapi banyak nilai akun mdY memiliki nilai pelanggan yang sama.

Pertanyaan 17 : Coba lihat kumpulan relasi dan dependensi berikut ini. Asumsikan bahwa setiap relasi 1.1 memperoleh cd melalui dekomposisi dari relasi dengan atribut ABCDEFGHI dan bahwa semua dekomposisi relasi ABCDEFGHI terdaftar untuk setiap pertanyaan. (Pertanyaan tidak tergantung pada masing-masing, tetapi dependensi yang diberikan pada ABCDEFGHI berbeda.) Untuk setiap (sub),'

(a) bangun bentuk normal terkuat dari relasinya.
(b) Jika tidak ada dalam BCNF, dekomposisi menjadi (koleksi relasi BCNF.

1. R 1 (A,C,B,D,E),A → B, C→ D
2. R 2 (A, B, F), AC → E, B →F
3. R = (A,D), D →G, G → H
4. R 4 (D, C, H, G), A →I, I → A
5. R 5 (A, I, C, E)

Jawaban :

1. INF BCNF Dekomposisi: AB, CD, ACE
2. INF BCNF Dekomposisi: AB, BF
3. BCNF
4. BCNF
5. BCNF

Pertanyaan 18 : Misalkan Anda diberikan relasi R dengan empat atribut ABCD. Untuk setiap himpunan F D berikut, dengan asumsi hanya itu dependensi yang berlaku untuk R, lakukan hal berikut: (a) Identifikasi kunci kandidat (1') untuk R. (b) Identifikasikan bentuk normal terbaik yang R memenuhi (lNF, 2NF, 3NF, atau BCNF). (c) Jika R adalah BCNF, dekomposisi menjadi satu set hubungan BCNF yang mempertahankan dependensi.

1. C → D, C →A, B →C
2. B →CD→A
3. ABC →D, D →A

4. A→B,BC→D,A→C
5. AB →C, AB →D, C →A, D →B

Jawaban :

Jawaban no 1 :

(a) Kandidat utama : B
(b) R adalan 2NF tapi bukan 3NF
(c) C → D dan C → A keduanya menyebabkan pelanggaran BCNE Salah satu cara untuk mendapatkan (lossless) join melestarikan dekomposisi adalah dengan menguraikan R menjadi AC, BC, dan CD.

Jawaban no 2

(a) Kandidat utama : BD
(b) R adalah 1NF tapi bukan 2NF
(c) Baik B → C dan D → A menyebabkan pelanggaran BCNF. Dekomposisi : AD, B C, BD (diperoleh dengan penguraian pertama menjadi AD, BCD) adalah BCNF dan lossless dan join-preserving.

Jawaban no 3:

(a) Kandidat utama : ABC, BCD
(b) R adalah 3NF tapi bukan BCNF
(c) ABCD tidak ada dalam BCNF karena D → A dan D bukan kunci. Namun, jika kita membagi R.sebagai AD, BCD kita tidak dapat mempertahankan ketergantungan ABC → D. Jadi tidak ada dekomposisi BCNF.

Jawaban no 4:

(a) Kandidat utama : A
(b) R adalah 2NF tapi bukan 3NF (Karena FD = BC→D)
(c) BC → D melanggar BCNF karena BC tidak mengandung kunci. Jadi kita bagi R seperti pada: BCD, ABC.

Jawaban no 5:

(a) Kandidat utama: Ab, BC, CD, AD
(b) R adalah 3NF tapi bukan BCNF (karena FD = C → A)
(c) C → A dan D → B keduanya menyebabkan pelanggaran. Jadi terurai menjadi: A C, B CD tetapi ini tidak mengamati AB → C dan AB → D, dan BCD masih belum BCNF karena D → B. Jadi kita perlu menguraikan lebih lanjut menjadi: AC, BD, CD. Namun, ketika kami mencoba untuk menghidupkan kembali dependensi fungsional yang hilang dengan menambahkan ABC dan ABD, kami melihat bahwa hubungan ini tidak dalam bentuk BCNF. Oleh karena itu, tidak ada dekomposisi BCNF.

3.13 PERTANYAAN REVIEW

1. Apakah yang Anda maksud: ketergantungan fungsional? Jelaskan dengan contoh dan diagram ketergantungan fungsional.
2. Apa pentingnya ketergantungan fungsional dalam desain database?
3. Apa karakteristik utama dari dependensi fungsional?
4. Jelaskan aksioma Armstrong. Apa itu aturan turunan?

5. Mari kita asumsikan bahwa berikut ini diberikan :

   Himpunan atribut R = ABCDEFGH

   Himpunan FD dari F = {AB → C,AC → B,AD → E,B → D,BC → A,E → G}

   Manakah dari dekomposisi R = ABCDEG berikut, dengan himpunan dependensi F yang sama,

   (a) mempertahankan ketergantungan dan

   (b) lossless-join

   (c) {AB, BC,ABDE, EG}

   (d) { ABC,ACDE,ADG}

6. Apa sifat gabungan lossless atau non-aditif dari dekomposisi? Mengapa itu penting?
7. Apa yang Anda pahami dengan istilah normalisasi? Menjelaskan proses normalisasi data. Apa yang dicapainya?
8. Jelaskan tujuan dari normalisasi data.
9. Apa perbedaan bentuk normal?
10. Definisikan apa itu yang disebut sebagai 1NF, 2NF dan 3NF.
11. Diberikan sebuah relasi R (A, B, C, D, E) dan F = (A → B, BC → D, D → BC, DE → 1)), sintesiskan satu set skema relasi 3 NF.
12. Definisikan bentuk normal Boyce-Codd (BCNF) Apa bedanya dengan 3 NF? Mengapa dianggap lebih kuat dari 3 NF? Berikan contoh untuk mengilustrasikannya.
13. Mengapa 4 NF lebih disukai daripada BCNF?
14. Relasi R (A, B, C) memiliki FD AB → C dan C → A. Apakah R ada di 3 NF atau di BCNF? Jelaskan jawaban Anda
15. Jelaskan hal berikut:
    (a) Mengapa R_2 ada di 2NF tetapi tidak 3NF, di mana R_2 = ({A, B, C, D, E}, {AB → CE,E → AB, C → D})
    (b) Mengapa R_3 ada di 3NF tapi bukan BCNF, dimana R_3 = ({A, B, C, D}, {A → C, D → B})

BAB 4
KONSEP PEMROSESAN TRANSAKSI

4.1 KONSEP TRANSAKSI

Kumpulan operasi yang membentuk satu unit kerja logis disebut transaksi. Transaksi adalah unit eksekusi program yang mengakses dan mungkin memperbarui berbagai item data. Kita dapat mengatakan bahwa transaksi terdiri dari semua operasi yang dijalankan antara transaksi awal & akhir transaksi. Transaksi memiliki empat properti berikut.

(1) **Atomicity**: Entah semua operasi transaksi tercermin dengan benar dalam database atau tidak sama sekali, yaitu, jika semuanya bekerja dengan benar tanpa kesalahan, maka semuanya akan dikomit ke database. Jika ada bagian dari transaksi yang gagal, seluruh transaksi akan dibatalkan.

(2) **Konsistensi**: Properti konsistensi menyiratkan bahwa jika database berada dalam keadaan konsisten sebelum dimulainya transaksi, maka setelah pelaksanaan transaksi, database juga akan berada dalam keadaan konsisten.

(3) **Isolasi**: Meskipun beberapa transaksi dapat dieksekusi secara bersamaan, sistem menjamin bahwa, untuk setiap pasangan transaksi Ti & Tj, tampaknya Ti bahwa Tj menyelesaikan eksekusi sebelum Ti dimulai atau Tj memulai eksekusi setelah Ti selesai.

(4) **Durability**: Durability memastikan bahwa, setelah transaksi selesai, pembaruan transaksi tersebut tidak hilang, bahkan jika ada kegagalan sistem.

Keempat properti ini disebut properti ACID dari Transaksi.

4.2 DATA AKSES TRANSAKSI

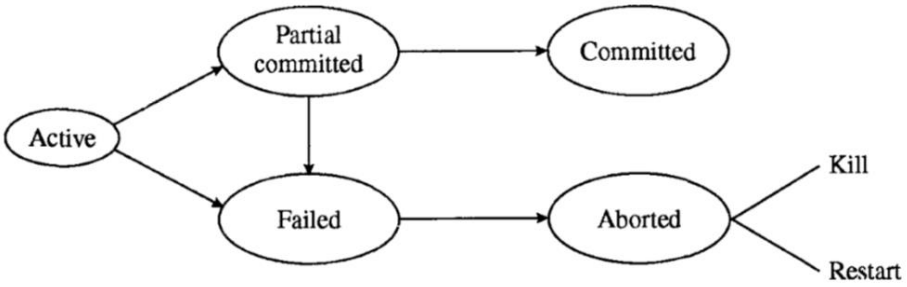
Transaksi akses data menggunakan dua operasi.

(i) **Read(X)** : Operasi Read (X) mentransfer item data X dari database ke buffer lokal milik transaksi yang mengeksekusi operasi read.

(ii) **Write (X)** : Yang mentransfer item data X dari buffer lokal dari transaksi yang mengeksekusi penulisan kembali ke database. Contoh: Misalkan T_1 adalah transaksi yang mentransfer Rp. 500 ribu dari rekening A ke rekening B. Transaksi ini dapat didefinisikan sebagai :

```
TI          read (A)
            A: = A-SOD;
            write(A);
            read (B);
            B: = B+500;
            write (R);
```

4.3 WILAYAH TRANSAKSI



Gambar 4.1 Wilayah Transkasi

Transaksi harus dalam salah satu status berikut:

- **Active**: Status ini adalah status awal, transaksi mengatakan dalam status ini saat sedang dieksekusi.
- **Partial Committe :** Setelah pernyataan akhir dieksekusi.
- **Failed**: Gagal, setelah ditemukan bahwa eksekusi normal tidak dapat lagi dilanjutkan.
- **Aborted** : Dibatalkan, setelah transaksi dibatalkan dan database telah dikembalikan ke keadaannya sebelum transaksi dimulai.
- **Committed**: Setelah berhasil diselesaikan. yaitu, Transaksi yang menyelesaikan eksekusinya dengan sukses dikatakan berkomitmen.
- Setelah transaksi telah dilakukan, kami tidak dapat membatalkan efeknya dengan membatalkannya.
- Suatu transaksi dikatakan dihentikan jika telah dilakukan atau dibatalkan.
- Sebuah transaksi dimulai dalam keadaan aktif. Ketika menyelesaikan pernyataan terakhirnya, ia memasuki kondisi komitmen sebagian. Pada titik ini, transaksi telah menyelesaikan eksekusinya, tetapi masih ada kemungkinan bahwa transaksi tersebut harus dibatalkan.

4.4 EKSEKUSI BERSAMAAN

Sistem pemrosesan transaksi biasanya memungkinkan beberapa transaksi berjalan secara bersamaan memungkinkan beberapa transaksi memperbarui data secara bersamaan, menyebabkan beberapa komplikasi dengan konsistensi data. Memastikan konsistensi terlepas dari pelaksanaan transaksi secara bersamaan membutuhkan kerja ekstra. Lebih mudah untuk bersikeras bahwa transaksi berjalan secara serial. yaitu, Satu per satu, masing-masing dimulai hanya setelah yang sebelumnya selesai.

Ada dua alasan untuk mengizinkan konkurensi:

(i) Peningkatan throughput & pemanfaatan sumber daya.

(ii) Mengurangi waktu tunggu

Jadwal

- Urutan eksekusi dalam urutan kronologis disebut jadwal.
- Jadwalnya serial; setiap jadwal serial terdiri dari urutan instruksi dari berbagai transaksi.

4.5 SERIALLZABILLTY

Sebuah jadwal non serial dikatakan serializable, jika konflik ekuivalen atau view-ekuivalen dengan jadwal serial. Misalnya:

Tabel 4.1 Contoh Jadwal Non Serial

T_1	T_2
read (A)	
write (A)	
	read (A)
	write (A)
read (B)	
write (B)	
	read (B)
	write (B)

Jenis Serializability

- Conflict Serializability
- View Serializability

Conflict Serializability

Mari kita pertimbangkan! jadwal S di mana ada dua instruksi berurutan Ii & Ij dari transaksi Ti & Tj masing-masing (i ≠j) Jika Ii & Ij merujuk ke item data yang berbeda, maka kita dapat menukar Ii & Ij tanpa mempengaruhi hasil instruksi apa pun dalam jadwal . Jika Ii & Ij mengacu pada item data yang sama Q. Maka urutan mungkin menjadi masalah. Ada empat kasus yang muncul.

Kasus 1 : Jika Ii = read (Q), Ij = read (Q)

Urutan Ii & Ij tidak penting.

Kasus 2 : Jika Ii = read (Q), Ij = tulis (Q)

Jika Ii mendahului Ij maka Ti tidak memread nilai Q yang ditulis oleh Tj pada instruksi Ij. Jadi ketertiban adalah masalah.

Kasus 3 : Jika Ii = write (Q), Ij = read (Q).

Urutannya adalah soal.

Kasus 4 : Jika Ii = write (Q), Ij = write (Q)

Tabel 4.2 *Conflict Serializability*

T_1	T_2
read (A)	
write (A)	
	read (A)
	write (A)
read (B)	
write (B)	
	read (B)
	write (B)

Urutannya tidak penting. Tetapi untuk beberapa kasus mungkin menjadi masalah.

Dalam jadwal ini; penulisan (A) T_1 bertentangan dengan pembacaan (A) T_2. Sedangkan penulisan (A) T_2 tidak bertentangan dengan pembacaan (B) T_1. Karena A & B adalah dua item yang berbeda. Kita dapat menukar ke instruksi yang tidak bertentangan.

- tukar baca (B) dari T_1 dengan membaca (A) dari T_2
- tukar write (B) dari T_1 dengan write (A) dari T_2
- tukar write (B) dari T_1 dengan membaca (A) dari T_2

Tabel 4.3 Setelah bertukar Jadwal di atas

T_1	T_2
read (A)	
write (A)	
	read (A)
read (B)	
	write (A)
write (B)	
	read (B)
	write (B)

Konsep kesetaraan konflik mengarah pada konsep serializability konflik. Kami mengatakan bahwa jadwal S adalah konflik serial jika konflik setara dengan jadwal serial. Contoh: Misalkan jadwal serial S.

Tabel 4.4 Contoh Jadwal Serial S

T_1	T_2
read (A)	
A : = A – 50	
write (A)	
read (B)	
B : = B + 50	
write (B)	
	read (A)
	temp : = A * 0.1
	A : = A – temp
	write (A)
	read (B)
	B : = B + temp
	write (B)

Setelah swapping S menjadi jadwal S^1 :

Tabel 4.5 Jadwal Serial S swapping S menjadi jadwal S^1

T_1	T_2
read (A)	
A : = A – 50	
write (A)	
	read (A)
	temp : = A * 0.1
	A : = A – temp
	write (A)
read (B)	
B : = B + 50	
write (B)	
	read (B)
	B : = B + temp
	write (B)

Jadwal S^1 ekuivalen dengan jadwal S. Jadi ini adalah contoh dari serializability konflik.

View Serializability

Pertimbangkan dua jadwal S dan S^l, di mana beberapa set transaksi berpartisipasi dalam kedua penjadwal.

Tabel 4.6 Penjadwalan S dan S'

T_1	T_2
read (A)	
A : = A – 50	
write (A)	
read (B)	
B : = B + 50	
write (B)	
	read (A)
	temp : = A * 0.1
	A : = A – temp
	write (A)
	read (B)
	B : = B + temp
	write (B)

Jadwal S dan S^l dikatakan ekuivalen tampilan jika tiga kondisi terpenuhi.

(1) Untuk setiap item data Q, jika transaksi Ti membaca nilai awal Q pada jadwal S, maka transaksi T_i harus, pada jadwal S^1, juga membaca nilai awal Q.

(2) Untuk setiap item data 0 Q jika transaksi T_i mengeksekusi read (Q) pada schedule S, dan jika nilai tersebut dihasilkan oleh operasi write (Q) yang dieksekusi oleh transaksi Tj, maka operasi read (Q) dari transaksi T_i harus pada schedule S^1, juga membaca nilai Q yang dihasilkan oleh operasi write (Q) yang sama dari transaksi T_j.

(3) Untuk setiap item data Q, transaksi yang melakukan operasi write akhir (Q) pada jadwal S harus melakukan operasi write akhir (Q) pada jadwal S^l.

Konsep view equivalence mengarah pada konsep view serializability. Jadwal S^l adalah:

Tabel 4.7 Jadwal S^l (konsep view serializability)

T_1	T_2
read (A)	
A : = A – 50	
write (A)	
	read (A)
	temp : = A * 0.1
	A : = A – temp
	write (A)
read (B)	
B : = B + 50	
write (B)	
	read (B)
	B : = B + temp
	write (B)

Jadi jadwal S & S1 dapat dilihat secara serial, karena jadwal S & S^l adalah tampilan yang setara

Pengujian Serializability

Saat merancang skema kontrol konkurensi, kita harus menunjukkan bahwa jadwal yang dihasilkan oleh skema dapat serial. **Pengujian untuk serializability konflik**: Pertimbangkan

jadwal S. Kami membangun grafik berarah yang disebut "Grafik Prioritas" dari S. Himpunan tepi grafik memiliki salah satu dari tiga kondisi.

Ti - Tj (edge)

(i) Ti mengeksekusi write (Q) sebelum Tj mengeksekusi read (Q).

(ii) Ti mengeksekusi read (Q) sebelum Tj mengeksekusi write (Q).

(iii) Ti mengeksekusi write (Q) sebelum Tj mengeksekusi write (Q).

(A) (B)

Jika sebuah edge Ti → Tj ada pada graf preseden maka, dalam setiap jadwal serial S 1 yang ekuivalen dengan S. Ti harus muncul sebelum Tj. *Contoh* : Pada graf (A) Graf prioritas untuk jadwal 1, berisi sisi tunggal $T_1 \rightarrow T_2$, karena semua instruksi T 1 dieksekusi sebelum instruksi T_2 dieksekusi. *Demikian pula* : Grafik (B) menunjukkan, grafik prioritas untuk jadwal 2 dengan sisi tunggal $T_2 \rightarrow T_1$. Karena semua instruksi T_2 dieksekusi sebelum instruksi T_1 dieksekusi.

"Jika grafik prioritas untuk S memiliki siklus, maka jadwal S tidak konflik serializable".

"Jika graf tidak mengandung siklus, maka jadwal S adalah konflik seriaIizable".

- Untuk menguji serializability konflik, kita perlu membuat grafik prioritas dan menjalankan algoritma pendeteksian siklus.
- Algoritma deteksi siklus berdasarkan pencarian depth-First. misalnya.

Grafik prioritas berisi tepi $T_1 \rightarrow T_2$, karena T_I dieksekusi read (A) sebelum T_2 dieksekusi write (A). Itu juga berisi tepi $T \rightarrow T_I$ karena T_2 mengeksekusi read (B) sebelum T_I mengeksekusi write (13). Karena grafik prioritas mengandung siklus maka, itu tidak konflik serializable. **Pengujian untuk view serializability** : Pengujian untuk melihat serializability rumit. Faktanya, telah ditunjukkan bahwa masalah pengujian untuk melihat serializablitiy itu sendiri adalah NP-Complete. Jadi tidak ada algoritma yang efisien untuk menguji serializability tampilan.

- Skema kontrol konkurensi dapat menggunakan kondisi yang cukup untuk melihat serializability, Tapi melihat jadwal serializability mungkin tidak memenuhi kondisi yang cukup.

Catatan: Kita dapat menguji jadwal yang diberikan untuk serializabilitas konflik dengan membuat grafik prioritas untuk jadwal tersebut dan dengan mencari tidak adanya siklus dalam grafik.

4.6 PEMULIHAN KEMBALI

Jika transaksi Ti gagal, apapun alasannya. Kita perlu membatalkan (memutar kembali) efek dari transaksi ini untuk memastikan properti atomisitas dari transaksi tersebut. Dalam sistem yang memungkinkan eksekusi konkuren, perlu juga untuk memastikan bahwa setiap transaksi Tj yang bergantung pada Ti (yaitu Tj telah membaca data yang ditulis oleh Ti) juga dibatalkan. Untuk mencapai kepastian ini, kita perlu menempatkan batasan pada jenis jadwal yang diizinkan dalam sistem

Jadwal yang Dapat Dipulihkan

Jadwal yang dapat dipulihkan adalah jadwal di mana, untuk setiap pasangan transaksi Ti & Tj sedemikian rupa sehingga Tj membaca item data yang sebelumnya ditulis oleh Ti. Operasi komit Ti muncul sebelum operasi komit Tj.

Tabel 4.8 Jadwal yang DiPulihkan

T_1	T_2
read (A)	
write (A)	
	read (A)
read (B)	

Dalam transaksi yang diberikan, T_2 hanya melakukan satu pembacaan instruksi (A). Misalkan sistem mengizinkan T_2 untuk melakukan commit segera setelah mengeksekusi instruksi read (A). Jadi T_2 melakukan sebelum T_1 melakukannya. Sekarang anggaplah T_1 gagal sebelum melakukan, kita harus membatalkan T_2 untuk memastikan atomositas transaksi. Namun, T_2 telah berkomitmen dan tidak dapat dibatalkan. Jadi situasi T_1 ini tidak mungkin dipulihkan dengan benar dari kegagalan T_1. **Catatan**: Sebagian besar sistem Database mengharuskan semua jadwal dapat dipulihkan.

Jadwal Cascadeless

Bahkan jika jadwal dapat dipulihkan, untuk memulihkan dengan benar-dari kegagalan transaksi Ti. Kami mungkin harus melakukan roll-back beberapa transaksi. Situasi seperti itu terjadi jika transaksi telah membaca data yang ditulis oleh Ti.

- Fenomena, di mana satu kegagalan transaksi menyebabkan serangkaian pembatalan transaksi, disebut "Cascading Rollback".

Tabel 4.9 Jadwal Cascadeless

T_1	T_2	T_3
read (A)		
read (B)		
write (A)		
	read (A)	
	write (A)	
		read (A)

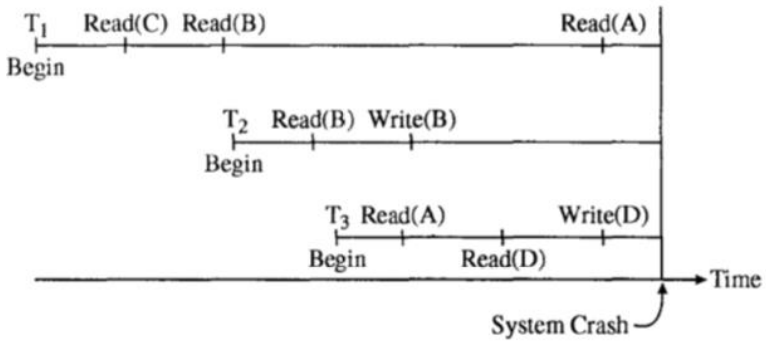
Pada skedul ini, Transaksi T_1 menulis nilai A yang dibaca oleh transaksi T_2. Transaksi T_2 menulis nilai A yang dibaca oleh transaksi T 3. Jika T 1 gagal, TI harus melakukan roll back karena T_2 tergantung pada T_1. Jadi T_2 juga rollback dan karena T_3 tergantung pada T_2 maka T_3 juga rollback. Jadi setelah kegagalan Transaksi T 1 a}l transaksi T_2 & T_3 juga rollback dengan T_1. Oleh karena itu, transaksi yang diberikan adalah cascading rollback.

4.8 PEMULIHAN TRANSAKSI

Sebuah transaksi dimulai dengan eksekusi yang berhasil dari pernyataan TRANSAKSI BEGIN dan diakhiri dengan eksekusi yang berhasil dari pernyataan COMMIT atau ROLLBACK. Kita

sekarang dapat melihat bahwa transaksi bukan hanya unit kerja tetapi juga unit pemulihan. Jika transaksi berhasil dilakukan maka sistem akan menjamin bahwa pembaruannya akan diinstal secara permanen di database, bahkan jika sistem macet pada saat berikutnya. Sangat mungkin, untuk data bahwa sistem mungkin macet setelah komit dihormati tetapi sebelum pembaruan ditulis secara fisik ke database, jadi hilanglah pada saat crash.

Bahkan jika itu terjadi, prosedur restart sistem akan tetap menginstal pembaruan tersebut di database. Dengan demikian prosedur restart akan memulihkan setiap transaksi yang berhasil diselesaikan tetapi tidak berhasil mendapatkan pembaruan mereka yang ditulis secara fisik sebelum crash.



Gambar 4.2 Prosedure Restart Program

Pemulihan Sistem: Sistem harus siap untuk pulih, tidak hanya dari kegagalan murni lokal seperti terjadinya kondisi luapan dalam transaksi individu, tetapi juga dari, "Kegagalan Global" seperti pemadaman listrik.

Klasifikasi Kegagalan

Pemulihan transaksi adalah proses memulihkan transaksi ke keadaan yang benar (konsisten) jika terjadi kegagalan. Kegagalan tersebut dapat diakibatkan oleh sistem crash karena kesalahan perangkat keras atau perangkat lunak, kegagalan media seperti head crash, atau kesalahan perangkat lunak pada aplikasi seperti kesalahan logis pada program yang mengakses transaksi. Jumlah teknik pemulihan yang didasarkan pada sifat atomisitas transaksi. Pemulihan transaksi membalikkan semua perubahan yang telah dibuat transaksi ke database sebelum dibatalkan.

a. **Kegagalan Lokal** : Kegagalan lokal hanya mempengaruhi transaksi di mana kegagalan benar-benar terjadi.
b. **Kegagalan Global :** Kegagalan Global mempengaruhi semua transaksi yang sedang berlangsung pada saat kegagalan. Kegagalan Global terbagi dalam dua kategori besar:

- Kegagalan Sistem
- Kegagalan Media

Kegagalan Sistem : Ada berbagai jenis kegagalan yang mungkin terjadi pada suatu sistem.

(i) **Kegagalan Transaksi:** Dalam kegagalan transaksi, ada dua jenis kesalahan yang mungkin terjadi.

(a) **Logical Error:** Transaksi tidak dapat lagi melanjutkan eksekusi normalnya karena beberapa kondisi internal seperti input yang salah, data tidak ditemukan, batas sumber daya terlampaui.

(b) **Kesalahan Sistem:** Sistem telah memasuki keadaan yang tidak diinginkan sebagai akibatnya transaksi tidak dapat dilanjutkan dengan ego eksekusi normalnya Deadlock terjadi di sistem, kelaparan di sistem.

(ii) **Sistem Crash :** Ada kerusakan perangkat keras, atau bug dalam perangkat lunak database atau sistem operasi, yang menyebabkan hilangnya konteks penyimpanan volatil, dan menghentikan pemrosesan transaksi. Dengan demikian kegagalan sistem, yang mempengaruhi semua transaksi yang sedang berlangsung tetapi tidak secara fisik merusak database.

- Kegagalan sistem kadang-kadang disebut crash lunak. **Kegagalan Media:** Yang menyebabkan kerusakan pada database, atau sebagian darinya dan mempengaruhi setidaknya transaksi-transaksi yang saat ini menggunakan bagian tersebut. Kegagalan media terkadang disebut hard crash. **Kegagalan Disk:** Kegagalan Disk adalah contoh Kegagalan Media.
- Sebuah blok disk kehilangan isinya sebagai akibat dari head crash atau kegagalan selama operasi transfer data.

Jenis Pemulihan Transaksi

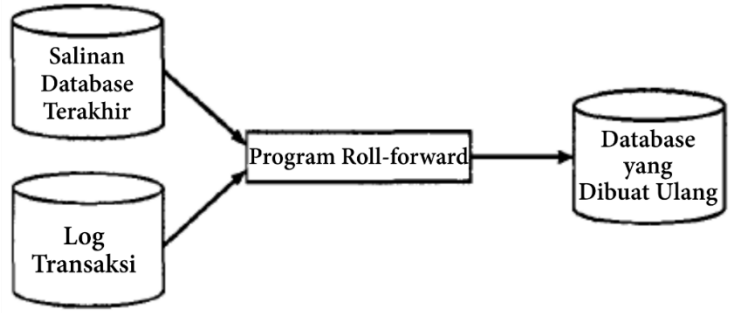
Jika terjadi jenis kegagalan apa pun, transaksi harus dibatalkan atau berkomitmen untuk menjaga integritas data. Log transaksi memainkan peran penting untuk pemulihan database .dan membawa database dalam keadaan yang konsisten jika terjadi kegagalan. Transaksi mewakili unit dasar pemulihan dalam sistem Database. Manajer pemulihan menjamin sifat atomisitas dan daya tahan transaksi jika terjadi kegagalan. Selama pemulihan dari kegagalan, mhnager pemulihan memastikan bahwa semua efek dari transaksi tertentu dicatat secara permanen dalam database atau tidak ada yang dicatat. Sebuah transaksi dimulai dengan eksekusi sukses dari pernyataan BEGIN TRANSACTION. Itu berakhir dengan eksekusi yang berhasil dari pernyataan COMMIT atau ROLLBACK. Dua jenis pemulihan transaksi berikut digunakan:

- Pemulihan ke depan
- Pemulihan mundur.

Forward Recovery (REDO)

Pemulihan ke depan (juga disebut roll-forward) adalah prosedur pemulihan, yang digunakan jika terjadi kerusakan fisik, misalnya crash paket disk (penyimpanan sekunder), kegagalan saat menulis data ke buffer database, atau kegagalan saat mentransfer buffer ke penyimpanan sekunder. Hasil antara transaksi ditulis dalam buffer database. Buffer database menempati area di memori utama. Dari buffer ini, data ditransfer ke dan dari penyimpanan sekunder database. Operasi pembaruan dianggap permanen hanya ketika buffer ditransfer ke penyimpanan sekunder. Operasi pembilasan (transfer) dapat dipicu oleh operasi COMMIT dari transaksi atau secara otomatis jika buffer menjadi penuh. Jika terjadi kegagalan antara menulis ke buffer dan membilas buffer ke penyimpanan sekunder, manajer pemulihan harus menentukan status transaksi yang dilakukan WRITE pada saat kegagalan. Jika transaksi sudah mengeluarkan COMMIT-nya, maka recovery manager akan mengulang (roll forward) sehingga transaksi tersebut diupdate ke database. Pengulangan pembaruan transaksi ini

juga dikenal sebagai roll-forward. Pemulihan ke depan menjamin properti durabilitas transaksi.



Gambar 4.3 Forward Recovery atau Redo

Untuk membuat ulang disk yang hilang karena alasan yang dijelaskan di atas, sistem mulai membaca salinan terbaru dari data yang hilang dan log transaksi dari perubahannya. Sebuah program kemudian mulai membaca entri log, mulai dari yang pertama direkam setelah salinan database dibuat dan berlanjut hingga yang terakhir direkam sebelum disk dihancurkan. Untuk setiap entri log ini, program mengubah nilai data yang bersangkutan dalam salinan database ke nilai setelah yang ditunjukkan dalam entri log. Ini berarti bahwa pemrosesan apa pun yang terjadi dalam transaksi yang menyebabkan entri log dibuat, hasil bersih dari database setelah transaksi itu akan disimpan. Operasi untuk setiap transaksi dilakukan yang menyebabkan perubahan dalam database sejak salinan diambil, dalam urutan yang sama dengan transaksi ini pada awalnya dijalankan. Ini membawa salinan database ke tingkat terbaru dari database yang dihancurkan.

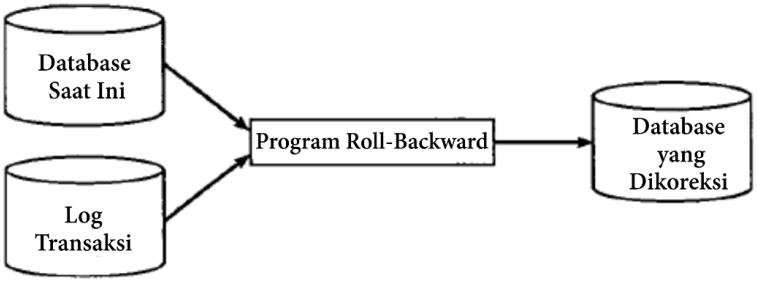
Gambar tersebut mengilustrasikan contoh sistem pemulihan ke depan. Ada sejumlah variasi metode pemulihan ke depan yang digunakan. Dalam satu variasi, perubahan mungkin telah dilakukan pada bagian data yang sama sejak salinan database terakhir dibuat. Dalam kasus ini, hanya yang terakhir dari perubahan tersebut pada saat disk dihancurkan yang perlu digunakan dalam memperbarui salinan database dalam operasi rolling forward.

Backward Recovery or UNDO

Pemulihan mundur (juga disebut roll-mundur) adalah prosedur pemulihan, yang digunakan jika terjadi kesalahan di tengah operasi normal pada database. Kesalahan tersebut dapat berupa manusia yang memasukkan nilai, atau program yang berakhir secara tidak normal dan meninggalkan beberapa perubahan pada database yang seharusnya dibuat. Jika transaksi tidak dilakukan pada saat kegagalan, itu akan menyebabkan inkonsistensi dalam database karena untuk sementara, program lain mungkin telah membaca data yang salah dan memanfaatkannya. Kemudian manajer pemulihan harus membatalkan (memutar kembali) semua efek dari database transaksi. Pemulihan ke belakang menjamin sifat atomisitas transaksi.

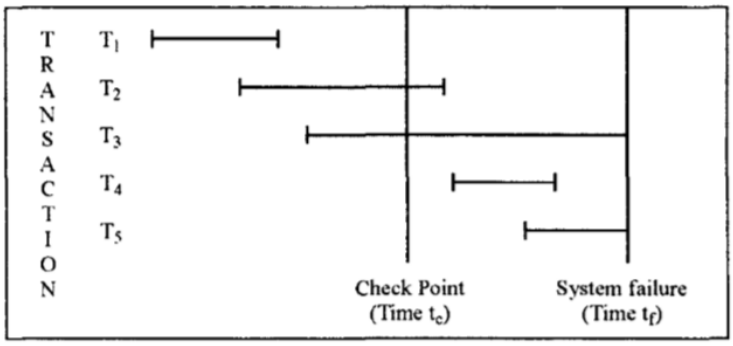
Gambar mengilustrasikan contoh metode pemulihan mundur. Dalam kasus pemulihan mundur, pemulihan dimulai dengan database dalam keadaan saat ini dan log transaksi diposisikan pada entri terakhir yang dibuat di dalamnya. Kemudian sebuah program membaca 'mundur' melalui log, mengatur ulang setiap nilai data yang diperbarui

dalam database ke "sebelum gambar" seperti yang direkam dalam log hingga mencapai titik di mana kesalahan dibuat. Dengan demikian, program 'UNDOES' setiap transaksi dalam urutan terbalik dari yang dibuat.



Gambar 4.4 *Backward Recovery or UNDO*

Contoh, Pada saat restart, sistem melalui prosedur berikut untuk mengidentifikasi semua transaksi tipe T_2 - T_5.



Gambar 4.5 Prosedur Transaksi berbanding Waktu

(1) Mulailah dengan dua daftar transaksi, daftar UNDO dan daftar REDO. Atur daftar UNDO sama dengan daftar semua transaksi yang diberikan di pos pemeriksaan terbaru yang diberikan dalam catatan pos pemeriksaan terbaru; atur daftar REDO menjadi kosong.
(2) Pencarian maju melalui log, mulai dari catatan titik pemeriksaan.
(3) Jika entri log TRANSAKSI AWAL ditemukan untuk transaksi T, tambahkan T ke daftar UNDO.
(4) Jika entri log COMMIT ditemukan untuk transaksi T, pindahkan T dari daftar UNDO ke daftar REDO.
(5) Ketika akhir log tercapai, daftar UNDO dan REDO mengidentifikasi masing-masing, transaksi tipe T_3 dan T_5 dan transaksi tipe T_2 dan T_4.

Sistem sekarang bekerja mundur melalui log, membatalkan transaksi dalam daftar UNDO. Memulihkan database ke keadaan yang konsisten dengan membatalkan pekerjaan disebut pemulihan mundur.

4.8 PEMULIHAN BERBASIS LOG

Struktur yang sangat penting digunakan untuk merekam modifikasi database adalah log. Log adalah urutan catatan log, merekam semua aktivitas pembaruan dalam database. Ada banyak jenis catatan log. Catatan log pembaruan memiliki bidang ini.

- **Pengidentifikasi transaksi:** Ini adalah pengidentifikasi unik dari transaksi yang melakukan operasi write.
- **Pengidentifikasi Item Data:** Ini adalah pengidentifikasi unik dari data yang ditulis. Ini adalah lokasi pada disk dari item data.
- **Nilai Lama:** Nilai lama adalah nilai item data sebelum ditulis.
- **Nilai Baru:** Ini adalah nilai yang akan dimiliki item data setelah penulisan.

Ada berbagai catatan log yang ada selama pemrosesan transaksi seperti memulai transaksi, melakukan atau membatalkan transaksi.

Catatan log ini adalah :

- < Ti Start> Transaksi Ti dimulai.
- <Ti, Xj, VI, V2> Transaksi Ti telah melakukan write pada item data Xj. Xj memiliki nilai VI sebelum penulisan dan akan memiliki nilai V 2 setelah penulisan
- < Ti commit> Transaksi Ti telah dilakukan.
- < Ti abort> Transaksi Ti telah dibatalkan.
- catatan log berguna untuk pemulihan dari kegagalan sistem dan disk, log harus berada di penyimpanan stabil untuk saat ini, kami berasumsi bahwa setiap catatan log ditulis ke akhir log pada penyimpanan stabil segera setelah dibuat

Pelaksanaan transaksi Ti berlangsung sebagai berikut. Sebelum Ti memulai eksekusinya, sebuah record <Ti start> ditulis ke log. Operasi write (x) oleh Ti menghasilkan penulisan record baru ke log. Akhirnya, ketika Pi melakukan sebagian, catatan <Ti commit> ditulis ke log. Sebagai Contoh " Pertimbangkan T_1 adalah transaksi yang mentransfer RP. 50 ribu dari rekening A ke rekening B.

T_I : read (A);
A = A-50;
write (A);
read (B);
B = B + 50;
write (B)

Biarkan Transaksi T_2 menarik Rp. 100 ribu dari rekening C.

T2 : read (C);
C: = C - 100;
write (C).

Misalkan nilai rekening A, B & C sebelum eksekusi terjadi Rp. 2000, Rp. 500 dan Rp. 1000 masing-masing. Log sebagai hasil eksekusi T_1 & T_2

<T_I start>
<T_I, A, 1950>
<T_I,B 550>
<T_I commit>
<T_2 start>
<T_2, C, 900>
<T2 commit>

Bagian dari log database.

Log	Database
<T_I start>	
<T_I, A, 1950>	
<T_1, B, 550>	
<T_I commit>	
	A = 1950
	B = 550
<T_2 start>	
<T_2, C, 900>	
<T_2 commit>	
	C = 900

Status log & database

4.9 CHECK POINT

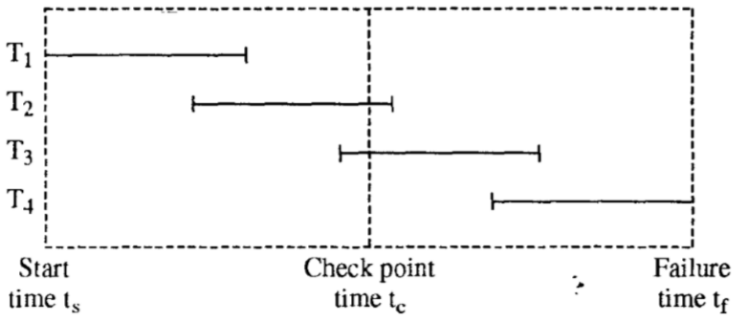
Titik pemeriksaan adalah titik sinkronisasi antara database dan file log transaksi. Semua tombol dipaksa ditulis ke penyimpanan sekunder di titik pemeriksaan. Check point juga disebut syncpoints atau save point. Kami menggunakan pos pemeriksaan untuk jumlah catatan log yang harus dipindai sistem saat pulih dari kerusakan.

Ketika terjadi kegagalan sistem, kita harus berkonsultasi dengan log untuk menentukan transaksi yang perlu dilakukan ulang & yang perlu dibatalkan. Itu perlu untuk mempertimbangkan hanya transaksi berikut selama pemulihan.

(1) Transaksi yang dimulai setelah pos pemeriksaan terakhir.

(2) Satu transaksi, jika ada, yang aktif pada saat check point terakhir.

Contoh:



Gambar 4.6 Check point Transaksi

Mari kita asumsikan bahwa log transaksi digunakan dengan pembaruan segera. Perhatikan juga bahwa timeline transaksi T_I, T_2, T_3 & T_4 ditunjukkan pada Gambar. Ketika sistem gagal pada waktu tr, log transaksi hanya perlu dipindai sejauh checkpoint terbaru dll. Transaksi T_1 tidak apa-apa , kecuali jika ada kegagalan disk yang menghancurkannya dan mungkin catatan lain sebelum titik pemeriksaan terakhir. Dalam hal ini, database dimuat ulang dari salinan cadangan yang dibuat pada titik pemeriksaan terakhir. Dalam kedua kasus, transaksi T_2 & T_3 diulang dari log transaksi, dan transaksi T_4 dibatalkan dari log transaksi.

4.10 DEADLOCKS

Deadlock : Deadlock adalah situasi dimana suatu proses atau sekumpulan proses diblokir, menunggu suatu kejadian yang tidak akan pernah terjadi. Kita dapat mengatakan, "menyatakan di mana tak satu pun dari transaksi dapat berjalan dengan eksekusi normalnya. Situasi ini disebut Deadlock".

Tabel 4.10 Contoh Tabel deadlock

T_1	T_2
Lock – X (B)	
read (B)	
B : = B – 50	
write (B)	
	Lock – S (A)
	read (A)
	lock – S (B)
Lock – X (A)	

Karena T_1 yang memegang kunci mode eksklusif di B & T_2 meminta kunci mode bersama di B, T_2 menunggu T_1 untuk membuka kunci B. Demikian pula, karena T_2 memegang kunci mode bersama di A & T_1 meminta kunci mode eksklusif di A, T_1 sedang menunggu T_2 untuk membuka kunci. Pada transaksi ini deadlock.

Deadlock Handling/Penanganan Deadlock

Ketika terjadi deadlock, sistem harus melakukan roll back pada transaksi tersebut. Setelah transaksi dibatalkan, item data yang dikunci oleh transaksi tersebut akan terbuka. Ada dua metode penting untuk menangani Deadlock.

- Pencegahan Deadlock.
- Deteksi & Pemulihan Deadlock.

Pencegahan Deadlock: Kita dapat menggunakan metode pencegahan deadlock untuk memastikan bahwa sistem tidak akan pernah masuk dalam keadaan deadlock. Ada dua pendekatan dasar untuk pencegahan Deadlock.

(i) **Satu pendekatan** memastikan bahwa tidak ada penantian siklik yang dapat terjadi dengan memesan permintaan untuk kunci, atau meminta semua kunci untuk diakuisisi bersama.

(ii) **Pendekatan lain** lebih dekat ke pemulihan Deadlock & melakukan rollback transaksi alih-alih menunggu kunci.

Ada dua skema pencegahan Deadlock yang berbeda menggunakan Timestamp.

(1) **Skema Wait-die** adalah teknik non-preemptive. Ketika transaksi Ti meminta item data yang saat ini dipegang oleh Tj, Ti diperbolehkan menunggu hanya jika memiliki stempel waktu yang lebih kecil dari Tj. Artinya Ti lebih tua dari Ti, sebaliknya Tj digulung kembali (mati). Contoh : Misalkan transaksi T_1, T_2 & T_3 memiliki timestamp masing-masing 5, 10 & 15. Jika T_1 meminta item data yang dipegang oleh T_2 maka T_1 akan menunggu. Jika T_3 meminta data .item yang dipegang oleh T_2, maka T_3 akan digulung kembali (Dies).

(2) **Skema wait-wound** adalah teknik preemptive. Ini adalah kebalikan dari skema wait-die. Ketika transaksi Ti meminta item data yang saat ini dipegang oleh Tj, Ti

diperbolehkan menunggu hanya jika memiliki stempel waktu yang lebih besar dari Tj. Artinya Ti lebih muda dari Tj, sebaliknya Tj terguling (Tj terluka oleh Ti). *Contoh* : Transaksi T_1, T_2 & T_3 memiliki stempel waktu masing-masing 5, 10 & 15. Jika T_1 meminta item data yang dipegang oleh T_2, maka item data yang akan di-preempt dari T_2 & T_2 akan di-roll back.

Jika T_3 meminta item data yang dipegang oleh T_2, maka T_3 akan menunggu.

a. Setiap kali sistem memutar kembali transaksi, penting untuk memastikan bahwa tidak ada kelaparan. **Catatan** : Baik skema Wait-die & Wound-wait menghindari kelaparan, setiap saat, ada transaksi dengan timestamp terkecil. Transaksi ini tidak dapat diminta untuk melakukan roll back di kedua skema tersebut.

Perbedaan antara skema Wait-die & Wound-wait:

(i) Skema Wait-die merupakan teknik non preemptive, sedangkan skema wound-wait merupakan teknik preemptive.

(ii) Dalam skema Wait-die, transaksi yang lebih tua harus menunggu yang lebih muda untuk menunggu, sedangkan dalam skema wound-wait, transaksi yang lebih lama tidak pernah menunggu transaksi yang lebih muda.

(iii) Dalam skema wait-die, jika sebuah transaksi Ti mati dan dibatalkan, maka Ti dapat menerbitkan kembali urutan permintaan yang sama ketika di-restart. Jika item data tersebut masih dipegang oleh Tj, maka Ti akan mati lagi.

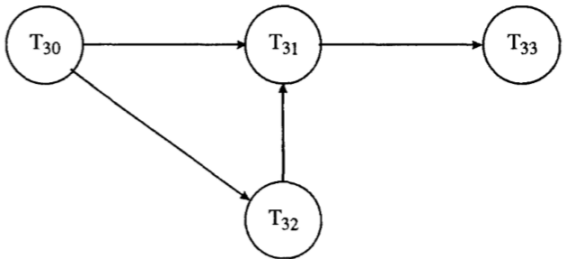
Dengan demikian Ti mungkin mati beberapa kali sebelum memperoleh item data yang dibutuhkan. Sedangkan dalam skema Wound-wait transaksi Ti terluka dan terguling karena Tj meminta item data yang dipegangnya. Ketika Ti di-restart & meminta item data yang sekarang dipegang oleh Tj, Ti menunggu. Dengan demikian, mungkin ada demam kembali dalam skema menunggu luka.

Deteksi dan Pemulihan Deadlock : Metode penting lainnya untuk penanganan Deadlock adalah deteksi Deadlock dan metode Pemulihan. Di mana sistem memeriksa apakah keadaan Deadlock benar-benar ada. Ada dua situasi yang muncul dalam metode ini: (i) Bagaimana kita mendeteksi Deadlock? (ii) Lalu bagaimana Recovery dari deadlock?

Deteksi Deadlock

Deadlock dapat dideskripsikan dalam bentuk graf berarah yang disebut "Grafik Tunggu". Graf ini terdiri dari himpunan simpul & sisinya adalah G = (V, E) Dimana V adalah himpunan titik & E adalah himpunan sisi. ''Sebuah Deadlock terjadi dalam sebuah sistem jika dan hanya jika grafik tunggu berisi sebuah siklus".

- Setiap transaksi yang terlibat dalam siklus tersebut dikatakan sebagai Deadlock.
- Untuk mendeteksi Deadlock, sistem perlu mempertahankan untuk menunggu grafik dan secara berkala untuk memanggil algoritma yang mencari siklus dalam grafik.

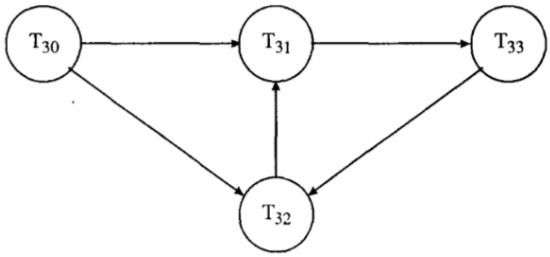


Gambar 4.7 Situasi Siklus pada Deadlock

Berikut situasi pada Gambar.

- Transaksi T_{30} menunggu transaksi T_{31} & T_{32}.
- Transaksi T_{32} menunggu transaksi T_{31}.
- Transaksi T_{31} menunggu transaksi T_{33}

Karena grafik tidak memiliki siklus, sistem tidak dalam keadaan deadlock. Misalkan sekarang transaksi T_{33} meminta item yang dipegang oleh T_{32}.



Gambar 4.7 Situasi Siklus pada Deadlock ketika Transaksi T_{33} meminta item yang dipegang oleh T_{32}.

Tepi $T_{33} \rightarrow T_{32}$ ditambahkan ke graf tunggu, menghasilkan graf Baru (2) berisi siklus. Karena Gambar (2) berisi siklus.

$T_{31} \rightarrow T_{33} \rightarrow T_{32} \rightarrow T_{31}$

Jadi ini menyiratkan bahwa aksi trem T_{31}, T_{32} & T_{33} semuanya menemui jalan buntu.

Pemulihan Deadlock

Ketika algoritma deteksi menentukan bahwa ada Deadlock, sistem harus pulih dari Deadlock. Ada dua solusi paling umum untuk memulihkan Deadlock.

(1) Batalkan transaksi satu per satu untuk memecahkan Deadlock.
(2) Batalkan semua transaksi yang berpartisipasi dalam Deadlock untuk memecahkan Deadlock.

Ada tiga tindakan yang perlu dilakukan.

(1) **Pemilihan Korban**: Dalam satu set transaksi deadlock, kita harus menentukan transaksi mana yang harus diputar kembali untuk memecahkan Deadlock. Kami harus mengembalikan transaksi yang akan kehilangan pemain minimum.
(2) **Roll back**: Setelah kami memutuskan bahwa transaksi tertentu harus dibatalkan. Kita harus menentukan seberapa jauh transaksi ini harus di-rollback? Solusi sederhananya adalah rollback total, Batalkan transaksi dan mulai ulang.
(3) **Starvation** : Starvation terjadi ketika suatu transaksi tidak dapat dilanjutkan untuk jangka waktu yang tidak terbatas sementara transaksi lain dalam sistem berlanjut

secara normal. Ini dapat terjadi jika skema menunggu untuk item yang dikunci tidak adil, memberikan prioritas pada beberapa transaksi di atas yang lain. Solusi kelaparan lainnya untuk memungkinkan beberapa transaksi memiliki prioritas di atas yang lain tetapi meningkatkan prioritas transaksi semakin lama menunggu, hingga akhirnya mendapatkan prioritas & hasil tertinggi.

4.11 KONSEP PHANTOM DEADLOCK

Deadlock yang terdeteksi tetapi tidak benar-benar kebuntuan disebut Deadlock Phantom.

- Pembatalan otomatis dapat menyebabkan Deadlock ini.
- Deadlock phantom terjadi ketika pengamat eksternal dapat melihat Deadlock yang sebenarnya tidak ada.

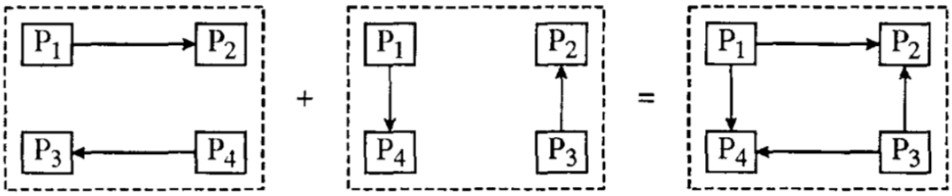
Berikut empat syaratnya.

- R_I disimpan di S_I
- R_2 disimpan di S_2
- T_I berjalan di S_3
- T_3 berjalan di S_4

Dua transaksi berjalan bersamaan

T_2 : lock R_I	T_2 : unlock	T_2 : lock R_2.
T_1 : lock R_I	T_1 : unlock R_I	T_I : unlock R_2
T_2 : kunci R_2	T_I : kunci R_2	

False Deadlock: Bagaimana jika setiap situs mempertahankan pandangan lokalnya tentang status sistem, dan mengirimkannya secara berkala ke situs kontrol?



Gambar 4.12 Konsep Phantom Deadlock

4.12 PENYELESAIAN MASALAH

Pertanyaan 1: Jelaskan mengapa eksekusi transaksi harus atomik. Jelaskan sifat ACID, perhatikan transaksi berikut.

```
Ti :    read (A);
        A: = A- 50;
        write (A);
        read (B);
        B: = B + 50;
        write (B)
```

Jawaban :

Mempertimbangkan bahwa transaksi Ti memulai eksekusi ketika database dalam keadaan konsisten. Jika atomisitas transaksi Ti tidak dipastikan, maka -kegagalannya di tengah

pelaksanaannya dapat membuat database dalam keadaan tidak konsisten. Jadi transaksi harus atom. Penjelasan properti ACID sehubungan dengan transaksi berikut :

```
Ti :        read (A);
            A: = A-50;
            write (A);
            read (B);
            B: = B + 50;
            write (B);
```

(i) **Consistency** : Syarat konsistensi disini adalah jumlah A & B tidak berubah oleh eksekusi transaksi yaitu jika database konsisten sebelum eksekusi transaksi, maka database tetap konsisten setelah eksekusi transaksi .

(ii) **Atomicity**: Misalkan, sesaat sebelum eksekusi transaksi Ti nilai Account A & Bare Rs 2000 & Rs 3000 masing-masing. Misalkan kegagalan terjadi setelah operasi write (A) tetapi sebelum operasi nilai write (B). Dalam hal ini, nilai rekening A & B adalah Rp. 1.950 & Rp. 3.000 masing-masing. Sistem menghancurkan Rp. 50 sebagai akibat dari kegagalan ini. Dengan demikian keadaan seperti itu adalah keadaan yang tidak konsisten. Kita harus memastikan bahwa inkonsistensi tersebut tidak terlihat dalam sistem database. Itulah alasan untuk persyaratan atomisitas. Jika atomisitas hadir, semua tindakan transaksi tercermin dalam database, atau tidak ada.

(iii) **Durability** : Properti Durability menjamin bahwa, setelah transaksi berhasil diselesaikan, semua pembaruan tidak hilang, bahkan jika ada kegagalan sistem. Jika transaksi Ti berhasil dilakukan, maka jumlah A & B sebelum eksekusi dan setelah eksekusi selalu sama (sama).

(iv) **Isolasi**: Properti isolasi dari suatu transaksi memastikan bahwa eksekusi transaksi secara bersamaan menghasilkan status sistem yang setara dengan status yang dapat diperoleh jika transaksi ini dieksekusi satu per satu dalam urutan tertentu.

Pertanyaan 2: transaksi gagal & masuk ke keadaan dibatalkan. Sebutkan kondisi di mana kita dapat memulai kembali transaksi & mematikan transaksi.

Jawaban : Suatu transaksi memasuki keadaan gagal setelah sistem menentukan bahwa transaksi tersebut tidak dapat lagi dilanjutkan dengan eksekusi normalnya. Transaksi seperti itu harus dibatalkan. Kemudian, ia memasuki keadaan dibatalkan. Pada titik ini, sistem memiliki dua pilihan:

(i) **Restart**: Ini dapat memulai kembali transaksi, tetapi hanya jika transaksi dibatalkan sebagai akibat dari beberapa kesalahan perangkat keras atau perangkat lunak yang tidak dibuat melalui logika internal transaksi. Transaksi yang dimulai kembali dianggap sebagai transaksi baru.

(ii) **Kill** : Dapat mematikan transaksi. Itu biasanya. melakukannya karena beberapa kesalahan logika internal yang dapat diperbaiki hanya dengan menulis ulang program aplikasi, atau karena inputnya buruk, atau karena data yang diinginkan tidak ditemukan dalam database. misalnya., Misalkan pengembang program telah menulis sebuah program untuk menemukan jumlah dua rekening tetapi

setelah eksekusi transaksi hasilnya adalah perkalian dua rekening, maka dalam situasi seperti itu transaksi akan dihentikan.

Pertanyaan 3: Perhatikan transaksi berikut :

T_I :	*read (A);*
	read (B);
if A =0 then	*B:=B+1*
	write (B);
	T_2: read (B);
	read (A);
if B = 0 then	*A: A = 1;*
	write (A);

Tambahkan instruksi kunci dan buka kunci ke transaksi T_1 dan T_2, sehingga mereka mematuhi protokol penguncian dua fase.

Jawaban : Lock -X = Exclusive lock

Lock -s = Shared lock

Tabel 4.11 Tabel Pengamatan Jawaban soal nomor 3

T_1	T_2
Lock – S (A)	
Lock – X (B)	
read (A)	
read (B)	
if A = 0 then B : = B + 1	
write (B)	
Unlock – S (A)	
Unlock – X (B)	
	Lock – S (B)
	Lock – X (A)
	read (A)
	read (B)
	if B = 0 then A : = A + 1
	write (A)
	Unlock – X (A)
	Unlock – S (B)

Pertanyaan 4: Perhatikan dua transaksi berikut

T_1 :	*read (A)*
	read (B)
if A = 0 then	*B : = B + 1*
	write (B)
	T_2 : read (B)
	read (A)
if B: = 0 then	*A:=A+I*
	write (A)

Biarkan persyaratan konsistensi menjadi A = 0 V B = 0 dengan A = B = 0 nilai awal.

(i) Tunjukkan bahwa setiap eksekusi serial yang melibatkan dua transaksi ini menjaga konsistensi

(ii) Tunjukkan eksekusi bersamaan T_I & T_2 yang menghasilkan jadwal yang tidak dapat serial.

(iii) Apakah ada eksekusi T_I & T_2 yang menghasilkan jadwal serial?

Jawaban :

(i) Eksekusi serial transaksi T_1 diikuti T_2 = 1 dan A = 0, yang menjaga konsistensi database dan eksekusi serial transaksi T_2 diikuti T_1 menghasilkan B = 0 & A = 1, yang menjaga konsistensi database.

(ii) Perhatikan jadwal berikut yang memiliki pelaksanaan serentak T_1 & T_2

Tabel 4.11 Tabel Pengamatan Jawaban soal nomor 4

T_1	T_2
read (A)	
	read (B)
read (B)	
if A = 0 then	
B : = B + 1	
write (B)	
	read (A)
	if B = 0 then
	A : = A + 1
	write (A)

Jika semua instruksi T_2 bertukar sebelum membaca (A) instruksi Tl maka penulisan (A) dari T_2 bertentangan dengan pembacaan (A) dari T_1. Sehingga kita tidak dapat menghasilkan jadwal serial T_2 diikuti oleh T_1 serta jika semua instruksi T_1 bertukar sebelum membaca (B) instruksi T_2 kemudian menulis (B) dari T 1 bertentangan dengan membaca (B) dari T_2. Sehingga kami tidak dapat menghasilkan jadwal serial T_1 diikuti oleh T_2. Oleh karena itu eksekusi T 1 & T 2 secara bersamaan di atas menghasilkan jadwal yang tidak dapat serial.

(iii) Tidak, tidak ada eksekusi serentak T_1 & T_2 yang menghasilkan jadwal serial, karena tidak menghasilkan jadwal serial.

Pertanyaan 5: Sebutkan apakah jadwal berikut ini konflik bersambung atau tidak. Justifikasi jawaban Anda.

Tabel 4.12 Tabel soal nomor 5

T_1	T_2
read (A)	
write (A)	
	read (B)
	write (B)
read (B)	
write (B)	
	read (A)
	write (A)

Penjelasan: Pada jadwal ini, penulisan (B) dari T_2 bertentangan dengan Baca (A) dari T_1, sedangkan penulisan (A) dari T_1 tidak bertentangan dengan Read (B) dari T_2, karena kedua instruksi mengakses data yang berbeda item. Jadi kita bisa menukar instruksi yang tidak bertentangan.

- Tukar write (A) dari T_1 dengan Read (B) dari T_2
- Tukar write (B) dari T_1 dengan Read (A) dari T_2

Setelah Tukar:

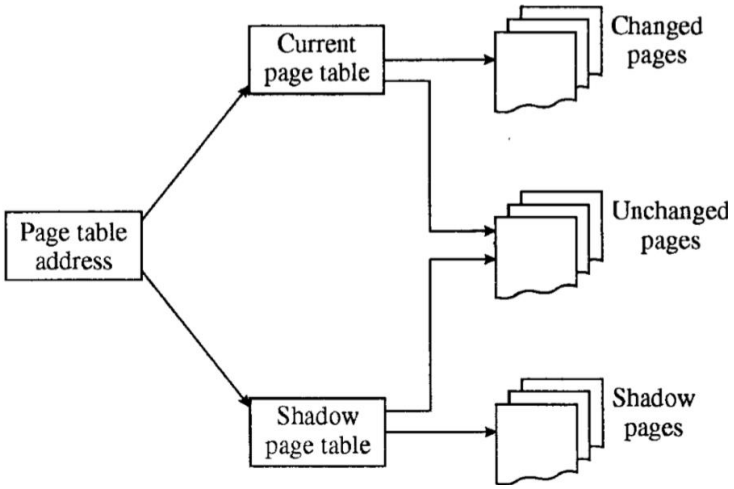
Tabel 4.13 Tabel Hasil Penjabaran dari Soal No 5

T_1	T_2
read (A)	
	read (B)
wtite (A)	
	write (B)
read (B)	
	read (A)
write (B)	
	write (A)

Dengan demikian kita dapat mengatakan jadwal di atas adalah konflik serializable.

Pertanyaan 6 : Write catatan singkat tentang shadow paging.

Jawaban: Shadow paging diperkenalkan oleh Lorie pada tahun 1977 sebagai alternatif skema pemulihan berbasis log. Teknik shadow paging tidak memerlukan penggunaan log transaksi di lingkungan pengguna tunggal.



Gambar 4.13 Skema Shadow Paging

Dalam skema halaman bayangan, database dianggap terdiri dari unit logis penyimpanan halaman disk ukuran tetap (atau blok disk). Halaman dipetakan ke dalam blok penyimpanan fisik melalui tabel halaman, dengan satu entri untuk setiap halaman logis dari database. Entri ini berisi nomor blok penyimpanan fisik tempat halaman ini disimpan. Dengan demikian, skema shadow paging adalah salah satu kemungkinan bentuk alokasi halaman tidak langsung. Teknik shadow paging mempertahankan dua tabel halaman selama masa transaksi yaitu:

(i) Tabel halaman saat ini

(ii) Tabel halaman bayangan. Halaman bayangan adalah tabel halaman asli dan alamat transaksi database menggunakan tabel halaman saat ini. Pada jenis transaksi, kedua tabel sama dan keduanya menunjuk ke blok penyimpanan fisik yang sama. Tabel halaman bayangan tidak pernah diubah setelahnya dan digunakan untuk memulihkan database jika terjadi kegagalan sistem.

Namun, entri tabel halaman saat ini dapat berubah selama pelaksanaan transaksi. Tabel halaman saat ini digunakan untuk merekam semua pembaruan ke database. Ketika transaksi selesai, tabel halaman saat ini menjadi tabel halaman bayangan.

Keuntungan dari shadow paging:

(i) Biaya pemeliharaan file log transaksi dihilangkan.

(ii) tidak perlu untuk membatalkan atau mengulang operasi, pemulihan secara signifikan lebih cepat.

Kerugian dari shadow paging:

(i) Fragmentasi atau hamburan data.

(ii) Perlunya pengumpulan sampah secara berkala untuk mendapatkan kembali blok-blok yang tidak dapat diakses

4.13 REVIEW PERTANYAAN

1. Menjelaskan pengertian transaksi. Menggambar dan menjelaskan diagram tahapan transaksi.
2. Jelaskan sifat ACID dari transaksi? Jelaskan serializability dengan contoh yang sesuai.
3. Jelaskan mengapa eksekusi transaksi harus atomik. Jelaskan sifat ACID dengan memperhatikan transaksi berikut.

```
Tl :    read (A);
        A: =A - 50;
        write (A);
        read (B);
        B: =B + 50;
        write (B);
```

4. Transaksi gagal dan masuk ke status dibatalkan. Sebutkan kondisi di mana kita dapat memulai kembali transaksi dan mematikan transaksi.
5. Mengapa pelaksana sistem Database menaruh banyak perhatian pada properti ACID?
6. Apakah yang Anda maksud: serializability Bedakan antara serializability konflik dan lihat jadwal serializability.
7. Apakah yang Anda maksud: transaksi Apa yang Anda maksud dengan atomisitas transaksi? Jelaskan jenis-jenis kegagalan yang dapat menyebabkan aborsi transaksi. Jelaskan fitur yang menonjol dari modifikasi database yang ditangguhkan dan skema modifikasi database langsung untuk pemulihan.
8. Apakah yang Anda maksud: jadwal Kapan jadwal disebut serializable? Apa itu jadwal serialisasi konflik? Tunjukkan, apakah jadwal berikut ini ekuivalen dengan konflik atau tidak.

T_1	T_2
R(A)	
W(A)	
	R(B)
	W(B)
R(C)	
W(C)	

Petunjuk: Sebuah jadwal dikatakan serializable, lebih dari satu set S transaksi berkomitmen yang efeknya pada setiap instance database konsistensi adalah gunintee-d identik dengan beberapa jadwal serial lengkap di atas S. Itu adalah instance database yang dihasilkan dari eksekusi jadwal yang diberikan identik dengan instance database yang dihasilkan dari eksekusi transaksi dalam beberapa urutan serial.

9. Nyatakan kondisi ketika dua jadwal dianggap setara dengan tampilan. Nyatakan apakah jadwal berikut dapat dilihat secara serial. Justifikasi jawaban Anda.

T	T_2	T_3
Read(A)		
	write(A)	
write(A)		
		write(A)

10. Bagaimana informasi pos pemeriksaan digunakan dalam operasi pemulihan setelah sistem macet?
11. Tunjukkan bagaimana teknik pemulihan kesalahan mundur diterapkan ke DBMS?
12. Jelaskan cara kerja dari lock manager
13. Apa itu Deadlock? Bagaimana Deadlock terdeteksi? Hitung metode untuk pemulihan dari Deadlock.
14. Jelaskan serialisasi? Kapan konflik jadwal dapat diserialisasi? Apa yang dapat dipulihkan? Apa itu jadwal cascadeles?
15. Apa itu Deadlock? Bagaimana Deadlock dapat dihindari?
16. Jelaskan teknik wait_die dan wound_wait untuk pencegahan deadlock.
17. Apa perbedaan antara teknik wait_die dan wound_wait untuk pencegahan Deadlock?
18. Apa itu grafik tunggu? Di mana itu digunakan? Jelaskan dengan sebuah contoh. 19. Diskusikan berbagai jenis kegagalan transaksi yang mungkin terjadi di lingkungan database.
19. Apa itu pemulihan Database? Apa yang dimaksud dengan pemulihan ke depan dan ke belakang? Jelaskan dengan sebuah contoh.
20. Apa itu pos pemeriksaan? Bagaimana informasi pos pemeriksaan digunakan dalam operasi pemulihan setelah sistem macet?
21. Apa saja jenis kerusakan yang dapat terjadi pada database? Jelaskan.
22. Berapa banyak teknik yang digunakan untuk pemulihan dari kegagalan non-fisik atau transaksi?

    Jawaban: Dua teknik berikut digunakan untuk pemulihan dari kegagalan non-fisik atau transaksi:

    a. Pembaruan yang ditangguhkan
    b. Pembaruan segera
23. Jelaskan fitur yang menonjol dari modifikasi Database yang ditangguhkan dan skema modifikasi Database langsung dari pemulihan.

    Jawaban: **Modifikasi Database Ditangguhkan (Pembaruan):** Dalam hal teknik pembaruan yang ditangguhkan, pembaruan tidak ditulis ke Database sampai setelah transaksi mencapai titik COMMIT. Dengan kata lain, Database pembaruan ditangguhkan (atau ditunda) hingga transaksi menyelesaikan eksekusinya dengan sukses dan mencapai titik komitnya. Selama eksekusi transaksi, pembaruan dicatat hanya di log transaksi dan di buffer cache. Setelah transaksi mencapai titik komit dan log transaksi ditulis secara paksa ke disk, pembaruan dicatat dalam database. Jika transaksi gagal sebelum mencapai titik ini, itu tidak akan mengubah database dan

tidak perlu membatalkan perubahan. Dalam hal pembaruan yang ditangguhkan, file log transaksi digunakan dengan cara berikut:

(i) Ketika transaksi T dimulai, transaksi dimulai (atau < T, BEGIN)) ditulis ke log transaksi.

(ii) Selama eksekusi transaksi T, catatan log baru yang berisi semua data log yang ditentukan sebelumnya.

(iii) Kapan semua tindakan yang terdiri dari transaksi T berhasil dilakukan, kami mengatakan bahwa transaksi T sebagian dilakukan dan catatan "<T, COMMIT>" ditulis ke log transaksi. Setelah transaksi T sebagian dilakukan, catatan yang terkait dengan transaksi T di log transaksi digunakan dalam mengeksekusi pembaruan aktual dengan menulis ke catatan yang sesuai dalam database

(iv) Jika transaksi T dibatalkan, catatan log transaksi diabaikan untuk transaksi T dan penulisan tidak dilakukan.

Pembaruan Segera: Dalam hal teknik pembaruan segera, semua pembaruan ke database diterapkan segera saat terjadi tanpa menunggu untuk mencapai titik COMMIT dan catatan semua perubahan disimpan dalam log transaksi. Dalam kasus pembaruan segera, file log transaksi digunakan dengan cara berikut:

(i) Ketika transaksi T dimulai, transaksi dimulai (atau "<T, BEGIN>") ditulis ke log transaksi.

(ii) Ketika operasi tulis dilakukan, catatan yang berisi data yang diperlukan ditulis ke file log transaksi.

(iii) Setelah log transaksi ditulis, pembaruan ditulis ke buffer database.

(iv) Pembaruan database itu sendiri ditulis ketika buffer selanjutnya di-flush (ditransfer) ke penyimpanan sekunder.

(v) Ketika transaksi T melakukan, transaksi commit (" <T, COMMIT>") catatan transaksi ditulis ke log transaksi.

(vi) Jika log transaksi mengungkapkan catatan "<T, BEGIN>" tetapi tidak mengungkapkan "< T, COMMIT>", transaksi T dibatalkan. Nilai lama dari item data yang terpengaruh dipulihkan dan transaksi T dimulai ulang.

(vii) Jika log transaksi berisi kedua catatan sebelumnya, transaksi T diulang. Transaksi tidak dimulai ulang.

BAB 5
TEKNIK KONTROL KONKURENSI

- Ketika beberapa transaksi dieksekusi secara bersamaan dalam database disebut eksekusi transaksi secara bersamaan.
- Ketika banyak transaksi dijalankan secara bersamaan dalam database, namun properti isolasi mungkin gagal, oleh karena itu sistem harus mengontrol interaksi antara transaksi saat ini. Kontrol ini dicapai dengan mekanisme yang disebut mekanisme kontrol konkurensi.

5.1 TEKNIK MENGUNCI UNTUK KONTROL CONCURRENCY

Kunci

Kunci adalah variabel yang terkait dengan item data yang menggambarkan status item sehubungan dengan kemungkinan operasi yang dapat diterapkan pada transaksi. • Protokol penguncian adalah seperangkat aturan yang menyatakan kapan suatu transaksi dapat mengunci atau membuka kunci setiap item data dalam database. • Setiap transaksi harus mengikuti aturan berikut :

(i) Sebuah transaksi T harus mengeluarkan operasi Lock-item (A) sebelum operasi read-item (A) atau write-item (A) dilakukan di T.

(ii) A transaksi T harus mengeluarkan operasi unlock-item (A) setelah semua operasi read-item (A) dan write-item (A) selesai di T.

(iii) Transaksi T tidak akan mengeluarkan operasi lock-item (A) jika sudah memegang kunci pada item (A).

Aturan ini diberlakukan oleh manajer kunci antara item kunci (A) & item buka kunci (A). Ada banyak mode di mana item data dapat dikunci.

(i) **Shared Mode (S**) : Jika suatu transaksi Ti telah memperoleh shared mode lock pada item data (A), maka Ti dapat membaca, tetapi tidak dapat menulis A. Shared mode dilambangkan dengan S.

(ii) **Exclusive-Mode (X)** : Jika suatu transaksi Ti telah memperoleh kunci mode eksklusif pada item A, maka Ti dapat membaca & menulis A. Mode eksklusif dilambangkan dengan X.

Tabel 5.1 Matriks kompatibilitas kunci.

	S	X
S	True	False
X	False	False

Sebuah transaksi meminta kunci bersama pada item data A dengan mengeksekusi instruksi lock-S(A). Demikian pula, transaksi meminta kunci eksklusif melalui instruksi lock-X(A). Misalnya :

Tl : lock-X (B);

```
        read (B);
        B: = B - 50;
        write (B)
        unlock (B);
        lock-X (A);
        read (A);
        A: = A + 50;
        write (A);
        unlock (A);
T 2 :   lock-S (A);
        read (A);
        unlock (A);
        lock-S (B)
        read (B);
        unlock (B);
```

Protokol Penguncian Dua Fase

Suatu transaksi dikatakan mengikuti protokol penguncian dua fase jika semua operasi penguncian mendahului operasi pembukaan kunci pertama dalam transaksi. Protokol ini dibagi menjadi dua fase.

- **Fase Tumbuh**: Suatu transaksi dapat memperoleh kunci baru, tetapi tidak dapat melepaskan kunci apa pun.
- **Fase Menyusut** : Sebuah transaksi dapat mengaktifkan kembali penguncian tetapi tidak dapat memperoleh penguncian baru.

Kondisi Protokol Locking 2 Fase:

Sebuah transaksi sedang dalam fase pertumbuhan. Transaksi memperoleh kunci sesuai kebutuhan. Setelah transaksi melepaskan kunci, ia memasuki fase menyusut dan tidak dapat mengeluarkan permintaan penguncian lagi. Ada berbagai versi protokol penguncian dua fase.

1. **Locking 2 fase dinamis :** Di sini transaksi mengunci item data segera sebelum operasi apa pun diterapkan pada item data. Setelah menyelesaikan semua operasi pada semua item d ta, itu melepaskan semua kunci. Contoh:

```
T :     lock-X(A);
        read (A);
        A: = A-50;
        write (A);
        lock-X(B);
        read (B);
        B: = B + 50;
        write (B);
        unlock (A);
        unlock (B);
```

2. **Locking dua fase statis (Konservatif):** Dalam ~cheme ini, semua item data~ dikunci sebelum operasi apa pun pada item tersebut dan dilepaskan setiap perubahan operasi terakhir yang dilakukan pada item data apa pun. Contoh:

```
T :     lock-X(A);
        read (A);
        A: = A-50;
        write (A);
        lock-X(B);
        read (B);
        B: = B + 50;
        write (B);
        unlock (A);
        unlock (B);
```

3. **Locking dua fase yang ketat:** Dalam ilmu komputer, penguncian 2-PH yang ketat adalah metode penguncian yang digunakan dalam sistem bersamaan.

Dua aturan dari 2PL yang ketat adalah:

(i) Jika transaksi T ingin membaca/menulis suatu objek, ia harus meminta kunci bersama/eksklusif pada objek tersebut.

(ii) Semua kunci eksklusif yang dipegang oleh transaksi T dilepaskan ketika T melakukan atau membatalkan (& tidak sebelumnya).

Tabel 5.2 Tabel Aturan 2PL

T_1	T_2
Lock – S (A)	
read (A)	
unlock (A)	
	Lock – S (A)
	read (A)
	Lock – X (B)
	read (B)
	write (B)
	unlock (A)
	unlock (B)

2PL yang ketat mencegah transaksi membaca data yang tidak dikomit, menimpa data yang tidak dikomit, dan pembacaan yang tidak dapat diulang. Dengan demikian mencegah cascading roll back karena kunci eksklusif harus dipegang sampai komit transaksi.

4. **Locking 2P yang Ketat**: Yang mengharuskan semua penguncian ditahan sampai transaksi dilakukan. Kami dapat dengan mudah memverifikasi bahwa, yang penguncian dua fase yang ketat, transaksi dapat diserialkan dalam urutan di mana mereka melakukan.

Tabel 5.3 Penguncian 2 Fase

T_1	T_2
Lock – X (A)	
read (A)	
A : = A + 50	
write (A)	
unlock (A)	
	Lock – X (A)
	read (A)
	temp : = A * 0.3
	A = A + temp
	write (A)
	unlock (A)

5.2 KONTROL CONCURRENCY BERDASARKAN PROTOKOL TIMESTAMP

Timestamp adalah pengidentifikasi unik yang dibuat oleh DBMS untuk mengidentifikasi transaksi.

Nilai stempel waktu ditetapkan dalam urutan transaksi yang dikirimkan ke sistem, sehingga stempel waktu dapat digunakan sebagai waktu mulai. Teknik ini tidak menggunakan kunci, sehingga tidak dapat terjadi deadlock. Algoritme mengaitkan dengan setiap item Database X dua nilai stempel waktu (TS).

(1) **Read-TS(X) :** Stempel waktu baca item X. Ini adalah stempel waktu transaksi yang berhasil membaca item X.
Misalnya:

$$\text{read-TS(X)} = \text{TS (T)}$$

dimana T adalah transaksi termuda yang berhasil membaca X.

(2) **Write-TS(X) :** Timestamp write item X. Ini adalah timestamp terbesar dari semua transaksi yang berhasil menulis item X.
Misalnya:

$$\text{Write-TS (X)} = \text{TS (T)}$$

dimana T adalah transaksi termuda yang berhasil menulis X.

- Setiap kali beberapa transaksi T mengikat untuk mengeluarkan operasi read-item (X) atau write-item (X), dasar algoritma membandingkan timestamp T dengan read-TS (X) & write-TS(X) untuk memastikan bahwa urutan waktu pelaksanaan transaksi tidak dilanggar.
- Setiap kali operasi yang bertentangan melanggar urutan stempel waktu dalam dua kasus berikut.

Transaksi Ti mengeluarkan operasi read (X):

(a) Jika write - TS(X) > TS(T), Maka T perlu membaca nilai X yang sudah ditulis sebelumnya. Oleh karena itu, operasi read ditolak & T dibatalkan.

(b) Jika write-TS(X):5 TS(T), Kemudian operasi read dijalankan & read-TS(X) diatur ke TS(T) yang lebih besar dan read-TS(X) saat ini .

Aturan Read

```
if (TS (Ti) ≥ W-TS(X))
        {
                R-TS(X) = MAX (R-TS(X), TS(i)));
                return OK;
        }
        else
        {
                return REJECT;
        }
```

Transaksi T mengeluarkan operasi write (X):

a) Jika read-TS(X) > TS(T)
    a. Maka nilai X yang dihasilkan T diperlukan sebelumnya, dan sistem mengasumsikan bahwa, nilai tersebut akan tidak akan pernah diproduksi. Oleh karena itu, sistem 'REJECT' operasi write & 'Roll back' T.
b) Jika write-TS(X) > TS(T),
    a. Kemudian T mencoba untuk menulis nilai absolut dari X. Oleh karena itu sistem 'REJECT' operasi write ini dan rollback T.
c) Jika tidak, sistem akan mengeksekusi operasi write dan Atur write-TS(X) ke TS(T).

Aturan Write:

```
if (TS(Ti)      ≥ W-TS (X) &&
                TS(Ti) ~ R-TS(X))
        {
                W-TS(X) = MAX (W-TS(X), TS(i));
                return OK;
        }
        Else
        {
                return REJECT;
                }
```

Contoh 1 :

```
T1      read (B);
        read (A);
        display (A + B);
T2      read (B);
        B: B - 50;
        write (B);
        read (A);
        A: = A + 50;
        display (A + B);
```

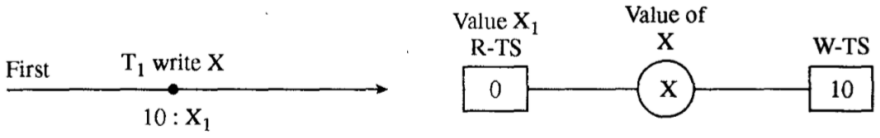
Dalam jadwal di bawah protokol stempel waktu ini, kami akan mengasumsikan bahwa transaksi diberi stempel waktu segera sebelum instruksinya.

Tabel 5.4 Tabel Aturan Read

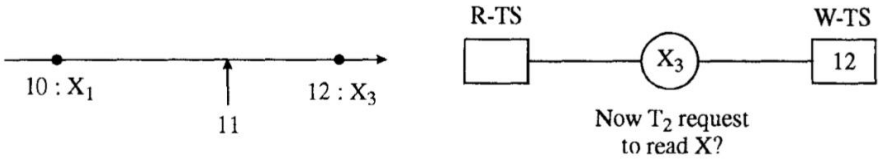
T_1	T_2
read (B)	
	read (B)
	B : = B – 50
	write (B)
read (A)	
	read (A)
display (A + B)	
	A : = A + 50
	write (A)
	display (A + B)

Jadi dalam jadwal ini TS(T_1) < TS (T_2) dan jadwal tersebut dimungkinkan di bawah protokol Timestamp. Contoh 2: T_I : W(X) T_2 : R(X) T_3: W(X)

- T_1 Mulai, diberikan TS 10 .... Kemudian T_2 Mulai, diberikan TS 11.
- beberapa saat kemudian, T3 dimulai, diberikan TS12.



Kemudian, T_3 over menulis X dengan X_3



Keuntungan dari protokol timestamp:

- Bebas deadlock karena tidak ada transaksi yang menunggu.
- Menghindari manajer kunci kontrol.
- Menarik dalam sistem database terdistribusi.

Kekurangan protokol timestamp:

- T abort/restart dapat menurunkan performa di bawah persaingan tinggi.
- Overhead penyimpanan per item data untuk stempel waktu.
- Perbarui overhead untuk mempertahankan stempel waktu.
- Berpotensi memperbarui selama setiap membaca/menulis ke item data.

Catatan : Protokol urutan stempel waktu memastikan serializabilitas konflik, karena operasi diproses dalam urutan stempel waktu.

5.3 PROTOKOL BERBASIS VALIDASI (OPTIMIS)

Skema validasi adalah metode kontrol konkurensi dalam kasus di mana sebagian besar transaksi adalah transaksi hanya baca dan, dengan demikian, tingkat konflik di antara transaksi ini rendah.

- Teknik validasi atau sertifikasi juga dikenal sebagai teknik kontrol konkurensi optimis.

- Dalam teknik validasi concurrency control, tidak ada pengecekan saat transaksi sedang dieksekusi. Selama eksekusi transaksi, semua pembaruan diterapkan ke salinan lokal dari item data yang disimpan untuk transaksi.
- Jika serializability tidak dilanggar, transaksi dilakukan dan database diperbarui dari salinan lokal; jika tidak, transaksi dibatalkan dan dimulai kembali nanti.

Ada tiga fase untuk protokol kontrol konkurensi validasi.

(1) **Fase Read:** Suatu transaksi dapat membaca nilai-nilai yang dikomit. Namun, pembaruan hanya diterapkan pada salinan lokal item data yang disimpan di ruang kerja transaksi.

(2) **Fase validasi:** Pengecekan dilakukan untuk memastikan bahwa serializability tidak akan dilanggar jika pembaruan transaksi diterapkan ke database.

(3) **Fase Write:** Jika fase validasi berhasil, pembaruan transaksi diterapkan ke database, sebaliknya, pembaruan dibuang dan transaksi dimulai kembali.

Untuk melakukan uji validasi, kita perlu mengetahui kapan berbagai fase transaksi T berlangsung. Ada tiga stempel waktu yang berbeda dengan transaksi T.

1) **Start** (T): Waktu saat transaksi T memulai eksekusinya.
2) **Validation** (T): Waktu ketika transaksi T menyelesaikan fase read dan memulai fase validasi.
3) **Finish** (T): Waktu transaksi T menyelesaikan fase write-nya. Uji validasi untuk transaksi Tj mensyaratkan bahwa, untuk semua transaksi Ti yang TS(Ti) < TS (Tj) , salah satu dari dua kondisi berikut harus dipenuhi
    a. Finish (Ti) < start (Tj) Karena Ti mengkompilasi eksekusinya sebelum Tj dimulai.
    b. Mulai (Tj) < Selesai (Ti) < Validasi (Tj) Kondisi ini memastikan penulisan Ti & Tj tidak tumpang tindih; karena penulisan Ti tidak mempengaruhi pembacaan Tj.

Tabel 5.5 Contoh Protokol Berbasis Validasi

T_1	T_2
read (B)	
	read (B)
	B : = B – 50
	read (A)
	A : = A + 50
read (A)	
<validate>	
display (A + B)	
	<validate>
	write (B)
	write (A)

Keuntungan Metode Optimis

Kontrol konkurensi optimis memiliki keuntungan sebagai berikut:

- Teknik ini sangat efisien ketika konflik jarang terjadi. Konflik sesekali mengakibatkan transaksi mundur.
- Rollback hanya melibatkan salinan data lokal, database tidak terlibat dan dengan demikian tidak akan ada rollback berjenjang.

Masalah Metode Optimis

Control concurrency optimis memiliki masalah berikut:

- Konflik mahal untuk menangani, karena transaksi yang bertentangan harus dibatalkan.
- Transaksi yang lebih lama lebih cenderung memiliki konflik dan mungkin berulang kali dibatalkan karena konflik dengan transaksi pendek.

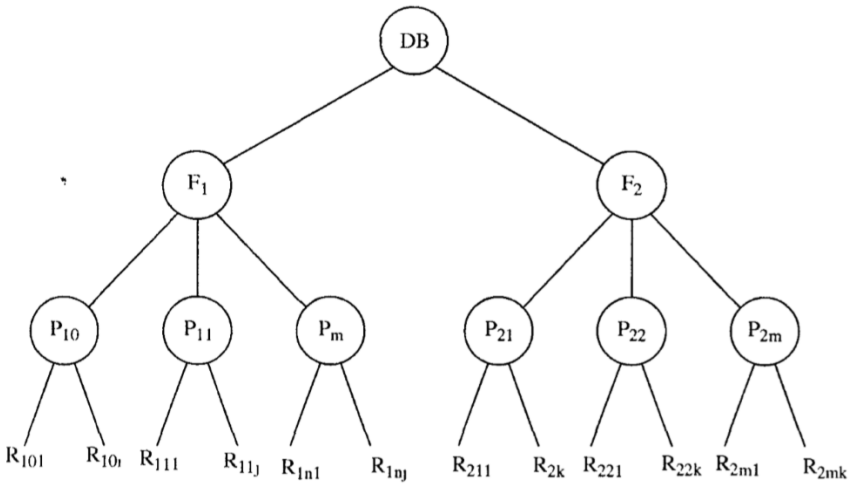
Penerapan Metode Optimis

- Hanya cocok untuk lingkungan di mana ada sedikit konflik dan tidak ada transaksi yang lama.
- Dapat diterima untuk sebagian besar sistem Database Baca atau Kueri yang memerlukan sangat sedikit transaksi pembaruan.

5.4 MENGUNCI GANDA GRANULARITAS

- Dalam ilmu komputer, penguncian granularitas ganda, kadang-kadang disebut metode penguncian 'John Rayner', adalah metode penguncian yang digunakan dalam DBMS & RDBMS.
- Dalam penguncian granularitas ganda, kunci adalah kumpulan objek yang berisi objek. MGL mengeksploitasi sifat hierarkis dari hubungan berisi.
- Item database dapat dipilih salah satu dari berikut ini:
    - Sebuah catatan database.
    - Nilai bidang dari database/catatan.
    - Seluruh file.
    - Seluruh Database.
    - Sebuah blok disk.
- Ukuran item data sering disebut item data granularity. Granularity halus berarti ukuran item kecil sedangkan granularity kasar berarti ukuran item besar. Contoh : Sebuah database mungkin memiliki file, yang berisi halaman, yang selanjutnya berisi catatan. Ini dapat dianggap sebagai pohon objek, di mana setiap node berisi anak-anaknya. Kunci mengunci simpul dan turunannya.

Hirarki Granularitas



Gambar 5.1 Hirarki Granularitas

- Penguncian granularitas ganda biasanya digunakan dengan penguncian dua fase Non-Strict untuk menjamin serializability. Di mana kunci dapat diminta pada suatu level.

Ada tiga jenis Intention lock.

- Intention-shared (IS) menunjukkan bahwa shared lock (S) akan diminta pada beberapa node turunan (S).
- Intention-exclusive (IX) menunjukkan bahwa pada exclusive lock (X) akan diminta pada beberapa node turunan (S).
- Shared-Intention-exclusive (SIX) menunjukkan bahwa node saat ini terkunci dalam mode bersama tetapi kunci eksklusif (S) akan diminta pada beberapa node turunan (S).

Tabel 5.6 Tabel Intention Lock

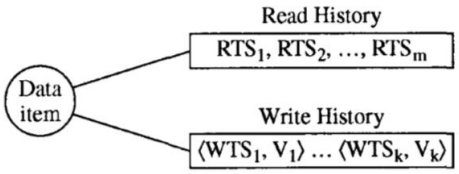
	IS	IX	S	SIX	X
IS	Yes	Yes	Yes	Yes	No
IX	Yes	Yes	No	No	No
S	Yes	No	Yes	No	No
SIX	Yes	No	No	No	No
X	No	No	No	No	No

Untuk mengunci node di (S atau X), MGL memiliki kunci transaksi dari semua ancestornya dengan IS (atau IX), jadi jika transaksi mengunci node di S (atau X), tidak ada transaksi lain yang dapat mengakses ancestornya di X (atau S & X).

5.5 SKEMA MULTI-VERSI

Dalam skema kontrol konkurensi multiversi.

- Setiap operasi write (X) membuat "versi" baru dari item X.
- TS (Xn) tidak menimpa nilai lama dari item data X.
- Operasi read tidak pernah ditolak.



Stempel waktu-W & Stempel waktu-R dari Versi Ke-N : Dalam metode ini, beberapa versi X_I, X_2, X_3 ... X_n dari setiap item data X dipertahankan untuk setiap versi, nilai versi X_n.

(i) **Read-TS(X_n) OR [R-TS (X_n)]** : Stempel waktu read X_n adalah yang terbesar dari semua stempel waktu transaksi yang berhasil membaca versi X_n.

Misalnya:

read by TS(X_n) dengan timestamp TS

read Vj dimana

j = Max {if TSi < TS}

Add TS to read history;

(ii) **Write-TS (X_n) OR [W-TS (X_n)]** : Stempel waktu write Xn adalah stempel waktu transaksi yang menulis nilai versi Xn.

Ketika suatu transaksi T diizinkan untuk mengeksekusi operasi write-item (X), versi baru X_{n+1} dari item X dibuat dengan read-TS (X_{n+1}) & write-TS (X_{n+1}) diatur ke TS(T). Yakni menulis val melalui TS (X_n) dengan

TS\if (ada k

TS< R-TS_k < W-TS_j

where j = min {i\TS<W-TS_i}

{

REJECT;

}

else,

{

Add < TS, Val> to write history;

}

- Ketika sebuah transaksi T diperbolehkan membaca nilai versi Xn, nilai read-TS(Xn) diset lebih besar dari read-TS(Xn) dan TS(T) saat ini.

Untuk memastikan serializability, dua aturan berikut digunakan.

- Jika transaksi T mengeluarkan operasi write (X), dan versi n dari x memiliki TS(Xn) write tertinggi dari semua versi X yang juga kurang dari atau sama dengan TS(T),

Misalnya:

write-TS(Xn):S; TS(T)

& read-TS (Xn) > TS(T), Then

abort & rollback transaction T;

otherwise, create a new version Xj of X with

read-TS(Xj) = write-TS(Xj) = TS(T).

- Jika transaksi T mengeluarkan operasi read (X), cari versi n dari X yang memiliki, write-TS(Xn) tertinggi dari semua versi X yaitu < = TS (T)

Misalnya :

write-TS(Xn) :s; TS(T). Kemudian kembalikan nilai dari Xn ke T, & atur nilai read-TS(Xn) melebihi TS(T) & current read-TS(Xn).

5.6 LOCKING DUA FASA MULTI-VERSI

Protokol penguncian dua fase multiversi mencoba untuk menggabungkan keuntungan dari kontrol konkurensi multiversi dengan keuntungan dari penguncian dua fase. Protokol ini membedakan antara transaksi read-only & transaksi update.

- Transaksi pembaruan melakukan penguncian dua fase yang ketat, yaitu, mereka menahan semua kunci hingga akhir transaksi.

Dengan demikian mereka dapat diserialkan sesuai dengan urutan komit mereka. Setiap versi item data memiliki stempel waktu tunggal. Stempel waktu dalam hal ini bukanlah stempel waktu berbasis jam yang sebenarnya.

- Ketika transaksi read-only Ti mengeluarkan read (Q), nilai yang dikembalikan adalah konten versi yang stempel waktunya adalah stempel waktu terbesar yang kurang dari TS(Ti).

- Saat transaksi pembaruan membaca item, ia mendapatkan kunci bersama pada item tersebut, dan membaca versi terbaru item tersebut.
- Ketika transaksi pembaruan ingin menulis item, pertama-tama ia mendapatkan kunci eksklusif pada item tersebut, dan kemudian membuat versi baru dari item data. Ketika transaksi pembaruan Ti menyelesaikan tindakannya, ia melakukan pemrosesan komit.

Penjadwal penguncian dua fase multiversi berfungsi sebagai berikut:

a) Setiap pembacaan memerlukan kunci read pada item yang sedang dibaca.
b) Setiap penulisan memerlukan kunci write pada item yang sedang ditulis.
c) Kunci write mencegah .....
    a. Membaca item tetapi bukan versi sebelumnya.
    b. Membuat versi baru item.

Varian yang sedikit lebih fleksibel dari dua penjadwal 2PL .....

- Read tidak hanya item versi terbaru.
- Tapi itu bisa menyebabkan cascading aborts.

Penjadwal menggunakan tiga jenis kunci

- kunci Read yang bertabrakan dengan kunci sertifikasi (tetapi bukan kunci write)
- kunci Write yang bertabrakan dengan kunci write & sertifikasi (tetapi bukan kunci read)
- kunci sertifikasi yang bertabrakan dengan kunci read, write, & sertifikasi.

5.7 PEMULIHAN DENGAN TRANSAKSI SAAT INI

Kami membahas di sini bagaimana kami dapat memodifikasi dan memperluas skema pemulihan berbasis log untuk menangani beberapa transaksi bersamaan.

1. **Interaksi dengan kontrol konkurensi :** Skema pemulihan sebagian besar bergantung pada skema kontrol konkurensi yang digunakan, untuk mengembalikan transaksi yang gagal, kita harus membatalkan pembaruan yang dilakukan oleh transaksi. Misalnya: Misalkan transaksi T 1 harus dibatalkan, dan item tanggal Q yang diperbarui oleh T 1 harus dikembalikan ke nilai lamanya. Menggunakan skema berbasis log untuk pemulihan, kami memulihkan nilai dengan menggunakan informasi UNDO dalam catatan log.
2. **Transaction Rollback:** Kami mengembalikan transaksi yang gagal Ti dengan menggunakan log. Sistem memindai kata log kembali untuk setiap catatan log dari formulir < T 1, T 2, V 1> V 2 > yang ditemukan di log, sistem mengembalikan item data X2, ke nilai lamanya V l' Memindai log berakhir ketika catatan log <T1, start> ditemukan.
3. **Titik pemeriksaan** : Kami menggunakan titik pemeriksaan untuk mengurangi jumlah catatan log yang harus dipindai sistem saat pulih dari kerusakan.
   Itu perlu untuk mempertimbangkan hanya transaksi berikut selama pemulihan.
    - Transaksi yang dimulai setelah pos pemeriksaan terbaru.

- Satu transaksi, jika ada yang aktif pada saat checkpoint terbaru.

Dalam sistem pemrosesan transaksi bersamaan, kami mengharuskan catatan log pos pemeriksaan dalam bentuk < pos pemeriksaan L>, di mana L adalah daftar transaksi yang aktif pada saat pos pemeriksaan.

4. **Restart Recovery:** Ketika sistem pulih dari crash, itu membangun dua daftar:
    a. UNDO-Lists terdiri dari transaksi yang akan : dibatalkan.
    b. Daftar-REDO terdiri dari transaksi yang akan dilakukan ulang. Setelah daftar-REDO dan daftar-UNDO dibuat, pemulihan berjalan sebagai berikut:
        i. Sistem memindai ulang log dari catatan terbaru mundur & melakukan UNDO untuk setiap catatan log. Pemindaian berhenti ketika catatan < T start > telah ditemukan untuk setiap transaksi T dalam daftar UNDO
        ii. Sistem menemukan catatan < pos pemeriksaan L > terbaru di log.
        iii. Sistem memindai log maju dari catatan < pos pemeriksaan> terbaru, & melakukan REDO & setiap catatan log milik transaksi T yang ada di daftar-REDO.

5.8 DATABASE DISTRIBUSI

Sistem Terdistribusi: Sistem terdistribusi adalah kumpulan sistem otonom yang berkomunikasi melalui jaringan komunikasi. **Konsep Database Terdistribusi :** Database terdistribusi adalah kumpulan dari banyak Database yang saling berhubungan secara logis yang didistribusikan melalui jaringan komputer. Sistem manajemen Database terdistribusi adalah sistem perangkat lunak yang mengelola Database terdistribusi sambil membuat distribusi transparan kepada pengguna.

Dalam sistem database terdistribusi, baik pemrosesan data maupun transaksi dibagi antara satu atau lebih komputer (CPU) yang terhubung oleh jaringan, masing-masing komputer memainkan peran khusus dalam sistem. Komputer dalam sistem terdistribusi berkomunikasi satu sama lain melalui berbagai media komunikasi, seperti jaringan saluran telepon berkecepatan tinggi. Mereka tidak berbagi memori utama atau disk. Sebuah sistem database terdistribusi memungkinkan aplikasi untuk mengakses data dari database lokal dan remote. Sistem database terdistribusi menggunakan arsitektur Client/Server untuk memproses permintaan informasi. Komputer dalam sistem terdistribusi dapat bervariasi dalam ukuran dan fungsi, mulai dari workstation hingga sistem mainframe. Komputer dalam sistem database terdistribusi disebut dengan sejumlah nama yang berbeda, seperti situs atau node.

Properti Database Terdistribusi: Sistem database terdistribusi harus membuat dampak distribusi data transparan. Sistem database terdistribusi harus memiliki properti berikut.

- Independensi data terdistribusi.
- Atomitas transaksi terdistribusi.

Independensi Data Terdistribusi: Properti independensi data terdistribusi memungkinkan pengguna untuk mengajukan pertanyaan tanpa menentukan di mana relasi referensi atau salinan atau fragmen relasi berada. Prinsip ini merupakan perpanjangan alami dari independensi data fisik dan logis. Selanjutnya, kueri yang menjangkau beberapa situs harus

dioptimalkan secara sistematis dengan cara berbasis biaya, dengan mempertimbangkan biaya komunikasi dan perbedaan dalam biaya komputasi lokal.

Atomisitas Transaksi Terdistribusi: Properti atomisitas transaksi terdistribusi memungkinkan pengguna untuk menulis transaksi yang mengakses dan memperbarui data di beberapa situs sama seperti mereka akan menulis transaksi melalui data lokal murni. Secara khusus, efek transaksi di seluruh situs harus terus bersifat atom. Itu semua perubahan tetap ada jika transaksi dilakukan, dan tidak ada yang bertahan jika dibatalkan.

Klasifikasi Database Terdistribusi

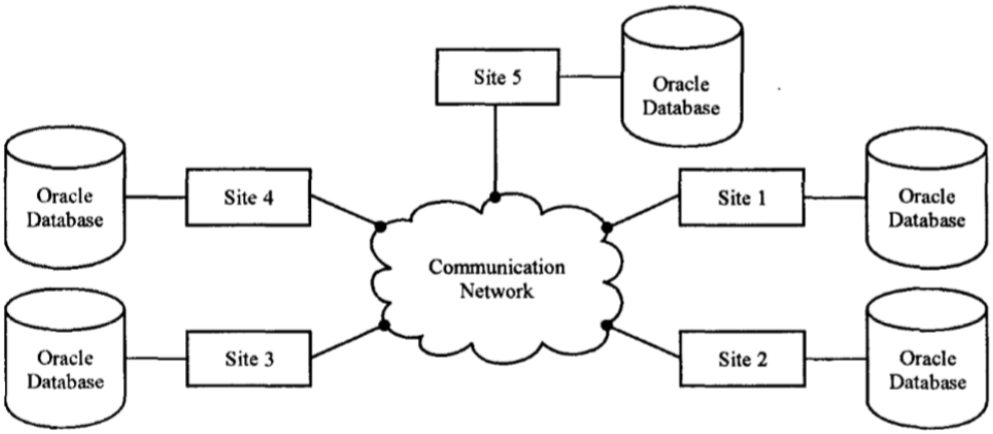
Kita dapat mengklasifikasikan database terdistribusi sebagai:

- Homogen
- Hetrogen

Database Terdistribusi Homogen

Dalam database terdistribusi homogen, semua situs memiliki perangkat lunak sistem manajemen identik yang setuju untuk bekerja sama dalam memproses permintaan pengguna. Dalam sistem seperti itu, situs lokal menyerahkan sebagian dari otonomi mereka dalam hal hak mereka untuk mengubah skema atau perangkat lunak DBMS. . DDBS homogen adalah bentuk paling sederhana dari database terdistribusi dimana terdapat beberapa situs, masing-masing menjalankan aplikasi mereka sendiri pada perangkat lunak DBMS yang sama. Semua situs memiliki perangkat lunak DBMS yang identik, semua pengguna menggunakan perangkat lunak yang sama, menyadari satu sama lain, setuju untuk bekerja sama dalam memproses permintaan pengguna. Semua aplikasi dapat melihat skema yang sama dan menjalankan transaksi yang sama. Artinya, ada transparansi lokasi di DDBS homogen.

Arsitektur database terdistribusi homogen.

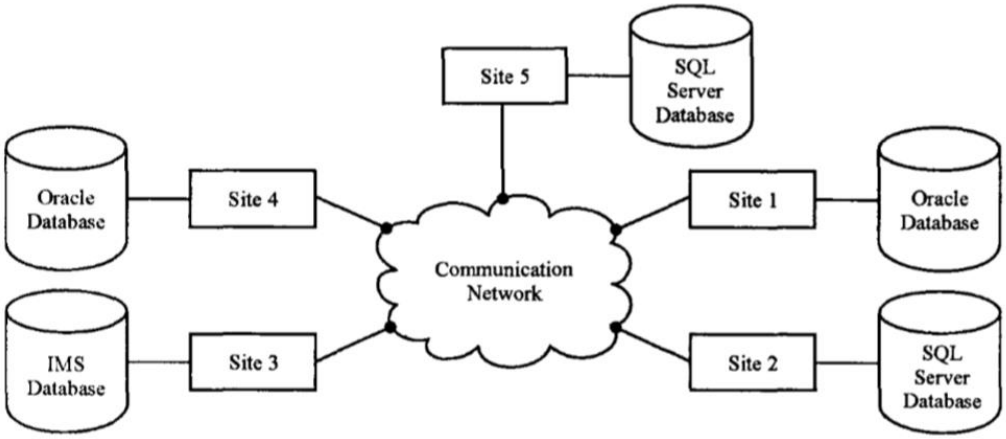


Gambar 5.2 Arsitektur database terdistribusi homogen.

Database Terdistribusi Heterogen

Dalam database terdistribusi heterogen, situs yang berbeda dapat menggunakan skema yang berbeda dan perangkat lunak sistem manajemen database yang berbeda. Situs-situs ini mungkin tidak saling mengenal dan mungkin hanya menyediakan fasilitas terbatas untuk kerjasama dalam pemrosesan transaksi.

Arsitektur DDBS heterogen



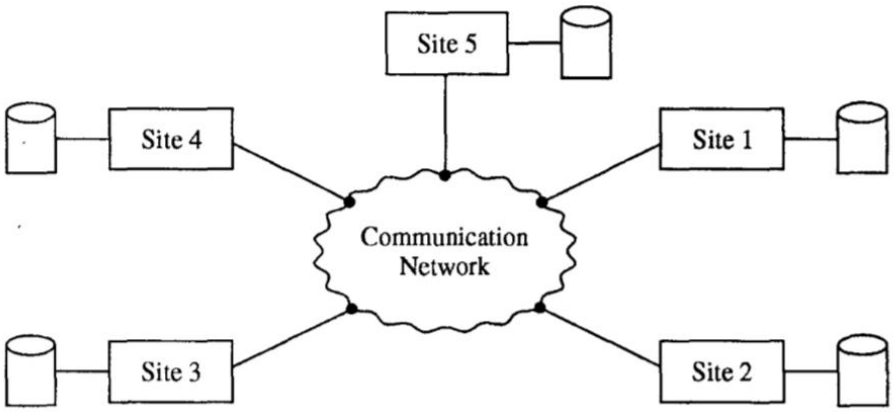
Gambar 5.3 Arsitektur DDBS heterogen

Sistem Database terdistribusi heterogen juga disebut sebagai sistem Database multi atau sistem Database federasi. Sistem database heterogen memiliki standar yang diterima dengan baik untuk protokol gateway untuk mengekspos fungsionalitas DBMS ke aplikasi eksternal. Protokol gateway membantu menutupi perbedaan dalam mengakses server database, dan menjembatani perbedaan antara server yang berbeda dalam sistem terdistribusi.

Fungsi Database Terdistribusi

1. Sistem manajemen Database terdistribusi (DDBMS) harus dapat menyediakan fungsi tambahan berikut dibandingkan dengan DBMS terpusat.
2. Kemampuan melacak data, distribusi data, fragmentasi, dan replikasi dengan memperluas katalog DDBMS.
3. Kemampuan manajemen data yang direplikasi untuk mengakses dan menjaga konsistensi item data yang direplikasi.
4. Kemampuan untuk mengelola pemrosesan kueri terdistribusi untuk mengakses situs jarak jauh dan transmisi kueri dan data di antara berbagai situs melalui jaringan komunikasi.
5. Kemampuan manajemen transaksi terdistribusi dengan merancang strategi eksekusi untuk kueri dan transaksi yang mengakses data dari beberapa situs.
6. Harus memiliki independensi fragmentasi, yaitu pengguna harus disajikan dengan tampilan data di mana fragmen digabungkan kembali secara logis melalui JOIN dan UNION yang sesuai.
7. Harus perangkat keras independen.
8. Memberikan otonomi daerah.
9. Manajemen katalog terdistribusi.
10. Harus lokasi independen.
11. Harus sistem operasi independen.
12. Harus mandiri jaringan.
13. Harus DBMS independen.
14. Manajemen pemulihan Database terdistribusi yang efisien jika terjadi kerusakan situs dan kegagalan komunikasi.

15. Manajemen keamanan data yang tepat dengan memberikan hak akses resmi kepada pengguna saat menjalankan transaksi terdistribusi



Gambar 5.4 Arsitektur database terdistribusi

Keuntungan dari Database Terdistribusi

Berikut ini adalah keuntungan dari database terdistribusi:

1. **Pengelolaan data terdistribusi dengan berbagai tingkat 'Iransparency :** Distribusi transparan berarti menyembunyikan detail di mana setiap file (tabel) disimpan secara fisik dalam sistem. Ini adalah jenis transparansi berikut yang dimungkinkan di DDB
   a. **Fragmentasi 'Iransparency:** Pengguna tidak perlu mengetahui bagaimana suatu relasi telah terfragmentasi. Ini adalah dua jenis:
      i. Fragmentasi horizontal mendistribusikan relasi ke dalam set baris.
      ii. Fragmentasi vertikal mendistribusikan relasi ke dalam kumpulan kolom.
   b. **Replikasi 'Iransparency:** Replikasi berarti salinan data. Artinya, dalam transparansi replikasi, salinan data dapat disimpan di beberapa situs untuk ketersediaan, kinerja & keandalan yang lebih baik.
   c. **Jaringan 'Iransparency:** Dalam transparansi jaringan, kebebasan bagi pengguna dari rincian operasional jaringan. **Transparansi penamaan** : penamaan menyiratkan bahwa setelah nama ditentukan, objek bernama dapat diakses dengan jelas tanpa spesifikasi tambahan **Transparansi lokasi** : pengguna tidak diharuskan mengetahui lokasi fisik data
2. **Peningkatan Keandalan & Ketersediaan:** Kedua adalah keuntungan yang paling penting dari database terdistribusi.
   a. Keandalan berarti, " Probabilitas bahwa suatu sistem berjalan (tidak turun) pada titik waktu tertentu".
   b. Availability berarti "Probabilitas bahwa sistem terus-menerus tersedia selama interval waktu".
3. **Keamanan:** Transaksi terdistribusi dilakukan dengan pengelolaan keamanan data yang baik
4. **Pemrosesan kueri terdistribusi:** Kemampuan untuk mengakses situs jarak jauh & mengirimkan kueri dan data di antara berbagai situs melalui jaringan komunikasi

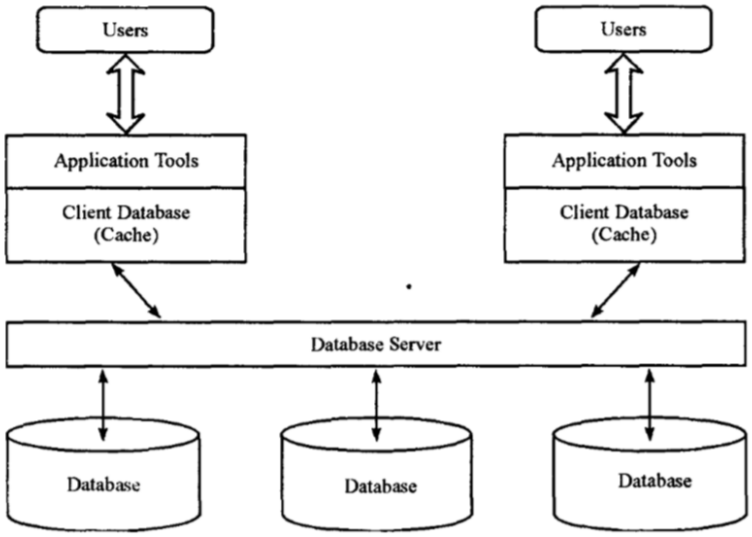
5. **Pemulihan Database Terdistribusi:** Kemampuan DDB untuk memulihkan dari kerusakan situs individual.
6. **Manajemen Data yang Direplikasi**: Kemampuan untuk memutuskan salinan item data yang direplikasi mana yang akan diakses untuk menjaga konsistensi item data yang direplikasi.
7. **Melacak data** : Kemampuan DDB untuk melacak fragmentasi, distribusi, & replikasi data dengan memperluas katalog DDBMS.
8. Peningkatan skalabilitas.
9. Ekspansi lebih mudah.
10. Peningkatan kinerja
11. Evaluasi paralel.

Kekurangan Database Terdistribusi

1. Masalah teknis menghubungkan mesin yang berbeda.
2. Biaya dan Kompleksitas Perangkat Lunak: Perangkat lunak yang lebih kompleks diperlukan untuk lingkungan Database terdistribusi.
3. Kesulitan dalam Kontrol Integritas Data : Produk sampingan dari meningkatnya kompleksitas dan kebutuhan akan koordinasi adalah paparan tambahan terhadap pemutakhiran yang tidak tepat dan masalah integritas data lainnya.
4. Overhead Pemrosesan : Berbagai situs harus bertukar pesan dan melakukan perhitungan tambahan untuk memastikan koordinasi yang tepat antar situs.
5. Kegagalan Jaringan Komunikasi.
6. Kehilangan pesan.
7. Pemulihan kegagalan lebih kompleks.
8. Peningkatan kompleksitas dalam desain dan implementasi sistem.
9. Masalah keamanan data yang direplikasi di beberapa lokasi dan jaringan.
10. Peningkatan transparansi mengarah pada kompromi antara kemudahan penggunaan dan biaya overhead untuk menyediakan transparansi.
11. Potensi bug yang lebih besar.

Arsitektur Database Terdistribusi

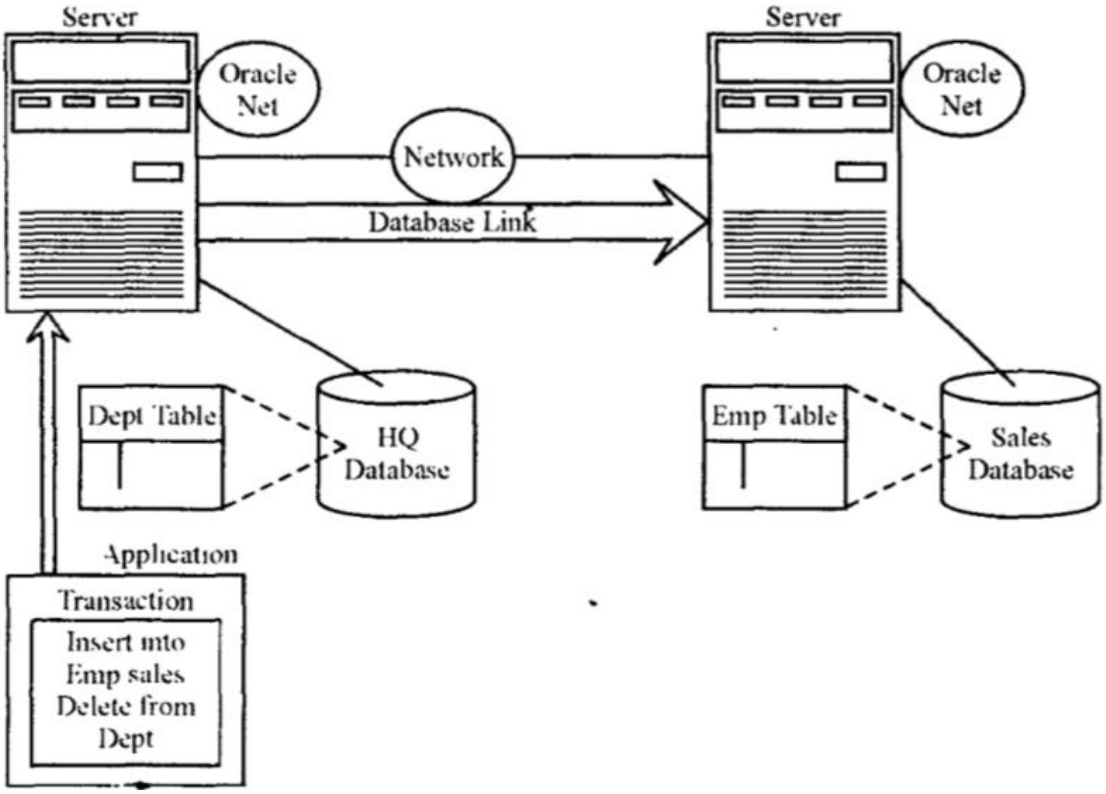
Database terdistribusi menggunakan arsitektur Client/Server untuk memproses permintaan informasi. **Arsitektur Client/Server:** Arsitektur Client/Server adalah arsitektur di mana beban kerja terkait DBMS dibagi menjadi dua komponen logis yaitu klien dan server, yang masing-masing biasanya dijalankan pada sistem yang berbeda. Klien adalah pengguna sumber daya sedangkan server adalah penyedia sumber daya. Aplikasi dan alat diletakkan pada satu atau lebih platform klien dan terhubung ke sistem manajemen Database yang berada di server. Aplikasi dan alat bertindak sebagai 'klien' dari DBMS, membuat permintaan untuk layanannya. Arsitektur client/sever dapat digunakan untuk mengimplementasikan DBMS di mana client adalah pemroses transaksi dan server adalah pemroses data. Aplikasi klien mengeluarkan pernyataan SQL untuk akses data, seperti yang mereka lakukan dalam aplikasi klien komputasi terpusat untuk terhubung ke server, mengirim pernyataan SQL dan menerima hasil atau kode pengembalian kesalahan setelah server memproses pernyataan SOL.



Gambar 5.5 Arsitektur Database Client/Server.

Arsitektur Client/Server terdiri dari komponen utama sebagai berikut :

- Klien berupa workstation cerdas sebagai titik kontak pengguna.
- Server DBMS sebagai sumber daya umum yang melakukan tugas khusus untuk perangkat yang meminta layanan mereka.
- Jaringan komunikasi yang menghubungkan klien dan server.
- Aplikasi perangkat lunak yang menghubungkan klien, server, dan jaringan untuk membuat arsitektur logis tunggal.



Gambar 5.6 Sistem database terdistribusi database oracle

Seorang klien dapat terhubung secara langsung atau tidak langsung ke server database. Koneksi langsung terjadi ketika klien terhubung ke server dan mengakses informasi dari database yang terdapat di server. Sebaliknya, koneksi tidak langsung terjadi ketika klien terhubung ke server dan kemudian mengakses informasi yang terdapat dalam database di server yang berbeda.

Contoh: Jika Anda terhubung ke database HQ dan mengakses tabel dept pada database ini seperti pada gambar, Anda dapat mengeluarkan yang berikut ini:

```
SELECT * FROM dept;
```

Kueri ini langsung karena Anda tidak mengakses objek di database jarak jauh. Jika Anda terhubung ke database HQ tetapi mengakses tabel emp pada database penjualan jarak jauh seperti pada gambar, Anda dapat mengeluarkan yang berikut:

```
SELECT * FROM emp@sales;
```

Kueri ini tidak langsung karena objek yang Anda akses tidak ada di database yang terhubung langsung dengan Anda.

Keuntungan Arsitektur Database Client/Server

1. Dalam kebanyakan kasus, arsitektur klien / server memungkinkan peran dan tanggung jawab sistem komputasi untuk didistribusikan di antara beberapa komputer independen yang diketahui satu sama lain hanya melalui jaringan. Ini menciptakan keuntungan tambahan untuk arsitektur ini: kemudahan perawatan yang lebih besar.
2. Ini berfungsi dengan beberapa pengguna yang berbeda dengan kemampuan yang berbeda.
3. Arsitektur ini relatif sederhana untuk diimplementasikan, karena pemisahan fungsionalitasnya yang bersih karena server terpusat.
4. Peningkatan kinerja dengan lebih banyak kekuatan pemrosesan yang tersebar di seluruh organisasi.
5. Semua data disimpan di server, yang umumnya memiliki kontrol keamanan yang jauh lebih besar daripada kebanyakan klien. Server dapat mengontrol akses dan sumber daya dengan lebih baik, untuk menjamin bahwa hanya klien dengan izin yang sesuai yang dapat mengakses dan mengubah data.
6. Karena data disimpan secara terpusat, pembaruan terhadap data tersebut jauh lebih mudah untuk dikelola daripada apa yang mungkin dilakukan di bawah peer to peer.
7. Mengurangi total biaya kepemilikan.
8. Meningkatkan produktivitas.
9. Karena klien tidak memainkan peran utama dalam model ini, mereka membutuhkan lebih sedikit administrator.
10. Meningkatkan keamanan.

Kekurangan

1. Kemacetan lalu lintas di jaringan telah menjadi masalah sejak awal paradigma Client/Server. Karena jumlah permintaan klien simultan ke server tertentu, server dapat menjadi kelebihan beban.

2. Paradigma Client/Server tidak memiliki kekokohan jaringan peer to peer yang baik. Di bawah Client/Server, jika server kritis gagal, permintaan klien tidak dapat dipenuhi.

Desain Sistem Database Terdistribusi

Desain sistem database terdistribusi adalah tugas yang kompleks. Oleh karena itu, diperlukan penilaian yang cermat terhadap strategi dan tujuan. Beberapa strategi dan tujuan yang umum untuk desain sistem database terdistribusi adalah sebagai berikut:

- **Fragmentasi Data:** Yang diterapkan pada sistem Database relasional untuk mempartisi hubungan antar situs jaringan.
- **Data Allocatifln:** Di mana setiap fragmen disimpan di situs dengan distribusi opsional.
- **Replikasi Data:** Yang meningkatkan ketersediaan dan meningkatkan kinerja sistem.
- **Transparansi Lokasi:** Yang memungkinkan pengguna untuk mengakses data tanpa mengetahui, atau peduli dengan, situs tempat data berada. Lokasi data disembunyikan dari pengguna.
- **Transparansi Replikasi:** Artinya ketika ada lebih dari satu salinan data, satu salinan dipilih saat mengambil data dan semua salinan lainnya diperbarui saat perubahan dilakukan.
- **Kemandirian Konfigurasi:** Yang memungkinkan organisasi untuk menambah atau mengganti perangkat keras tanpa mengubah komponen perangkat lunak DBMS yang ada. Ini memastikan perluasan sistem yang ada ketika perangkat kerasnya saat ini jenuh.
- **DBMS non-homogenitas:** Yang membantu dalam mengintegrasikan database yang dikelola oleh DBMS yang berbeda di situs yang berbeda pada komputer yang berbeda.

Fragmentasi data Replikasi data dan alokasi data adalah teknik yang paling umum digunakan yang digunakan selama proses desain DDBS untuk memecah database menjadi unit logis dan menyimpan data tertentu di lebih dari satu situs.

Pemrosesan Transaksi dalam Sistem Terdistribusi

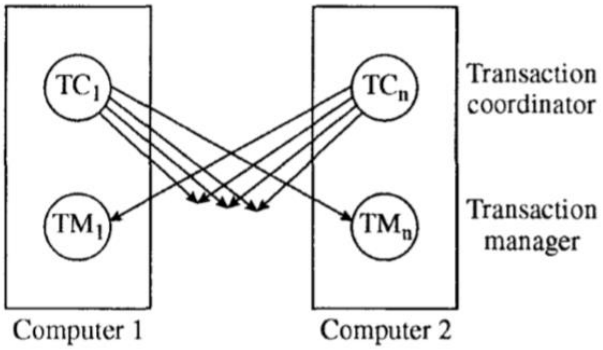
Sistem Transaksi: Sistem pemrosesan transaksi adalah sistem dengan Database besar dan sejumlah besar pengguna secara bersamaan menjalankan transaksi Database. **Pemrosesan Transaksi:** Dalam DBMS terdistribusi, transaksi tertentu dikirimkan di beberapa situs, tetapi dapat mengakses data di situs lain. Ketika sebuah transaksi dikirimkan di beberapa situs, manajer transaksi di situs tersebut memecahnya menjadi kumpulan satu atau lebih subtransaksi yang dijalankan di situs yang berbeda.

Ada dua jenis transaksi.

- **Transaksi Lokal :** Transaksi lokal adalah transaksi yang mengakses dan memperbarui data hanya dalam satu database lokal.
- **Transaksi Global:** Transaksi global adalah transaksi yang mengakses dan memperbarui data di beberapa database lokal.

Struktur Sistem: Setiap situs memiliki manajer transaksi lokalnya sendiri, yang berfungsi untuk memastikan sifat ACID dari transaksi yang dijalankan di situs tersebut. Dalam sistem transaksi setiap situs berisi dua subsistem.

a. **Manajer Transaksi:** Manajer transaksi mengelola pelaksanaan transaksi yang mengakses data yang disimpan di situs lokal.
b. **Koordinator Transaksi:** Koordinator transaksi mengoordinasikan pelaksanaan berbagai transaksi yang dimulai di situs itu.



Gambar 5.7 Pemrosesan Transaksi Terdisibusi

Tanggung Jawab Manajer Transaksi

Berikut adalah tanggung jawab masing-masing manajer transaksi:

- Mempertahankan log untuk tujuan pemulihan.
- Berpartisipasi dalam skema kontrol konkurensi yang sesuai untuk mengoordinasikan eksekusi bersamaan dari transaksi yang dieksekusi di situs itu.

Tanggung Jawab Koordinator

Berikut ini adalah tanggung jawab koordinator :

- Memulai eksekusi transaksi.
- Memecah transaksi menjadi beberapa subtransaksi dan mendistribusikan subtransaksi ini ke situs yang sesuai untuk dieksekusi.
- Mengkoordinasikan penghentian transaksi, yaitu transaksi dapat dilakukan di semua situs atau dibatalkan di semua situs.

Mode Kegagalan Sistem: Sistem terdistribusi mungkin mengalami jenis kegagalan yang sama dengan sistem terpusat.

Berikut adalah jenis-jenis kegagalan:

- Kegagalan sebuah situs.
- Kehilangan pesan.
- Kegagalan link komunikasi.
- Partisi Jaringan

Fragmentasi Data

Sistem mempartisi relasi menjadi beberapa fragmen dan menyimpan setiap fragmen di tempat yang berbeda. Misalkan relasi r terfragmentasi, r dibagi dalam sejumlah fragmen r_I, r_2, r_3 ..... , r_n. Fragmen ini berisi informasi yang cukup untuk memungkinkan rekonstruksi relasi asli r.

Jenis-Jenis Fragmentasi: Berikut ini adalah fragmentasi :

1. **Fragmentasi Horizontal:** Fragmentasi Horizontal membagi relasi dengan menugaskan setiap tuple (baris) dari r ke satu atau lebih fragmen. Kita dapat membangun ri fragmentasi dengan menggunakan seleksi

$$r_i = \sigma_{Pi}(r)$$

dimana Pi adalah predikat. Kita dapat membangun relasi r dengan mengambil gabungan semua fragmen; itu adalah:

$$r = r_1 U\ r_2 U\ \ldots\ldots\ldots\ U r_n$$

Tabel 5.7 Tabel Siswa Relasi dan Fragment Siswa Horizontal (Gambar paling Bawah).

Roll No.	Name	Address	City	State
1.	Dani	H–327	Jakarta Selatan	Indonesia
2.	Lawren	F–300	Bandung	Indonesia
3.	Agus	L–62	Jakarta	Indonesia
4.	Joni	M–129	Bandung	Indonesia

Fragment Student1

Roll No.	Name	Address	City	State
1.	Dani	H–327	Jakarta Selatan	Indonesia
2.	Lawren	F–300	Bandung	Indonesia

Fragment Student2

Roll No.	Name	Address	City	State
3.	Agus	L–62	Jakarta	Indonesia
4.	Joni	M–129	Bandung	Indonesia

2. **Fragmentasi Vertikal:** Fragmentasi vertikal membagi relasi dengan dekomposisi skema R dari relasi r. Setiap fragmentasi ri dari r di-dermed oleh

$$r_i = \pi_{Ri}(r)$$

dimana R adalah atribut. Fragmentasi harus dilakukan sedemikian rupa sehingga kita dapat merekonstruksi relasi r dari fragmen dengan mengambil natural join.

$$r = r_1 \bowtie r_2 \bowtie \ldots\ldots\ldots \bowtie r_n$$

Tabel 5.8 Tabel Relasi Siswa

Roll No.	Name	Address	City	State
1.	Dani	H–327	Jakarta Selatan	Indonesia
2.	Lawren	F–300	Bandung	Indonesia
3.	Agus	L–62	Jakarta	Indonesia
4.	Joni	M–129	Bandung	Indonesia

Tabel 5.9 Hasil Fragmentasi Vertikal 1 dan 2

Roll No.	Name	Address
1.	Dani	H–327
2.	Lawren	F–300
3.	Agus	L–62
4.	Joni	M–129

Roll No.	City	State
1.	Jakarta Selatan	Indonesia
2.	Bandung	Indonesia
3.	Jakarta	Indonesia
4.	Bandung	Indonesia

3. **Fragmentasi Campuran**: Kita dapat mencampur dua jenis fragmentasi (Fragmentasi Horisontal & Vertikal) menghasilkan fragmentasi campuran. Representasi matematis dari fragmentasi campuran adalah:

$$r_1 = \pi_{R_i} (\sigma_{P_i} (r))$$

Berikut ini adalah kasus-kasus yang muncul.

Kasus 1 : jika Pi ≠ Benar & Ri = ATTRS (R),

Kemudian, kita mendapatkan fragmen vertikal.

Kasus 2: Jika Pi = Benar & Ri ≠ ATTRS (R),

Kemudian, kami mendapatkan- fragmen horizontal.

Kasus 3: Jika Pi ≠ Benar & Ri ≠ ATTRS(R),

Kemudian, kami mendapatkan fragmen campuran

Replikasi & Alokasi Data

Replikasi Data berarti salinan data. Replikasi data adalah teknik yang memungkinkan penyimpanan data tertentu di lebih dari satu situs. Sistem memelihara beberapa replika identik (salinan) dari relasi dan menyimpan setiap salinan di situs yang berbeda, Biasanya, replikasi data diperkenalkan untuk meningkatkan ketersediaan sistem ketika salinan tidak tersedia karena kegagalan situs, itu harus dimungkinkan untuk mengakses salinan lain.

Basis Data Terdistribusi Sepenuhnya Replikasi: Replikasi seluruh Database di setiap situs dalam sistem terdistribusi, disebut Database terdistribusi yang sepenuhnya direplikasi.

Keuntungan: Ini adalah keuntungan berikut:

- Ini dapat meningkatkan ketersediaan karena sistem dapat terus beroperasi selama setidaknya satu situs aktif.
- Ini juga meningkatkan kinerja pengambilan untuk kueri global, karena hasil kueri semacam itu dapat diperoleh secara lokal dari situs siapa pun.

Kekurangan: Kerugian dari replikasi penuh adalah:

- Ini dapat memperlambat operasi pembaruan secara drastis, karena pembaruan logis tunggal harus dilakukan pada setiap salinan database untuk menjaga salinan tetap konsisten.

- Replikasi penuh membuat kontrol konkurensi dan teknik pemulihan lebih mahal daripada jika tidak ada replikasi.
    - **Skema Replikasi :** Deskripsi replikasi fragmen disebut skema replikasi.
    - **Parrtial Replication** : Dalam Replikasi parsial, beberapa fragmen database dapat direplikasi sedangkan yang lain mungkin tidak

Alokasi Data

Alokasi data menggambarkan proses memutuskan tentang penempatan (atau penempatan) data ke beberapa situs. Berikut ini adalah strategi penempatan data yang digunakan untuk sistem database terdistribusi.

- Terpusat
- Dipartisi atau terfragmentasi
- Direplikasi

Dalam hal strategi terpusat, seluruh database tunggal dan DBMS disimpan di satu situs. Namun, pengguna secara geografis didistribusikan di seluruh jaringan. Lokalitas referensi paling rendah karena semua situs, kecuali situs pusat, harus menggunakan jaringan untuk semua akses data. Dengan demikian, biaya komunikasi menjadi tinggi. Karena seluruh Database berada di satu situs, ada kehilangan seluruh sistem Database jika terjadi kegagalan situs pusat. Oleh karena itu, keandalan dan ketersediaannya rendah.

Dalam strategi dipartisi atau terfragmentasi, database dibagi menjadi beberapa bagian yang terpisah (fragmen) dan disimpan di beberapa situs. Jika item data terletak di lokasi yang paling sering digunakan, lokalitas referensi tinggi. Karena tidak ada replikasi, biaya penyimpanan rendah. Kegagalan sistem pada situs tertentu akan mengakibatkan hilangnya data situs tersebut. Oleh karena itu, keandalan dan ketersediaan lebih tinggi daripada strategi terpusat.

Namun, keandalan dan ketersediaan secara keseluruhan masih rendah. Biaya komunikasi rendah dan kinerja keseluruhan baik dibandingkan dengan strategi terpusat. Dalam strategi replikasi, salinan dari satu atau lebih fragmen database disimpan di beberapa situs. Dengan demikian, lokalitas referensi, keandalan dan ketersediaan dan kinerja dimaksimalkan. Tapi, komunikasi dan biaya penyimpanan sangat tinggi. Dalam alokasi data atau data distributi Oil setiap salinan fragmen harus ditetapkan ke situs tertentu dalam sistem terdistribusi.

Alokasi Non Redundant: Jika setiap fragmen disimpan tepat di satu situs, maka semua fragmen harus terputus-putus kecuali untuk pengulangan kunci utama di antara fragmen vcctical atau campuran, ini disebut alokasi tidak berlebihan.

Pilihan situs, tingkat replikasi tergantung pada kinerja dan tujuan ketersediaan sistem dan pada jenis dan frekuensi transaksi yang diajukan di setiap situs. Misalnya: Jika ketersediaan tinggi diperlukan dan transaksi dapat dikirimkan di situs mana pun dan jika sebagian besar transaksi hanya dapat diambil, database yang sepenuhnya direplikasi adalah pilihan yang baik.

Ikhtisar Kontrol Konkurensi

Untuk tujuan kontrol konkurensi, banyak masalah muncul di lingkungan DBMS terdistribusi yang tidak ditemui di lingkungan DBMS terpusat. Ini termasuk yang berikut:

- **Berurusan dengan banyak salinan item data** : Metode kontrol konkurensi bertanggung jawab untuk menjaga konsistensi di antara salinan ini.
- **Kegagalan situs individu:** DDBMS harus terus beroperasi dengan situs yang sedang berjalan jika memungkinkan, ketika satu atau lebih situs individu gagal.
- **Kegagalan Link Komunikasi:** Sistem harus mampu menangani kegagalan satu atau lebih link komunikasi yang menghubungkan situs.
- **Deadlock terdistribusi**: Deadlock dapat terjadi di antara beberapa situs, jadi teknik untuk menangani deadlock harus diperluas untuk mempertimbangkan hal ini.
- **Komit Terdistribusi :** Masalah dapat muncul dengan melakukan transaksi yang mengakses database yang disimpan di beberapa situs jika beberapa situs gagal selama proses komit.
- **Teknik Kontrol Konkurensi Terdistribusi :** Teknik kontrol konkurensi terdistribusi harus menangani ini dan masalah lainnya.

Ini adalah teknik berikut:

1. **Protokol Penguncian:** Protokol penguncian dapat digunakan dalam lingkungan terdistribusi. Manajer kunci berurusan dengan data yang direplikasi. Kami menyajikan skema yang mungkin berlaku untuk lingkungan di mana data dapat direplikasi di beberapa situs.
    a. **Pendekatan Manajer Kunci Tunggal:** Dalam pendekatan manajer kunci tunggal, sistem mempertahankan satu manajer kunci yang berada di satu lokasi yang dipilih, pertimbangkan Si. Semua permintaan kunci dan buka kunci dibuat di situs Si.
        i. Ketika sebuah transaksi perlu mengunci item data, ia mengirimkan permintaan kunci ke Si.
        ii. Manajer kunci menentukan apakah kunci dapat segera diberikan. Jika kunci dapat diberikan, manajer kunci mengirim pesan ke situs tempat permintaan kunci dimulai. Jika tidak, permintaan akan ditunda hingga dapat dikabulkan.

Keuntungan : Skema ini memiliki keuntungan sebagai berikut :

- **Implementasi Sederhana:** Skema ini memerlukan dua pesan untuk menangani permintaan kunci, dan satu pesan untuk menangani permintaan buka kunci.
- **Penanganan Deadlock Sederhana** : Karena semua permintaan penguncian dan pembukaan kunci dilakukan di satu situs, algoritme penanganan Deadlock dapat diterapkan langsung ke lingkungan ini.

Kekurangan : Kerugian dari skema ini adalah :

- Bottleneck: Situs Si menjadi bottle neck, karena semua permintaan harus diproses di sana.
- Kerentanan : Jika situs Si gagal, pengontrol konkurensi hilang. Pemrosesan harus dihentikan, atau skema pemulihan harus digunakan agar situs cadangan dapat mengambil alih manajemen kunci dari Si.

b. **Manajer kunci terdistribusi:** Di mana fungsi manajer kunci didistribusikan ke beberapa situs.

Setiap situs memelihara manajer lpck lokal yang berfungsi untuk mengelola permintaan kunci dan buka kunci untuk item data yang disimpan di situs tersebut. Ketika sebuah transaksi menginginkan kunci data item X, yang tidak direplikasi dan berada di situs Si, sebuah pesan dikirim ke manajer kunci di situs Si yang meminta kunci. Jika item data X dikunci dalam mode yang tidak kompatibel, maka permintaan akan ditunda hingga dapat diberikan.

Setelah ditentukan bahwa permintaan kunci dapat diberikan, manajer kunci mengirim pesan kembali ke pemrakarsa yang menunjukkan bahwa dia telah mengabulkan permintaan kunci. **Keuntungan :**

- Implementasi sederhana.
- Mengurangi sejauh mana koordinator adalah hambatan.

Pemulihan Terdistribusi

Proses pemulihan dalam database terdistribusi cukup terlibat. Kami hanya memberikan gambaran yang sangat singkat tentang beberapa masalah di sini. Dalam beberapa kasus bahkan cukup sulit untuk menentukan apakah sebuah situs sedang down tanpa bertukar banyak pesan dengan situs lain. Sebagai Contoh: Misalkan situs X mengirim pesan ke situs Y dan mengharapkan tanggapan dari Y tetapi tidak menerimanya.

Ada beberapa kemungkinan penjelasan:

- Pesan tidak terkirim ke Y karena kegagalan komunikasi.
- Situs Y tidak aktif dan tidak dapat merespons.
- Situs Y berjalan dan mengirim respons, tetapi respons tidak terkirim.

Masalah lain dengan pemulihan terdistribusi adalah komit terdistribusi. Ketika transaksi memperbarui data di beberapa situs, itu tidak dapat dilakukan sampai yakin bahwa efek transaksi di setiap situs tidak dapat hilang. Ini berarti bahwa setiap situs harus terlebih dahulu mencatat efek lokal dari transaksi secara permanen di log situs lokal pada disk. Protokol komit dua fase sering digunakan untuk memastikan kebenaran komit terdistribusi.

Protokol Komitmen Dua Fase

Protokol ini mengasumsikan bahwa salah satu proses yang bekerja sama bertindak sebagai koordinator, dan proses lainnya sebagai kohort.

- Pada awal transaksi, koordinator mengirimkan pesan awal transaksi ke setiap kohort.
- Jika transaksi yang dijalankan pada kohort berhasil, ia menulis 'UNDO' dan 'REDO' log pada penyimpanan stabil dan mengirimkan pesan 'SETUJU' ke koordinator.
- Jika tidak, ia akan mengirimkan pesan ABORT ke koordinator.

Tahap I : Pada Koordinator:

(1) Koordinator mengirimkan pesan COMMIT REQUEST ke setiap kohor yang meminta kohort untuk berkomitmen.

(2) Koordinator menunggu jawaban dari semua kohor.

Di Kelompok: Saat menerima pesan COMMIT-REQUEST, kelompok mengambil tindakan berikut.

Tahap II: Pada Koordinator :

(1) Jika semua kohor menjawab SETUJU dan koordinator juga setuju, maka koordinator menuliskan record 'COMMIT' ke dalam log. Kemudian mengirimkan pesan COMMIT ke semua kohort. Jika tidak, koordinator mengirimkan pesan ABORT ke semua kohort.
(2) Koordinator kemudian menunggu pengakuan dari masing-masing kohor.
(3) Jika pengakuan tidak diterima dari kelompok mana pun dalam jangka waktu tertentu, koordinator mengirim ulang pesan COMMIT/ABORT ke kelompok itu.
(4) Jika semua pengakuan diterima, koordinator menulis catatan LENGKAP ke log.

Di Kelompok:

(1) Saat menerima pesan COMMIT, Cohort melepaskan semua sumber daya dan kunci yang dipegangnya untuk mengeksekusi transaksi, dan mengirimkan pengakuan.
(2) Saat menerima pesan ABORT, kohort membatalkan transaksi menggunakan catatan log UNDO, melepaskan semua sumber daya dan kunci yang dipegangnya untuk melakukan transaksi dan mengirim pengakuan.

Penanganan Kegagalan

Protokol komit dua fase merespons dengan cara yang berbeda untuk berbagai jenis kegagalan.

1. **Kegagalan Situs yang Berpartisipasi:** Jika koordinator Ci mendeteksi bahwa sebuah situs telah gagal, ia akan mengambil tindakan berikut:
    a. Jika situs gagal sebelum merespons dengan pesan T siap ke Ci, koordinator mengasumsikan bahwa situs tersebut merespons dengan pesan ABORT T.
    b. Jika situs gagal setelah koordinator menerima pesan READY T dari situs, koordinator mengeksekusi sisa protokol komit dengan cara biasa, mengabaikan kegagalan situs.

    Ketika situs yang berpartisipasi Si pulih dari kegagalan, sebagian besar memeriksa lognya untuk menentukan nasib transaksi-transaksi yang berada di tengah-tengah eksekusi ketika kegagalan terjadi. Kami mempertimbangkan setiap kasus yang mungkin.
        - Log berisi catatan < COMMIT T >. Dalam hal ini, situs mengeksekusi REDO (T).
        - Log berisi catatan < ABORT T >. Dalam hal ini, situs mengeksekusi UNDO (T).
        - Log berisi catatan < READY T >. Dalam hal ini, situs harus berkonsultasi dengan Ci untuk menentukan nasib T.
        - Log tidak berisi catatan kontrol (ABORT, COMMIT, READY) tentang T.
2. **Kegagalan Koordinator:** Jika koordinator gagal di tengah pelaksanaan protokol komit untuk transaksi T, maka situs yang berpartisipasi harus memutuskan nasib T. Dalam kasus tertentu, situs yang berpartisipasi tidak dapat memutuskan apakah akan COMMIT atau ABORT T, & oleh karena itu situs ini harus menunggu pemulihan koordinator yang gagal.
    a. Jika situs aktif berisi catatan <COMMIT T > di lognya, maka T harus di-commit.
    b. Jika situs aktif berisi catatan < ABORT T ,> di lognya, maka T harus dibatalkan .

c. Jika beberapa situs aktif tidak berisi catatan < READY T > dalam lognya, maka koordinator Ci yang gagal tidak dapat memutuskan untuk melakukan T, karena situs yang tidak memiliki catatan < READY T > dalam lognya tidak dapat mengirimkan file READY T pesan ke Ci. Namun, koordinator mungkin telah memutuskan untuk ABORT T, tetapi tidak untuk COMMIT T. Daripada menunggu Ci pulih, lebih baik untuk membatalkan T.
d. Jika tidak ada kasus sebelumnya yang berlaku, maka semua situs aktif harus memiliki catatan < READY T > di log mereka, tetapi tidak ada catatan kontrol tambahan seperti < ABORT T > atau < COMMIT T >.

5.9 PENYELESAIAN MASALAH

Pertanyaan 1: Tuliskan metode mekanisme *write-ahead-logging* untuk pemulihan data.

Jawaban: ***Write-Ahead Logging*** **(WAL)**) adalah pendekatan standar untuk pencatatan transaksi. Konsep sentral WA adalah bahwa perubahan pada file data (di mana tabel dan indeks berada) harus ditulis hanya setelah perubahan tersebut dicatat, yaitu ketika catatan log telah dipindahkan ke penyimpanan permanen. Jika kita mengikuti prosedur ini, kita tidak perlu mem-flush halaman data ke disk pada setiap komit transaksi, karena kita tahu bahwa jika terjadi crash kita akan memulihkan database menggunakan log.

- Setiap perubahan yang belum diterapkan ke halaman data pertama-tama akan dilakukan ulang dari catatan log, ini adalah pemulihan roll-forward, juga dikenal sebagai REDO.
- Dan kemudian perubahan yang dilakukan oleh transaksi yang tidak dikomit akan dihapus dari halaman data, ini adalah pemulihan roll mundur UNDO.

Misalnya: Pertimbangkan protokol *Write-Ahead Logging* (WAL) berikut untuk Algoritma pemulihan yang memerlukan UNDO & REDO :

- Gambar sebelum suatu item tidak dapat ditulis ulang oleh gambar sesudahnya dalam database pada disk sampai semua catatan log tipe UNDO untuk transaksi pemutakhiran-sampai saat ini telah ditulis secara paksa ke disk.
- Operasi komit transaksi tidak dapat diselesaikan sampai semua catatan log tipe REDO & tipe UNDO untuk transaksi tersebut telah ditulis paksa ke disk.

Kelebihan WAL :

- *Write-Ahead-Logging* adalah teknik untuk menyediakan atomisitas dan daya tahan dalam sistem database.
- Dalam sistem yang menggunakan WAL, semua modifikasi ditulis ke log sebelum diterapkan ke database. Biasanya informasi REDO dan UNDO disimpan dalam log.
- WAL adalah jumlah penulisan disk yang berkurang secara signifikan, karena hanya file log yang perlu di-flush ke disk pada saat transaksi dilakukan.
- Keuntungan selanjutnya adalah konsistensi halaman data.

WAL menyimpan seluruh konten halaman data dalam log jika diperlukan untuk memastikan konsistensi halaman setelah pemulihan kerusakan.

Pertanyaan 2: Tunjukkan bagaimana teknik pemulihan mundur (*Backward-Recovery*) diterapkan pada DBMS?

Jawaban: Teknik *Backward-Recovery*: Pada saat restart, sistem melalui prosedur berikut untuk mengidentifikasi semua transaksi 'JYpe T 2 - T 5

1. Mulailah dengan dua daftar transaksi, daftar UNDO dan daftar REDO. Atur daftar UNDO sama dengan daftar -dari semua transaksi yang diberikan dalam catatan pos pemeriksaan terbaru; atur daftar- REDO menjadi kosong.
2. Cari maju melalui log, mulai dari catatan pos pemeriksaan.
3. Jika entri log TRANSAKSI BEGIN ditemukan untuk transaksi T, tambahkan T ke daftar UNDO.
4. Jika entri log COMMIT ditemukan untuk transaksi T, pindahkan T dari daftar UNDO ke daftar REDO.
5. Ketika akhir log tercapai, daftar UNDO & REDO mengidentifikasi, masing-masing, transaksi tipe T 3 & T 5 & transaksi tipe T 2 & T 4.

Sistem sekarang bekerja mundur melalui log, membatalkan transaksi dalam daftar UNDO. Memulihkan database ke kondisi yang konsisten dengan membatalkan pekerjaan disebut Pemulihan Mundur.

Pertanyaan 3: Perhatikan Transaksi Berikut

T_1 : read (A);
read (B);
if A = 0 then B : = B + 1;
write (B);
T_2 : read (A);
read (B);
if B = 0 then A : = A + 1;
write (A);

Tambahkan instruksi *lock & unlock* ke transaksi T_1 & T_2 sehingga mereka mematuhi protokol penguncian dua fase.

Jawaban :

Lock-X = Exclusive lock

Lock-S = Shared lock

Tabel 5.10 *lock & unlock* ke transaksi T_1 & T_2

T_1	T_2
Lock – S (A)	
Lock – X (B)	
read (A)	
read (B)	
if A = 0 then B : = B + 1	
write (B)	
Unlock – S (A)	
Unlock – X (B)	
	Lock – S (B)
	Lock – X (A)
	read (A)
	read (B)
	if B = 0 then A : = A + 1
	write (A)
	Unlock – X (A)
	Unlock – S (B)

Pertanyaan 4: Sebutkan apakah jadwal berikut ini konflik bersambung atau tidak. Justifikasi jawaban Anda.

Tabel 5.11 Tabel Pertanyaan 4

T_1	T_2
Read (A)	
Write (A)	
	Read (A)
	Write (A)
Read (B)	
Write (B)	
	Read (B)
	Write (B)

Jawaban: Dalam skedul ini, penulisan (A) dari T 1 bertentangan dengan Read (A) dari T 2, sedangkan penulisan (A) dari T_2 tidak bertentangan dengan Read (B) dari T1. Karena kedua instruksi ini mengakses item data yang berbeda.

Kita dapat menukar instruksi yang tidak bertentangan.

- Tukar Read (B) dari T_I dengan Read (A) dari T_2
- Tukar write (B) dari T_I dengan write (A) dari T_2.
- Tukar write (B) dari T_I dengan Read (A) dari T_2. Setelah bertukar jadwal.

Tabel 5.12 Tabel Jawaban dari Soal Pertanyaan 4

T_1	T_2
Read (A)	
Write (A)	
	Read (A)
Read (B)	
	Write (A)
Write (B)	
	Read (B)
	Write (B)

Konsep kesetaraan konflik mengarah pada konsep serializability konflik. Dengan demikian kita dapat mengatakan bahwa itu adalah kesetaraan konflik. Oleh karena itu konflik tersebut dapat diserialisasikan.

Pertanyaan 5: Manakah dari jadwal berikut ini yang bisa berkonflik? Untuk setiap jadwal serial, tentukan jadwal serial yang setara.

(1) r_1 (x); r_3 (x); w_1(X), r_2(x); w_3(x);

(2) r_1 (x); r_3 (x); w_3(X), w_1(x); r_2(x);

(3) r_3 (x); r_2 (x); w_3(X), r_1(x); w_1(x);

Jawaban :

T_1	T_2	T_3
$r_1(x)$		
		$r_3(x)$
$w_1(x)$		
	$r_2(x)$	
		$w_3(x)$

Kita dapat menukar yang berikut ini :

- Tukar $r_1(x)$ dari T_I dengan r3(x) dari T_3.
- Tukar $r_2(x)$ dari T_2 dengan $r_3(x)$ dari T_3.

Sekarang setelah bertukar kita mendapatkan jadwal

T_1	T_2	T_3
		$r_3(x)$
$r_1(x)$		
$w_1(x)$		
	$r_2(x)$	
		$w_3(x)$

Oleh karena itu jadwal yang diberikan adalah konflik sefializable. (2)

T_1	T_2	T_3
$r_1(x)$		
		$r_3(x)$
		$w_3(x)$
$w_1(x)$		
	$r_2(x)$	

Kita bisa Tukar r_1 (x) dari T_1 dengan r_3(x) dari T_3. Setelah menukar jadwal.

T_1	T_2	T_3
		$r_3(x)$
$r_1(x)$		
		$w_3(x)$
$w_1(x)$		
	$r_2(x)$	

Jadi jadwal yang diberikan adalah jadwal konflik serial karena kita dapat menukar $r_I(x)$ dari T_I dengan $r_3(x)$ dari T_3.

T_1	T_2	T_3
		$r_3(x)$
	$r_2(x)$	
		$w_3(x)$
$r_1(x)$		
$w_1(x)$		

Kita dapat menukar yang berikut ini:

- Tukar $r_3(x)$ dari T_3 dengan $r_2(x)$ dari T_2.
- Tukar $r_2(x)$ dari T_2 dengan $r_I(x)$ dari T_1.

Setelah bertukar jadwal.

T_1	T_2	T_3
	$r_2(x)$	
		$r_3(x)$
		$w_3(x)$
$r_1(x)$		
$w_1(x)$		

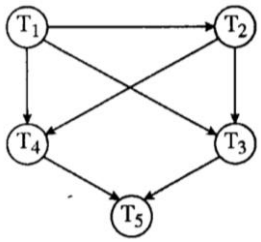
Sekarang jadwal serial baru. T_1, T_2 & T_3 adalah.

$r_2(x)$; $r_3(x)$; $w_3(x)$, $r_I(x)$; $w_I(x)$;

Oleh karena itu jadwal konflik serializable.

Pertanyaan 6: Perhatikan grafik prioritas pada Gambar. Apakah konflik jadwal com!sponding dapat serial? Jelaskan jawabanmu.

Grafik Prioritas



Jawaban : Dalam graf prioritas yang diberikan, himpunan sisi dari graf memiliki salah satu dari tiga kondisi.

$T_1 \rightarrow T_2$ (tepi)

- T_1 mengeksekusi write (Q) sebelum T_2 mengeksekusi read (Q)
- T_1 mengeksekusi read (Q) sebelum T_2 mengeksekusi write (Q)
- T_2 mengeksekusi write (Q) sebelum T_1 mengeksekusi write (Q)

Sebuah Tepi $T_2 \rightarrow T_3$

- T2 dieksekusi write (Q) sebelum T_3 dieksekusi read (Q)
- T2 mengeksekusi read (Q) sebelum T_3 mengeksekusi write (Q)
- T2 mengeksekusi write (Q) sebelum T_3 mengeksekusi write (Q)

Sebuah Tepi $T_3 \rightarrow T_5$

- T_3 mengeksekusi write (Q) sebelum T_5 mengeksekusi read (Q).
- T_3 mengeksekusi read (Q) sebelum T_5 mengeksekusi write (Q).
- T_3 mengeksekusi write (Q) sebelum T_5 mengeksekusi write (Q).

Sebuah Tepi $T_I \rightarrow T_4$

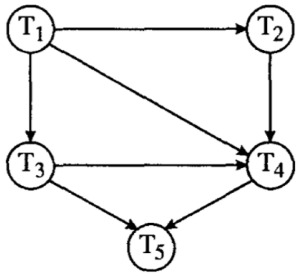
- T_1 mengeksekusi write (Q) sebelum T_4 mengeksekusi read (Q).
- T_I mengeksekusi read (Q) sebelum T_4 mengeksekusi write (Q).
- T_I mengeksekusi write (Q) sebelum T_4 mengeksekusi write (Q).

Sebuah Tepi $T_I \rightarrow T_3$

- T_I mengeksekusi write (Q) sebelum T_3 mengeksekusi read (Q).
- T_I mengeksekusi read (Q) sebelum T_3 mengeksekusi write (Q).
- T_I mengeksekusi write (Q) beforl:; T_3 dieksekusi write (Q).

Sebuah Edge $T_2 \rightarrow T_4$: Karena semua instruksi T_2 dieksekusi sebelum instruksi T_4 dieksekusi. Demikian pula: Sebuah Edge T_4 -T_5 Karena semua instruksi dari T_4 dieksekusi sebelum instruksi dari T_5 dieksekusi. Kita tahu bahwa "Jika grafik prioritas berisi siklus, maka jadwal tidak dapat diserialisasi konflik". Dari uraian di atas grafik prioritas tidak memuat siklus. Oleh karena itu jadwal yang sesuai dari grafik prioritas adalah konflik serial.

Pertanyaan 7: Perhatikan grafik prioritas pada Gbr. Apakah konflik jadwal ko"esponding dapat bersambung? Jelaskan jawaban Anda.



Jawaban : Himpunan sisi pada graf memiliki salah satu dari tiga kondisi.

Ti-Tj edge

- Ti mengeksekusi write (Q) sebelum Tj mengeksekusi read (Q).
- Ti mengeksekusi read (Q) sebelum Tj mengeksekusi write (Q).
- Ti mengeksekusi write (Q) sebelum Tj mengeksekusi write (Q).

Tepi T_1 →Tepi T_2 **:** Karena semua instruksi T 1 dieksekusi sebelum instruksi T2 dieksekusi.

Sisi T_1 →T_3 **:** Karena semua instruksi T 1 dieksekusi sebelum instruksi T 3 dieksekusi.

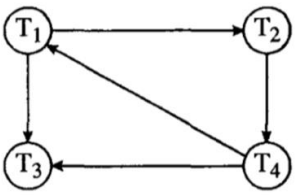
Sisi T_I →T_4 **:** Karena semua instruksi T_I dieksekusi sebelum instruksi T_4 dieksekusi.

Sisi T_2 →T_4 : Karena semua instruksi T_2 dieksekusi sebelum instruksi T_4 dieksekusi.

Sisi T_3 → T_5 : Karena semua instruksi T_2 dieksekusi sebelum instruksi T_5 dieksekusi.

Dari uraian di atas, grafik prioritas yang diberikan tidak mengandung siklus. Karena jika grafik prioritas berisi siklus, maka jadwal yang sesuai tidak dapat diserialisasi konflik. Oleh karena itu jadwal yang sesuai adalah konflik serial.

Pertanyaan 8 : Perhatikan grafik prioritas pada Gambar. Apakah konflik jadwal yang sesuai dapat serial? Jelaskan jawabanmu.



Jawaban: Himpunan sisi pada graf memiliki salah satu dari tiga kondisi.

Sebuah tepi Ti →Tj

- Ti mengeksekusi write (Q) sebelum Tj mengeksekusi read (Q).
- Ti mengeksekusi read (Q) sebelum Tj mengeksekusi write (Q).
- Ti mengeksekusi write (Q) sebelum Tj mengeksekusi write (Q).

Sisi $T_1 \rightarrow T_2$: Karena semua instruksi T_1 dieksekusi sebelum instruksi T_2 dieksekusi.

Sisi $T_1 \rightarrow T_3$: Karena semua instruksi T_1 dieksekusi sebelum instruksi T_3 dieksekusi.

Sisi $T_2 \rightarrow T_4$: Karena semua instruksi T_2 dieksekusi sebelum instruksi T_4 dieksekusi.

Sisi $T_4 \rightarrow T_1$: Semua instruksi T4 dieksekusi sebelum instruksi Tl dieksekusi.

Karena grafik prioritas berisi siklus antara T_1, T_2 & T_4 maka jadwal yang sesuai tidak dapat serial konflik.

Pertanyaan 9 : Jelaskan operasi 'Blind-Writes' dengan bantuan contoh.

T_1	T_2	T_3
Read (A)		
	Write (A)	
Write (A)		
		Write (A)

Pada jadwal di atas, transaksi T_2 & T_3 melakukan operasi write (A) tanpa melakukan operasi read (A). Tulisan semacam ini disebut tulisan buta. **Catatan**: Penulisan buta muncul dalam jadwal apa pun yang dapat diserialisasikan tetapi tidak dalam konflik yang dapat diserialisasi.

Pertanyaan 10: Perhatikan kedua relasi tersebut.

R_I (A, B, C, D)

R_2 (C, D, E, F)

Ukuran relasi R_I adalah 20.000 tupel yang memiliki 10 record/blok penyimpanan sekunder & ukuran R_2 adalah 10.000 tuple yang memiliki 25 record/blok penyimpanan sekunder. Buffer yang dialokasikan ke RI adalah 10 & ke R2 adalah 20. Temukan jumlah transfer blok yang mungkin diperlukan untuk menghitung join dari relasi ini. Asumsikan metode penggabungan apa pun, tetapi nyatakan metode yang diasumsikan.

Jawaban: Diketahui:

R_I (A, B, C, D)

R_2 (C, D, E, F)

Ukuran R_I = 20,000 tuples

bf R_I = 10

Ukuran R_2 = 10,000 tuples

bf R_2 = 25

b (ukuran penyangga tersedia) = b_1 = 10

b_2=20

$$Total\ jumlah\ blok = \left[\frac{R1}{bf\ R1}\right] + \left[\frac{1}{b1} x\ \frac{1}{b2} x\ \frac{R2}{bf\ R2}\right]$$

$$= \left[\frac{20000}{10}\right] + \left[\frac{1}{10} x\ \frac{20000}{10} x\ \frac{1}{20} x \frac{1000}{25}\right]$$

= 2000 + 4000

= 6000 blok

Ukuran join alami pada R_1 & $R_2 \leq |R_1| * |R_2|$

Pertanyaan 11: Estimasi biaya dan optimasi transfer tuple untuk join di database terdistribusi.

Jawaban: Dalam sistem terdistribusi, beberapa faktor tambahan semakin memperumit pemrosesan kueri. Yang pertama adalah biaya transfer data melalui jaringan. Data ini mencakup file perantara yang ditransfer ke situs lain untuk diproses lebih lanjut, serta file hasil akhir yang mungkin harus ditransfer ke situs tempat hasil kueri diperlukan.

Meskipun ini terhubung melalui jaringan area lokal berkinerja tinggi, mereka menjadi cukup signifikan dalam jenis jaringan lain. Oleh karena itu, algoritma pengoptimalan kueri DDBMS mempertimbangkan tujuan pengurangan jumlah transfer data sebagai kriteria pengoptimalan dalam memilih strategi eksekusi kueri terdistribusi. Kami mengilustrasikan ini dengan contoh kueri sederhana. Misalkan hubungan EMPLOYEE dan DEPARTMENT terdistribusi pada gambar.

SITE 1

EMPLOYEE

FNAME	MNAME	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	DNO

10.000 catatan Setiap catatan panjangnya 100 byte Bidang SSN panjangnya 9 byte FNAME bidang panjangnya 15 byte, panjang LNAME 15 byte, bidang DNO panjangnya 4 byte

SITE 2

DEPARTMENT

DNAME	DNUMBER	MGRSNN	MGRSTARTDATE

100 record, setiap record panjangnya 35 byte field DNUMBER panjangnya 4 byte, DNAME panjangnya 10 byte, field MGRSSN panjangnya 9 byte. Kami akan mengasumsikan dalam contoh ini bahwa tidak ada relasi yang terfragmentasi. Berdasarkan gambar, ukuran relasi EMPLOYEE adalah 100 X 10.000 = 10^6 byte, dan ukuran relasi DEPARTMENT adalah 35 X 100 = 3500 byte. Pertimbangkan kueri Q "Untuk setiap karyawan, ambil nama karyawan dan nama departemen tempat karyawan tersebut bekerja"

Hasil dari query ini akan mencakup 10.000 record, dengan asumsi bahwa setiap karyawan berhubungan dengan sebuah departemen. Misalkan setiap record dalam hasil query panjangnya 40 byte. Kueri dikirimkan di situs berbeda 3, yang disebut situs hasil karena hasil

kueri diperlukan di sana. Baik hubungan EMPLOYEE maupun DEPARTMENT berada di situs 3. Ini adalah tiga strategi sederhana untuk mengeksekusi kueri terdistribusi ini:

1. Transfer relasi EMPLOYEE dan DEPARTMENT ke situs hasil, dan lakukan join di situs 3. Dalam hal ini total 1.000.000 + 3500 = 1.003.500 byte harus ditransfer.
2. Transfer relasi EMPLOYEE ke situs 2, jalankan join di situs 2, dan kirim hasilnya ke situs 3. Ukuran hasil query adalah 40 X 10.000 = 400.000 byte jadi 400.000 + 1.000.000 = 1.400.000 byte harus ditransfer.
3. Transfer relasi DEPARTMENT ke situs 1, jalankan join di situs 1, dan kirim hasilnya ke situs 3.

Dalam hal ini 400.000 + 3500 = 403.500 byte harus ditransfer. Dalam meminimalkan jumlah transfer data adalah kriteria optimasi kami, kami harus memilih strategi 3.

Misalkan situs hasil adalah situs 2, maka kita memiliki dua strategi sederhana :

1. Transfer relasi EMPLOYEE ke situs 2, jalankan kueri, dan sajikan hasilnya kepada pengguna di situs 2. Namun jumlah byte yang sama harus 1.000.000 ditransfer untuk Q dan Q^{I}.
2. Transfer relasi DEPARTMENT ke situs 1, jalankan query di situs 1, dan kirim kembali hasilnya ke situs 2. Dalam hal ini 400.000 + 3500 = 403.500 byte harus ditransfer untuk Q dan 4000 + 3500 = 7500 byte untuk Q^{I}.

5.10 REVIEW PERTANYAAN

1. Apa yang Anda maksud dengan eksekusi transaksi secara bersamaan? Apa itu kunci? Jelaskan protokol penguncian dua fase secara singkat.
2. Apa itu kontrol konkurensi? Apa tujuannya?
3. Apa saja jenis kunci yang berbeda?
4. Apa itu stempel waktu? Diskusikan teknik pemesanan stempel waktu untuk kontrol konkurensi
5. Buat daftar fitur yang menonjol dari protokol penguncian dua fase. Buktikan bahwa penguncian dua fase memastikan serializability .
6. Perhatikan transaksi berikut:

   T_I Read (A)
   Read (B)
   If A = ° then B : = B + 1
   Write (B)
   T_2 Read (B)
   Read (A)
   If B = 0 then A : = A + 1
   Write (A)

   a. Tambahkan instruksi kunci dan buka kunci ke transaksi Tl dan T2, sehingga mereka mematuhi protokol penguncian dua fase.
   b. Apakah pelaksanaan transaksi tersebut dapat mengakibatkan Deadlock?
7. Apa itu penguncian multi-perincian? Apa perbedaan antara penguncian implisit dan eksplisit dalam penguncian multi-perincian?

8. Apa perbedaan antara kunci eksklusif dan kunci bersama? Jelaskan dengan contoh.
9. Jelaskan skema kontrol konkurensi berdasarkan protokol timestamping.
10. Jelaskan teknik kontrol konkurensi validasi.
11. Apakah yang Anda maksud: skema multiversi Jelaskan, apa yang dimaksud dengan Timestamp-W dan Timestamp-R dari objek data versi ke-N.
12. Apa metode optimistis dari kontrol konkurensi? Diskusikan fase yang berbeda di mana transaksi bergerak selama kontrol optimis.
13. Daftar keuntungan, masalah dan aplikasi metode optimistis kontrol konkurensi.
14. Apa itu Database terdistribusi? Jelaskan dengan diagram yang rapi.
15. Apa keuntungan dan kerugian utama dari database terdistribusi.
16. Apa yang dimaksud dengan arsitektur sistem database terdistribusi? Apa saja jenis arsitektur yang berbeda? Diskusikan masing-masing dengan sketsa yang rapi.
17. Apa saja macam-macam database terdistribusi? Diskusikan secara rinci.
18. Apa itu DDBS homogen? Jelaskan secara detail.
19. Apa itu DDBS heterogen? Jelaskan secara detail.
20. Apa itu fragmenting relasi? Apa jenis utama dari fragmen data? Mengapa fragmentasi IS merupakan konsep yang berguna dalam desain database terdistribusi?
21. Apa itu fragmentasi data horizontal? Jelaskan dengan sebuah contoh.
22. Apa itu fragmentasi data vertikal? Jelaskan dengan sebuah contoh.
23. Apa itu fragmentasi data campuran? Jelaskan dengan sebuah contoh.
24. Apa itu replikasi data? Mengapa replikasi data berguna dalam DDBMS? Apa unit khas data yang direplikasi?
25. Apa itu alokasi data? Membahas.
26. Apa yang dimaksud dengan replikasi data? Apa kelebihan dan kekurangannya?
27. Jelaskan perbedaan antara berikut ini:
    a. DDBMS homogen dan DDBMS heterogen.
    b. Fragmentasi horizontal dan fragmentasi vertikal.
28. Jelaskan perbedaan antara fragmentasi, transparansi, transparansi replikasi dan transparansi lokasi.
29. Pertimbangkan kegagalan yang terjadi selama dua fase komit untuk transaksi di lingkungan terdistribusi. Jelaskan protokol komit dua fase.
30. Jelaskan pengertian sistem transaksi! Bagaimana proses transaksi dilakukan dalam database terdistribusi
31. Jelaskan algoritme: Protokol read-sebelum-tulis.
32. Tulislah catatan singkat tentang hal-hal berikut:
    a. Basis data terdistribusi
    b. Fragmentasi data
    c. Alokasi data
    d. Replikasi data
    e. Stempel waktu
    f. Dua fase komit protokol.

LAMPIRAN A

LAB 1

Pertanyaan 1: Buat tabel siswa yang strukturnya adalah:

Column Name	Data Type	Size
Roll-No	Number	6
Name	Char	20
Age	Number	4
Sex	Char	7
Branch	Char	10
Ph-no	Number	10
Address	Varchar	30
City	Char	15
State	Char	15
Pin code	Number	6

Jawaban: CREATE TABLE STUDENT (Nomor Roll-no (6), Name char (20), Age number (4), Sex char (7), Branch char (10), Ph-no number (10), Address varchar (30), Karakter kota (15), Karakter negara (15), Nomor kode pin (6);

Output: Tabel dibuat

Pertanyaan 2: Buat tabel EMPLOYEE yang strukturnya :

Column Name	Data Type	Size
Emp-id	Varchar	6
Ename	Char	20
Sex	Char	7
Address	Varchar	30
City	Char	15
State	Char	15
Dept-no	Varchar	7
Salary	Number	(9, 2)

Jawaban: CREATE TABLE EMPLOYEE (Emp-id varchar (6), Ename char (20), Sex char (7), Address varchar (30), City char (15), State char (15), Dept-no varchar (7), Nomor Gaji (9, 2);

Output : Tabel dibuat.

Pertanyaan 3 : Buat tabel CLIENT-MAST yang strukturnya:

Column Name	Data Type	Size
Client-no	Varchar	5
Name	Char	15
Address1	Varchar	20
Address2	Varchar	20
City	Char	15
State	Char	15
Pin code	Number	6
Ph-no	Number	10
Bal-due	Number	10, 2

Jawaban: CREATE TABLE CLIENT-MAST (Client-no varchar (5), Name char (15), Address1 varchar (20), Address2 varchar (20), City char (15), State char (15), Nomor kode pin (6), nomor tlp (10), nomor tilang (10, 2));

Output : Tabel dibuat.

Pertanyaan 4: Buat PRODUCT_MAST yang strukturnya adalah:

Column Name	Data Type	Size
Product-no	Varchar	5
Description	Varchar	30
Profit-perc	Number	4, 2
Unit	Varchar	10
Qty-available	Number	9
Sell-price	Number	9, 2
Cost-price	Number	9, 2

Jawaban : CREATE TABLE PRODUCT MAST (Product-no varchar (5), Description varchar (30), Profit-perc number (4, 2), Unit varchar (10), Oty-available number (9), Sell-price number (22), Nomor harga-biaya (9, 2));

Output : Tabel dibuat.

LAB 2

Pertanyaan 1: Masukkan nilai ke tabel siswa:

INSERT KE DALAM NILAI SISWA
(&No Roll-no, &Nama, &Usia, &Jenis Kelamin, &Cabang, &No Telp, &Alamat, &Kota, &Negara Bagian, &Kode PIN);

Jawaban :

Masukkan nilai untuk Roll-no : 1
Masukkan nilai untuk Nama : Agus
Masukkan nilai untuk Usia : 28
Masukkan nilai untuk Sex : Laki-Laki
Masukkan nilai untuk Cabang : Siliwangi
Masukkan nilai untuk Ph-no : 0812300000
Masukkan nilai untuk Alamat : no. 10
Masukkan nilai untuk Kota : Semarang
Masukkan nilai untuk Negara : Indonesia

Masukkan nilai untuk kode Pin : 101010

Output: 1 baris dibuat.

Pertanyaan 2: Masukkan nilai ke tabel EMPLOYEE :

INSERT INTO EMPLOYEE NILAI

(&Emp-id, &Name, &Sex, &~ddress, &City, &State, &Dept-no, &Gaji);

Jawaban :

Masukkan nilai untuk Emp-id : C0001
Masukkan nilai untuk : Agus Wibowo
Nama Masukkan nilai untuk Jenis Kelamin : Laki-laki
Masukkan nilai untuk Alamat : 01
Masukkan nilai untuk Kota : Semarang
Masukkan nilai untuk Tahap : IDN
Masukkan nilai untuk Dept-no : CS10
Masukkan nilai untuk Gaji : 15.000.000

Output: 1 baris dibuat

Pertanyaan 3 : INSERT INTO CLIENT-MAST VALUES

(&No-Klien, &Nama, &Alamat, &Alamat2, &Kota, &Negara Bagian, &Kode Pin, &NoNo Telp, &Batal jatuh tempo);

Pertanyaan 4: INSERT INTO PRODUCT-MAST VALUES

(&NoProduct-no, &Description, &Profit, &Unit, &Qty-available, &Sell-price, &Cost-price)

LAB 3

Ambil catatan dari tabel:

Pertanyaan 1 :

(a) Cari tahu roll-no, nama-nama semua siswa.
(b) Ambil kembali seluruh isi tabel siswa.
(c) Mengambil daftar nama, alamat, cabang, kota semua siswa.
(d) Daftar semua siswa yang negara bagiannya adalah Seamarang.

Jawaban :

(a) SELECT roll-no, nama dari siswa;
(b) SELECT * dari siswa;
(c) SELECT nama, alamat, cabang, kota dari siswa;
(d) SELECT * dari siswa
WHERE State = Semarang;

Pertanyaan 2 :

(a) Cari tahu nama-nama klien-no semua klien.
(b) Ambil seluruh isi tabel CLIENT-MAST.
(c) Daftar semua klien yang berlokasi di INDONESIA.
(d) Cari klien-no, nama, alamat, alamat2 klien yang memiliki bal-due lebih besar.

Jawaban :

(a) SELECT client-no, name from CLIENT-MAST;

(b) SELECT * FROM CLIENT-MAST;

(c) SELECT * FROM CLIENT-MAST
WHERE State = ‘Semarang’

(d) PILIH client-no, name, address1, address2 dari CLIENT-MAST
WHERE Bal-due > 500;

Pertanyaan 3:

(a) Ubah kota klien-no 'l15A' menjadi ‘SEMARANF”.

(b) Ubah saldo piutang klien-no '118A' menjadi Rp. 5000

(c) Buat daftar berbagai produk yang tersedia dari tabel tiang produk.

(d) Hapus semua produk dari PRODUCT-MAST di mana kuantitas yang tersedia sama dengan 200.

(e) Hapus dari CLIENT-MAST di mana status kolom menyimpan nilai 'u.P.'

Jawaban :

(a) UPDATE CLIENT-MAST SET City = ‘Semarang’
WHERE Client-no = 'l1SA';

(b) UPDATE CLIENT-MAST SET Bal-due = 5000
WHERE Client-no = '118A';

(c) SELECT deskripsi FROM PRODUCT-MAST;

(d) DELETE FROM PRODUCT-MAST
WHERE Oty-available = 200;

(e) DELETE FROM CLIENT-MAST
WHERE State = 'Up';

Pertanyaan 4:

(a) Tambahkan kolom 'mob-num' dengan tipe data 'number' dan size = '10' pada tabel CLIENT-MAST.

(b) Ubah ukuran kolom harga pokok produk menjadi 10, 2.

(c) Hancurkan tabel PRODUCT-MAST beserta datanya.

(d) Hancurkan tabel client-mast beserta datanya.

(e) Ubah nama tabel siswa menjadi tiang siswa.

(f) Ubah nama tabel PRODUCT-MAST menjadi PRO-MASTER

Jawaban :

(a) ALTER TABLE CLIENT-MAST ADD (Nomor mob (10));

(b) ALTER TABLE PRODUCT-MAST
MODIFY (Nomor harga-biaya (10, 2));

(c) DROP TABLE PRODUCT-MAST;

(d) DROP TABLE CLIENT-MAST

(e) RENAME student menjadi student-mast

(f) RENAME RPODUCT-MAST menjadi PRO-MASTER;

LAB 4

Pertanyaan 1: Ambil isi kolom produksi-no, deskripsi, profit dan hitung 5% dari nilai yang terdapat pada kolom harga jual dan 105% dari nilai yang terdapat pada kolom harga jual untuk setiap baris dari tabel PRODUCT -TIANG KAPAL.

Jawaban : PILIH Produk-no, deskripsi, keuntungan, Harga Jual * 0,05, Harga Jual * 1,05 DARI PRODUK-MAST

Output :

Product-no	Description	Profit	Sell-price * 0.05	Sell-price * 1.05
110A	Hard disk	300	500	9500
111A	DVD writer	400	200	7200
112A	CD writer	200	300	6300
113A	Monitors	700	800	13800
114A	Mouse	100	40	1140

Pertanyaan 2 : Ambil isi kolom product-no, description & count 5% & 105% dari harga jual field untuk setiap baris yang diambil. Ganti nama sell-price * 0,05 sebagai kenaikan dan sell-price * 1,05 sebagai new-price.

Jawaban : PILIH Produk-no, deskripsi, harga jual * kenaikan 0,05, harga jual * harga baru 1,05, DARI PRODUK-MAST;

Output :

Product-no	Description	Increase	New-price
110A	Hard disk	300	9500
111A	DVD writer	400	7200
112A	CD writer	200	6300
113A	Monitors	700	13800
114A	Mouse	100	1140

Pertanyaan 3 : Ambil isi kolom product-no, description, profit, sell-price, cost-price dari tabel PRODUCT-MAST dimana nilai yang terdapat pada field profit antara 500 & 1000 keduanya sudah termasuk.

Jawaban :

```
SELECT produk-no, deskripsi, keuntungan, harga jual, harga pokok
FROM PRODUK-MAST
WHERE Laba > = 500 DAN Laba < = 1000;
```

Pertanyaan 4: Ambil informasi karyawan seperti emp-id, ename, alamat, kota, negara bagian, dept-no dan gaji untuk semua karyawan yang gaji lapangannya 10000 atau 15000.

Jawaban :

```
SELECT Emp-id, ename, address, city, state, dept-no, salary
FROM KARYAWAN(EMPLOYEE)
WHERE (Gaji = 10.000 ATAU Gaji = 15000);
```

Pertanyaan 5: Ambil informasi karyawan yang ditentukan untuk karyawan yang TIDAK berada di 'SEMARANG' atau 'SALATIGA'

Jawaban :

```
SELECT Emp-id, ename, address, city, state, dept-no, salary
FROM KARYAWAN
WHERE NOT (Kota = 'SEMARANG' ATAU Kota = 'SALATIGA');
```

Pertanyaan 6: Ambil roll-no, nama, umur, cabang, ph-no, alamat, kota, negara bagian dari tabel student dimana nilai yang terdapat pada field roll-no antara 15 dan 50 keduanya termasuk

Jawaban : SELECT rol no, nama, umur, cabang, no tlp, alamat, kota, negara bagian
FROM STUDENT
WHERE Roll-no 15 DAN 50; .

Pertanyaan 7 : Ambil emp-id, ename, address, city, salary dari tabel EMPLOYEE dimana nilai yang terdapat pada field emp-id tidak antara 115 dan 130 keduanya inklusif.

Penjelasan : SELECT emp-id, ename, alamat, kota, gaji
FROM EMPLOYEE
WHEREemp-id NOT BETWEEN 115 AND 130;

Pertanyaan 8 : Ambil roll-no, nama, alamat, kota dan negara bagian dari tabel siswa dimana nama siswa adalah Azeda atau Ina atau Nico atau Kenny atau Ibnu.

Jawaban : SELECT Roll-no, name, address, city, state
FROM student
WHERE Name IN ('Azeda, 'Ina', 'Nico', 'Kenny', 'Ibnu');

LAB 5

Pertanyaan 1 : Buat tabel CLIENT-MAST sedemikian rupa sehingga isi kolom client-no adalah uniquekey di seluruh kolom.

Jawaban : CREATE TABLE CLIENT-MAST
(Client-no, varchar (7) UNIQUE, Name char (15), Address varchar (30), City char (15), State char (15), Bal-due number (9, 2));

Pertanyaan 2: Buat tabel siswa sehingga isi kolom roll-no adalah primary key di seluruh kolom.

Jawaban : CREATE TABLE Student
(Roll-no number (7) PRIMARY KEY, Name char (15), Branch char (15), Address varchar (30), City char (15), State char (15));

Pertanyaan 3: Buat tabel SALES-MAST dimana terdapat composite primary key pada kolom order-no dan product-no. Karena batasan membentang di seluruh kolom, jelaskan di tingkat tabel.

Column Name	Data Type	Size
Order-no	Varchar	5
Product-no	Varchar	5
Rate	Number	(8, 2)
Product-name	Char	15
Qty-order	Number	7

Jawaban : CREATE TABLE SALES-MAST
(Order-no varchar (5), Product-no varchar (5), Rate number (8, 2), Product-name char (15), Qty-order number (7), PRIMARY KEY (Order-no, product-no));

Pertanyaan 4: Buat tabel CLIENT-MAST dengan batasan sebagai berikut:

(i) Nilai data yang disisipkan ke dalam nama kolom harus huruf kecil saja.

(ii) Nilai data yang disisipkan pada kolom client-no harus dimulai dengan huruf kecil 'a'.

(iii) Hanya izinkan "Indonesia", "Semarang", 'Greater Jakarta" sebagai nilai sah untuk kota kolom.

Jawaban : CREATE TABLE CLIENT-MAST
(Client-no varchar (5), Name char (15), Address varchar (30), City char (15), State char (15), Bal-due number (9, 2), CHECK (Client-no like 'a%'), CHECK (Name = lower (name)), CHECK (City IN ('Indonesia', 'Jakarta', 'Greater Jakarta'));

Pertanyaan 5 : Buat tabel CLIENT-MAST dengan batasan centang sebagai berikut:

(i) Nilai data yang dimasukkan ke dalam kolom client-no harus diawali dengan huruf kapital W.

(ii) Nilai data yang dimasukkan ke dalam nama kolom harus di atas kasus saja.

(iii) Hanya izinkan "Jakarta", "Indonesia", "Greater Jakarta", "''.

Jawaban : CREATE TABLE CLIENT-MAST
(Client-no varchar (7), Name char (15), Address varchar (30), City char (15), State char (15), Bal-due number (10,2), CHECK (Client-no like 'A%'), CHECK (Name = Upper (Name)), CHECK (City IN ('Jakarta', 'Indonesia', 'Greater Jakarta', ''));

Pertanyaan 6: Membuat tabel sales-order 1 table dengan primary key sebagai deltorder-no dan product-no, foreign key deltorder-no, referensi kolom order-no pada tabel sales-order,

Jawaban : CREATE TABLE Sales-order 1
(Deltorder-no varchar (6), REFERENCES sales-order, Product-no varchar (5), Qty number (7), Rate number (8, 2), PRIMARY KEY (Deltorder-no, Product-no));

Pertanyaan 7 : Buat tabel seperti di bawah ini : Nama Tabel: CLIENT-MAST

Jawaban : CREATE TABLE CLIENT-MAST
(Client-no varchar (6) PRIMARY KEY, Name char (20) NOT NULL, Address varchar (30), City Char (15), State char (15), Bal-due number (8, 2), CONSTRAINT CK_Client CHECK (Client-no like 'A%'));

Pertanyaan 8: Buat tabel seperti di bawah ini : Nama Tabel: PRODUCT_MAST

Table Name : PRODUCT_MAST

Column Name	Data Type	Size	Attribtues
Product-no	Varchar	6	Primary key/First letter must start with A
Product-name	Char	20	NOT NULL
Sell-price	Number	8, 2	NOT NULL cannot be 0
Cost-price	Number	8, 2	NOT NULL cannot be 0
Profit-no	Number	4, 2	NOT NULL

Jawaban : CREATE TABLE PRODUCT_MAST

(Product-no varchar (6) PRIMARY KEY, Product-name char (20) NOT NULL, Sell-price number (8, 2) NOT NULL, Cost-price number (8,2) NOT NULL, Profit-number (4, 2) NOT NULL, CONSTRAINT CK-product CHECK (Product-no like 'A%'), CONSTRAINT CK-sell CHECK (Sell-price < > 0), CONSTRAINT CK-cost CHECK (Cost-price < > 0));

Pertanyaan 9: Nama Tabel: SALES-MAST

Column Name	Date Type	Size	Attributes
ID	Varchar	6	Primary key/First letter must start with 'M'
Name	Char	20	NOT NULL
Address	Varchar	30	NOT NULL
City	Char	15	
State	Char	15	
Salary	Number	(8, 2)	NOT NULL cannot be 0

Jawaban : CREATE TABLE SALES-MAST

(ID varchar (6) PRIMARY KEY, Name char (20) NOT NULL, Address varchar (30) NOT NULL, City char (15), State char (15), Salary number (8, 2) NOT NULL, CONSTRAINT CK-sales CHECK (ID like 'M%'), CONSTRAINT CK-sales CHECK (Salary < > 0));

LAB 6

Pertanyaan 1 : Masukkan data berikut ke dalam tabel masing-masing.

Data untuk tabel CLIENT-MAST

Client-no	Name	Address	City
A0001	Dani Sasmoko	H-327	Jepara
A0002	Lawren Setyono	F-119	Jakarta
A0003	Agus Sasmoko	L-62	Jakarta
A0004	Joni Setyono	L-179	Bandung
A0005	Ridlo Setyono	H-310	Jakarta Selatan
A0006	Widodo Setyono	H-431	Kudus

State	Bal-due
Indonesia	3000.00
Indonesia	5000.00
Indonesia	4500.80
Indonesia	5000.90
Indonesia	1500.35
Indonesia	1000.00

Jawaban :

INSERT INTO CLIENT-MAST

(Client-no, Name, Address, City, State, Bal-due)

VALUES ('A000l', ‘Agus Wibowo’, 'H-327', ‘Jepara’, ‘Indonesia’ 3000.00);
INSERT INTO CLIENT-MAST
(Client-no, Name, Address, City, State, Bal-due)
VALUES ('A0002', Lawren Setyono, 'F-119', ‘Jakarta’, ‘Indonesia’, 5000.00);
INSERT INTO CLIENT-MAST
(Client-no, Name, Address, City, State, Bal-due)
VALUES ('A0003', ‘Agus, 'L-62', ‘Jakarta’, ‘Indonesia’, 4500.80);
INSERT INTO CLIENT-MAST
(Client-no, Name, Address, City, State, Bal-due)
VALUES ('A0004', ‘Joni Setyono’, 'L-179', ‘Bandung’, ‘Indonesia’, 5000.90); INSERT INTO CLIENT-MAST
(Client-no, Name, Address, City, State, Bal-due)
VALUES ('A0005', 'ridlo setyono’, 'H-31O', ‘Jakarta Selatan’, ‘Indonesia’.', 1500.30);
INSERT INTO CLIENT-MAST
(Client-no, Name, Address, City, State, Bal-due)
VALUES ('A0006', ‘Widodo Setyono’, 'H-431' ‘Kudus’, ‘Indonesia’ 1000.00);

Pertanyaan 2: Masukkan data berikut ke dalam tabel masing-masing : Data untuk tabel PRODUCT-MAST

Product-no	Prod-name	Profit-per	Sell-price	Cost-price
A0001	Monitors	6	12000	11280
A0002	HDD	4	8000	8000
A0003	CD writer	2.5	5250	5100
A0004	Mouse	5	1050	1000
A0005	Keyboard	10	3150	3050
A0006	Floppics drive	5	1050	1000

Jawaban :

INSERT INTO PRODUCT-MAST
(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (AOOO1', 'Monitors', 6, 12000, 11280);
Output : 1 baris dibuat
INSERT INTO PRODUCT-MAST
(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (A0002', 'HDD', 4, 8400, 8000);
Output : 1 baris dibuat.
INSERT INTO PRODUCT-MAST
(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (A0003', 'CD writer', 2.5, 5250, 5100);
Output : 1 baris dibuat.
INSERT INTO PRODUCT-MAST
(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (A0004', 'Mouse', 5, 1050, 1000);

Output : 1 baris dibuat.

INSERT INTO PRODUCT-MAST

(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (A0005', 'Keyboard', to, 3150, 3050);

Output : 1 baris dibuat.

INSERT INTO PRODUCT-MAST

(Product-no, Prod-name, Profit-per, Sell-price, Cost-price) VALUES (A0006', 'Floppies Drive', 5, 1050, 1000);

Output : 1 baris dibuat.

Pertanyaan 3: Masukkan data berikut ke dalam tabel masing-masing : Data SALES-MAST

ID	Name	Address	City	State	Salary
M0001	Dani	H-327	Jepara	Indonesia	30,000
M0002	Agus	L-62	Jakarta	Indonesia	30,000
M0003	Lawren	H-431	Jakarta Sel.	Indonesia	25,000
M0004	Faruq	F-530	Jakarta Sel.	Indonesia	25,000
M0005	Ridlo	G-300	Jakarta	Indonesia	25,000
M0006	Widodo Joni	N-310	Jepara	Indonesia	20,000

Jawaban :

INSERT INTO SALES-MAST

(ID, Name, Address, City, State, Salary)

VALUES ('MOOOI', ‘Dani’, 'H-327',’Jepara’, ‘Indonesia’, 30000);

Output : 1 baris dibuat

INSERT INTO SALES-MAST

(10, Name, Address, City, State, Salary)

VALUES ('M0002', ‘Dani’, 'L-62', ‘Jakarta’, ‘Indonesia’ 30000);

Output : 1 baris dibuat

INSERT INTO SALES-MAST

(10, Name, Address, City, State, Salary)

VALUES ('M0003', ‘Lawren, 'H-431', ‘Jakarta Selatan’, ‘Indonesia’, 25000);

Output : 1 baris dibuat.

INSERT INTO SALES-MAST

(10, Name, Address, City, State, Salary)

VALUES ('M0004', ‘Faruq’, 'F-530',’Jakarta Selatan’, ‘Indonesia’', 25000);

Output : 1 baris dibuat.

INSERT INTO SALES-MAST

(ID, Name, Address, City, State, Salary)

VALUES ('MOOOS', ‘Ridlo’, 'G-300', Jakarta, ‘Indonesia’, 25000);

Output : 1 baris dibuat.

INSERT INTO SALES-MAST

(10, Name, Address, City, State, Salary)
VALUES ('M0006', ‘Joni’, 'N-310', ‘Jepara’, ‘Indonesia’, 20000);

Output : 1 baris dibuat

LAB 7

Pertanyaan 1: Berikan setiap karyawan bonus 20%. Hitung jumlah 20% berdasarkan nilai yang ada di kolom gaji tabel karyawan dan perbarui nilai yang ada di kolom gaji bersih.

Jawaban : UPDATE Employee
SET Net-salary = Net-salary * Salary * 0.20;

Pertanyaan 2: Update tabel SALES-MAST ubah isi field name menjadi Andi dan isi field address menjadi G-119 Andi Apartments untuk record diidentifikasi dengan field ID yang berisi nilai 'M0001'.

Jawaban : UPDATE SALES-MAST
SET Name = 'Dani Kuma Andi’, Address = 'G-119 Andi Apartments'
WHERE ID = ‘M0001’;

Pertanyaan 3: Tambahkan field no ponsel, yaitu field yang dapat menampung angka hingga 10 digit pada tabel siswa.

Jawaban : ALTER TABLE Student
ADD (Mobile-no number (10));

Pertanyaan 4: Ubah field mobile-no tabel siswa menjadi nilai baru maksimal 12 digit.

Jawaban : ALTER TABLE Student
MODIFY (Mobile-no number (12));

Pertanyaan 5: Cari tahu semua informasi tentang salesman yang namanya diawali huruf 'Sa' dari tabel SALES-MAST

Jawaban : SELECT * FROM SALES-MAST
WHERE Name LIKE 'Sa%';

Pertanyaan 6: Ambil semua informasi tentang siswa yang namanya dimulai dengan huruf 'V' dari tabel siswa.

Jawaban : SELECT * FROM Student
where Name LIKE 'V%';

Pertanyaan 7: Retrieve Roll-no, Name, Address, City, State tentang siswa yang karakter kedua namanya 'a' atau 'i'.

Jawaban : SELECT Roll-no, Name, Address, City, State from student
WHERE Name LIKE '-a%' OR Name LIKE '-i%';

LAB 8

Pertanyaan 1 : Buat tabel SUPPLIER_MAST dari CLIENT-MAST. Pilih semua bidang, ganti nama client-no dengan supplier-no dan beri nama dengan sname.

Jawaban : CREATE TABLE SUPPLIER-MAST
(Supplier-no, Sname, Address, City, State, Bal-due)
AS SELECT Client-no, Name, Address, City, State, Bal-due FROM CLIENT-MAST;

Pertanyaan 2 : Buat tabel EMP-NAME dari karyawan. Pilih semua bidang, ganti nama emp-id dengan EMP-ID dan Ename dengan Emp-name.

Jawaban : CREATE TABLE EMP-MASTER
(Emp-ID, Emp-name, Sex, Address, City, State, Dept-no, Salary)
AS SELECT Emp-Id, Ename, Sex, Address, City, State, Dept-no, Salary FROM Employee;

Q**Pertanyaan 3:** Masukkan record ke dalam tabel SUPPLIER-MAST dari tabel CLIENT-MAST.

Jawaban : INSERT INTO SUPPLIER-MAST
SELECT Client-no, Name, Address, City, State, Bal-due
FROM CLIENT-MAST;

Pertanyaan 4 : Memasukkan record ke dalam tabel EMP-MASTER dari table employee.

Jawaban : INSERT INTO EMP-MASTER
SELECT Emp-id, Ename, Sex, Address, City, State, Salary
FROM Employee;

LAB 9

Pertanyaan 1 : Ambil semua pesanan yang dilakukan oleh nama Klien 'Agus Wibowo' dari tabel SALES-ORDER.

Table Name : SALES-ORDER

Order No	Client_No	Order_Date
B0001	A0001	10-Jan-07
B0002	A0001	12-Feb-07
B0003	A0003	24-May-07
B0004	A0004	30-Jun-07
B0005	A0005	12-July-07
B0006	A0006	15-Aug-07

Table Name : CLIENT-MAST

Client-No	Name	Bal-due
A0001	Dani Sasmoko	500
A0002	Agus Sasmoko	1000
A0003	Lawren Setyono	300
A0004	Ridlo Setyono	700
A0005	Faruq Setyono	800
A0006	Widodo Setyono	200

Jawaban : SELECT * FROM SALES-ORDER
WHERE Client-no = (SELECT Client-no FROM CLIENT- MAST/WHERE Name ‘Agus Wibowo’);

Output :

Order-No	Client-No	Order-date
B0001	A0001	10-Jan-07
B0002	A0001	12-Feb-07

Pertanyaan 2: Cari tahu semua produk yang tidak dijual dari tabel PRODUCT-MAST, berdasarkan produk yang benar-benar dijual seperti yang terlihat pada tabel SALES-ORDER.

Nama Tabel: SALES-ORDER

Detorder-no	Product-no	Qty-order
D0001	A0001	10
D0001	A0002	4
D0001	A0003	5
D0002	A0005	8
D0002	A0001	7
D0003	A0002	5
D0004	A0005	4
D0005	A0003	6
D0006	A0005	7

Nama Tabel: PRODUCT-MAST

Product-no	Prod-name
A0001	Monitors
A0002	HDD
A0003	CD writer
A0004	Mouse
A0005	Keyboard
A0006	Floppies drive
A0007	Floppies

Jawaban : SELECT Product-no, Prod-name
FROM PRODUCf-MAST
WHERE Product-no NOT IN
(SELECT Product-no FROM SALES-ORDER);

Output :

Product-no	Prod-name
A0004	Mouse
A0006	Floppies drive
A0007	Floppies

Pertanyaan 3 : Ambil nomor produk dan jumlah total yang dipesan untuk setiap produk dari tabel SALES-ORDER.

Nama Tabel: SALES-ORDER

Detorder-no	Product-no	Qty-order
D0001	A0001	7
D0001	A0004	9
D0001	A0006	8
D0002	A0002	3
D0002	A0002	4
D0003	A0005	6
D0004	A0003	4
D0005	A0001	3
D0005	A0006	2
D0006	A0004	9

Jawaban : SELECT Product-no, Sum (Qty-order)
"Total Qty Ordered"
FROM SALES-ORDER
GROUP BY Product-no;

Output :

Product-no	Total Qty-order
A0001	10
A0002	7
A0003	4
A0004	18
A0005	6
A0006	10

Pertanyaan 4: Ambil no produk dan jumlah total yang dipesan untuk no produk A0002', A0006' dari tabel SALES-ORDER.

Nama Tabel: SALES-ORDER

Detorder	Product-no	Qty-order
D0001	A0001	7
D0002	A0004	9
D0002	A0006	8
D0001	A0002	3
D0003	A0002	4
D0002	A0005	6
D0004	A0003	4
D0005	A0001	3
D0005	A0006	2
D0006	A0004	9

Jawaban : SELECT Product-no, Sum (Qty-order)
"Total Qty order"
FROM SALES-ORDER
GROUP BY Product-no
HAVING Product-no = A0002' OR

Product-no = A0006'

Output:

Product-no	Total Qty Order
A0002	7
A0006	10

Pertanyaan 5 : Ambil nomor pesanan, client-no dan salesman-no dimana seorang klien telah melayani lebih dari satu salesman dari tabel SALES-ORDER.

Order-no	Client-no	Salesman-no
D0001	A0006	S0002
D0002	A0002	S0001
D0003	A0007	S0004
D0004	A0005	S0003
D0005	A0002	S0003
D0006	A0007	S0002

Jawaban : SELECT firsLOrder-no, first.Client-no, firsLSalesman-no
FROM SALES-ORDER first,
SALES-ORDER second
WHERE first.Client-no = second.Client-no
AND first.Salesman-no < > second.Salesman-no;

Table Name : First

Order-no	Client-no	Salesman-no
D0001	A0006	S0002
D0002	A0002	S0001
D0003	A0007	S0004
D0004	A0005	S0003
D0005	A0002	S0003
D0006	A0007	S0002

Table Name : Second

Order-no	Client-no	Salesman-no
D0001	A0006	S0002
D0002	A0002	S0001
D0003	A0007	S0004
D0004	A0005	S0003
D0005	A0002	S0003
D0006	A0007	S0002

Output :

Order-no	Client-no	Salesman-no
D0005	A0002	S0003
D0002	A0002	S0001
D0006	A0007	S0002
D0003	A0007	S0004

LAB 10

Pertanyaan1 : Ambil nama semua klien dan salesman di kota 'Indonesia' dari tabel CLIENT-MAST dan SALES-MAST

Table Name : CLIENT-MAST

Client-no	Name	City
A0001	Dani Sasmoko	Bandung
A0002	Lawren Setyono	Jakarta
A0003	Agus Sasmoko	Bandung
A0004	Ridlo Setyono	Jakarta Sel.
A0005	Faruq Setyono	Bandung
A0006	Widodo Setyono	Jakarta Sel.
A0007	Joni Setyono	Bandung
A0008	Depak	Jakarta Sel.

Table Name : SALES-MAST

Salesman-no	Name	City
B0001	Angga Setyono	Salatiga
B0002	Nizam Setyono	Salatiga
B0003	Yafa Sasmoko	Bandung
B0004	Faris Setyono	Lmpung
B0005	Nakir Setyono	Jakarta
B0006	Wahyu Setyono	Lampung
B0007	Anwar Setyono	Salatiga
B0008	Zulfa Setyono	Jakarta Sel.

Jawaban :

```
SELECT Client-no "ID", Name
FROM CLIENT-MAST
WHERE City = 'Indonesia'
Union:      SELECT Salesman-no "ID", Name
            FROM SALES-MAST WHERE City = 'Indonesia';
```

Output :

ID	Name
A0001	Dani Sasmoko
A0003	Agus Sasmoko
A0005	Faruq Setyono
A0007	Joni Setyono
B0003	Imam Sasmoko

Pertanyaan 2: Cari nama salesman di 'Indonesia' yang usahanya menghasilkan minimal satu transaksi penjualan.

Table Name : SALES-MAST

Salesman-no	Name	City
B0001	Angga Setyono	Salatiga
B0002	Nizam Setyono	Salatiga
B0003	Yafa Sasmoko	Bandung
B0004	Nakir Setyono	Bandung
B0005	Wahyu Setyono	Lampung
B0006	Anwar Setyono	Lampung
B0007	Zulfa Setyono	Jakarta Sel.

Table Name : SALES-ORDER

Order-no	Order-date	Salesman-no
S0001	10-Apr-07	B0001
S0002	28-Apr-07	B0002
S0003	05-May-07	B0003
S0004	12-June-07	B0004
S0005	15-July-07	B0005
S0006	18-Aug-07	B0006

Jawaban :

```
SELECT Salesman-no, Name FROM SALES-MAST
    WHERE City = 'Indonesia'
    INTERSECT
SELECT SALES-MAST.Salesman-no, Name
    FROM SALES-MAST, SALES-ORDER
    WHERE SALES-MAST.Salesman-no
    = SALES-ORDER.Salesman-no;
```

Output :

Salesman-no	Name
A0003	Yafa Sasmoko
A0004	Nakir Setyono

Pertanyaan 3 : Ambil semua nomor produk non-moving item dari tabel PRODUCT-MAST.

Nama Tabel: SALES-ORDER

Output :

Order-no	Product-no
S0001	A0001
S0001	A0004
S0001	A0006
S0002	A0002
S0002	A0005
S0003	A0003
S0004	A0001
S0005	A0006
S0005	A0004
S0006	A0006

Table Name: PRODUCT-MAST

Product-no	Prod-name
A0001	Monitors
A0002	Mouse
A0003	HDD
A0004	CD writer
A0005	DVD writer
A0006	Keyboard
A0007	Floppies driver
A0008	Floppies

Jawaban : SELECT Product-no FROM PRODUCT-MAST
MINUS
SELECT Product-no FROM SALES-ORDER

LAB 11

Pertanyaan 1:

(i) Temukan nama semua klien yang memiliki huruf kedua 'a' di namanya

(ii) Cari tahu Nama, Kota, Negara dari semua klien yang memiliki 'V' sebagai huruf pertama dalam nama mereka.

(iii) Temukan daftar semua klien yang tinggal di 'Indonesia' atau 'Jakarta'.

(iv) Temukan informasi dari tabel SALES-ORDER untuk pemesanan yang dilakukan di bulan Maret.

Jawaban :

(i)
```
SELECT Name FROM CLIENT-MAST
        WHERE Name LIKE '-a%';
```

(ii)
```
SELECT Name City, State
        FROM CLIENT-MAST
            WHERE Name LIKE 'V%';
```

(iii)
```
SELECT Client-no, Name, Address, City, State
        FROM CLIENT-MAST
            WHERE City IN ('Indonesia', 'Jakarta');
```

(iv)
```
SELECT * FROM SALES-MAST
        WHERE to-char (Order-date, 'MON') = 'Mar';
```

Pertanyaan 2:

(i) Hitung jumlah pesanan dari tabel SALES-ORDER

(ii) Hitung harga jual rata-rata semua produk dari tabel PRODUCT-MAST.

(iii) Tentukan harga jual maksimum dan minimum. Ubah nama output sebagai MAX-PRICE dan MIN-PRICE masing-masing.

(iv) Hitung jumlah produk yang memiliki harga jual lebih besar atau sama dengan 2000. Dari tabel PRODUCT-MAST.

Jawaban :

(i)
```
SELECT COUNT (Order-no)
        FROM SALES-ORDER;
```

(ii)
```
SELECT AVG (Sell-price)
```

```
                FROM PRODUCT-MAST;
(iii)   SELECT MAX (Sell-price) MAX-PRICE
                MIN (Sell-price) MIN-PRICE
                        FROM PRODUCT-MAST
(iv)    SELECT COUNT (Product-no)
                FROM PRODUCT-MAST
                        WHERE Sell-price> = 2000;
```

Pertanyaan 3:

(i) Menampilkan nomor pesanan dan hari dimana klien melakukan pemesanan untuk Tabel sale order

(ii) Menampilkan bulan (dalam abjad) dan tanggal saat pesanan harus dikirimkan untuk tabel SALES-ORDER.

(iii) Tampilkan tanggal pemesanan dalam format 'DD-MM-YY' pada tabel SALES-ORDER.

(iv) Temukan tanggalnya, 15 hari setelah tanggal hari ini.

Jawaban :

```
(i)     SELECT Order-no, to-char (Order-date, 'day')
                FROM SALES-ORDER
(ii)    SELECT to-char (dely-date, 'month'), dely-date
                FROM SALES-ORDER
                        ORDER BY to-char (dely-date, 'month');
(iii)   SELECT to-char (Order-date, 'DD-MM-YY')
                FROM SALES-ORDER;
(iv)    SELECT Sysdate + 15
                FROM dual;
```

LAB 12

Pertanyaan 1: Tulis kode blok PL/SQL yang pertama kali menyisipkan record dalam tabel EMPLOYEE. Perbarui gaji Dani dan Lawrence sebesar Rp. 12000 dan Rp. 3000. Kemudian periksa untuk melihat bahwa total gaji tidak melebihi Rp. 30000. Jika total gaji lebih besar dari 30000 maka batalkan pembaruan yang dilakukan pada gaji Dani & Lawren.

Table Name : EMPLOYEE

Emp-no	Ename	Salary
E0001	Agus	5000
E0002	Dani	10000
E0003	Joni	5000
E0004	Lawren	2000

Jawaban :

```
DECLARE
        total-salary number (10);
BEGIN
        INSERT INTO EMPLOYEE
```

```
        VALUES ('E005', 'Raja', 1500);
    SAVEPOINT no-update;
    UPDATE EMPLOYEE SET Salary = Salary + 12000
        WHERE Ename = 'Dani';
    UPDATE EMPLOYEE SET
        Salary = Salary + 3000
        WHERE Ename = 'Lawrence';
    /* Pemilihan total gaji dari tabel EMPLOYEE * /
    SELECT Sum (Salary) INTO total-salary
        FROM EMPLOYEE;
    IF Total-salary > 30000 THEN
        ROLL BACK To Savepoint no-update;
        END IF;
        COMMIT;
    END;
```

Pertanyaan 2: Manajer SDM memutuskan untuk menaikkan gaji karyawan sebesar 20%. Tulis blok PL/SQL untuk menerima nomor karyawan dan memperbarui gaji karyawan tersebut menampilkan pesan yang sesuai berdasarkan keberadaan catatan di tabel EMPLOYEE.

Jawaban :

```
    BEGIN
        UPDATE EMPLOYEE SET Salary = Salary * 0.20
        WHERE Emp-code = & Emp-code;
    IF SQL % FOUND THEN
        dbms _ output.puUine ('Employee record modified successfully');
        Else
        dbms _ output.puUine ('Employee no does not exist');
        END IF;
    END;
```

Pertanyaan 3: Manajer HR memutuskan untuk menaikkan gaji karyawan sebesar 20%. Tulis blok PL/SQL untuk menerima nomor karyawan dan memperbarui gaji karyawan tersebut. Menampilkan pesan yang sesuai berdasarkan adanya record NOT found di tabel employee.

Jawaban :

```
    BEGIN
    UPDATE EMPLOYEE SET Salary = Salary * 0.20
        WHERE emp-code = & emp-code;
    IF SQL % NOT FOUND THEN
        dbms_output.puUine ('Employee no does not exist');
    Else
        dbms_output.puUine ('Employee record modified successfully);
        END IF;
    END.
```

LAB 13

Pertanyaan 1: Manajer SDM telah memutuskan untuk menaikkan gaji untuk semua karyawan di departemen nomor 30 sebesar 0,25. Setiap kali kenaikan gaji tersebut diberikan kepada karyawan, catatan untuk hal yang sama disimpan dalam tabel EMPLOYEE-RAISE. Ini termasuk nomor karyawan, tanggal saat kenaikan gaji diberikan dan kenaikan gaji yang sebenarnya. Tulis blok PL/SQL t~. perbarui gaji setiap karyawan dan masukkan catatan dalam tabel EMPLOYEE-RAISE.

Jawaban :

```
DECLARE
CURSOR C-emp IS SELECT emp-code, salary FROM EMPLOYEE
                WHERE Dep-no = 30;
        Str-emp-code EMPLOYEE.emp-code % type;
        num-salary EMPLOYEE.salary % type;
BEGIN
        OPEN, C-emp;
IF C-emp % IS OPEN THEN
        LOOP
        FETCH C-emp INTO Str-emp-code, num-salary;
        exit when C-emp % NOT FOUND;
        UPDATE EMPLOYEE SET
                Salary = num-salary + (num-salary * 0.25)
        WHERE emp-code = Str-emp-code;
INSERT INTO EMPLOYEE-RAISE
VALUES (Str-emp-code, sysdate, num-salary * 0.25);
END LOOP;
COMMIT;
CLOSE C-emp;
        ELSE
        dbms_output.putJine ('Unable to open cursor');
END IF;
END.
```

Pertanyaan 2: Manajer HR memutuskan untuk menaikkan gaji untuk semua karyawan di departemen nomor 30 sebesar 0,25. Setiap kali kenaikan gaji tersebut diberikan kepada KARYAWAN, catatan untuk hal yang sama disimpan dalam tabel EMP-RAISE. Ini termasuk nomor karyawan, tanggal saat kenaikan gaji diberikan dan kenaikan gaji sebenarnya. Tulis blok PL/SQL untuk memperbarui gaji setiap karyawan dan masukkan catatan ke dalam tabel EMP-RAISE.

Jawaban :

```
DECLARE
        CURSOR e-emp IS SELECT
        emp-code, salary
 FROM EMPLOYEE
```

```
        WHERE Dept-no = 30;
Str-emp-code EM,PLOYEE.emp-code % type;
num-salary EMPLOYEE. salary % type;
BEGIN
        OPEN c-cmp;
        LOOP
        FETCH e-emp INTO str-emp-code, num-salary;
IF e-emp % FOUND THEN
        UPDATE EMPLOYEE SET
        Salary = nurn-salary + (num-salary * 0.25)
        WHERE emp-code = Str-emp-code;
INSERT INTO EMP-RAISE VALUES
        (Str-emp-code, sysdate, nurn-salary * 0.25);
ELSE
        Exit;
        END IF;
        END LOOP;
        COMMIT;
        CLOSE e-cmp;
END.
```

LAMPIRAN B

Centang Jawaban yang Sesuai

1. Manakah dari berikut ini yang berhubungan dengan informasi?
    (a) data
    (b) komunikasi
    (c) pengetahuan
    (d) semua jawaban benar
2. Data adalah:
    (a) sepotong fakta
    (b) metadata
    (c) informasi
    (d) tidak satu pun dari ketiganya
3. Manakah dari berikut ini yang merupakan elemen Database:
    (a) data
    (b) kendala dan skema
    (c) hubungan
    (d) semua jawaban benar
4. Apa yang mewakili korespondensi antara elemen data:
    (a) data
    (b) kendala
    (c) hubungan
    (d) skema
5. Manakah dari berikut ini yang merupakan keuntungan menggunakan sistem database:
    (a) keamanan penegakan
    (b) menghindari redundansi
    (c) mengurangi ketidakkonsistenan
    (d) Semua jawaban benar
6. Manakah dari berikut ini yang merupakan karakteristik data dalam database:
    (a) independen
    (b) aman
    (c) bersama
    (d) semua jawaban benar
7. Nama database sistem yang berisi deskripsi data dalam database adalah :
    (a) kamus data
    (b) metadata
    (c) tabel
    (d) tidak satupun
8. Berikut jenis metadatanya :
    (a) Operasional

(b) Datamart
(c) EDW
(d) semua jawaban benar

9. Katalog sistem adalah database yang dibuat sistem yang menjelaskan:
   (a) objek database
   (b) informasi kamus data
   (c) informasi akses pengguna
   (d) semua jawaban benar
10. Hubungan dapat berupa jenis berikut:
   (a) hubungan satu-ke-satu
   (b) hubungan satu-ke-banyak
   (c) hubungan banyak-ke-banyak
   (d) Semua jawaban benar
11. Dalam sistem berorientasi file ada :
   (a) inkonsistensi data
   (b) duplikasi data
   (c) ketergantungan data
   (d) Semua jawaban benar
12. Dalam sistem Database ada :
   (a) peningkatan produktivitas
   (b) Skala ekonomi
   (c) peningkatan biaya keseluruhan
   (d) Semua jawaban benar
13. Dalam sistem database ada:
   (a) ukuran besar DBMS
   (b) peningkatan kompleksitas
   (c) peningkatan keamanan
   (d) Semua jawaban benar
14. DML berikut akses fungsional ke database:
   (a) mengambil data dan/atau catatan
   (b) menambah (atau menyisipkan) catatan
   (c) menghapus catatan dari file database
   (d) semua jawaban benar
15. 4 GL memiliki komponen berikut bawaan di dalamnya :
   (a) bahasa query
   (b) pembuat laporan
   (c) spreadsheet
   (d) Semua jawaban benar
16. Apa yang membedakan aspek fisik penyimpanan data dari aspek logis representasi data?
   (a) data
   (b) skema

(c) kendala
(d) hubungan

17. Skema apa yang mendefinisikan bagaimana dan di mana data diatur dalam penyimpanan data fisik?
    (a) eksternal
    (b) internal
    (c) konseptual
    (d) tidak satupun dari ini
18. Manakah dari skema berikut yang mendefinisikan tampilan atau tampilan database untuk pengguna tertentu?
    (a) Eksternal
    (b) internal
    (c) konseptual
    (d) tidak satupun dari ini
19. Kumpulan data yang dirancang untuk digunakan oleh orang yang berbeda disebut?
    (a) database
    (b) RDBMS
    (c) DBMS
    (d) tidak satu pun dari ini
20. Manakah dari berikut ini yang merupakan karakteristik data dalam database:
    (a) bersama
    (b) aman
    (c) independen
    (d) semua
21. DBMS berorientasi objek mampu menyimpan:
    (a) data dan teks
    (b) gambar dan gambar
    (c) suara dan video
    (d) semua di atas
22. Manakah dari berikut ini yang merupakan fitur berorientasi objek?
    (a) pewarisan
    (b) polimorfisme
    (c) abstraksi
    (d) Semua jawaban benar
23. Kekebalan skema konseptual terhadap perubahan dalam skema internal disebut sebagai:
    (a) independensi data fisik
    (b) independensi data logis
    (c) a dan b benar
    (d) tidak satupun dari ini
24. Model data fisik digunakan untuk:
    (a) menentukan keseluruhan struktur logis dari database

(b) menggambarkan data dan hubungannya
(c) deskripsi tingkat yang lebih tinggi dari struktur penyimpanan dan mekanisme akses
(d) Semua jawaban benar

25. Model data berorientasi objek digunakan untuk:
(a) menentukan keseluruhan struktur logis dari database
(b) menggambarkan data dan hubungannya
(c) deskripsi tingkat yang lebih tinggi dari struktur penyimpanan dan mekanisme akses
(d) Semua jawaban benar

26. Model data relasional pertama kali diperkenalkan oleh :
(a) SPARC
(b) kabel EF
(c) ANSI
(d) chern

27. Model data ER pertama kali diperkenalkan oleh :
(a) SPARC
(b) ANSI
(c) Kabel EF
(d) Chem

28. Bapak dari sistem Database relasi adalah :
(a) Pascal
(b) C.J Date
(c) Dr. Edgar F. Cord
(d) tidak satu pun dari ini

29. Siapa yang menulis makalah berjudul sebuah model relasional data untuk bank data bersama yang besar"?
(a) FR McFadder
(b) Tanggal CJ
(c) Dr. .edgar F. cord
(d) tidak satupun dari ini

30. Apa terminologi RDBMS untuk sebuah baris?
(a) tuple
(b) relasi
(c) atribut
(d) domain

31. Berapa kardinalitas tabel dengan 1000 baris dan 10 kolom?
(a) 10
(b) 100
(c) 1000
(d) tidak satupun dari ini

32. Berapa kardinalitas tabel dengan 5000 baris dan 50 kolom?
    (a) 10
    (b) 50
    (c) 500
    (d) 5000
33. Manakah dari kunci berikut dalam tabel yang dapat secara unik mengidentifikasi baris dalam tabel?
    (a) kunci utama
    (b) kunci alternatif
    (c) kunci kandidat
    (d) Semua jawaban benar
34. Sebuah tabel dapat memiliki hanya satu :
    (a) kunci primer
    (b) kunci alternatif
    (c) kunci kandidat
    (d) Semua jawaban benar
35. Disebut apakah semua kunci kandidat selain kunci primer?
    (a) kunci sekunder
    (b) kunci alternatif
    (c) kunci yang memenuhi syarat
    (d) tidak satupun dari ini
36. Apa nama atribut atau kombinasi atribut dari satu relasi yang nilainya diperlukan untuk mencocokkan nilai kunci utama dari beberapa relasi lainnya?
    (a) kunci kandidat
    (b) kunci utama
    (c) kunci asing
    (d) kunci pencocokan
37. Apa terminologi RDBMS untuk kolom?
    (a) tuple
    (b) relasi
    (c) atribut
    (d) domain
38. Apa terminologi RDBMS untuk tabel?
    (a) tuple
    (b) atribut
    (c) relasi
    (d) domain
39. Apa terminologi RDBMS untuk nilai legal yang dapat dimiliki suatu atribut? (a)
    (a) tupel
    (b) relasi
    (c) atribut
    (d) domain

40. Apa terminologi RDBMS untuk no. dari tupel dalam suatu relasi?
    (a) derajat
    (b) hubungan
    (c) atribut
    (d) kardinalitas
41. Apa terminologi RDBMS untuk no. atribut dalam sebuah relasi?
    (a) derajat
    (b) relasi
    (c) atribut
    (d) kardinalitas
42. Manakah dari aspek data berikut yang merupakan pusat dari model Database relasional?
    (a) manipulasi data
    (b) integritas data
    (c) struktur data
    (d) Semua jawaban benar
43. Apa unit data terkecil dalam model relasional?
    (a) tipe data
    (b) field
    (c) nilai data
    (d) tidak satupun dari~e
44. Ketergantungan fungsional adalah :
    (a) hubungan banyak-ke-banyak antara dua set atribut
    (b) satu-ke -satu hubungan antara dua set atribut
    (c) banyak-ke-satu hubungan antara dua set atribut
    (d) tidak satupun dari ini
45. Dekomposisi membantu dalam menghilangkan beberapa masalah desain yang buruk seperti:
    (a) redundansi
    (b) inkonsistensi
    (c) anomali
    (d) Semua jawaban benar
46. Kata loss in lossless mengacu pada:
    (a) kehilangan informasi
    (b) kehilangan atribut
    (c) kehilangan hubungan
    (d) tidak satupun dari ini
47. himpunan atribut X akan sepenuhnya bergantung secara fungsional pada himpunan atribut Y jika kondisi berikut dipenuhi:
    (a) X bergantung secara fungsional pada Y
    (b) X tidak bergantung secara fungsional pada setiap subset dari Y
    (c) keduanya (a) dan (b)

(d) tidak satupun dari ini

48. Normalisasi adalah proses dari:
    (a) menguraikan himpunan relasi
    (b) reduksi berturut-turut skema relasi
    (c) memutuskan atribut mana dalam relasi yang akan dikelompokkan bersama
    (d) Semua jawaban benar
49. Proses normalisasi dikembangkan oleh :
    (a) EF Codd
    (b) R. Fagin
    (c) RF Boyce
    (d) Collin White
50. Bentuk normal adalah :
    (a) keadaan relasi yang dihasilkan dari penerapan aturan sederhana mengenai FD
    (b) kondisi bentuk normal tertinggi yang dipenuhi
    (c) indikasi sejauh mana telah dinormalisasi
    (d) Semua jawaban benar
51. Manakah dari berikut ini yang merupakan proses formal untuk memutuskan atribut mana yang harus dikelompokkan bersama dalam suatu relasi?
    (a) optimasi
    (b) normalisasi
    (c) penyetelan
    (d) tidak satupun dari ini.
52. Dalam 1 NF:
    (a) semua domain sederhana
    (b) (a) dan (b) benar
    (c) dalam domain sederhana, semua elemen adalah atom
    (d) tidak satupun dari ini
53. 2NF selalu dalam :
    (a) 1 NF
    (b) MVD
    (c) BCNF
    (d) tidak satupun dari ini
54. Suatu relasi R dikatakan dalam 2 NF :
    (a) jika berada dalam 1 NF
    (b) setiap atribut non-prime key dari R secara fungsional bergantung pada setiap kunci relasi R
    (c) jika dalam BCNF
    (d) keduanya (a) dan (b)
55. Suatu relasi R dikatakan dalam 3 NF jika ;
    (a) relasi R dalam 2 NF
    (b) atribut non-prima saling bebas

(c) secara fungsional bergantung pada kunci prima
(d) Semua jawaban benar

56. Gagasan ketergantungan multi-nilai diperkenalkan oleh:
(a) EF Codd
(b) RE Boyce
(c) R. Fagin
(d) tidak satupun dari ini

57. Perluasan BCNF adalah :
(a) Bentuk normal Boyd-Codd
(b) Bentuk normal Boyce-Codd
(c) Normal Boyce-Ceromwell bentuk
(d) tidak satupun dari ini

58. 4 NF berkaitan dengan ketergantungan b/w unsur-unsur senyawa kunci terdiri dari :
(a) satu atribut
(b) dua atribut
(c) tiga atau lebih atribut
(d) tidak satupun dari ini

59. Ketika semua kolom dalam suatu relasi mendeskripsikan dan bergantung pada kunci utama, relasi tersebut dikatakan berada dalam :
(a) 1NF
(b) 2NF
(c) 3NF
(d) 4NF

60. Manakah dari berikut ini yang merupakan aktivitas mengkoordinasikan aksi-aksi proses yang beroperasi secara paralel dan mengakses data bersama?
(a) manajemen transaksi
(b) manajemen pemulihan
(c) kontrol konkurensi
(d) tidak satu pun dari ini

61. Manakah dari berikut ini yang merupakan kemampuan DBMS untuk mengelola berbagai transaksi yang terjadi dalam sistem?
(a) manajemen transaksi
(b) manajemen pemulihan
(c) kontrol konkurensi
(d) tidak satu pun dari ini

62. Manakah dari berikut ini yang merupakan properti transaksi?
(a) isolasi
(b) atomisitas
(c) daya tahan
(d) Semua jawaban benar

63. Manakah dari berikut ini yang memastikan konsistensi transaksi?
(a) pemrogram aplikasi

(b) kontrol konkurensi
(c) manajemen pemulihan
(d) manajemen transaksi

64. Manakah dari berikut ini yang menjamin ketahanan transaksi?
(a) pemrogram aplikasi
(b) kontrol konkurensi
(c) manajemen pemulihan
(d) manajemen transaksi

65. Dalam fase menyusut, transaksi:
(a) melepaskan semua tugas
(b) baik (a) dan (b)
(c) tidak dapat memperoleh kunci baru
(d) tidak satupun dari berikut ini

66. Manakah dari berikut ini yang memastikan atomisitas suatu transaksi?
(a) pemrogram aplikasi
(b) kontrol konkurensi
(c) manajemen pemulihan
(d) manajemen transaksi

67. Manakah dari berikut ini yang memastikan isolasi transaksi?
(a) pemrogram aplikasi
(b) kontrol konkurensi
(c) manajemen pemulihan
(d) manajemen transaksi

68. Manakah dari berikut ini yang merupakan status transaksi?
(a) aktif
(b) dibatalkan
(c) melakukan
(d) Semua jawaban benar

69. Kontrol konkurensi memiliki masalah berikut;
(a) pembaruan yang hilang
(b) pembacaan kotor
(c) pembacaan yang tidak dapat diulang
(d) Semua jawaban benar.

70. Manakah dari berikut ini yang bukan merupakan perintah SQL manajemen transaksi?
(a) commit
(b) pilih
(c) savepoint
(d) rollback

71. Manakah dari pernyataan berikut yang setelah itu Anda tidak dapat mengeluarkan perintah commit?
(a) sisipkan
(b) pilih

(c) perbarui
(d) hapus

72. Manakah dari berikut ini yang merupakan kontrol konkurensi berbasis validasi?
(a) validasi
(b) write
(c) read
(d) Semua jawaban benar

73. Penguncian dapat terjadi pada level berikut:
(a) level halaman
(b) level Database
(c) level baris
(d) Semua jawaban benar

74. Suatu transaksi dapat mencakup operasi akses Database dasar berikut:
(a) item read (X)
(b) item write (X)
(c) keduanya (a) dan (b)
(d) tidak satu pun dari ini

75. Yang mana berikut ini yang bukan merupakan strategi penanganan Deadlock?
(a) Timeout
(b) Pemusnahan Deadlock
(c) Pencegahan Deadlock
(d) Deteksi Deadlock

76. Dalam fase berkembang, transaksi memperoleh semua kunci yang diperlukan:
(a) dengan mengunci data
(b) tanpa membuka kunci data
(c) dengan membuka data
(d) tidak satupun dari ini

77. Manakah dari berikut ini yang merupakan metode kontrol konkurensi optimis:
(a) berbasis validasi
(b) pemesanan stempel waktu
(c) berbasis kunci
(d) tidak satu pun dari ini

78. Variasi dasar metode konkurensi berbasis stempel waktu kontrol adalah:
(a) pemesanan stempel waktu total
(b) pemesanan stempel waktu parsial
(c) pemesanan stempel waktu multiversi
(d) Semua jawaban benar

79. Dalam metode optimis, setiap transaksi bergerak melalui fase berikut:
(a) fase baca
(b) validasi fase
(c) fase write
(d) SO ini.

80. Manakah dari berikut ini yang bukan merupakan teknik pemulihan:
    (a) shadow paging
    (b) pembaruan yang ditangguhkan
    (c) write-ahead logging
    (d) pembaruan segera
81. Manakah dari berikut ini yang merupakan salinan file database fisik:
    (a) log transaksi
    (b) pencadangan fisik
    (c) pencadangan logis
    (d) tidak satu pun dari ini
82. Manakah dari kegagalan berikut yang disebabkan oleh kegagalan perangkat keras:
    (a) operasi
    (b) desain
    (c) fisik
    (d) tidak satupun dari ini
83. Dari berikut ini adalah jenis kegagalan yang paling berbahaya:
    (a) perangkat keras
    (b) jaringan
    (c) media
    (d) perangkat lunak
84. Kegagalan perangkat keras dapat mencakup:
    (a) kesalahan memori disk crash
    (b) kesalahan disk penuh
    (c) Semua jawaban benar
85. Kegagalan perangkat lunak dapat mencakup kegagalan yang terkait dengan perangkat lunak seperti:
    (a) sistem operasi
    (b) perangkat lunak DBMS
    (c) program aplikasi
    (d) Semua jawaban benar
86. Manakah dari berikut ini yang merupakan fasilitas yang disediakan oleh DBMS untuk membantu proses pemulihan:
    (a) manajer pemulihan
    (b) fasilitas lagging
    (c) mekanisme cadangan
    (d) Semua jawaban benar
87. Saat menggunakan pemulihan berbasis log transaksi skema, mungkin meningkatkan kinerja serta menyediakan mekanisme pemulihan dengan:
    (a) menulis log yang sesuai ke disk selama eksekusi transaksi
    (b) menulis catatan log ke disk ketika setiap transaksi dilakukan
    (c) tidak pernah menulis catatan log ke disk

(d) menunggu untuk menulis catatan log sampai banyak transaksi melakukan dan menulis kemudian sebagai batch

88. Untuk mengatasi kegagalan media (atau disk), perlu:
    (a) menyimpan salinan database yang berlebihan
    (b) untuk tidak pernah membatalkan transaksi
    (c) DBMS hanya melakukan transaksi dalam satu lingkungan pengguna
    (d) Semua jawaban benar
89. Teknik paging bayangan mempertahankan:
    (a) tabel dua halaman
    (b) tabel empat halaman
    (c) tiga halaman tabel
    (d) tabel lima halaman
90. Teknik checkpoint digunakan untuk membatasi :
    (a) volume informasi log jumlah pencarian
    (b) pemrosesan selanjutnya yang diperlukan untuk melaksanakan file log transaksi
    (c) Semua jawaban benar
91. Manakah dari teknik pemulihan berikut tidak memerlukan log:
    (a) shadow paging
    (b) pembaruan segera
    (c) pembaruan yang ditangguhkan
    (d) tidak satu pun dari ini
92. Cadangan Database disimpan di tempat yang aman, biasanya;
    (a) di gedung yang berbeda
    (b) terlindung dari bahaya seperti kebakaran, pencurian, banjir
    (c) potensi bencana lainnya
    (d) Semua jawaban benar
93. Hilangnya ketersediaan berarti bahwa ;
    (a) data tidak dapat diakses oleh pengguna
    (b) sistem tidak dapat diakses oleh pengguna
    (c) baik data maupun sistem tidak dapat digunakan oleh pengguna
    (d) tidak satu pun dari ini
94. Manakah dari berikut ini yang merupakan izin untuk mengakses objek bernama dengan cara yang ditentukan;
    (a) peran
    (b) hak istimewa
    (c) izin
    (d) Semua jawaban benar
95. Hilangnya integritas data berarti bahwa:
    (a) data dan sistem tidak dapat diakses oleh pengguna
    (b) data yang dihasilkan tidak valid dan rusak
    (c) hilangnya perlindungan atau kerahasiaan atas data penting organisasi

(d) hilangnya perlindungan data dari individu

96. Manakah dari berikut ini yang bukan merupakan bagian dari keamanan database:
    (a) data
    (b) perangkat keras dan perangkat lunak
    (c) orang
    (d) peretas eksternal
97. Kontrol akses diskresi (juga disebut skema keamanan) didasarkan pada konsep ;
    (a) hak akses
    (b) kebijakan di seluruh sistem
    (c) baik (a) dan (b)
    (d) tidak satu pun dari ini
98. Hilangnya kerahasiaan berarti bahwa ;
    (a) data dan sistem tidak dapat diakses oleh penggunaan
    (b) data yang tidak valid dan rusak telah dihasilkan
    (c) hilangnya perlindungan atau pemeliharaan kerahasiaan atas data penting organisasi
    (d) hilangnya perlindungan data dari individu
99. Hilangnya privasi berarti bahwa:
    (a) data dan sistem tidak dapat diakses oleh pengguna
    (b) data yang tidak valid dan rusak telah dihasilkan
    (c) hilangnya perlindungan dan pemeliharaan kerahasiaan atas data penting organisasi
    (d) hilangnya perlindungan data dari individu
100. Masalah hukum dan etika yang terkait dengan:
    (a) hak akses pengguna individu atau kelompok pengguna untuk mengakses informasi tertentu
    (b) penegakan berbagai fungsi keamanan di tingkat sistem, misalnya di tingkat perangkat keras fisik, di tingkat Tingkat DBMS atau pada tingkat sistem operasi
    (c) penegakan kebijakan keamanan organisasi sehubungan dengan izin akses ke berbagai klasifikasi data
    (d) tidak satupun dari ini
101. Masalah terkait sistem yang terkait dengan
    (a) hak akses pengguna individu atau kelompok pengguna untuk mengakses informasi tertentu
    (b) penegakan berbagai fungsi keamanan pada tingkat sistem, misalnya pada tingkat perangkat keras fisik, pada tingkat DBMS atau pada tingkat sistem operasi
    (c) penegakan kebijakan keamanan organisasi sehubungan dengan izin akses ke berbagai klasifikasi data
    (d) tidak satu pun dari ini.
102. Manakah dari berikut ini yang merupakan hak istimewa database:

(a) hak untuk membuat tabel atau relasi
(b) hak untuk memilih baris dari tabel pengguna lain
(c) hak untuk membuat sesi
(d) Semua jawaban benar

103. ORDBMS dapat menangani:
(a) objek kompleks
(b) tipe yang ditentukan pengguna
(c) tipe data abstrak
(d) Semua jawaban benar

104. 4BMS relasional objek (ORDBMS) juga disebut:
(a) DBMS relasional yang disempurnakan
(b) DBMS relasional umum
(c) DBMS berorientasi objek
(d) Semua jawaban benar

105. Contoh objek kompleks adalah :
(a) data kompleks non-konvensional dalam desain teknik
(b) comp lex data non-konvensional dalam informasi genom biologis
(c) data non-konvensional kompleks dalam gambar arsitektur
(d) Semua jawaban benar

106. Produk ORDBMS yang dikembangkan oleh ORACLE dikenal sebagai:
(a) database universal
(b) postgres
(c) informix
(d) Tidak satu pun dari ini

107. Sistem Database terdistribusi memungkinkan aplikasi untuk mengakses data dari:
(a) basis data lokal
(b) basis data jarak jauh
(c) basis data lokal dan jarak jauh
(d) tidak satu pun dari ini

108. Dalam DDBS homogen :
(a) ada beberapa situs, masing-masing menjalankan aplikasinya sendiri pada perangkat lunak DBMS yang sama
(b) semua situs memiliki perangkat lunak DBMS yang identik
(c) semua pengguna (atau klien) menggunakan perangkat lunak yang identik
(d) Semua jawaban benar

109. Secara heterogen DDBS :
(a) situs yang berbeda berjalan di bawah kendali DBMS yang berbeda, pada dasarnya secara otonom
(b) situs yang berbeda terhubung entah bagaimana untuk memungkinkan akses ke data dari beberapa situs
(c) situs yang berbeda dapat menggunakan skema yang berbeda dan DBMS yang berbeda

(d) Semua jawaban benar

110. Komponen utama arsitektur Client/Server:
    (a) jaringan komunikasi
    (b) server
    (c) perangkat lunak aplikasi
    (d) Semua jawaban benar
111. Manakah dari berikut ini yang bukan manfaat arsitektur Client/Server:
    (a) pengurangan biaya operasi
    (b) kemampuan beradaptasi
    (c) independensi bentuk pelat
    (d) tidak satu pun dari ini.
112. Manakah dari berikut ini yang merupakan komponen DDBS:
    (a) jaringan komunikasi
    (b) server
    (c) klien
    (d) semua
113. Manakah dari arsitektur komputasi yang digunakan oleh DDBS:
    (a) komputasi Client/Server
    (b) komputasi mainframe
    (c) komputasi pribadi
    (d) tidak satupun dari ini
114. Dalam arsitektur server yang berkolaborasi:
    (a) ada beberapa server Database
    (b) setiap server mampu menjalankan transaksi terhadap data lokal
    (c) transaksi dijalankan di beberapa server
    (d) Semua jawaban benar
115. Fragmentasi data adalah:
    (a) teknik memecah database menjadi unit logis, yang dapat ditugaskan untuk penyimpanan di berbagai lokasi
    (b) proses penentuan lokasi (atau penempatan) data ke situs server
    (c) teknik yang memungkinkan penyimpanan data tertentu berpindah dari satu situs
    (d) tidak satu pun dari ini
116. Fragmentasi horizontal dihasilkan dengan menentukan:
    (a) operasi predikat aljabar relasional
    (b) operasi proyeksi relasional aljabar
    (c) operasi seleksi dan proyeksi aljabar relasional
    (d) tidak satupun dari ini
117. Fragmentasi vertikal dihasilkan dengan menetapkan a :
    (a) operasi predikat aljabar relasional
    (b) operasi proyeksi aljabar relasional
    (c) seleksi dan proyeksi aljabar relasional operasional

(d) tidak satu pun dari ini

118. Dalam pemrosesan kueri terdistribusi, operasi semijoin digunakan untuk:
    (a) mengurangi ukuran relasi yang perlu ditransmisikan
    (b) mengurangi biaya komunikasi
    (c) keduanya (a) dan (b)
    (d) tidak satu pun dari ini
119. Dalam DDBS fungsi manajer kunci adalah:
    (a) didistribusikan ke beberapa lokasi
    (b) terpusat di satu lokasi
    (c) tidak ada manajer kunci yang digunakan
    (d) tidak satupun dari
120. Dalam sistem terdistribusi, deteksi Deadlock membutuhkan generasi dari :
    (a) menunggu lokal untuk grafik
    (b) menunggu global untuk grafik
    (c) keduanya (a) dan (b)
    (d) tidak satupun dari ini
121. Dalam sistem database terdistribusi, metode pencegahan deadlock dengan membatalkan transaksi dapat digunakan seperti:
    (a) timestamping
    (b) metode wait-die
    (c) metode wound-wait
    (d) Semua jawaban benar
122. Manakah dari berikut ini yang fungsi DBMS terdistribusi:
    (a) pemulihan data terdistribusi
    (b) pemrosesan kueri terdistribusi
    (c) manajemen data yang direplikasi
    (d) Semua jawaban benar

KUNCI JAWABAN

KUNCI JAWABAN PILIHAN GANDA

1. (a)	2. (a)	3. (d)	4. (c)	5. (d)	6. (d)
7. (b)	8. (d)	9. (d)	10 (d)	11 (d)	12 (d)
13. (d)	14. (d)	15. (d)	16. (a)	17. (b)	18. (a)
19. (a)	20. (d)	21. (d)	22. (d)	23. (a)	24. (d)
25. (b)	26. (b)	27. (d)	28. (c)	29. (c)	30. (a)
31. (c)	32. (d)	33. (d)	34. (a)	35. (d)	36. (c)
37. (c)	38. (b)	39. (d)	40. (d)	41. (a)	42. (d)
43. (b)	44. (c)	45. (d)	46. (a)	47. (c)	48. (d)
49. (a)	50. (d)	51. (b)	52. (c)	53. (a)	54. (d)
55. (d)	56. (c)	57. (c)	58. (c)	59. (b)	60. (a)
61. (b)	62. (d)	63. (b)	64. (c)	65. (c)	66. (c)
67. (b)	68. (d)	69. (d)	70. (c)	71. (b)	72. (d)
73. (d)	74. (c)	75. (b)	76. (b)	77. (a)	78. (d)
79. (d)	80. (c)	81. (b)	82. (b, c)	83. (c)	84. (d)
85. (d)	86. (d)	87. (a)	88. (a)	89. (a)	90. (a)
91. (d)	92. (d)	93. (a)	94. (b)	95. (b)	96. (c)
97. (a)	98. (c)	99. (d)	100. (a)	101. (b)	102. (d)
103. (d)	104. (a)	105. (d)	106. (d)	107. (c)	108. (d)
109. (d)	110. (d)	111. (d)	112. (d)	113. (a)	114. (d)
115. (a)	116. (a)	117. (b)	118. (c)	119. (a)	120. (c)
121. (d)	122. (d)				

SOAL BENAR SALAH

1. Data disebut juga metadata.
2. Data adalah bagian dari fakta
3. Data adalah bagian informasi yang berbeda.
4. Dalam DBMS, file data adalah file yang menyimpan informasi database.
5. Skema eksternal mendefinisikan bagaimana dan di mana data diatur dalam penyimpanan data fisik.
6. Kumpulan data yang dirancang untuk digunakan oleh pengguna yang berbeda disebut database.
7. Dalam sebuah database, integritas data dapat terjaga.
8. Data dalam database tidak dapat dibagi.
9. DBMS menyediakan bahasa pendukung yang digunakan untuk definisi dan manipulasi data dalam database. 10. Katalog data dan kamus data sama.
10. Katalog data diperlukan untuk mendapatkan informasi tentang struktur database
11. Sebuah database tidak dapat menghindari inkonsistensi data.
12. Menggunakan database redundency dapat dikurangi.
13. Pembatasan keamanan tidak dapat diterapkan dalam sistem database.
14. Data dan metadata adalah sama. 16. Metadata juga dikenal sebagai data tentang data

15. Katalog sistem adalah tempat penyimpanan informasi yang menjelaskan data dalam database.
16. Informasi yang disimpan dalam katalog disebut metadata.
17. DBMS mengelola akses database konkuren dan mencegah dari masalah kehilangan informasi kehilangan integritas.
18. Bahasa definisi tampilan digunakan untuk menentukan tampilan pengguna (skema eksternal) dan pemetaannya ke skema konseptual.
19. Bahasa definisi penyimpanan data digunakan untuk menentukan skema konseptual dalam database.
20. Structured query language (SQL) dan query by example (OBE) adalah contoh bahasa generasi keempat.
21. Transaksi tidak dapat memperbarui catatan, menghapus catatan, memodifikasi satu set catatan dan sebagainya tidak.

JAWABAN

1. Benar
2. Benar
3. Benar
4. Benar
5. Salah
6. Benar
7. Benar
8. Salah
9. Benar
10. Benar
11. Benar
12. Salah
13. Benar
14. Salah
15. Salah
16. Benar
17. Benar
18. Benar
19. Benar
20. Benar
21. Salah
22. Benar
23. Salah

ISILAH PADA TITIK-TITIK YANG KOSONG

1. ... adalah sumber daya yang paling penting dari sebuah organisasi.
2. Data adalah mentah ... sedangkan informasi adalah ...
3. ... adalah perangkat lunak yang menyediakan layanan untuk mengakses database.
4. Dua bahasa penting dalam sistem Database adalah (a) ... dan (b)
5. Untuk mengakses informasi dari suatu Database, diperlukan suatu ...
6. DBMS yang merupakan singkatan dari...
7. SQL singkatan dari ...
8. 4G L singkatan dari ...
9. Tiga struktur data untuk aplikasi data warehouse adalah
    a. ...
    b. ... dan
    c. ...
10. DDL adalah singkatan dari...
11. DML singkatan dari ...
12. Data turunan disimpan di...
13. Empat komponen kamus data adalah
    a. ...
    b. ...
    c. ...

   d. ...
14. Empat jenis kunci yang digunakan adalah
   a. ...
   b. ...
   c. ... dan
   d. ...
15. Dua jenis kamus data adalah (a) ... dan (b) ...
16. CODASYL adalah singkatan dari ...
17. LPTF adalah singkatan dari ...
18. DBTG singkatan dari ...
19. Pada pertengahan tahun 1960-an, DBMS tujuan umum pertama yang dirancang oleh Charles Bachman di General Electric, USA disebut ...
20. Penerima pertama ilmu komputer yang setara dengan hadiah Nobel, yang disebut Association of Computing Machinery (ACM) Turing Award, untuk pekerjaan di bidang Database, pada tahun 1973 adalah...
21. Ketika DBMS melakukan commit, perubahan yang dilakukan oleh transaksi dilakukan...

JAWABAN

1. Data,
2. Fakta, data yang diolah/diorganisasikan/diringkas,
3. DBMS,.
4. Jawaban :(a) Bahasa deskripsi data (DDL), (b) bahasa manipulasi data (DML)
5. DBMS,
6. Sistem Manajemen Database,
7. Bahasa Kueri Terstruktur
8. Bahasa Generasi Keempat,
9. Jawaban :
   a. Data Operasional,
   b. Data yang Direkonsiliasi,
   c. Data Turunan,
10. Bahasa Definisi Data,
11. Bahasa Manipulasi Data,
12. Masing-masing data mart (Data warehouse yang dipilih, terbatas, dan diringkas),
13. Jawaban:
   a. Entitas
   b. Atribut
   c. Hubungan
   d. Kunci,
14. Jawaban:
   a. Kunci utama
   b. Kunci sekunder

c. Kunci super
d. Kunci gabungan,

15. Jawaban :
    a. Kamus data aktif
    b. kamus data pasif,
16. Konferensi Bahasa Sistem Data,
17. Gugus Tugas Pemrosesan Daftar,
18. Gugus Tugas Database,
19. Integrated Data Store (IDS)
20. Bachman,
21. Permanen.

SOAL BENAR/SALAH

1. Dalam sistem manajemen database, file data adalah file yang menyimpan informasi database.
2. Skema eksternal mendefinisikan bagaimana dan di mana data diatur dalam penyimpanan data fisik.
3. Dalam terminologi Database jaringan, suatu hubungan adalah himpunan.
4. Sebuah fitur dari database relasional adalah bahwa database tunggal dapat tersebar di beberapa tabel.
5. SQL dari adalah bahasa generasi keempat.
6. DBMS berorientasi objek cocok untuk aplikasi multimedia serta data dengan hubungan yang kompleks.
7. Sebuah OODBMS memungkinkan database terintegrasi penuh yang menyimpan data, teks, suara, gambar dan video.
8. Model hierarki mengasumsikan bahwa struktur pohon adalah hubungan yang paling sering terjadi
9. Model database hirarkis adalah model data tertua.
10. Data dalam database tidak dapat dibagi.
11. Perbedaan utama antara model data yang berbeda terletak pada metode mengungkapkan hubungan dan kendala antara elemen data.
12. Dalam database, data disimpan sedemikian rupa sehingga tidak tergantung pada program pengguna yang menggunakan data tersebut.
13. Rencana (atau perumusan skema) database dikenal sebagai skema.
14. Skema fisik berkaitan dengan pemanfaatan struktur data yang ditawarkan oleh DBMS untuk membuat skema tersebut dipahami oleh komputer.
15. Skema logis, berkaitan dengan cara di mana database konseptual akan diwakili di komputer sebagai database yang tersimpan.
16. Subskema bertindak sebagai unit untuk menegakkan penilaian terkontrol ke database.
17. Proses transformasi permintaan dan hasil antara tiga level disebut pemetaan.

18. Pemetaan konseptual/internal mendefinisikan korespondensi antara tampilan konseptual dan database yang disimpan.
19. Pemetaan eksternal/konseptual mendefinisikan korespondensi antara pandangan eksternal tertentu dan pandangan konseptual
20. Model data adalah proses abstraksi yang memusatkan aspek esensial dan inheren dari aplikasi organisasi sementara mengabaikan detail yang berlebihan atau tidak disengaja.
21. Model data berorientasi objek adalah model data logis yang menangkap semantik objek yang didukung dalam pemrograman berorientasi objek.
22. Sistem database terpusat secara fisik terbatas pada satu lokasi.
23. Arsitektur sistem database paralel terdiri dari satu central processing unit (CPU) dan disk penyimpanan data secara paralel.
24. Sistem database terdistribusi mirip dengan arsitektur client/server.

ISILAH BAGIAN TITIK-TITIK YANG KOSONG

1. Model data relasional menyimpan data dalam bentuk ...
2. ... mendefinisikan berbagai tampilan database.
3. Model ... mendefinisikan struktur data yang disimpan dalam hal model database yang digunakan.
4. Model data berorientasi objek memelihara hubungan melalui ...
5. Model data ... mewakili entitas sebagai kelas.
6. ... mewakili korespondensi antara berbagai elemen data.
7. Untuk mengakses informasi dari database diperlukan ...
8. Subskema adalah ... dari skema.
9. Kekebalan skema konseptual (atau eksternal) terhadap perubahan skema internal disebut sebagai ...
10. Kekebalan skema eksternal (atau program aplikasi) terhadap perubahan skema konseptual 10 disebut sebagai ...
11. Proses transformasi permintaan dan hasil antara tiga level disebut
12. Pemetaan konseptual/internal mendefinisikan korespondensi antara ... melihat dan
13. Pemetaan eksternal/konseptual mendefinisikan korespondensi antara tampilan ... tertentu dan tampilan ...
14. Model data hierarkis diwakili oleh pohon ...
15. Sistem Manajemen Informasi (IMS) dikembangkan bersama oleh ... dan ...
16. Model data jaringan diformalkan oleh ... pada akhir...
17. Tiga komponen dasar model jaringan adalah
    a. ...
    b. ...
    c. ...
18. Model data relasional pertama kali diperkenalkan oleh ...
19. Arsitektur client/server sistem database memiliki dua komponen logis yaitu ...dan...

JAWABAN

KUNCI JAWABAN BENAR/SALAH :

1. Benar	7. Benar	13. Benar	19. Benar
2. Salah	8. Benar	14. Benar	20. Benar
3. Benar	9. Benar	15. Benar	21. Benar
4. Benar	10. Salah	16. Salah	22. Benar
5. Benar	11. Benar	17. Benar	23. Salah
6. Benar	12. Benar	18. Benar	24. Benar

KUNCI JAWABAN SOAL ISIAN :

1. Tabel
2. Level Eksternal
3. Fisik
4. Entitas dan kelas
5. Berorientasi objek
6. Diagram ER
7. DBMS
8. Mewarisi
9. Data fisik independen
10. Independensi data logis
11. Konseptual, tersimpan database
12. Eksternal, konseptual
13. Terbalik
14. IBM, North American Aviation
15. DBTG/CODASYL, 1960-an
16. Jawaban :
    a. tipe catatan
    b. item data (atau field)
    c. linds
17. EF-Diagram
18. Klien, server.

SOAL BENAR/SALAH

1. Pada tahun 1980 Dr. E.F. Codd bekerja dengan Oracle Corporation.
2. DB2, System R dan ORACLE adalah contoh DBMS relasional.
3. Dalam terminologi RDBMS, tabel disebut relasi.
4. Model relasional didasarkan pada konsep inti relasi.
5. Kardinalitas tabel berarti jumlah kolom dalam tabel.
6. Dalam terminologi RDBMS, atribut berarti kolom atau bidang.
7. Domain adalah sekumpulan nilai atom.
8. Nilai data diasumsikan atom, yang berarti bahwa mereka tidak memiliki struktur internal sejauh model yang bersangkutan.

9. Sebuah tabel tidak boleh memiliki lebih dari satu atribut yang dapat mengidentifikasi baris secara unik.
10. Candidate key adalah atribut yang secara unik dapat mengidentifikasi sebuah baris dalam sebuah tabel.
11. Sebuah tabel hanya dapat memiliki satu kunci alternatif.
12. Sebuah tabel hanya dapat memiliki satu kunci kandidat.
13. Kunci asing dan kunci utama harus didefinisikan pada domain dasar yang sama.
14. Suatu relasi selalu memiliki pengenal unik.
15. Kunci utama melakukan fungsi identifikasi unik dalam model Database relasional.
16. Pada kenyataannya, NULL bukanlah nilai, melainkan ketiadaan nilai.
17. Basis data relasional adalah kumpulan terbatas dari relasi dan relasi dalam hal domain, atribut, dan tupel.
18. Atomic berarti bahwa setiap nilai dalam domain tidak dapat dibagi lagi dengan model relasional.
19. Superkey adalah sebuah atribut, atau sekumpulan atribut, yang secara unik mengidentifikasi sebuah tuple dalam sebuah relasi.
20. Codd mendefinisikan formula yang terbentuk dengan baik (WFF).

ISILAH BAGIAN TITIK-TITIK YANG KOSONG

1. Model relasional didasarkan pada konsep inti ...
2. Landasan teknologi database relasional diletakkan oleh ...
3. Dr. E.F Codd, dalam makalah berjudul ... meletakkan prinsip-prinsip dasar RDBMS.
4. Upaya pertama pada implementasi besar model relasional Codd adalah ...
5. Dalam terminologi RDBMS, sebuah record disebut ...
6. Derajat tabel berarti banyaknya ... dalam sebuah tabel.
7. Domain adalah sekumpulan nilai ...
8. Unit data terkecil dalam model relasional adalah individu ...
9. ... adalah himpunan semua nilai data yang mungkin.
10. Banyaknya atribut dalam suatu relasi disebut ... dari relasi tersebut.
11. Banyaknya tupel atau baris dalam suatu relasi disebut ... dari tabel
12. Sebuah tabel hanya dapat memiliki satu ... kunci.
13. Semua nilai yang muncul di kolom tabel harus diambil dari yang sama ...
14. Kalkulus relasional tupel awalnya diusulkan oleh ... di...

KUNCI JAWABAN

BENAR/SALAH

1. Salah
2. Benar
3. Benar
4. Benar
5. Salah
6. Benar
7. Benar
8. Benar
9. Salah
10. Salah
11. Salah
12. Salah
13. Benar
14. Benar
15. Benar
16. Benar
17. Benar
18. Benar
19. Benar
20. Benar

KUNCI JAWABAN ISIAN

1. Relasi dan teori himpunan,
2. Dr. EF Codd,
3. Model Relasional Data untuk Bank Data Bersama Besar,
4. Sistem R,
5. Tupel
6. Jumlah kolom,
7. Nilai hukum atau atom,
8. Bidang,
9. Bidang,
10. Derajat,
11. Kardinalitas,
12. Kunci Primer,
13. Relasi,
14. Dr. Codd, 1972.

SOAL BENAR/SALAH

1. Edgar F. Codd mengusulkan seperangkat aturan yang dimaksudkan untuk menentukan karakteristik dan kemampuan penting dari sistem relasional apa pun
2. Aturan Independensi Data Logis Codd menyatakan bahwa operasi pengguna dan program aplikasi harus independen dari setiap perubahan dalam struktur logis tabel dasar asalkan tidak melibatkan informasi yang hilang.
3. Seluruh bidang RDBMS berasal dari makalah Dr. E.F. Codd.
4. ISBL tidak memiliki operator agregat misalnya rata-rata, rata-rata dan sebagainya.
5. ISBL tidak memiliki fasilitas untuk penyisipan, penghapusan atau modifikasi tupel.
6. QUEL adalah bahasa kalkulus relasional tupel dari sistem Database relasional INGRESS (Graphics Interactive and Retrieval System)
7. QUEL mendukung operasi aljabar relasional seperti persimpangan, minus atau serikat.
8. RDBMS komersial pertama adalah DB2 IBM.
9. RDBMS komersial pertama adalah INGRES IBM.
10. SEQUEL dan SQL adalah sama
11. SQL adalah bahasa query relasional.
12. SQL pada dasarnya bukan bahasa format frec.
13. Pernyataan SQL dapat dipanggil baik secara interaktif dalam sesi terminal tetapi tidak dapat disematkan dalam program aplikasi.
14. Dalam tipe data SQL setiap objek data harus dideklarasikan oleh programmer saat menggunakan bahasa pemrograman.
15. Klausa HAVING setara dengan klausa WHERE digunakan untuk menentukan kriteria pencarian atau kondisi pencarian ketika klausa GROUP BY ditentukan.
16. Klausa HAVING digunakan untuk menghilangkan grup seperti halnya WHERE digunakan untuk menghilangkan baris.
17. Jika HAVEING ditentukan, klausa ORDER BY juga harus ditentukan.

18. Perintah ALTER TABLE memungkinkan kita untuk menghapus kolom dari sebuah tabel.
19. Bahasa definisi data SQL menyediakan perintah untuk mendefinisikan skema relasi, menghapus relasi dan memodifikasi skema relasi
20. Di SQL, tidak mungkin membuat tabel sementara lokal atau global dalam suatu transaksi.
21. Semua tugas yang berhubungan dengan pengelolaan data relasional tidak dapat dilakukan dengan menggunakan SQL saja.
22. Perintah DCL memungkinkan pengguna memasukkan data ke dalam database, memodifikasi dan menghapus data dalam database.
23. DML terdiri dari perintah yang mengontrol akses pengguna ke objek database.
24. Jika tidak ada yang ditentukan, kumpulan hasil disimpan dalam urutan menurun, yang merupakan default.
25. ,*, digunakan untuk mendapatkan semua kolom dari tabel tertentu.
26. Pernyataan CREATE TABLE membuat tabel dasar baru.
27. Tabel berbasis bukan tabel bernama otonom.
28. DDL digunakan untuk membuat, mengubah dan menghapus objek database.
29. Pernyataan administrasi data SQL (DAS) memungkinkan pengguna untuk melakukan audit dan analisis pada operasi dalam database.
30. Pernyataan COMMIT mengakhiri transaksi dengan sukses, membuat perubahan Database menjadi permanen.
31. Administrasi data Perintah memungkinkan pengguna untuk melakukan audit dan analisis pada operasi dalam database.
32. Laporan kontrol transaksi mengelola semua perubahan yang dibuat oleh pernyataan DML.
33. DOL memungkinkan pengguna untuk menanyakan satu atau lebih tabel untuk mendapatkan informasi yang mereka inginkan.
34. Dalam SQL tertanam, pernyataan SQL digabungkan dengan bahasa pemrograman host.
35. Kata kunci DISTINCT ilegal untuk MAX dan MIN.
36. Aplikasi yang ditulis dalam SQL dapat dengan mudah dipindahkan ke seluruh sistem.
37. Ouery-By-Example (OBE) adalah bahasa kalkulus domain dua dimensi.
38. OBE awalnya dikembangkan oleh M.M. Zloof di T.J. Pusat Penelitian Waston.
39. OBE merupakan pendekatan visual untuk mengakses informasi dalam database melalui penggunaan template query.
40. Permintaan tindakan membuat tabel OBE adalah kueri tindakan saat melakukan tindakan pada tabel atau tabel yang ada untuk membuat tabel baru.
41. OBE berbeda dari SQL karena pengguna tidak harus menentukan kueri terstruktur secara eksplisit.
42. Dalam OBE, pengguna tidak perlu mengingat nama atribut atau relasi, karena mereka ditampilkan sebagai bagian dari template.
43. Permintaan tindakan penghapusan OBE menghapus satu atau lebih dari satu catatan dari tabel atau lebih dari satu tabel.

ISILAH PADA TITIK-TITIK YANG KOSONG

1. Bahasa berbasis sistem informasi (ISBL) adalah bahasa query berbasis aljabar relasional murni, dikembangkan di ... di Inggris pada tahun ...
2. ISBL pertama kali digunakan dalam sistem manajemen Database interaktif eksperimental yang disebut
3. Dalam ISBL, untuk mencetak nilai ekspresi, perintah didahului oleh...
4. ... adalah kumpulan perintah standar yang digunakan untuk berkomunikasi dengan RDBMS.
5. Untuk mengkueri data dari tabel dalam database, kita menggunakan pernyataan ...
6. Bentuk QUEL yang diperluas adalah ...
7. QUEL adalah bahasa kalkulus relasional tupel dari sistem Database relasional yang disebut ...
8. QUEL didasarkan pada ...
9. INGRES dalam sistem manajemen Database relasional yang dikembangkan di ...
10. ... adalah definisi data dan bahasa manipulasi data untuk INGRES.
11. Pernyataan definisi data yang digunakan dalam QUEL
    a. ...
    b. ...
    c. ...
    d. ...
    e. ...
12. Pernyataan dasar pengambilan data dalam QUEL adalah ...
13. SEQUEL adalah bahasa query prototipe pertama dari ...
14. SEQUEL diimplementasikan dalam prototipe IBM yang disebut ...
15. SQL pertama kali diimplementasikan pada database relasional yang disebut ...
16. Operasi DROP SQL digunakan untuk ... tabel dari skema.
17. Bahasa definisi data SQL menyediakan perintah untuk
    a. ...
    b. ...
    c. ...
18. ... adalah contoh perintah atau pernyataan bahasa definisi data.
19. ... adalah contoh perintah atau pernyataan bahasa manipulasi data
20. Klausa ... mengurutkan atau mengurutkan hasil berdasarkan pada data dalam satu atau lebih kolom dalam urutan menaik atau menurun.
21. Klausa .... menentukan kueri ringkasan.
22. ... adalah contoh perintah atau pernyataan bahasa kontrol data.
23. Klausa ... menentukan tabel atau tabel dari mana data harus diambil.
24. Klausa ... mengarahkan SQL untuk memasukkan hanya baris data tertentu dalam kumpulan hasil.
25. ... adalah contoh perintah atau pernyataan sistem administrasi data
26. ... adalah contoh pernyataan pengendalian transaksi.

27. Pernyataan administrasi data (DAS) SQL memungkinkan pengguna untuk melakukan (a) ... dan (b) ... pada operasi dalam database.
28. Lima fungsi agregat yang disediakan oleh SQL adalah
    a. ...
    b. ...
    c. ...
    d. ...
    e. ...
29. Portabilitas atau SQL yang disematkan adalah ...
30. Query-BY-Example (OBE) adalah bahasa ... dua dimensi.
31. OBE awalnya dikembangkan oleh ... di TJ IBM. Pusat Riset Watson.
32. QBE ... membuat tabel baru dari semua atau sebagian data dalam satu atau lebih tabel.
33. OBE's ... dapat digunakan untuk memperbarui atau memodifikasi nilai dari satu atau lebih record dalam satu atau lebih dari satu tabel dalam database.
34. Pada QBE, query dirumuskan dengan mengisi ... atau relasi yang ditampilkan pada layar MS Access.

KUNCI JAWABAN

KUNCI JAWABAN BENAR/SALAH

1. Benar	12. Salah	23. Salah	34. Benar
2. Benar	13. Salah	24. Salah	35. Salah
3. Benar	14. Benar	25. Benar	36. Benar
4. Benar	15. Benar	26. Benar	37. Benar
5. Benar	16. Benar	27. Benar	38. Benar
6. Benar	17. Salah	28. Benar	39. Benar
7. Benar	18. Benar	29. Benar	40. Benar
8. Salah	19. Benar	30. Benar	41. Benar
9. Benar	20. Salah	31. Benar	42. Benar
10. Benar	21. Salah	32. Benar	43. Benar
11. Benar	22. Salah	33. Benar	

KUNCI JAWABAN ISIAN

1. IBM's Peterlee Centre, 1973
2. Peterlee Relational Test Vehicle (PRTV),
3. LIST,
4. SQL,
5. SELECT,
6. Ouery Language,
7. INGRESS,
8. Beberapa bahasa kalkulus relasional sistem database relatinal INGRESS
9. IBM,

10. SQL,
11. (a) CREATE, (b) RETREIVE, (c) DELETE, (d) SORT, (e) PRINT
12. RETREIVE,
13. IBM,
14. Sistem R,
15. SistemR,
16. Menghapus,
17. (a) mendefinisikan skema relasi, (b) menghapus relasi, (c) memodifikasi skema relaiton,
18. CREATE ALTER DROp,
19. INSERT, DELETE UPDAT,
20. ORDER OLEH,
21. GROUP BY
22. HIBAH, REVOKE,
23. FROM,
24. WHERE,
25. START AUDIT, STOP AUDIT,
26. COMMIT, ROLLBACK,
27. (a) audit, (b) analysis,
28. (a) AVG, (b) SUM, (c) MIN, (d) MAX, (e),
29. Sangat tinggi,
30. Kalkulus domain,
31. MM Zloo,
32. Make-table,
33. Perintah U.

SOAL BENAR/SALAH

1. Model E-R pertama kali diperkenalkan oleh Dr. E.F. Codd.
2. Pemodelan E-R adalah model data konseptual tingkat tinggi yang dikembangkan untuk memfasilitasi desain database
3. Model ER bergantung pada sistem manajemen datab!lse (DBMS) dan platform perangkat keras tertentu.
4. Hubungan biner ada ketika asosiasi dipertahankan dalam satu entitas.
5. Tipe entitas yang lemah tidak bergantung pada keberadaan entitas lain.
6. Tipe entitas adalah sekelompok objek dengan properti yang sama, yang diidentifikasi oleh perusahaan sebagai memiliki keberadaan yang independen.
7. Kemunculan entitas disebut juga dengan instance entitas.
8. Instance entitas adalah objek yang dapat diidentifikasi secara unik dari tipe entitas.
9. Hubungan adalah persekutuan antara dua atau lebih entitas yang menjadi kepentingan perusahaan.
10. Partisipasi adalah opsional jika keberadaan entitas memerlukan keberadaan entitas terkait dalam hubungan tertentu.

11. Jenis entitas tidak memiliki keberadaan independen.
12. Atribut dipandang sebagai item dunia nyata atom.
13. Domain bisa terdiri dari lebih dari satu domain.
14. Derajat suatu relasi adalah jumlah entitas yang terkait atau partisipan 10 relasi tersebut.
15. Konektivitas hubungan menggambarkan kendala pada pemetaan kejadian entitas terkait dalam hubungan.
16. Dalam hal keberadaan wajib, kemunculan" entitas itu tidak perlu ada.
17. Atribut adalah properti dari suatu entitas atau tipe hubungan.
18. Dalam diagram ER, jika atributnya sederhana atau bernilai tunggal maka atribut tersebut terhubung menggunakan garis ganda
19. Dalam Diagram ER, jika atribut diturunkan ketika dihubungkan menggunakan garis ganda
20. Tipe entitas yang tidak bergantung pada keberadaan beberapa tipe entitas lain disebut tipe entitas kuat.
21. Entitas yang lemah juga disebut sebagai entitas anak, dependen atau subordinat.
22. Tipe entitas dapat menjadi objek dengan keberadaan fisik tetapi tidak dapat menjadi objek dengan keberadaan konseptual.
23. Atribut sederhana dapat dibagi lagi.
24. Dalam diagram E-R, nama entitas ditulis dengan huruf besar sedangkan nama atribut ditulis dengan huruf kecil.

ISILAH BAGIAN TITIK-TITIK YANG KOSONG

1. Model E-R diperkenalkan oleh ... di ...
2. Entitas adalah ... atau ... di dunia nyata.
3. Suatu hubungan adalah ... di antara dua atau lebih ... yaitu menarik bagi perusahaan.
4. Kejadian tertentu dari suatu hubungan disebut ...
5. Model database menggunakan (a) ... (b) ... (c) ... untuk membangun representasi dari sistem dunia nyata.
6. Relasi dihubungkan oleh ... ke entitas yang berpartisipasi dalam relasi.
7. Asosiasi antara tiga entitas disebut ...
8. Hubungan antara instance dari tipe entitas tunggal disebut ...
9. Asosiasi antara dua entitas disebut ...
10. Hitungan sebenarnya dari elemen yang terkait dengan konektivitas disebut .... dari konektivitas hubungan.
11. Atribut adalah properti dari ... atau tipe ...
12. Komponen atau entitas atau kualifikasi yang menjelaskannya disebut ... dari entitas.
13. Dalam diagram E-R, ... direpresentasikan dengan kotak persegi panjang dengan nama entitas di dalam kotak.
14. Komponen utama diagram ER adalah (a) ... (b) ... (c) ... (d) ...
15. Diagram ER menangkap (a) ... dan (b) ...
16. ... entitas juga disebut sebagai entitas induk, pemilik atau dominan.

17. ... adalah atribut yang terdiri dari satu komponen dengan keberadaan yang independen.
18. Pada diagram E-R ... digarisbawahi.
19. Setiap instance yang dapat diidentifikasi secara unik dari tipe entitas juga disebut sebagai ... atau
20. Hubungan ... ada ketika dua entitas diasosiasikan.
21. Dalam diagram E-R, jika atributnya adalah ... atribut komponennya yang ditampilkan adalah elips yang berasal dari atribut komposit.

KUNCI JAWABAN

KUNCI JAWABAN BENAR/SALAH

1. Salah	7. Benar	13. Benar	19. Salah
2. Benar	8. Benar	14. Benar	20. Benar
3. Salah	9. Benar	15. Benar	21. BenarSalah
4. Salah	10. Benar	16. Salah	22. Salah
5. Salah	11. Salah	17. Benar	23. Salah
6. Benar	12. Benar	18. Benar	24. Salah

KUNCI JAWABAN SOAL ISISAN

1. PP Chen,
2. Objek, benda,
3. Asosiasi, entitas,
4. Konektivitas,
5. (a) entitas, (b) Atribut, (c) relasi,
6. Garis,
7. Relasi ternary,
8. Rekursif relasi,
9. Relasi biner,
10. Kardinalitas,
11. Entitas, relasi,
12. Atribut atau item data,
13. Himpunan entitas,
14. (a) himpunan entitas, (b) himpunan relasi, (c) atribut, (d) pemetaan caradinalitas,
15. Data (b) organisasi data,
16. Tipe entitas kuat,
17. Atribut sederhana,
18. Kunci utama,
19. Kemunculan entitas, instance entitas,
20. Biner,
21. Komposit,

SOAL BENAR/SALAH

1. Subclass adalah sub-pengelompokan kemunculan entitas dalam tipe entitas yang memiliki atribut atau hubungan umum yang berbeda dari sub-pengelompokan lainnya.
2. Dalam kasus supertipe, objek dalam satu set dikelompokkan atau dibagi lagi menjadi satu atau lebih kelas dalam banyak sistem.
3. Superclass adalah tipe entitas generik yang memiliki hubungan dengan satu atau lebih subtipe.
4. Setiap anggota subclass juga merupakan anggota dari superclass.
5. Hubungan antara superclass dan subclass adalah hubungan satu-ke-banyak (1 : N).
6. Simbol berbentuk V dalam model ERR menunjukkan bahwa supertipe adalah subset dari subtipe.
7. Warisan atribut adalah properti dimana entitas supertipe mewarisi nilai dari semua atribut dari SUbtipe.
8. Spesialisasi adalah proses mengidentifikasi himpunan bagian dari himpunan entitas dari superclass atau supertype yang memiliki beberapa karakteristik yang membedakan. 9.
9. Spesialisasi meminimalkan perbedaan antara anggota suatu entitas dengan mengidentifikasi karakteristik yang membedakan dan unik dari setiap anggota
10. Generalisasi adalah proses mengidentifikasi beberapa karakteristik umum dari kumpulan himpunan entitas dan membuat himpunan entitas baru yang berisi entitas yang memproses karakteristik umum tersebut.
11. Generalisasi memaksimalkan perbedaan antara entitas dengan mengidentifikasi fitur umum.
12. Partisipasi total disebut juga partisipasi opsional.
13. Partisipasi total menetapkan bahwa setiap anggota (atau entitas) dalam supertipe (atau superkelas) harus berpartisipasi sebagai anggota dari beberapa subkelas dalam spesialisasi/generalisasi.
14. Batasan partisipasi dapat bersifat total atau sebagian.
15. Batasan partisipasi parsial menetapkan bahwa anggota supertipe tidak perlu menjadi bagian dari subkelas spesialisasi/generalisasinya
16. Kendala non-joint juga disebut kendala tumpang tindih.
17. Partisipasi parsial disebut juga partisipasi wajib.
18. Batasan terputus menentukan hubungan antara anggota subtipe dan menunjukkan . apakah mungkin bagi anggota supertipe untuk menjadi anggota dari satu, atau lebih dari satu, subtipe.
19. Batasan disjoint hanya diterapkan jika itu adalah supertipe.
20. Partisipasi parsial direpresentasikan menggunakan garis tunggal antara supertipe dan lingkaran spesialisasi/generalisasi.
21. Subtipe bukanlah entitas pada onwnya.
22. sebuah commot subtipe memiliki subtipe sendiri.

ISILAH BAGIAN TITIK KOSONG

1. Hubungan antara superclass dan subclass adalah ...
2. Simbol berbentuk V dalam model EER menunjukkan bahwa ... adalah ... dari ..
3. Warisan atribut adalah properti dimana ... entitas nilai yang melekat dari semua atribut ...
4. Model E-R yang didukung dengan konsep semantik tambahan disebut ...
5. Pewarisan atribut menghindari

KUNCI JAWABAN

KUNCI JAWABAN SALAH/BENAR

1. Benar
2. Benar
3. Benar
4. Benar
5. Benar
6. Salah
7. Salah
8. Benar
9. Benar
10. Benar
11. Benar
12. Salah
13. Benenar
14. Benar
15. Benar
16. Benar
17. Salah
18. Benar
19. Salah
20. Benar
21. Benar
22. Benar
23. Benar
24. Benar

SOAL BENAR/SALAH

1. Sebuah ketergantungan fungsional (FD) adalah properti dari informasi yang diwakili oleh relasi.
2. Ketergantungan fungsional memungkinkan perancang database untuk mengungkapkan fakta tentang perusahaan yang dimodelkan oleh perancang dengan database perusahaan.
3. Sebuah ketergantungan finctional adalah hubungan banyak-ke-banyak antara dua set atribut X dan Y dari tabel T.
4. Istilah ketergantungan fungsional penuh (FFD) digunakan untuk menunjukkan himpunan maksimum atribut dalam determinan ketergantungan fungsional (FD).
5. Ketergantungan fungsional dalam himpunan adalah berlebihan jika dapat diturunkan dari dependensi fungsional lain dalam himpunan.
6. Penutupan himpunan (juga disebut lengkap) dari ketergantungan fungsional mendefinisikan semua FD yang dapat diturunkan dari himpunan FD tertentu.
7. Dekomposisi fungsional adalah proses penguraian fungsi-fungsi organisasi menjadi tingkat-tingkat perincian yang semakin besar (lebih halus dan lebih halus).
8. Kata loss in lossless mengacu pada hilangnya atribut.
9. Ketergantungan dipertahankan karena setiap ketergantungan di F mewakili batasan pada database.
10. Jika dekomposisi tidak mempertahankan ketergantungan, beberapa ketergantungan hilang dalam dekomposisi.

ISILAH BAGIAN TITIK KOSONG

1. Sebuah ... adalah hubungan banyak ke satu antara dua himpunan ... dari suatu relasi.

2. Ruas kiri dan ruas kanan dari ketergantungan fungsional disebut (a) ... dan ... masing-masing.
3. Notasi panah '~' pada FD dibaca sebagai ...
4. Istilah ketergantungan fungsional penuh (FFD) digunakan untuk menunjukkan ... himpunan atribut dalam ... ketergantungan fungsional (FD).
5. Ketergantungan fungsional dalam himpunan adalah redundan jika dapat diturunkan dari yang lain ... dalam himpunan.
6. Penutupan suatu himpunan (juga disebut himpunan lengkap) dari ketergantungan fungsional mendefinisikan semua ... yang dapat diturunkan dari himpunan ...
7. Dekomposisi fungsional adalah proses ... fungsi-fungsi organisasi ke dalam tingkat detail yang semakin besar (awal dan awal).
8. Dekomposisi lossless-join adalah sifat dekomposisi, yang memastikan bahwa tidak ada... dihasilkan ketika operasi ... diterapkan pada hubungan dalam dekomposisi.
9. Kata loss in lossless mengacu pada ...
10. Aksioma Armstrong dan aturan turunan dapat digunakan untuk mencari ... FD.

KUNCI JAWABAN

KUNCI JAWABAN BENAR/SALAH

1. Benar
2. Benar
3. Salah
4. Salah
5. Benar
6. Benar
7. Benar
8. Salah
9. Benar
10. Benar

KUNCI JAWABAN ISIAN

1. Ketergantungan fungsional, atribut,
2. (a) determinan, (b) dependen,
3. Secara fungsional menentukan,
4. Minimum, determinan,
5. Ketergantungan fungsional,
6. FD, FD, 7. Melanggar,
7. Tupel palsu, gabung alami,
8. Kehilangan informasi,
9. Set non-redundan dan set lengkap (atau penutupan) Of

SOAL BENAR/SALAH

1. Normalisasi adalah proses penguraian sekumpulan relasi dengan anomali untuk menghasilkan relasi yang lebih kecil dan terstruktur yang mengandung minimal atau tidak ada redundansi
2. Suatu relasi dikatakan dalam 1NF jika nilai-nilai dalam domian dari setiap atribut relasi tersebut adalah non-atomik.
3. 1NF tidak mengandung informasi yang berlebihan.
4. 2NF selalu dalam 1NF.

5. 2NF adalah penghapusan dependensi fungsional parsial atau data yang berlebihan.
6. Ketika sebuah relasi R dalam 2NF dengan FDs A →Band B →CDEF (dengan A adalah satu-satunya kunci kandidat), didekomposisi menjadi dua relasi Rl (dengan A →B) dan R2 (dengan B →CDEF) , relasi R_I dan R_2
    a. selalu merupakan dekomposisi lossless dari R.
    b. Biasanya memiliki total ruang penyimpanan gabungan kurang dari R.
    c. tidak memiliki anomali hapus.
    d. akan selalu lebih cepat untuk mengeksekusi query daripada R.
7. Ketika sebuah relasi R dalam 3NF dengan FDs AB →C, relasi Rl (dengan AB →null, yaitu all key) dan R_2 (dengan C →R), hubungan R_I dan R_2.
    a. selalu merupakan dekomposisi lossless dari R.
    b. keduanya merupakan pelestarian ketergantungan. (c)
    c. keduanya dalam BCNF.
8. Ketika sebuah relasi R dalam BCNF dengan FDs A →BCD (dengan A adalah primary key) didekomposisi menjadi dua relasi R_I (dengan A →B) dan R_2 (dengan A →CD), maka dihasilkan dua relasi R_I dan R_2·
    a. selalu mempertahankan ketergantungan.
    b. biasanya memiliki total ruang penyimpanan gabungan kurang dari R.
    c. tidak memiliki anomali penghapusan.
9. Dalam 3NF, tidak ada atribut non-prima yang secara fungsional bergantung pada atribut non-prima lainnya
10. Dalam BCNF, sebuah relasi hanya boleh memiliki kunci kandidat sebagai determinan.
11. Dependensi lossless-join adalah properti dekomposisi, yang memastikan bahwa tidak ada tupel palsu yang dihasilkan ketika relasi dikembalikan melalui operasi join alami.
12. Ketergantungan multi-nilai adalah hasil dari INF, yang melarang atribut memiliki kumpulan nilai.
13. 5NF tidak memerlukan beberapa hubungan semantik terkait.
14. Normalisasi adalah proses formal untuk mengembangkan struktur data dengan cara menghilangkan redundansi dan meningkatkan integritas. 15.
15. 5NF juga disebut bentuk normal proyeksi-gabungan (PJNF)

ISILAH BAGIAN TITIK KOSONG

1. Normalisasi adalah proses ... sekumpulan relasi dengan anomali untuk menghasilkan relasi yang lebih kecil terstruktur dengan baik yang mengandung minimum atau tidak ada ...
2. ... adalah proses formal untuk memutuskan atribut mana yang harus dikelompokkan bersama.
3. Dalam proses ... kami menganalisis dan menguraikan hubungan yang kompleks dan mengubahnya menjadi hubungan yang lebih kecil, lebih sederhana, dan terstruktur dengan baik.
4. ... pertama kali mengembangkan proses normalisasi.
5. Suatu relasi dikatakan dalam INF jika nilai-nilai dalam domain dari setiap atribut dari relasi tersebut adalah...

6. Suatu relasi R dikatakan berada dalam 2NF jika berada di ... dan setiap atribut non-prime key dari R adalah ... pada setiap kunci relasi R.
7. 2NF hanya dapat dilanggar jika kunci adalah ... kunci atau satu yang terdiri dari lebih dari satu
8. Ketika atribut multi-nilai atau grup berulang dalam suatu relasi dihilangkan maka relasi tersebut dikatakan berada di...
9. Dalam 3NF, tidak ada atribut non-prima yang secara fungsional bergantung pada ...
10. Relasi R dikatakan dalam BCNF jika untuk setiap FD nontrivial: ... antara atribut X dan Y terdapat pada R.
11. Suatu relasi dikatakan berada pada ... ketika dependensi transitif dihilangkan.
12. Suatu relasi dalam BCNF jika dan hanya jika setiap determinan adalah ...
13. Setiap relasi di BCNF juga ada di ... dan akibatnya di ...
14. Perbedaan antara 3NF dan BCNF adalah bahwa untuk ketergantungan fungsional A →B, 3NF mengizinkan ketergantungan ini dalam suatu relasi jika B adalah ... atribut kunci dan A bukan sebuah ... kunci. Sedangkan BCNF menegaskan bahwa agar ketergantungan ini tetap ada dalam suatu relasi, A harus berupa kunci ...
15. 4NF dilanggar ketika suatu relasi tidak diinginkan ...
16. Suatu relasi dikatakan berada dalam 5NF jika setiap ketergantungan join adalah ... dari kunci relasinya.

KUNCI JAWABAN

1. Benar
2. Salah
3. Salah
4. Benar
5. Benar
6. Benar
7. Benar
8. Benar
9. Salah
10. Benar
11. Benar
12. Salah
13. Benar
14. Benar
15. Benar

KUNCI JAWABAN ISIAN

1. Mendekomposisi redundansi,
2. Normalisasi,
3. Normalisasi,
4. EF Codd,
5. Atomic,
6. 1NF, dependen secara fungsional penuh,
7. Atribut komposit,
8. 1NF
9. Kunci primer (atau relasional),
10. XY
11. 3NF
12. Kunci kandidat,
13. 3NF, 2NF
14. Primer, kandidat, kandidat,

15. MVD
16. Konsekuensi.

ISILAH DIBAGIAN TITIK KOSONG

1. Pemroses kueri mengubah kueri ... menjadi ... yang melakukan pengambilan dan manipulasi yang diperlukan dalam database.
2. Rencana pelaksanaan adalah serangkaian ... langkah-langkah.
3. Dalam fase pengecekan sintaks dari pemrosesan kueri sistem ... kueri dan memeriksa apakah mematuhi aturan ...
4. ... adalah prosesnya. mengubah kueri yang ditulis dalam SQL (atau bahasa tingkat tinggi apa pun) menjadi strategi eksekusi yang benar dan efisien yang diekspresikan dalam bahasa tingkat rendah.
5. Selama proses transformasi kueri, ... memeriksa sintaks dan memverifikasi apakah relasi dan atribut yang digunakan dalam kueri di-dermed dalam database.
6. Transformasi query dilakukan dengan mengubah query menjadi ... yang lebih efisien untuk dieksekusi.
7. Empat fase utama pemrosesan kueri adalah (a) ... (b) ... (c) ... dan (d)...
8. Dua jenis teknik optimasi query adalah (a) ... dan (b) ....
9. Di ... kueri diuraikan, divalidasi, dan dioptimalkan satu kali.
10. Tujuan dari ... adalah untuk mengubah kueri tingkat tinggi menjadi kueri aljabar relasional dan untuk memeriksa apakah kueri tersebut benar secara sintaksis dan semantik.
11. Lima tahapan dekomposisi query adalah (a) ... (b) ... (c) ... (d) ... (e) ...
12. Pada tahap ... kueri dianalisis secara leksikal dan sintaksis menggunakan parser untuk mengetahui kesalahan sintaksis.
13. Pada tahap ... query diubah menjadi bentuk normalisasi yang dapat lebih mudah dimanipulasi
14. Pada tahap ..., salah formulasi dan kueri yang kontradiktif ditolak.
15. .... menggunakan aturan transformasi untuk mengubah satu ekspresi aljabar relasional menjadi bentuk ekuivalen yang lebih efisien.
16. Komponen biaya utama optimasi query adalah (a) ... dan (b) ...
17. Pohon kueri juga disebut pohon ...
18. Biasanya aturan heuristik digunakan dalam bentuk ..... atau ..... struktur data.
19. Algoritma optimasi heuristik menggunakan beberapa aturan transformasi untuk mengubah ..... pohon kueri menjadi ..... dan .. .. pohon kueri
20. Penekanan minimalisasi biaya tergantung pada ..... dan ..... dari aplikasi database.
21. Proses evaluasi query di mana beberapa operasi relasional digabungkan menjadi sebuah pipeline operasi disebut .....

22. Jika hasil dari proses antara dalam sebuah query dibuat dan kemudian digunakan untuk evaluasi operasi tingkat berikutnya, eksekusi query semacam ini disebut ....

KUNCI JAWABAN

KUNCI JAWABAN ISIAN

1. Tingkat tinggi, rencana eksekusi,
2. Komplikasi kueri,
3. Parsing, sintaks,
4. Pemrosesan kueri,
5. Syntax analyzer menggunakan tata bahasa SQL sebagai input dan protion parser dari query processor,
6. Ekspresi aljabar (query aljabar relasional),
7. (a) Syntax analyzer, (b) Query dekomposer, (c) Query Optimizer (d) Generator kode kueri,
8. (a) Optimasi kueri heuristik, (b) Estimasi sistematis.
9. Pengoptimal kueri,
10. Pengurai kueri,
11. (a) Analisis kueri, (b) normalisasi kueri, (c) Analisis semantik, (d) Penyederhanaan kueri, (e) Restrukturisasi kueri,
12. Analisis kueri,
13. kueri normalisasi,
14. Penganalisis semantik,
15. Restrukturisasi kueri,
16. (a) jumlah liD, (b) waktu CPU,
17. Aljabar relasional,
18. kueri, grafik kueri,
19. Inisial (kanonik), dioptimalkan, dapat dieksekusi secara efisien ,
20. Ukuran, jenis,
21. Pemrosesan langsung,
22. Perwujudan.

SOAL SALAH/BENAR

1. Transaksi terdiri dari semua operasi yang dijalankan antara awal dan akhir transaksi.
2. Transaksi adalah unit program, yang dapat disematkan di dalam program aplikasi atau dapat ditentukan secara interaktif ~ sebuah bahasa query tingkat tinggi seperti SQL.
3. Perubahan yang dibuat ke database oleh transaksi yang dibatalkan harus dihormati atau dibatalkan.
4. Transaksi yang dilakukan atau dibatalkan dikatakan dihentikan.
5. Transaksi atomik adalah transaksi di mana semua tindakan yang terkait dengan transaksi dijalankan sampai selesai, atau tidak ada yang dilakukan.

6. Efek dari transaksi yang berhasil diselesaikan secara permanen dicatat dalam database dan tidak boleh hilang karena kegagalan berikutnya.
7. Tingkat transaksi dapat dipulihkan.
8. Transaksi Level 1 adalah persyaratan konsistensi minimum yang memungkinkan transaksi dipulihkan jika terjadi kegagalan sistem.
9. Log adalah catatan semua transaksi dan perubahan terkait ke database.
10. Konsistensi transaksi Level 2 terisolasi dari pembaruan transaksi lainnya.
11. DBMS secara otomatis memperbarui log transaksi saat menjalankan transaksi yang memodifikasi database.
12. Transaksi berkomitmen yang telah melakukan pembaruan mengubah database menjadi keadaan konsisten baru.
13. Tujuan dari concurrency control adalah untuk menjadwalkan Qr mengatur transaksi sedemikian rupa untuk menghindari gangguan.
14. Masalah analisis yang salah juga dikenal sebagai pembacaan kotor atau pembacaan yang tidak dapat diulang.
15. Status database yang konsisten di mana semua batasan integritas data terpenuhi.
16. Eksekusi serial selalu meninggalkan database dalam keadaan yang konsisten meskipun hasil yang berbeda dapat dihasilkan tergantung pada urutan eksekusi.
17. Rollback berjenjang tidak diinginkan.
18. Penguncian dan urutan stempel waktu adalah teknik optimis, karena dirancang berdasarkan asumsi bahwa konflik jarang terjadi.
19. Jenis kunci lock adalah kuncci read dan write.
20. Dalam penguncian dua fase, setiap transaksi dibagi menjadi (a) fase tumbuh dan (b) fase menyusut.
21. Masalah pembacaan kotor terjadi ketika satu transaksi memperbarui item database dan kemudian transaksi gagal karena suatu alasan.
22. Ukuran item yang dikunci menentukan perincian kunci.
23. Tidak ada Deadlock dalam metode stempel waktu kontrol konkurensi.
24. Transaksi yang mengubah isi database harus mengubah database dari satu keadaan konsisten ke keadaan lain.
25. Suatu transaksi dikatakan dalam keadaan berkomitmen jika telah dilakukan sebagian, dan dapat dipastikan tidak akan pernah dibatalkan.
26. Konsistensi transaksi Level 3 menambahkan pembacaan yang konsisten sehingga pembacaan record yang berurutan akan selalu memberikan nilai yang sama
27. Masalah pembaruan yang hilang terjadi ketika dua transaksi yang mengakses item database yang sama memiliki operasinya dengan cara yang membuat nilai beberapa item Database salah.
28. Serialisability menggambarkan eksekusi beberapa transaksi secara bersamaan.

ISILAH BAGIAN TITIK KOSONG

1. Transaksi adalah ..... pekerjaan yang mewakili peristiwa dunia nyata dari setiap organisasi atau perusahaan, sedangkan kontrol konkurensi adalah manajemen pelaksanaan transaksi bersamaan.
2. ..... adalah kegiatan koordinasi tindakan proses yang beroperasi secara paralel, mengakses data bersama, dan karena itu, berpotensi mengganggu satu sama lain.
3. Cara sederhana untuk mendeteksi keadaan Deadlock adalah sistem membangun dan memelihara ..... grafik.
4. Transaksi adalah urutan tindakan ..... dan ..... yang dikelompokkan bersama untuk membentuk database.
5. ..... adalah kemampuan suatu DBMS untuk mengelola berbagai transaksi yang terjadi di dalam sistem.
6. Transaksi atom adalah transaksi di mana ..... dengan transaksi dijalankan sampai selesai atau ..... dilakukan .
7. Sifat ACID dari suatu transaksi adalah (a) ..... (b) ..... (c) .. .. dan (d)
8. ..... berarti bahwa eksekusi transaksi secara terpisah menjaga konsistensi database.
9. The ..... dari DBMS memastikan atomisitas setiap transaksi.
10. Log transaksi adalah ..... dari semua ..... dan perubahan yang sesuai dengan .....
11. Memastikan daya tahan adalah tanggung jawab ..... dari DBMS
12. Sifat isolasi transaksi berarti bahwa data yang digunakan selama pelaksanaan transaksi tidak dapat digunakan oleh ..... sampai .yang pertama selesai
13. Status database yang konsisten adalah satu 'yang semua ..... kendala terpenuhi.
14. Transaksi yang mengubah isi database harus mengubah database dari satu ..... ke yang lain.
15. Properti isolasi adalah tanggung jawab ..... atau DBMS.
16. Transaksi yang berhasil menyelesaikan eksekusinya disebut .....
17. Konsistensi transaksi level 2 terisolasi dari ..... transaksi lainnya.
18. Ketika sebuah transaksi belum berhasil menyelesaikan eksekusinya, kami mengatakan bahwa transaksi tersebut telah
19. ..... adalah jadwal di mana operasi dari sekelompok transaksi bersamaan disisipkan.
20. Tujuan dari ..... adalah untuk menemukan jadwal non-serial.
21. Situasi di mana kegagalan transaksi tunggal menyebabkan serangkaian rollback disebut .....
22. .....adalah ukuran item data yang dipilih sebagai unit proteksi oleh program kontrol konkurensi.
23. Teknik kontrol konkurensi optimis disebut juga ..... skema konkurensi.
24. Satu-satunya cara untuk membatalkan efek dari transaksi yang dilakukan adalah dengan mengeksekusi .....
25. Kumpulan operasi yang membentuk satu unit kerja logis disebut .....

26. Serialisability harus dijamin untuk mencegah ..... dari transaksi yang mengganggu satu sama lain.
27. Grafik prioritas digunakan untuk menggambarkan .....
28. Kunci mencegah akses ke ..... oleh transaksi kedua sampai transaksi pertama menyelesaikan semua tindakannya.
29. Kunci bersama/eksklusif (atau Read/Write) menggunakan ..... kunci.
30. Kunci bersama ada ketika transaksi bersamaan diberikan ..... akses berdasarkan kunci umum.
31. Penguncian fase-thro adalah metode pengendalian .....di mana semua operasi penguncian mendahului operasi pembukaan pertama.
32. Dalam fase pertumbuhan, transaksi yang diperoleh terkunci tanpa ..... data apa pun.
33. Dalam fase menyusut, transaksi melepaskan ..... dan tidak dapat memperoleh kunci ..... apa pun.

KUNCI JAWABAN

KUNCI JAWABAN BENAR/SALAH

1. Benar
2. Benar
3. Benar
4. Benar
5. Benar
6. Benar
7. Salah
8. Benar
9. Benar
10. Benar
11. Benar
12. Benar
13. Benar
14. Salah
15. Benar
16. Benar
17. Benar
18. Benar
19. Benar
20. Benar
21. Benar
22. Benar
23. Benar
24. Benar
25. Benar
26. Benar
27. Benar
28. Benar

KUNCI JAWABAN ISIAN

1. Unit logis,
2. Kontrol konkurensi,
3. Grafik tunggu,
4. Read, Write,
5. 5. Kontrol konkurensi,
6. Semua tindakan yang terkait, tidak ada,
7. (a) Atomisitas, (b) Konsistensi, (c) Isolasi, (d) Daya Tahan
8. Isolasi,
9. Subsistem pemulihan transaksi,
10. Catatan, transaksi, Database,
11. Subsistem pemulihan,
12. Transaksi kedua,
13. Integritas data,
14. Status konsisten,
15. Kontrol konkurensi,
16. Berkomitmen,

17. Pembaruan,
18. Dibatalkan,
19. Jadwal non-serial,
20. Serializability,
21. Cascading rollback,
22. Granularity,
23. Validasi atau metode sertifikasi,
24. Rollback,
25. Transaksi,
26. Inkonsistensi
27. Serializability
28. Database record
29. Multiple-mode
30. READ
31. Pemrosesan serentak
32. Unlocking
33. Semua kunci, baru.

SOAL BENAR/SALAH

1. Kontrol konkurensi dan pemulihan Database saling terkait dan keduanya merupakan bagian dari manajemen transaksi.
2. Pemulihan Database adalah layanan yang disediakan oleh DBMS untuk memastikan bahwa Database dapat diandalkan dan tetap dalam keadaan konsisten jika terjadi kegagalan.
3. Pemulihan Database adalah proses Database ke keadaan yang benar (konsisten) jika terjadi kegagalan.
4. Pemulihan ke depan adalah prosedur pemulihan, yang digunakan jika terjadi kerusakan fisik.
5. Back ward recovery adalah prosedur pemulihan, yang digunakan jika terjadi kesalahan di tengah operasi normal pada database.
6. Kegagalan media adalah kegagalan yang paling berbahaya.
7. Pemulihan media dilakukan ketika ada head crash (rekaman tergores oleh jarum fonograf) pada disk
8. Proses pemulihan erat kaitannya dengan sistem operasi.
9. Teknik paging bayangan tidak memerlukan penggunaan log transaksi di lingkungan pengguna tunggal.
10. Dalam shadowing, baik sebelum-gambar dan sesudah-gambar disimpan pada disk, dengan demikian, menghindari kebutuhan untuk log transaksi untuk proses pemulihan.
11. Operasi REDO memperbarui database dengan nilai baru (gambar setelah) yang disimpan dalam log.

12. Operasi REDO menyalin nilai-nilai lama dari log ke database, dengan demikian, memulihkan database sebelum keadaan sebelum dimulainya transaksi.
13. Dalam hal teknik updata yang ditangguhkan, pembaruan tidak ditulis ke database sampai setelah transaksi mencapai titik COMMIT.
14. Dalam hal teknik pembaruan segera, semua pembaruan ke database diterapkan segera saat terjadi dengan menunggu untuk mencapai titik COMMIT dan catatan semua perubahan disimpan dalam log transaksi.
15. Checkpoint adalah titik sinkronisasi antara database dan file log transaksi.
16. Di pos pemeriksaan, semua buffer ditulis paksa ke penyimpanan sekunder.
17. Teknik pembaruan yang ditangguhkan juga dikenal sebagai algoritma UNDO/REDO.
18. Shadow paging adalah teknik di mana log transaksi tidak diperlukan.
19. Pemulihan mengembalikan database dari keadaan tertentu, biasanya tidak konsisten, ke keadaan sebelumnya yang konsisten.
20. Penugasan dan pengelolaan blok memori disebut buffer manager.

ISILAH BAGIAN TITIK KOSONG

1. ..... adalah proses mengembalikan database ke keadaan yang benar bahkan dari kegagalan.
2. Jika hanya transaksi yang harus dibatalkan, maka disebut .....
3. Ketika semua transaksi yang aktif harus dibatalkan, maka disebut .....
4. Jika semua halaman yang diperbarui oleh suatu transaksi segera ditulis ke disk saat transaksi melakukan ini .....
5. Jika halaman di-flush ke disk hanya saat penuh atau pada interval waktu tertentu, maka disebut .....
6. Teknik paging bayangan tidak memerlukan penggunaan log transaksi di lingkungan .....
7. Teknik shadow paging diklasifikasikan sebagai ..... algoritma.
8. Kontrol konkurensi dan pemulihan Database saling terkait dan keduanya merupakan bagian dari .....
9. Pemulihan diperlukan untuk melindungi database dari (a) ..... dan (b) .....
10. Kegagalan dapat disebabkan oleh (a) ..... , (b) ..... , (c) .....
11. Pemulihan memulihkan database dari status tertentu, biasanya ....., ke status .....
12. Cadangan Database disimpan di tempat yang aman, biasanya di (a) ..... dan (b) ..... .seperti kebakaran, pencurian, banjir dan potensi bencana lainnya.
13. Sistem crash karena kesalahan perangkat keras atau perangkat lunak, mengakibatkan hilangnya .....
14. Pada saat terjadi kegagalan, ada dua efek utama yang terjadi, yaitu (a) ..... dan (b) .....
15. Pemulihan media dilakukan bila ada ..... pada disk.
16. Dalam hal teknik pembaruan yang ditangguhkan, pembaruan tidak ditulis ke database sampai setelah transaksi tercapai.

17. Dalam hal teknik updata segera, pembaruan ke database diterapkan segera saat terjadi ..... untuk mencapai titik COMMIT dan catatan semua perubahan disimpan di .....
18. Teknik shadow paging mempertahankan dua tabel halaman selama umur transaksi yaitu (a) ..... dan (b) .....
19. Di pos pemeriksaan, semua buffer ..... ke penyimpanan sekunder.
20. Penugasan dan pengelolaan blok memori disebut ..... dan komponen sistem operasi yang melakukan tugas ini disebut .....

KUNCI JAWABAN

1. Benar	6. Benar	11. Benar	16. Benar
2. Benar	7. Benar	12. Salah	17. Salah
3. Benar	8. Benar	13. Benar	18. Benar
4. Benar	9. Benar	14. Benar	19. Benar
5. Benar	10. Benar	15. Benar	20. Benar

KUNCI JAWABAN ISIAN

1. Pemulihan Database
2. Rollback
3. Pembatalan global
4. Pendekatan paksa, penulisan paksa
5. Pendekatan tanpa paksaan,
6. Pengguna tunggal ,
7. NO-VNDO/NO-REDO
8. Manajemen transaksi,
9. Jawaban :
    a. inkonsistensi data
    b. kehilangan data
10. Jawaban :
    a. kegagalan perangkat keras,
    b. kegagalan perangkat lunak,
    c. kegagalan media,
    d. kegagalan jaringan,
11. Status tidak konsisten, konsisten,
12. (a) bangunan yang berbeda, (b) terlindung dari bahaya,
13. Memori utama,
14. Jawaban :
    a. hilangnya memori utama termasuk buffer database
    b. hilangnya salinan disk (penyimpanan sekunder) dari database,
15. Head crash (rekaman tergores oleh jarum fonograf),
16. COMMIT point
17. Tanpa menunggu, log transaksi

18. (a) tabel halaman saat ini, (b) tabel halaman bayangan,
19. Ditulis paksa,
20. Manajemen buffer, manajer buffer.

SOAL BENAR/SALAH

1. Dalam sistem database terdistribusi, setiap situs biasanya dikelola oleh DBMS yang bergantung pada situs lain.
2. Sistem Database terdistribusi muncul dari kebutuhan untuk menawarkan otonomi Database lokal di lokasi yang terdistribusi secara geografis
3. Tujuan utama arsitektur Client/Server adalah untuk memanfaatkan kekuatan pemrosesan pada desktop sambil mempertahankan aspek terbaik dari pemrosesan data terpusat
4. Properti atomisitas transaksi terdistribusi memungkinkan pengguna untuk mengajukan pertanyaan tanpa menentukan di mana hubungan referensi, atau salinan atau fragmen dari hubungan tersebut berada
5. Properti independensi data mendistribusikan memungkinkan pengguna untuk menulis transaksi yang mengakses dan memperbarui data di beberapa situs seperti halnya mereka akan menulis transaksi melalui data lokal murni.
6. Meskipun secara geografis tersebar, sistem database terdistribusi mengelola dan mengontrol seluruh database sebagai satu kumpulan data.
7. Dalam DDBS homogen, ada beberapa situs, masing-masing menjalankan aplikasinya sendiri pada perangkat lunak DBMS yang sama.
8. Dalam DDBS heterogen, situs yang berbeda berjalan di bawah kendali DBMS yang berbeda, pada dasarnya secara otonom dan terhubung beberapa cara untuk mengaktifkan akses ke data dari beberapa situs.
9. Sistem database terdistribusi memungkinkan aplikasi untuk mengakses data dari database lokal dan remote.
10. Sistem database homogen memiliki standar yang diterima dengan baik untuk protokol gateway untuk mengekspos fungsionalitas DBMS ke aplikasi eksternal.
11. Database terdistribusi tidak menggunakan arsitektur client/server.
12. Dalam arsitektur Client/Server, klien adalah penyedia sumber daya sedangkan server adalah pengguna sumber daya.
13. Dalam arsitektur Client/Server tidak memungkinkan satu permintaan untuk menjangkau beberapa server.
14. Fragmentasi horizontal dihasilkan dengan menentukan predikat yang melakukan pembatasan pada tupel dalam relasi.
15. Replikasi data digunakan untuk meningkatkan kinerja database lokal dan melindungi ketersediaan aplikasi.
16. Transparansi dalam replikasi data membuat pengguna tidak menyadari adanya salinan.
17. Server adalah mesin yang menjalankan perangkat lunak DBMS dan menangani fungsi-fungsi yang diperlukan untuk akses data bersama secara bersamaan.

18. Replikasi data meningkatkan kinerja operasi read dengan meningkatkan kecepatan pemrosesan di lokasi.
19. Replikasi data mengurangi ketersediaan data untuk transaksi read-only.
20. Dalam penguncian terdistribusi, DDBS memelihara manajer kunci di setiap situs yang fungsinya untuk mengelola permintaan penguncian dan pembukaan kunci untuk item data yang disimpan di situs tersebut.
21. Dalam sistem terdistribusi, setiap situs menghasilkan stempel waktu lokal yang unik baik menggunakan penghitung logis atau jam lokal dan menggabungkannya dengan pengenal situs.
22. Dalam kontrol pemulihan, atomisitas transaksi harus dipastikan.
23. Protokol komit dua fase menjamin bahwa semua server Database yang berpartisipasi dalam transaksi terdistribusi baik semua komit atau semua dibatalkan.
24. Penggunaan 2PC tidak transparan bagi pengguna.

ISILAH BAGIAN TITIK KOSONG

1. Database terdistribusi adalah Database yang disimpan secara fisik pada beberapa sistem komputer di ..... terhubung bersama melalui .....
2. Sistem database terdistribusi muncul dari kebutuhan untuk menawarkan otonomi database lokal di ..... lokasi
3. ..... adalah arsitektur yang memungkinkan sumber daya komputasi terdistribusi pada jaringan untuk berbagi sumber daya bersama di antara kelompok pengguna workstation cerdas.
4. Dua properti yang diinginkan dari database terdistribusi adalah (a) ..... dan (d) .....
5. ..... adalah database yang disimpan secara fisik dalam dua atau lebih sistem komputer.
6. Sistem database terdistribusi heterogen juga disebut sebagai ..... atau .....
7. Arsitektur Client/Server adalah arsitektur di mana beban kerja terkait DBMS dibagi menjadi dua komponen logis yaitu (a) ..... dan (b)
8. Arsitektur Klient/server terdiri dari empat komponen utama yaitu (a) ..... (b) .... (c) .... dan (d) .....
9. Tiga keuntungan utama dari database terdistribusi adalah (a) ..... (b) ..... dan (C)
10. Tiga kelemahan utama dari Database terdistribusi adalah (a) ..... (b) ..... dan (C)
11. Arsitektur database middleware disebut juga .....
12. Middleware pada dasarnya adalah lapisan ..... yang berfungsi sebagai server khusus dan mengkoordinasikan eksekusi ..... dan ..... di satu atau lebih server database independen.
13. Fragmen horizontal dari suatu relasi adalah himpunan bagian dari ..... dengan semua ..... dalam relasi tersebut.

14. Pada fragmentasi horizontal, operasi ..... dilakukan untuk merekonstruksi relasi aslinya.
15. Replikasi data meningkatkan kinerja operasi read dengan meningkatkan ..... di lokasi.
16. Replikasi data telah meningkatkan biaya untuk ..... transaksi.
17. Dalam pemrosesan kueri terdistribusi, operasi simijoin digunakan untuk mengurangi ..... dari relasi yang perlu ditransmisikan dan karenanya ..... transaksi.
18. Dalam situasi Deadlock database terdistribusi, LWFG singkatan dari .....
19. Dalam situasi Deadlock database mendistribusikan, GWFG adalah singkatan dari .....
20. Dalam DDBS, setiap salinan item data berisi dua nilai stempel waktu yaitu (a) ..... dan B) .....
21. Protokol komit dua fase memiliki dua fase yaitu (a) ..... dan (b) .....
22. Protokol 3PC menghindari pembatasan ..... dari protokol komit dua fase.

KUNCI JAWABAN BENAR/SALAH

1. Benar	6. Benar	11. Salah	16. Benar	21. Benar
2. Benar	7. Benar	12. Salah	17. Benar	22. Benar
3. Benar	8. Benar	13. Benar	18. Benar	23. Benar
4. Salah	9. Benar	14. Benar	19. Salah	24. Salah
5. Salah	10. Benar	15. Benar	20. Benar	

KUNCI JAWABAN ISIAN

1. Beberapa situs, jaringan komunikasi,
2. Terdistribusi secara geografis,
3. Arsitektur server klien,
4. Jawaban
    a. Memberikan otonomi lokal
    b. Lokasi harus independen,
5. Sistem database terdistribusi (DDBS),
6. Multi- sistem Database, sistem Database federasi (FOBS),
7. Jawaban :
    a. Klien
    b. server
8. Jawaban :
    a. Klien, menginformasikan workstation cerdas sebagai titik kontak pengguna,
    b. server DBMS sebagai sumber daya umum yang melakukan tugas khusus tugas untuk perangkat yang meminta layanan mereka,
    c. Jaringan komunikasi yang menghubungkan klien dan server,
    d. Aplikasi perangkat lunak yang menghubungkan klien, server, dan jaringan untuk membuat arsitektur logis tunggal,

9. Jawaban :
    a. berbagi data
    b. meningkat efisiensi
    c. peningkatan otonomi daerah,
10. Jawaban :
    a. Pemulihan kegagalan lebih kompleks
    b. Peningkatan biaya pengembangan perangkat lunak
    c. Kurangnya standar,
11. Middleware akses data,
12. Perangkat lunak, query, Transaksi,
13. Tuple (atau baris), atribut
14. UNION
15. Kecepatan pemrosesan
16. Perbarui transaksi pada
17. Ukuran, komunikasi
18. Grafik menunggu lokal
19. Grafik menunggu global
20. Jawaban :
    a. stempel waktu read
    b. stempel waktu write,
21. Jawaban :
    a. fase pemungutan suara
    b. tahap keputusan,
22. Pemblokiran.

LAMPIRAN C

B-TECH

UJIAN SEMESTER KELIMA.

Sistem Manajemen Database

1. **Coba salah satu dari empat pertanyaan berikut: (3 X 4 = 12)**
    a. Definisikan istilah berikut:
        i. Sistem Database
        ii. Pengguna Akhir
        iii. DML
        iv. DDL
    b. Bedakan antara file sistem pemrosesan dan DBMS.
    c. Apa perbedaan antara independensi data logis dan independensi data fisik?
    d. Buatlah diagram E-R untuk sebuah rumah sakit dengan sekumpulan pasien dan sekumpulan dokter medis. Kaitkan dengan setiap pasien, log dari berbagai tes dan pemeriksaan yang dilakukan.
    OR
    Gambarlah diagram hubungan E-R yang menunjukkan kardinalitas sebagai berikut: Seorang operator dapat bekerja pada banyak mesin dan setiap mesin memiliki banyak operator. Setiap mesin milik satu departemen tetapi departemen dapat memiliki banyak mesin.
    e. Kapan konsep entitas yang lemah berguna dalam pemodelan data? Tentukan istilah: tipe entitas pemilik dan tipe entitas lemah.
    f. Apa perbedaan antara DML prosedural dan non-prosedural.
2. **Coba salah satu dari dua pertanyaan berikut: (7 X 2 = 14)**
    a. Bagaimana kalkulus relasional berbeda dari aljabar relasional dan bagaimana persamaannya"? Jelaskan dengan beberapa contoh yang sesuai.
    b. Perhatikan skema berikut :
    DEALER (DEALER_ID, DEALER_NAME, DEALER_ADDRESS)
    PARTS (PART_ID, PART_NAME, COLOUR)
    CATALOG (DEALER_ID, PART_ID, COST)
    Tulis query berikut dalam Aljabar Relasional dan SQL:
        i. Temukan nama Dealer yang memasok suku cadang merah.
        ii. Temukan nama Dealer yang memasok suku cadang kuning dan hijau
        iii. Temukan nama Dealer yang memasok semua suku cadang
    c. Diskusikan berbagai operasi pembaruan pada relasi dan jenis batasan integritas yang harus diperiksa setiap operasi pembaruan.
3. **Kerjakan salah satu dari dua pertanyaan berikut: (6 X 2 = 12)**
    a. Tulislah catatan berikut:
        i. Ketergantungan Fungsional atau 4 NF
        ii. Bentuk Normal

b. Perhatikan skema S = (V, w, X, Y, Z). Misalkan dependensi fungsional berikut ini berlaku:

$Z \rightarrow Y$
$W \rightarrow Y$
$XY \rightarrow Z$
$Y \rightarrow WX$

Nyatakan apakah dekomposisi skema S berikut adalah dekomposisi gabungan lossless. Justifikasikan jawaban Anda :

i. $S_1 = (V, w, X)$
   $S_2 = (V, Y, Z)$
ii. $S_1 = (V, W, Z)$
   $S_2 = (X, Y, Z)$

c. Apa apakah kamu mengerti dengan bentuk normal kelima? Jelaskan dengan beberapa contoh yang sesuai. 4. Coba salah satu dari dua pertanyaan berikut: (6 X 2 = 12)

i. Tulis berikut:
   1. Dead lock
   2. Protokol penguncian dua fase

ii. Apa yang Anda pahami dengan serializability jadwal? Jelaskan dengan beberapa contoh yang sesuai.

iii. Bagaimana sistem transaksinya? Bagaimana Anda membuat pemulihan dari kegagalan transaksi? Jelaskan dengan beberapa contoh yang sesuai.

UJIAN SEMESTER KETIGA, Sistem Manajemen Database

1. Coba Empat bagian: (5 X 4 = 20)

a. Definisikan istilah berikut:
   i. Abstraksi data
   ii. Independensi data
   iii. Skema Database
   iv. Redundansi data
   v. DDL & DML

b. Membahas aturan Data Base Administrator (DBA) dalam Sistem Manajemen Database.

c. Jelaskan tiga tingkat arsitektur DBMS secara rinci.

d. Definisikan istilah Generalisasi, Spesialisasi dan Agregasi dengan contoh yang sesuai.

e. Gambarlah diagram E-R dari proses pendaftaran mahasiswa pada suatu mata kuliah tertentu. Ubah juga diagram E-R menjadi tabel.

2. Coba bagian Tho apa saja: (IO X 2 = 20)

a. Perhatikan tiga skema relasi berikut S, P dan SP di mana S# adalah kode pemasok, P# kode produk dan Oty adalah Jumlah dan yang lainnya memiliki arti masing-

masing . S(S#, SNAME, SCITY, TURNOVER) P(P#, WEIGHT, COLOR, BIAYA, HARGA JUAL) SP(S#, P#, OTY) Tulislah pernyataan SQL dan aljabar relasional yang sesuai untuk pertanyaan berikut.

i. Dapatkan semua rincian pemasok yang beroperasi dari INDONESIA dengan TURNOVER = RO.
ii. Dapatkan nomor bagian. bobot antara 25 dan 35.
iii. Dapatkan nama pemasok yang namanya dimulai dengan A.
iv. Untuk setiap bagian yang dipasok, dapatkan no. dan nama semua kota yang memasok suku cadang tersebut.
v. Dapatkan nama-nama pemasok yang memasok part no. 2.

b. Sebuah universitas memiliki banyak jurusan. Setiap departemen mungkin memiliki banyak mahasiswa purna waktu dan paruh waktu. Setiap departemen dapat menawarkan beberapa kursus untuk siswanya sendiri. Setiap departemen memiliki anggota staf yang mungkin penuh waktu atau paruh waktu. Merancang generalisasi, hierarki spesialisasi untuk universitas.

c. Definisikan istilah berikut:
i. Batasan Integritas
ii. Foreign Key
iii. Primaru Key
iv. Super Key
v. Candidate Key

d. Perhatikan tabel berikut:

P

A	B	C
a	b	c
b	c	a
c	a	b

Q

A	B	C
c	b	a
b	a	c
b	c	a

R

J
J_1
J_2
J_3

Lakukan operasi aljabar relasional berikut.

(i) P ∪ Q
(ii) P ∩ Q
(iii) P - Q dan Q - P
(iv) P x R dan Q x R

e. Apa yang Anda maksud dengan Tampilan? Diskusikan kelebihan dan kekurangan View secara rinci.

3. Coba bagian 'IWo apa saja: (10 X 2 = 20)

a. Bergabagi anomali
i. Diskusikan berbagai anomali yang terkait dengan sistem manajemen Database relasional dengan memberikan contoh yang sesuai.
ii. Perhatikan skema relasional berikut:
R (A, B, C, D, E, F, G, H) dengan FD
AB →C, BC →C, E →F, G →F, H →A, FG →H.

Apakah penguraian R menjadi Rl (A, B, C, D), R2 (A, B, C, E, F), R3 (A, D, F, G, H) losslcss? Apakah ini mempertahankan ketergantungan?

b. Apa itu ketergantungan gabungan? Apa bedanya dengan ketergantungan Multivalued dan Fungsional? Berikan contoh masing-masing ketergantungan gabungan dan multinilai. Diskusikan Bentuk Normal Keempat (4 NF) juga secara rinci

c. Apa yang dimaksud dengan Ketergantungan Fungsional? Jelaskan BCNF dengan contoh yang sesuai. Sebuah dekomposisi dalam BCNF mungkin lossless dan ketergantungan melestarikan". Apakah pernyataan ini benar? Berikan jawaban Anda dengan contoh yang sesuai.

4. Coba 'IWo-parts : (10 X 2 = 20)

a. Apa itu deadlock? Kapan apakah itu terjadi? Bagaimana mendeteksinya dalam sistem Database? Bagaimana cara menghindarinya? Diskusikan secara rinci.

b. Apa yang Anda maksud dengan sistem Transaksi? Sebutkan properti ACID dari transaksi. Diskusikan juga pemulihan dari kegagalan transaksi.

c. Apa yang Anda maksud dengan Serializability? Diskusikan konflik dan lihat serializability dengan contoh yang sesuai. Diskusikan juga pengujian serializability.

5. Coba dua bagian: (10 X 2 = 20)

a. Apa itu skema multi-versi Concurrency Kontrol? Jelaskan dengan bantuan sebuah contoh. Diskusikan berbagai protokol Timestamp untuk kontrol konkurensi juga.

b. Apa yang Anda maksud dengan Multiple Granularity? Diskusikan dengan contoh yang sesuai. Diskusikan protokol berbasis validasi juga dengan contoh yang sesuai.

c. Apa yang dimaksud dengan penguncian dua fase? th bantuan contoh. Apakah penguncian dua fase akan menghasilkan Deadlock? Buktikan jawaban Anda dengan bantuan sebuah contoh. Diskusikan Pemulihan dengan transaksi serentak juga.

UJIAN SEMESTER KETIGA, Sistem Manajemen Database

1. Coba salah satu dari empat bagian berikut ini: (5 X 4 = 20)

a. Bagaimana mungkin untuk mendapatkan lebih banyak informasi dari jumlah data yang sama dengan menggunakan pendekatan database sebagai lawan dari pendekatan file?

b. Tentukan redundansi. Bisakah redundansi data sepenuhnya dihilangkan ketika pendekatan Database digunakan?

c. Gambarlah diagram E.R. untuk department store, setelah menentukan entitas-entitas inrerest dan hubungan yang ada di antara entitas-entitas tersebut. Juga membangun representasi tabular dari entitas dan hubungan. Apakah ada atribut di setiap himpunan entitas yang secara unik mengidentifikasi dan mencontoh himpunan entitas?

d. Jelaskan perbedaan antara kendala total dan sebagian dengan contoh yang sesuai.
e. Bagaimana representasi asosiasi dan hubungan dalam model jaringan dan hierarki dapat berbeda?
f. Definisikan konsep agregasi dengan setidaknya dua contoh di mana konsep ini berguna.

2. Coba salah satu dari empat bagian berikut ini: (5 x·4 = 20)

a. Dalam hal apa kalkulus relasional berbeda dari aljabar relasional dan dalam arti apa keduanya serupa?
b. Apakah pandangan itu? Sebutkan dua alasan mengapa kita dapat memilih untuk mendefinisikan tampilan.
c. Misalkan R = (A, B, C) dan r1 r_2 keduanya merupakan relasi pada skema R. Ekspresi dalam kalkulus relasional domain untuk:
   i. π A (r_l)
   ii. $\pi_{A, B}$ (r_l)[><] π_B C(r^2)
   iii. r_l - r_2
d. Pertimbangkan database asuransi yang diberikan, di mana kunci utama digarisbawahi membangun kueri SQL yang diberikan untuk database relasional, orang
   (id driver #, nama, alamat)
   mobil (lisensi, model tahun)
   kecelakaan (nomor-laporan, data, lokasi)
   milik (id-pengemudi #, lisensi)
   berpartisipasi (nomor-pengemudi #, nomor laporan, jumlah kerusakan)
   i. Tambahkan kecelakaan baru ke database; asumsikan nilai apa pun untuk atribut yang diperlukan.
   ii. Perbarui jumlah kerusakan untuk mobil dengan nomor lisensi 'AABB2000" dalam kecelakaan dengan nomor laporan "AR 2197" menjadi Rp..0 3.00000.
e. Untuk relasi P dan 0 seperti yang diberikan. Lakukan operasi berikut dan tunjukkan relasi yang dihasilkan:

P

A	B	C	D
a_1	b_2	c_2	d_2
a_2	b_1	c_1	d_2
a_1	b_1	c_2	d_1
a_2	b_1	c_2	d_2
a_1	b_2	c_1	d_2
a_3	b_1	c_2	d_1
a_1	b_2	c_2	d_2
a_2	b_1	c_1	d_2
a_1	b_3	c_2	d_2

Q

B	C	D
b_1	c_1	d_2
b_3	c_1	d_2
b_2	c_2	d_1
b_1	c_1	d_2
b_3	c_2	d_2

(i) Temukan proyeksi 0 pada atribut (B, C).

(ii) Bagi P dengan relasi yang diperoleh dengan terlebih dahulu memilih tupel 0 di mana nilai B adalah bl atau bz dan kemudian proyeksikan 0 pada atribut (C, D).

f. Apakah pandangan itu? Jelaskan keuntungan kursor di SQL?

g. (A →BCDE, B → ACDE, C →ABDE), Berikan dekomposisi lossless dari R

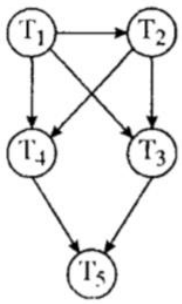
h. Jelaskan mengapa 4 NF lebih diinginkan daripada BCNE

i. Dengan menggunakan pengetahuan lingkungan perguruan tinggi, tentukan dependensi fungsional yang ada pada tabel berikut. Setelah ini ditentukan, ubah tabel ini menjadi kumpulan yang setara dengan tabel yang ada di 3 NF
Siswa [(Nomor Siswa, Nama Siswa, Nomor SKS, Nomor Penasehat, Nama Penasehat, Nomor Jurusan, Nama Jurusan), (Nomor Kursus, Deskripsi mata kuliah, istilah mata kuliah, nilai)J.

j. Jelaskan protokol komit dua fase. Bagaimana itu dilakukan pertunjukan dengan contoh?

k. Bedakan mekanisme check point dengan fasilitas logging.

l. Pertimbangkan grafik prioritas yang diberikan pada gambar dan periksa jenis serial yang mana. Jelaskan jawabanmu.



3. Coba salah satu dari dua bagian berikut ini: (10 X 2 = 20)

a. Dalam pemesanan stempel waktu, stempel waktu-w (Q) menunjukkan stempel waktu terbesar yang berhasil mengeksekusi penulisan (0). Jika kita mendefinisikannya sebagai stempel waktu transaksi terbaru untuk mengeksekusi penulisan (0) dengan sukses. Apakah ada perbedaan? Justifikasi jawaban Anda.

b. Diskusikan keuntungan dan kerugian dari stempel waktu terpusat dan stempel waktu terdistribusi.

c. Jelaskan pengertian transparansi dan otonomi? Jelaskan juga replikasi multimaster.

UJIAN SEMESTER KETIGA.

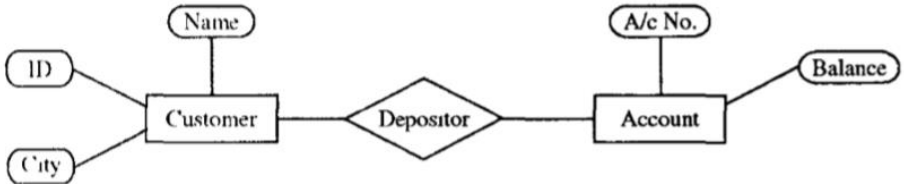
1. Sistem Manajemen Database 1. Coba salah satu dari empat bagian berikut ini: (5 X 4 = 20)

a. Jelaskan perbedaan antara sistem berorientasi file dan sistem berorientasi Database.

b. Buatlah model E-R untuk perusahaan asuransi mobil yang pelanggannya masing-masing memiliki satu atau lebih mobil. Setiap mobil telah mengaitkannya dengan nol hingga tidak. dari insiden yang direkam.
c. Daftar semua pengguna database. Jelaskan pengguna yang canggih dan khusus.
d. Apa yang dimaksud dengan tipe relasi rekursif? Jelaskan dengan sebuah contoh.
e. Jelaskan berbagai jenis model data? Bagaimana ini berbeda satu sama lain; jelaskan secara singkat.
f. Himpunan entitas yang lemah selalu dapat dibuat menjadi himpunan entitas yang kuat dengan menambahkan atribut-atributnya ke himpunan entitas pengidentifikasinya. Garis besar redundansi seperti apa yang akan terjadi jika kita melakukannya?

2. Coba salah satu dari empat bagian berikut ini: (5 X 4 = 20)

a. Gambarlah arsitektur dasar sistem DBMS (oracle Si).
b. Karyawan (emp_id, emp_name, emp_street, emp_city) bekerja (emp_id, company_id, gaji) terletak di (kode_perusahaan, nama-perusahaan, kota_perusahaan) Tulis pertanyaan dalam aljabar relasional:
    i. Temukan nama semua karyawan yang bekerja untuk sebuah perusahaan yang berlokasi di "Indonesia".
    ii. Temukan nama-nama karyawan yang bekerja untuk perusahaan yang berlokasi di kota tempat tinggal mereka.
    iii. Cari nama karyawan yang bekerja di perusahaan "TCS".
c. Apa yang dimaksud dengan batasan kunci asing? Mengapa batasan seperti itu penting?
d. Merancang Database relasional yang sesuai dengan diagram E-R
e. Perintah mana yang merupakan bagian DOL dari SQL? Tulis sintaks mereka. (I) Jelaskan dengan contoh operasi berikut: join, union, minus, update, insert.



3. Coba salah satu dari empat bagian berikut ini: (5 X 4 = 20)

a. Apa tujuan desain dari database relasional yang baik?
b. Buktikan dengan contoh yang sesuai bahwa BCNF lebih kuat dari 3NF.
c. Perhatikan skema R = (A, B, C, D, E). Misalkan hold FD berikut:
   F = {E → A, CD →E, A →BC, B →D}
   Nyatakan apakah mengikuti dekomposisi Rare lossless join decomposition atau tidak. Jelaskan jawaban anda :
   (1) {(A, B, C), (A, D, E)}
   (2) {(A, B, C), (C, D, E)}
d. Apa yang dimaksud dengan aksioma Armstrong untuk menemukan FD?

e. Pertimbangkan dua set dependensi fungsional:
   F_I = {A→C,AC→D,E→AD,E→H}
   F_2 = {A→CD, E → AH}
f. periksa apakah mereka setara. (f) Buktikan bahwa jika dalam skema relasi, no. atribut dalam a/primary key adalah satu, skema akan setidaknya dalam 2NF.

4. Coba salah satu dari dua hal berikut ini: (10 X 2 = 20)

a. Apa yang Anda maksud dengan jadwal? Kapan jadwal disebut serializable? Apa itu jadwal bersambung konflik? Tunjukkan apakah jadwal berikut ini setara konflik atau tidak. Justifikasi pernyataan Anda.

Schedule 1

T_1	T_2
Read (A)	
Write (A)	
	Read (A)
	Write (A)

Schedule 2

T_1	T_2
	Read (A)
Read (A)	
	Write (A)
Write (A)	

b. Jelaskan perbedaan berikut ini: Fragmentasi, Transparansi replikasi dan Transparansi lokasi.
c. Jelaskan alasan mengapa pemulihan transaksi interaktif lebih sulit daripada pemulihan transaksi batch.

5. Coba salah satu dari dua bagian berikut ini: (10 X 2 = 20)

a. Apa teknik penguncian yang berbeda untuk kontrol konkurensi?
b. Apa itu stempel waktu? Daftar semua protokol berbasis Timestamp, periksa apakah itu cascadeless dan apakah itu dapat dipulihkan.
c. Tulislah catatan singkat sebagai berikut :
   i. Estimasi biaya dan optimalisasi transfer tuple untuk bergabung dalam database terdistribusi.
   ii. Beberapa granularity dan skema multiversi.

LAMPIRAN D

Pertanyaan dan Jawaban Wawancara DBMS

Pertanyaan 1: Apa itu Database?

Jawaban: Database adalah kumpulan data yang koheren secara logis dengan beberapa makna yang melekat, mewakili beberapa aspek dunia nyata dan yang dirancang, dibangun, dan diisi dengan data untuk tujuan tertentu.

Pertanyaan 2: Apa itu DBMS?

Jawaban: Ini adalah kumpulan program yang memungkinkan pengguna untuk membuat dan memelihara database. Dengan kata lain itu adalah perangkat lunak tujuan umum yang menyediakan pengguna dengan proses mendefinisikan, membangun dan memanipulasi database untuk berbagai aplikasi.

Pertanyaan 3: Apa yang dimaksud dengan sistem Database?

Jawaban : Database dan perangkat lunak DBMS bersama-sama disebut sebagai sistem Database.

Pertanyaan 4: Kelebihan DBMS?

Jawaban:

(i) Redundansi dikendalikan.
(ii) Akses tidak sah dibatasi.
(iii) Menyediakan banyak antarmuka pengguna.
(iv) Menegakkan batasan integritas.
(v) Menyediakan backup dan recovery.

Pertanyaan 5: Kekurangan Sistem Pemrosesan File?

Jawaban:

(i) Redundansi dan inkonsistensi data.
(ii) Sulitnya mengakses data
(iii) Isolasi data.
(iv) Integritas data.
(v) Akses serentak tidak dimungkinkan.
(vi) Masalah keamanan.

Pertanyaan 6: Jelaskan tiga tingkat abstraksi data?

Jawaban: Ada tiga tingkat abstraksi:

(i) Tingkat fisik: Tingkat abstraksi terendah menjelaskan bagaimana data disimpan.
(ii) Level logis : Level abstraksi berikutnya yang lebih tinggi, menggambarkan data apa yang disimpan dalam database dan hubungan apa di antara data tersebut.
(iii) View level : Level abstraksi tertinggi hanya menggambarkan sebagian dari keseluruhan database.

Pertanyaan 7: Tentukan "aturan integritas"

Jawaban: Ada dua aturan Integritas.

(i) Integritas Entitas: Menyatakan bahwa "Kunci Utama tidak dapat memiliki nilai NULL"

(ii) Integritas Referensial: Menyatakan bahwa "Kunci Asing dapat berupa nilai NULL atau harus menjadi nilai Kunci Utama dari relasi lain.

Pertanyaan 8: Apakah ekstensi dan intensi itu?

Jawaban: Ekstensi - Ini adalah jumlah tupel yang ada dalam tabel pada setiap contoh. Ini tergantung waktu. Intensi - Ini adalah nilai konstan yang memberikan nama, struktur tabel, dan batasan yang diletakkan di atasnya.

Pertanyaan 9: Apa itu Sistem R? Apa dua subsistem utama?

Jawaban : System R dirancang dan dikembangkan selama periode 1974 -79 di IBM San Jose Research Center. Ini adalah prototipe dan tujuannya adalah untuk menunjukkan bahwa adalah mungkin untuk membangun Sistem Relasional yang dapat digunakan dalam lingkungan kehidupan nyata untuk memecahkan masalah kehidupan nyata, dengan kinerja setidaknya sebanding dengan sistem yang ada. Dua subsistemnya adalah

(i) Research Storage
(ii) System Relational Data System.

Pertanyaan 10: Bagaimana struktur data Sistem R berbeda dari struktur relasional?

Jawaban: Tidak seperti sistem Relasional di sistem R

(i) Domain tidak didukung
(ii) Penegakan keunikan kunci kandidat bersifat opsional
(iii) Penegakan integritas entitas bersifat opsional
(iv) Integritas referensial tidak diterapkan

Pertanyaan 11: Apa itu Independensi Data?

Jawaban: Independensi data berarti bahwa "aplikasi tidak tergantung pada struktur penyimpanan dan strategi akses data". Dengan kata lain, kemampuan untuk memodifikasi definisi skema di satu tingkat tidak boleh mempengaruhi definisi skema di tingkat yang lebih tinggi berikutnya. Dua tipe Independensi Data:

(i) **Independensi Data Fisik** : Modifikasi pada level fisik seharusnya tidak mempengaruhi level Logika.
(ii) **Independensi Data Logis:** Modifikasi pada level logika seharusnya tidak mempengaruhi level tampilan. Catatan: Independensi Data Logis lebih sulit dicapai

Pertanyaan 12: Apa itu view? Bagaimana kaitannya dengan independensi data?

Jawaban: Tampilan dapat dianggap sebagai tabel virtual, yaitu tabel yang tidak benar-benar ada dengan sendirinya tetapi diturunkan dari satu atau lebih tabel dasar yang mendasarinya. Dengan kata lain, tidak ada file tersimpan yang secara langsung mewakili tampilan, sebaliknya definisi tampilan disimpan dalam kamus data. Pertumbuhan dan restrukturisasi tabel dasar tidak tercermin dalam tampilan. Dengan demikian, tampilan dapat melindungi pengguna dari efek restrukturisasi dan pertumbuhan dalam database. Oleh karena itu memperhitungkan independensi data logis.

Pertanyaan 13: Apa itu Model Data?

Jawaban: Kumpulan alat konseptual untuk mendeskripsikan data, hubungan data semantik data dan atribut.

Pertanyaan 14: Apakah model E-R itu?

Jawaban: Model data ini didasarkan pada dunia nyata yang terdiri dari objek-objek dasar yang disebut entitas dan hubungan antara objek-objek tersebut. Entitas dijelaskan dalam database dengan satu set atribut.

Pertanyaan 15: Apa itu model Berorientasi Objek?

Jawaban: Model ini didasarkan pada kumpulan objek. Sebuah objek berisi nilai-nilai yang disimpan dalam variabel instan dengan di dalam objek. Sebuah objek juga berisi badan kode yang beroperasi pada objek tersebut. Badan kode ini disebut metode. Objek yang berisi jenis nilai dan metode yang sama dikelompokkan bersama ke dalam kelas.

Pertanyaan 16: Apa itu Entitas?

Jawaban: Itu adalah 'benda' di dunia nyata dengan keberadaan yang independen.

Pertanyaan 17: Apa yang dimaksud dengan tipe Entitas

Jawaban: Ini adalah kumpulan (set) entitas yang memiliki atribut yang sama.

Pertanyaan 18: Apa yang dimaksud dengan kumpulan tipe Entitas?

Jawaban: Ini adalah kumpulan dari semua entitas tipe entitas tertentu dalam database.

Pertanyaan 19: Apa yang dimaksud dengan Perpanjangan tipe entitas?

Jawaban: Kumpulan entitas dari tipe entitas tertentu dikelompokkan bersama menjadi satu set entitas.

Pertanyaan 20: Apa yang dimaksud dengan himpunan Entitas Lemah?

Jawaban: Himpunan entitas mungkin tidak memiliki atribut yang cukup untuk membentuk kunci utama, dan kunci primernya berkompromi dengan kunci parsial dan kunci primer entitas induknya, maka disebut himpunan Entitas Lemah.

Pertanyaan 21: Apa yang dimaksud dengan atribut?

Jawaban: Ini adalah properti tertentu, yang menggambarkan entitas.

Pertanyaan 22: Apa itu Skema Relasi dan Relasi?

Jawaban: Skema relasi yang dilambangkan dengan R (A_I, A_2, ... , A_n) terdiri dari nama relasi Rand daftar atribut A 1 yang dikandungnya. Sebuah relasi didefinisikan sebagai sekumpulan tupel. Misalkan r adalah relasi yang berisi kumpulan tupel (t_1, t_2, t_3, ... t_n). Setiap tupel adalah daftar terurut dari nilai-n t = (v_I, v_2, ...v_n).

Pertanyaan 23: Apa yang dimaksud dengan derajat suatu Relasi ?

Jawaban: Ini adalah jumlah atribut dari skema relasinya.

Pertanyaan 24: Apa itu Hubungan?

Jawaban. Ini adalah asosiasi antara dua atau lebih entitas.

Pertanyaan 25: Apa itu Himpunan Hubungan?

Jawaban : Koleksi (atau set) hubungan serupa.

Pertanyaan Pertanyaan 26: Apa itu tipe Relasi?

Jawaban: Jenis hubungan mendefinisikan satu set asosiasi atau satu set hubungan antara satu set tertentu jenis entitas.

Pertanyaan 27: Apa itu derajat tipe Hubungan?

Jawaban: Ini adalah jumlah tipe entitas yang berpartisipasi.

• Q.28. Apa itu DDL (Bahasa Definisi Data)? Jwb. Skema Database ditentukan oleh serangkaian definisi yang diungkapkan oleh bahasa khusus yang disebut DDL.

Pertanyaan 29 : Apa itu VDL (View Definition Language)?

Jawaban : Ini menentukan pandangan pengguna dan pemetaan mereka ke skema konseptual

Pertanyaan 30: Apa itu Bahasa Definisi Penyimpanan Data?

Jawaban : Struktur penyimpanan dan metode akses yang digunakan oleh sistem database ditentukan oleh sekumpulan definisi dalam tipe khusus DDL yang disebut bahasa definisi penyimpanan data.

Pertanyaan 31: Apa itu DML (Bahasa Manipulasi Data)?

Jawaban: Bahasa ini memungkinkan pengguna untuk mengakses atau memanipulasi data sebagaimana diatur oleh model data yang sesuai. DML Prosedural atau Tingkat Rendah: DML mengharuskan pengguna untuk menentukan data apa yang dibutuhkan dan cara mendapatkan data tersebut. DML Non-Prosedural atau Tingkat Tinggi: DML mengharuskan pengguna untuk menentukan data apa yang dibutuhkan tanpa menentukan cara mendapatkan data tersebut.

Pertanyaan 32. Apa itu VDL (View Definition Language)?

Jawaban: Ini menentukan pandangan pengguna dan pemetaan mereka ke skema konseptual.

Pertanyaan 33. Apa itu Kompilator DML?

Jawaban: Ini menerjemahkan pernyataan DML dalam bahasa kueri ke dalam instruksi tingkat rendah yang dapat dipahami oleh mesin evaluasi kueri.

Pertanyaan 34. Apa itu mesin evaluasi Query?

Jawaban: Ini mengeksekusi instruksi tingkat rendah yang dihasilkan oleh kompiler.

Pertanyaan 35: Apa itu Interpreter DDL?

Jawaban: Ini menafsirkan pernyataan DDL dan merekamnya dalam tabel yang berisi metadata.

Pertanyaan 36: Apa itu Record-at-a-time-?

Jawaban: DML tingkat rendah atau Prosedural dapat menentukan dan mengambil setiap record dari sekumpulan record. Pengambilan record ini dikatakan sebagai Record-at-a-time.

Pertanyaan 37: Apa itu Set-at-a-time atau Set-oriented?

Jwb. DML tingkat tinggi atau non-prosedural dapat menentukan dan mengambil banyak catatan dalam satu pernyataan DML. Pengambilan record ini dikatakan set-at-a-time atau Set-oriented.

Pertanyaan 38. Apa itu Aljabar Relasional?

Jwb. Ini adalah bahasa query prosedural. Ini terdiri dari satu set operasi yang mengambil satu atau dua relasi sebagai input dan menghasilkan relasi baru.

Pertanyaan 39. Apa itu Kalkulus Relasional?

Jwb. Ini adalah kalkulus predikat terapan yang dirancang khusus untuk Database relasional yang diusulkan oleh E.F. Codd, misalnya, bahasa yang didasarkan padanya adalah DSL ALPHA, QUEL.

Pertanyaan 40: Bagaimana kalkulus relasional berorientasi Tuple berbeda dari kalkulus relasional berorientasi domain?

Jwb. Kalkulus berorientasi tupel menggunakan variabel tupel, yaitu variabel yang nilainya hanya diizinkan adalah tupel dari relasi itu, misalnya QUEL. Kalkulus berorientasi domain

memiliki variabel domain, yaitu, variabel yang berkisar di atas domain yang mendasarinya, bukan di atas hubungan, misalnya, ILL, DEDUCE.

Pertanyaan 41. Apakah normalisasi itu?

Jwb. Ini adalah proses menganalisis skema relasi yang diberikan berdasarkan Dependensi Fungsional (FD) dan kunci utama untuk mencapai properti \

(i) Meminimalkan redundansi

(ii) Meminimalkan penyisipan, penghapusan dan pembaruan anomali.

Pertanyaan 42: Apa itu Ketergantungan Fungsional?

Jwb. Sebuah ketergantungan fungsional dilambangkan dengan X →Y antara dua set atribut X dan Y yang merupakan himpunan bagian dari R menentukan kendala pada tupel yang mungkin yang dapat membentuk keadaan relasi r dari R. Batasan adalah untuk dua tupel t1 dan t2 di r jika (1[X] = (2[X] maka mereka memiliki t 1[Y] = (2[Y]. Ini berarti nilai komponen X dari sebuah tupel secara unik menentukan nilai komponen Y.

Pertanyaa 43. Kapan ketergantungan fungsional F dikatakan minimal?

Jwb.

(i) Setiap ketergantungan dalam F memiliki satu atribut untuk ruas kanannya.

(ii) Kita tidak dapat mengganti ketergantungan X →A di F dengan ketergantungan Y →A di mana Y adalah subset yang tepat dari x dan masih memiliki himpunan ketergantungan yang setara dengan F.

(iii) Kita tidak dapat menghapus ketergantungan apapun dari F dan masih memiliki set dependensi yang setara dengan F.

Pertanyaan 44. Apakah ketergantungan multinilai itu?

Jwb. Ketergantungan multinilai dilambangkan dengan X →Y ditentukan pada skema relasi R, di mana X dan Yare kedua himpunan bagian dari R, menentukan kendala berikut pada setiap relasi r dari R: jika dua tupel t1 dan t2 ada di r sedemikian rupa sehingga (1[X] == (2[X] maka t3 dan t4 juga harus ada di r dengan properti berikut:

di mana

t 3 [X] = t4 [X] = t 1[X] = t 2 [X]

t3 [Y] = t 1 [Y] dan t4[Y] = t 2 [Y]

t3 [Z] = t2 [Z] dan t4 [Z] = t 1 [Z]

dimana [Z = (R - (XUY))]

Pertanyaan 45. Apa itu Lossless join property?

Jwb. Ini menjamin bahwa generasi tupel palsu tidak terjadi sehubungan dengan hubungan 5chemas setelah dekomposisi.

Pertanyaan 46. Apa itu 1 NF (Bentuk Normal)?

Jwb. Domain atribut harus menyertakan hanya nilai atom (sederhana, tidak dapat dibagi).

Pertanyaan 47. Apa itu Ketergantungan Fungsional Sepenuhnya?

Jawaban: Hal ini didasarkan pada konsep ketergantungan fungsional penuh. Sebuah ketergantungan fungsional X →Y adalah ketergantungan fungsional penuh jika penghapusan atribut A dari X berarti bahwa ketergantungan tidak berlaku lagi, yaitu {X-A}→1- Y.

Pertanyaan 48. Apa itu 2NF?

Jwb. Skema relasi R berada dalam 2NF jika berada dalam INF dan setiap atribut non-prima A dalam R secara fungsional bergantung pada kunci primer.

Pertanyaan 49. Apa itu 3NF?

Jwb. Skema relasi R berada dalam 3NF jika berada dalam 2NF dan untuk setiap FD X -+ A salah satu dari berikut ini benar

(i) X adalah Super-key dari R. ATAU

(ii) A adalah atribut prima dari R. Dengan kata lain, jika setiap atribut non prima tidak bergantung secara transitif pada kunci primer.

Pertanyaan 50: Apa itu BCNF (Bentuk Normal Boyce-Codd)?

Jwb. Skema relasi R berada dalam BCNF jika berada dalam 3NF dan memenuhi batasan tambahan bahwa untuk setiap FD X →A, X harus menjadi kunci kandidat.

Pertanyaan 51. Apa itu 4NF?

Jwb. Suatu skema relasi R dikatakan dalam 4NF jika untuk setiap ketergantungan multinilai X →Y yang menahan R, salah satu dari berikut ini benar (i) X adalah himpunan bagian atau sama dengan (atau) X →Y = R. (ii) X adalah kunci super.

Pertanyaan 52. Apa itu 5NF?

Jwb. Skema relasi R dikatakan 5NF jika untuk setiap ketergantungan gabungan {R 1, R 2 , ... , Rn} yang menampung R, salah satu berikut ini benar

(i) R i = R untuk beberapa i.

(ii) Ketergantungan gabungan diimplikasikan oleh himpunan PD, di atas R di mana ruas kiri adalah kunci dari R.

Catatan: ONF: Bentuk Normal Optimal Sebuah model terbatas hanya pada fakta-fakta sederhana (elemental), seperti yang diekspresikan dalam Object Role Model notasi.

DKNF : Domain-Key Normal Form Model yang bebas dari semua anomali modifikasi. Ingat, pedoman normalisasi ini bersifat kumulatif. Untuk database dalam 3NF, pertama-tama harus memenuhi semua kriteria database 2NF dan 1NF.

Pertanyaan 53. Apa itu RDBMS?

Jwb. Sistem Manajemen Database Relasional (RDBMS) adalah sistem manajemen Database yang memelihara catatan data dan indeks dalam tabel. Hubungan dapat dibuat dan dipelihara di seluruh dan di antara data dan tabel. Dalam database relasional, hubungan antar item data diekspresikan melalui tabel. Saling ketergantungan di antara tabel-tabel ini diekspresikan oleh nilai data dan bukan oleh pointer. Hal ini memungkinkan tingkat independensi data yang tinggi. Sebuah RDBMS memiliki kemampuan untuk menggabungkan kembali item data dari file yang berbeda, menyediakan alat yang kuat untuk penggunaan data.

Pertanyaan 54. Apa itu Stored Procedure?

Jwb. Prosedur tersimpan adalah sekelompok pernyataan SQL bernama yang sebelumnya telah dibuat dan disimpan di database server. Prosedur tersimpan menerima parameter input sehingga satu prosedur dapat digunakan melalui jaringan oleh beberapa klien menggunakan data input yang berbeda. Dan ketika prosedur diubah, semua klien secara otomatis mendapatkan versi baru. Prosedur tersimpan mengurangi lalu lintas jaringan dan

meningkatkan kinerja. Prosedur yang disimpan dapat digunakan untuk membantu memastikan integritas database. misalnya sp _ helpdb. sp _ berganti namab, sp _depends dll.

Pertanyaan 55. Apa itu kursor?

Jawaban : Kursor adalah objek database yang digunakan oleh aplikasi untuk memanipulasi data dalam kumpulan berdasarkan baris demi baris, alih-alih perintah SQL tipikal yang beroperasi pada semua baris dalam kumpulan pada satu waktu. Untuk bekerja dengan kursor, kita perlu melakukan beberapa langkah dalam urutan berikut:

- Hapus kursor
- Buka kursor
- Ambil baris dari kursor
- Proses baris yang diambil
- Tutup kursor
- Hapus alokasi kursor

Pertanyaan 56. Apa yang Dipicu?

Jwb. Pemicu adalah prosedur SQL yang memulai tindakan saat peristiwa (INSERT, DELETE, atau UPDATE) terjadi. Pemicu disimpan dan dikelola oleh DBMS. Pemicu digunakan untuk menjaga integritas referensial data dengan mengubah data secara sistematis. Pemicu tidak dapat dipanggil atau dieksekusi; DBMS secara otomatis mencari pemicu sebagai akibat dari modifikasi data ke tabel terkait. Pemicu dapat dilihat mirip dengan prosedur tersimpan karena keduanya terdiri dari logika prosedural yang disimpan di tingkat Database. Prosedur tersimpan, bagaimanapun, bukan event-drive dan tidak dilampirkan ke tabel tertentu seperti pemicunya. Prosedur tersimpan dieksekusi secara eksplisit dengan memanggil CALL ke prosedur sementara pemicu dieksekusi secara implisit. Selain itu, pemicu juga dapat mengeksekusi prosedur tersimpan. **Nasted Trigger**: Pemicu juga dapat berisi logika INSERT, UPDATE, dan DELETE di dalam dirinya sendiri, jadi ketika pemicu diambil karena modifikasi data, pemicu juga dapat menyebabkan modifikasi data lain, sehingga memicu pemicu lain. Pemicu yang berisi logika modifikasi data di dalamnya disebut pemicu bersarang.

Pertanyaan 57. Apa itu View?

Jwb. Tampilan sederhana dapat dianggap sebagai bagian dari tabel. Ini dapat digunakan untuk mengambil data, serta memperbarui atau menghapus baris. Baris yang diperbarui atau dihapus dalam tampilan diperbarui atau dihapus dalam tabel tempat tampilan dibuat. Perlu juga dicatat bahwa ketika data dalam tabel asli berubah, demikian juga data dalam tampilan, karena tampilan adalah cara untuk melihat bagian dari tabel asli. Hasil penggunaan tampilan tidak disimpan secara permanen dalam database. Data yang diakses melalui tampilan sebenarnya dibangun menggunakan perintah pilih T-SQL standar dan dapat berasal dari satu ke banyak tabel dasar yang berbeda atau bahkan tampilan lainnya.

Pertanyaan 58. Apa itu Indeks

Jwb. Indeks adalah struktur fisik yang berisi pointer ke data Indeks dibuat dalam tabel yang ada untuk menemukan baris lebih cepat dan efisien. Dimungkinkan untuk membuat indeks pada satu atau lebih kolom tabel, dan setiap indeks diberi nama. Pengguna tidak dapat melihat indeks, mereka hanya digunakan untuk mempercepat kueri. Indeks yang efektif

adalah salah satu cara terbaik untuk meningkatkan kinerja dalam aplikasi database. Pemindaian tabel terjadi ketika tidak ada indeks yang tersedia untuk membantu kueri. Dalam pemindaian tabel, SQL Server memeriksa setiap baris dalam tabel untuk memenuhi hasil kueri. Pemindaian tabel terkadang tidak dapat dihindari, tetapi pada tabel besar, pemindaian memiliki dampak yang luar biasa pada kinerja. Indeks berkerumun menentukan penyortiran fisik baris tabel database di media penyimpanan. Untuk alasan ini, setiap tabel database mungkin hanya memiliki satu indeks berkerumun. Indeks non-cluster dibuat di luar tabel database dan berisi daftar referensi yang diurutkan ke tabel itu sendiri.

Pertanyaan 59. Jelaskan properti persisten untuk database.

Jwb. Presisten berarti bahwa data berada pada penyimpanan yang stabil seperti disk magnetik. Misalnya: Organisasi perlu menyimpan data tentang pelanggan, pemasok, dan inventaris pada penyimpanan yang stabil karena data ini digunakan berulang kali. Variabel dalam program komputer tidak presisten karena berada di memori utama dan menghilang setelah program dihentikan. Kegigihan tidak berarti bahwa data bertahan selamanya. Ketika data tidak lagi relevan (seperti pemasok yang gulung tikar), data tersebut dihapus atau disimpan.

Pertanyaan 60: Jelaskan properti yang saling terkait untuk database.

Jwb. Saling terkait berarti bahwa data yang disimpan sebagai unit terpisah dapat dihubungkan untuk memberikan gambaran yang utuh. Misalnya, database pelanggan menghubungkan data pelanggan (nama, alamat, ...) dengan data pesanan (nomor pesanan, data pesanan, ...) untuk memfasilitasi pemrosesan pesanan. Database berisi entitas dan hubungan antar entitas. Entitas adalah sekelompok data biasanya tentang satu topik yang dapat diakses bersama-sama. Entitas dapat menunjukkan orang, tempat, benda, atau peristiwa.

Pertanyaan 61. Jelaskan properti bersama untuk database.

Jwb. Dibagikan berarti bahwa database dapat memiliki banyak kegunaan dan pengguna. Database menyediakan memori umum untuk beberapa fungsi dalam suatu organisasi. Misalnya, database kepegawaian dapat mendukung perhitungan penggajian, evaluasi kinerja, persyaratan pelaporan pemerintah, dan sebagainya. Banyak pengguna dapat menggunakan database secara bersamaan. Misalnya, banyak pelanggan dapat secara bersamaan melakukan reservasi maskapai. Kecuali dua pengguna mencoba mengubah bagian yang sama dari database pada saat yang sama, mereka dapat melanjutkan tanpa menunggu.

Pertanyaan 62. Apa hubungan antara akses nonprosedural dan pengembangan aplikasi (formulir atau laporan)? Bisakah akses nonprocedural digunakan dalam pengembangan aplikasi?

Jwb. Hubungan antara akses nonprosedural dan pengembangan aplikasi (form atau laporan) adalah bahwa akses non-prosedural digunakan dalam pengembangan aplikasi untuk menunjukkan kebutuhan data. Akses non-prosedural memungkinkan pembuatan formulir dan laporan tanpa pengkodean yang ekstensif. Sebagai bagian dari pembuatan formulir atau laporan, pengguna menunjukkan persyaratan data menggunakan bahasa non-prosedural (SQL) atau alat grafis.

Pertanyaan 63. Apa perbedaan antara formulir dan laporan?

Jwb. Formulir entri data menyediakan cara yang nyaman untuk memasukkan dan mengedit data, sementara laporan meningkatkan tampilan data yang ditampilkan atau dicetak.

Pertanyaan 64. Apa yang dimaksud dengan antarmuka bahasa prosedural?

Jwb. Antarmuka bahasa prosedural menambahkan kemampuan penuh bahasa pemrograman komputer. Akses non-prosedural dan alat pengembangan aplikasi, meskipun nyaman dan kuat, terkadang tidak cukup efisien atau tidak memberikan tingkat kontrol yang diperlukan untuk pengembangan aplikasi. Ketika alat-alat ini tidak memadai, DBMS menyediakan kemampuan penuh bahasa pemrograman. Antarmuka bahasa prosedural menggabungkan bahasa non-prosedural seperti COBOL atau Visual Basic.

Pertanyaan 65. Apa yang dimaksud dengan transaksi?

Jwb. Transaksi adalah unit kerja yang harus diproses dengan andal tanpa gangguan dari pengguna lain dan tanpa kehilangan data karena kegagalan. Contoh transaksinya adalah tarik tunai di ATM, melakukan reservasi maskapai penerbangan, dan mendaftar kursus.

Pertanyaan 66. Fitur apa yang disediakan DBMS untuk mendukung pemrosesan transaksi?

Jwb. Sebuah DBMS memastikan bahwa transaksi bebas dari gangguan dari pengguna lain, bagian dari transaksi tidak hilang karena kegagalan, dan transaksi tidak membuat database tidak konsisten. Pemrosesan transaksi sebagian besar merupakan urusan "di belakang layar". Pengguna tidak mengetahui detail tentang pemrosesan transaksi selain jaminan tentang keandalan.

Pertanyaan 67. Apa itu DBMS perusahaan?

Jwb. DBMS perusahaan mendukung database yang sering penting untuk berfungsinya suatu organisasi. DBMS perusahaan biasanya berjalan pada server yang kuat dan memiliki biaya tinggi.

Pertanyaan 68. Apa itu DBMS desktop?

Jwb. Sebuah DBMS desktop berjalan pada server pribadi dan kecil. Ini mendukung fitur pemrosesan transaksi terbatas tetapi memiliki biaya yang jauh lebih rendah daripada DBMS enterise. DBMS desktop mendukung database yang digunakan oleh tim kerja dan bisnis kecil.

Pertanyaan 69. Apa saja fitur yang menonjol dari DBMS generasi pertama?

Jwb. Struktur file dan antarmuka program berpemilik adalah fitur yang menonjol dari perangkat lunak Database generasi pertama.

Pertanyaan 70. Apa saja fitur yang menonjol dari DBMS generasi kedua?

Jwb. Jaringan dan hierarki catatan terkait bersama dengan antarmuka program standar adalah fitur yang menonjol dari perangkat lunak Database generasi kedua.

Pertanyaan 71. Apa saja fitur yang menonjol dari DBMS generasi ketiga?

Jwb. Bahasa non-prosedural, optimasi, dan pemrosesan transaksi adalah fitur yang menonjol dari perangkat lunak database generasi ketiga.

Pertanyaan 72. Apa saja fitur yang menonjol dari DBMS generasi keempat?

Jwb. Dukungan untuk data multi-media, database aktif, Data warehouse, dan pemrosesan terdistribusi adalah fitur yang menonjol dari perangkat lunak database generasi.

Pertanyaan 73. Untuk database yang Anda jelaskan di pertanyaan 1, buat tabel untuk menggambarkan perbedaan di antara level skema. Gunakan Tabel 1-4 sebagai panduan.

Jwb.
tbh 292

Pertanyaan 74. Dalam arsitektur client-server, mengapa kemampuan pemrosesan dibagi antara klien dan server? Dengan kata lain, mengapa server tidak melakukan semua pemrosesan?

Jwb. Untuk meningkatkan kinerja dan ketersediaan data, pemrosesan terdistribusi memungkinkan komputer yang tersebar secara geografis untuk bekerja sama saat menyediakan akses data. Pekerjaan dapat diseimbangkan antara server dan klien untuk memproses permintaan akses data secara efisien.

Pertanyaan 75. Dalam arsitektur client-server, mengapa data terkadang disimpan di beberapa komputer daripada di satu komputer?

Jwb. Karena data dapat disimpan di lokasi yang berbeda untuk manajemen dan keamanan, data terkadang disimpan di beberapa komputer, bukan di satu komputer.

Pertanyaan 76. Apa tujuan pemetaan dalam Arsitektur Tiga Skema? Apakah pengguna atau DBMS bertanggung jawab untuk menggunakan pemetaan?

Jwb. Tujuan pemetaan dalam Arsitektur Tiga Skema adalah untuk menggambarkan bagaimana skema pada tingkat yang lebih tinggi diturunkan dari skema pada tingkat yang lebih rendah. DBMS, bukan pengguna, bertanggung jawab untuk menggunakan pemetaan.

Pertanyaan 77. Jelaskan bagaimana Arsitektur Tiga Skema mendukung kemandirian data?

Jwb. Tiga Arsitektur Skema merupakan standar yang berfungsi sebagai pedoman tentang bagaimana kemandirian data dapat dicapai. Semangat Arsitektur Tiga Skema diimplementasikan secara luas di DBMS generasi ketiga dan keempat. Dalam Arsitektur Tiga Skema, DBMS menggunakan skema dan pemetaan untuk memastikan independensi data. Secara umum, aplikasi mengakses database menggunakan tampilan. DBMS mengubah permintaan aplikasi menjadi permintaan menggunakan skema konseptual daripada tampilan. DBMS kemudian mengubah permintaan skema konseptual menjadi permintaan menggunakan skema internal. Sebagian besar perubahan skema konseptual atau internal tidak mempengaruhi aplikasi karena aplikasi tidak menggunakan tingkat skema yang lebih rendah.

Pertanyaan 78. Apa perbedaan antara kunci utama dan kunci unik?

Jwb. Baik kunci utama dan unik menegakkan keunikan kolom yang mereka definisikan. Tetapi secara default, kunci utama membuat indeks berkerumun pada kolom, sedangkan unik membuat indeks tidak berkerumun secara default. Perbedaan utama lainnya adalah, kunci utama tidak mengizinkan NULL, tetapi kunci unik memungkinkan NULL.

Pertanyaan 79. Bagaimana menerapkan hubungan satu-ke-satu, satu-ke-banyak dan banyak-ke-**banyak saat mendesain tabel?**

Jwb.

- Hubungan satu-ke-Satu dapat diimplementasikan sebagai tabel tunggal dan jarang sebagai dua tabel dengan hubungan kunci utama dan kunci asing.
- Relasi One-to-Many diimplementasikan dengan membagi data menjadi dua tabel dengan relasi primary key dan foreign key.

- Hubungan Banyak ke Banyak diimplementasikan menggunakan tabel persimpangan dengan kunci dari kedua tabel yang membentuk kunci utama komposit dari tabel persimpangan.

Pertanyaan 80. Apa perbedaan antara perintah DELETE dan TRUNCATE?

Jwb. Perintah Delete menghapus baris dari tabel berdasarkan kondisi yang kami sediakan dengan klausa WHERE. Truncate sebenarnya akan menghapus semua baris dari sebuah tabel dan tidak akan ada data dalam tabel tersebut setelah kita menjalankan perintah truncate.

TRUNCATE

- TRUNCATE lebih cepat dan menggunakan lebih sedikit sumber daya sistem dan log transaksi daripada DELETE.
- TRUNCATE menghapus data dengan membatalkan alokasi halaman data yang digunakan untuk menyimpan data tabel, dan hanya alokasi halaman yang dicatat dalam log transaksi.
- TRUNCATE menghapus semua baris dari tabel, tetapi struktur tabel dan kolomnya, batasan, indeks, dan sebagainya tetap ada. Penghitung yang digunakan oleh identitas untuk baris baru diatur ulang ke benih untuk kolom.
- Anda tidak dapat menggunakan TRUNCATE TABLE pada tabel yang direferensikan oleh batasan FOREIGN KEY. Karena TRUNCATE TABLE tidak dicatat, itu tidak dapat mengaktifkan pemicu.
- TRUNCATE tidak dapat Digulung kembali.
- TRUNCATE adalah Perintah DDL.
- TRUNCATE me-reset identitas tabel.

DELETE

- DELETE menghapus baris satu per satu dan mencatat entri dalam log transasi untuk setiap baris yang dihapus. Jika Anda ingin mempertahankan penghitung identitas, gunakan DELETE sebagai gantinya. Jika Anda ingin menghapus definisi tabel dan datanya, gunakan pernyataan DROP TABLE.
- DELETE dapat digunakan dengan atau tanpa klausa WHERE
- DELETE Mengaktifkan Pemicu.
- DELETE dapat dibatalkan.
- DELETE adalah Perintah DML.
- DELETE dilakukan untuk mereset identitas tabel.

Pertanyaan 81. Kapan penggunaan perintah UPDATE STATISTICS?

Jwb. Perintah ini pada dasarnya digunakan ketika pemrosesan data yang besar telah terjadi. Jika sejumlah besar penghapusan, modifikasi atau Salin Massal ke dalam tabel telah terjadi, itu harus memperbarui indeks untuk memperhitungkan perubahan ini. UPDATE_STATISTICS memperbarui indeks pada tabel ini.

Pertanyaan 82. Jenis gabungan apa yang dimungkinkan dengan SQL Server?

Jwb. Gabung digunakan dalam kueri untuk menjelaskan bagaimana tabel yang berbeda terkait. Bergabung juga memungkinkan Anda memilih data dari tabel tergantung pada data dari tabel lain. jenis JOIN : INNER JOIN, OUTER JOIN, NATURAL JOIN, OUTER JOIN

diklasifikasikan lebih lanjut sebagai LEFT OUTER JOIN, RIGHT OUTER JOIN, dan FULL OUTER JOIN

Pertanyaan 83. Apa perbedaan antara HAVING CLAUSE dan WHERE CLAUSE?

Jwb. Menentukan kondisi pencarian untuk grup atau agregat. Memiliki hanya dapat digunakan dengan pernyataan SELECT. HAVING biasanya digunakan dalam klausa GROUP BY. Ketika GROUP BY tidak digunakan, HAVING berperilaku seperti klausa WHERE. Memiliki klausa pada dasarnya hanya digunakan dengan fungsi GROUP BY dalam kueri. Klausa WHERE diterapkan ke setiap baris sebelum menjadi bagian dari fungsi GROUP BY dalam kueri.

Pertanyaan 84. Apa itu sub-kueri? Menjelaskan sifat-sifat sub-query.

Jwb. Sub-kueri sering disebut sebagai sub-pilihan, karena memungkinkan pernyataan SELECT dieksekusi secara sewenang-wenang di dalam tubuh pernyataan SQL lainnya. Sub-kueri dijalankan dengan melampirkannya dalam satu set tanda kurung. Sub-kueri umumnya digunakan untuk mengembalikan satu baris sebagai nilai atom, meskipun mereka dapat digunakan untuk membandingkan nilai terhadap beberapa baris dengan kata kunci IN. Sub query adalah pernyataan SELECT yang bersarang di dalam pernyataan T-SQL lainnya.

Sebuah pernyataan SELECT sub query jika dieksekusi ind, terlepas dari pernyataan T-SQL, di mana ia bersarang, akan mengembalikan kumpulan hasil. Artinya, pernyataan SELECT sub kueri dapat berdiri sendiri dan tidak bergantung pada pernyataan di mana pernyataan itu bersarang. Pernyataan SELECT sub kueri dapat mengembalikan sejumlah nilai, dan dapat ditemukan dalam daftar kolom pernyataan SELECT, klausa FROM, GROUP BY, HAVING, dan atau ORDER BY dari pernyataan T-SO. Sub query di mana saja ekspresi dapat digunakan.

Properti Sub-Query

- Sub-query harus diapit dalam tanda kurung.
- Sub query harus diletakkan di sebelah kanan operator pembanding, dan
- Sub query tidak boleh berisi klausa ORDER-BY.
- Sebuah query dapat berisi lebih dari satu sub-query.

Pertanyaan 85. Apakah jenis sub-kueri?

Jwb. Sub kueri baris tunggal, di mana sub kueri hanya mengembalikan satu baris. Sub kueri beberapa baris, di mana sub kueri mengembalikan beberapa baris, dan Sub kueri beberapa kolom, di mana sub kueri mengembalikan beberapa kolom.

Pertanyan 86. Apa itu pengiriman kayu?

Jwb. Pengiriman log adalah proses otomatisasi pencadangan database dan file log transaksi pada server SQL produksi, dan kemudian memulihkannya ke server siaga. Edisi Perusahaan hanya mendukung pengiriman log. Dalam pengiriman log, file log transaksional dari satu server secara otomatis diperbarui ke database cadangan di server lain. Jika salah satu server gagal, server lain akan memiliki database yang sama yang dapat digunakan sebagai rencana Pemulihan Bencana. Fitur utama dari pengiriman log adalah akan secara otomatis membuat cadangan log transaksi sepanjang hari dan secara otomatis mengembalikannya ke server siaga pada interval dermed.

Pertanyaan 87. Apa perbedaan antara variabel lokal dan global?

Jwb. Tabel sementara lokal hanya ada selama durasi koneksi atau, jika didefinisikan di dalam pernyataan majemuk, selama durasi pernyataan majemuk. Tabel sementara global tetap

berada di database secara permanen, tetapi baris hanya ada dalam koneksi tertentu. Saat koneksi ditutup, data dalam tabel sementara global menghilang. Namun, definisi tabel tetap dengan database untuk akses ketika database dibuka waktu berikutnya.

Pertanyaan 88. Apa sifat dari tabel Relasional?

Jwb. Tabel relasi memiliki enam sifat:

- Nilai bersifat atomik.
- Nilai kolom memiliki jenis yang sama.
- Setiap baris unik
- Urutan kolom tidak signifikan. • Setiap kolom harus memiliki nama yang unik.

Pertanyaan 89. Apa itu De-normalisasi?

Jwb. De-normalisasi adalah proses mencoba untuk mengoptimalkan kinerja database dengan menambahkan data yang berlebihan. Jika kadang-kadang diperlukan karena DBMS saat ini mengimplementasikan model relasional dengan buruk. DBMS relasional yang sebenarnya akan memungkinkan database yang sepenuhnya dinormalisasi pada tingkat logis, sambil menyediakan penyimpanan fisik data yang diaktifkan untuk kinerja tinggi. De-normalisasi adalah teknik untuk berpindah dari bentuk normal yang lebih tinggi ke bentuk normal yang lebih rendah dari pemodelan database untuk mempercepat akses database.

Pertanyaan 90. Apa perbedaan antara indeks clustered dan non-clustered?

Jwb. Indeks berkerumun adalah jenis indeks khusus yang menyusun ulang cara catatan dalam tabel disimpan secara fisik. Oleh karena itu tabel hanya dapat memiliki satu indeks berkerumun. Node daun dari indeks berkerumun berisi halaman data. Indeks yang tidak berkerumun adalah jenis indeks khusus di mana urutan logis dari indeks tidak sesuai dengan urutan penyimpanan fisik dari baris pada disk. Node daun indeks non berkerumun tidak terdiri dari halaman data. Sebaliknya, node ieaf berisi baris indeks.

Pertanyaan 91: Apa saja konfigurasi indeks ditTerent yang dapat dimiliki sebuah tabel?

Jwb. Sebuah tabel dapat memiliki salah satu dari konfigurasi indeks berikut:

- Tidak ada indeks
- Sebuah indeks berkerumun
- Sebuah indeks berkerumun dan banyak indeks non berkerumun
- Sebuah indeks non berkerumun
- Banyak indeks non berkerumun

Pertanyaan 92: Mengapa Anda memilih sistem database daripada hanya menyimpan data dalam file sistem operasi? Kapan masuk akal untuk tidak menggunakan sistem database?

Jwb. Database adalah kumpulan data terintegrasi yang biasanya sangat besar sehingga harus disimpan pada perangkat penyimpanan sekunder seperti disk atau kaset. Data ini dapat dipelihara sebagai kumpulan file sistem operasi, atau disimpan dalam DBMS (sistem manajemen Database). Keuntungan menggunakan DBMS adalah:

- Independensi data dan akses yang efisien. Program aplikasi Database tidak bergantung pada detail representasi dan penyimpanan data. Skema konseptual dan eksternal memberikan independensi dari keputusan penyimpanan fisik dan keputusan desain logis masing-masing. Selain itu, DBMS menyediakan mekanisme

pengambilan penyimpanan yang efisien, termasuk dukungan untuk file yang sangat besar, struktur indeks, dan optimisasi kueri.

- Mengurangi waktu pengembangan aplikasi. Karena, DBMS menyediakan beberapa fungsi penting yang diperlukan oleh aplikasi, seperti kontrol konkurensi dan pemulihan kerusakan, fasilitas kueri tingkat tinggi, dll., hanya kode khusus aplikasi yang perlu ditulis. Bahkan ini difasilitasi oleh rangkaian alat pengembangan aplikasi yang tersedia dari vendor untuk banyak sistem manajemen Database.
- Integritas dan keamanan data. Mekanisme tampilan dan fasilitas otorisasi dari DBMS menyediakan mekanisme kontrol akses yang kuat. Selanjutnya, data hingga data yang melanggar semantik data dapat dideteksi dan ditolak oleh DBMS jika pengguna menentukan batasan integritas yang sesuai.
- Administrasi data. Dengan menyediakan payung umum untuk kumpulan besar data yang digunakan bersama oleh beberapa pengguna, DBMS memfasilitasi tugas pemeliharaan dan administrasi data. DBA yang baik dapat secara efektif melindungi pengguna akhir dari tugas-tugas penyesuaian representasi data, pencadangan berkala, dll.
- Akses bersamaan dan pemulihan kerusakan DBMS mendukung gagasan transaksi, yang secara konseptual merupakan program sekuensial pengguna tunggal. Pengguna dapat menulis transaksi seolah-olah program mereka berjalan dalam isolasi terhadap database. DBMS mengeksekusi tindakan transaksi dengan cara yang disisipkan untuk mendapatkan kinerja yang baik, tetapi menjadwalkannya sedemikian rupa untuk memastikan bahwa operasi yang bertentangan tidak diizinkan untuk dilanjutkan secara bersamaan. Selanjutnya, DBMS memelihara log terus menerus dari perubahan data, dan jika ada sistem crash, dapat mengembalikan database ke keadaan transaksi-konsisten. Artinya, tindakan dalam transaksi lengkap dibatalkan, sehingga status Database hanya mencerminkan tindakan transaksi yang diselesaikan. Jadi, jika setiap transaksi selesai, dijalankan sendiri, mempertahankan kriteria konsistensi, maka status Database setelah pemulihan dari kerusakan adalah konsisten. Jika keuntungan ini tidak penting untuk aplikasi yang ada, menggunakan kumpulan file mungkin merupakan solusi yang lebih baik karena peningkatan biaya dan overhead pembelian dan pemeliharaan DBMS.

Pertanyaan 93. Jelaskan perbedaan independensi data logis dan fisik.

Jwb. Independensi data logis berarti bahwa pengguna dilindungi dari perubahan struktur logis data, sedangkan independensi data fisik melindungi pengguna dari perubahan kekuatan fisik data.

Pertanyaan 94. Apa tanggung jawab DBA? Jika kita berasumsi bahwa DBA tidak pernah tertarik untuk menjalankan kuerinya sendiri, apakah DBA masih perlu memahami pengoptimalan kueri? Mengapa?

Jwb. DBA bertanggung jawab untuk:

- Merancang skema logis dan fisik, serta bagian skema eksternal yang banyak digunakan.
- Keamanan dan otorisasi

- Ketersediaan dan pemulihan data dari kegagalan.
- Penyetelan Database : DBA bertanggung jawab untuk mengembangkan Database, khususnya skema konseptual dan fisik, _ untuk memastikan kinerja yang memadai saat persyaratan pengguna berubah. DBA perlu memahami pengoptimalan kueri bahkan jika dia tidak tertarik untuk menjalankan kuerinya sendiri karena beberapa tanggung jawab ini (penandaan dan penyetelan Database) terkait dengan pengoptimalan kueri. Kecuali DBA memahami kebutuhan kinerja kueri yang digunakan secara luas, dan bagaimana DBMS akan mengoptimalkan dan mengeksekusi kueri ini, keputusan desain dan penyetelan yang baik tidak dapat dibuat.

JAWABLAH PERTANYAAN INI

1. Tentukan Database?
2. Apa itu DBMS?
3. Apa kebutuhan sistem database?
4. Tentukan tupel?
5. Apa tanggung jawab DBA?
6. Tentukan skema?
7. Definisikan entitas dan berikan contohnya?
8. Apa yang dimaksud dengan foreign key?
9. Apa perbedaan antara Kunci Unik dan Kunci Utama?
10. Tentukan meta data?
11. Apa kelemahan dari sistem database?
12. Apa yang dimaksud dengan entitas lemah? Berikan contoh
13. Apa itu kalkulus relasional domian?
14. Tentukan QueryLanguage?
15. Tentukan model data?
16. Apa saja 3 level abstraksi data?
17. Apa keuntungan dari model relasional?
18. Tentukan relasi dan himpunan relasi?
19. Tentukan atribut. Sebutkan jenis-jenisnya?
20. Apa yang dimaksud dengan himpunan entitas?
21. Apa saja jenis model data yang berbeda?
22. Apa perbedaan antara kunci kandidat dan kunci super?
23. Apa yang dimaksud dengan model relasional?
24. Apa saja komponen dari manajer penyimpanan?
25. Apa perbedaan antara atribut komposit dan atribut sederhana?
26. Bandingkan sistem database dan sistem file
27. Berikan perbedaan antara kunci primer, kunci kandidat, dan kunci super.
28. Apa yang dimaksud dengan atribut turunan?
29. Apa perbedaan antara himpunan entitas lemah dan kuat?
30. Apa perbedaan antara bahasa prosedural dan non-prosedural?

PERTANYAAN URAIAN

1. Apa kelemahan DBMS dibandingkan dengan sistem pemrosesan file? Jelaskan secara rinci?
2. Menggambar arsitektur sistem DBMS. Jelaskan masing-masing komponen secara rinci. 3. Menjelaskan berbagai jenis model data secara rinci.
3. Bandingkan model jaringan dan hierarki. Jelaskan dengan contoh?
4. Apa yang dimaksud dengan model E-R? Jelaskan dengan contoh
5. Apa saja macam-macam atribut? Jelaskan masing-masing dengan contoh?
6. Tulis catatan rinci tentang aljabar relasional.
7. Menjelaskan kalkulus relasional domain dan tupel secara rinci.
8. Apa perlunya model relasional? Jelaskan masing-masing dengan contoh?
9. Sebutkan jenis-jenis pengguna database dengan perannya masing-masing.
10. Apa peran DBA dalam DBMS?

JENIS PERTANYAAN SINGKAT

1. Apa itu data?
2. Jelaskan Database?
3. Apa itu sistem manajemen Database?
4. Apa itu RDBMS?
5. Tulis bentuk lengkap dari SQL?
6. Jelaskan model relasional?
7. Tulis tiga tampilan arsitektur tiga tingkat?
8. Jelaskan model data?
9. SQL adalah kombinasi dari ..... dan .....
10. Definisikan kamus data?
11. Tentukan istilah berikut: DDL dan DML?
12. Tentukan skema?
13. Tentukan contoh skema?
14. Apa unit dasar diagram ER?
15. Jelaskan kunci utama?
16. Jelaskan kunci asing?
17. Jelaskan kunci kandidat?
18. Jelaskan integritas data?
19. Jelaskan secara singkat integritas entitas?
20. Jelaskan secara singkat integritas referensial?
21. Derme entitas yang lemah?
22. Definisikan database relasional?
23. Apa itu spesialisasi?
24. Apa yang Anda maksud dengan pengenal?
25. Tentukan skema relasional?
26. Apa yang dimaksud dengan relasi unary?
27. Apa fungsi seleksi dan operasi proyeksi?

28. Apa itu abstraksi data?
29. Jelaskan secara singkat konsep metadata?
30. Apa yang Anda maksud dengan kardinalitas? (a) Thple (b) Domain
31. Definisikan istilah berikut: (a) Field (b) Record
32. Apa yang Anda maksud dengan informasi?
33. Apa itu redundansi data?
34. Jelaskan DCL?
35. Jelaskan DDL?
36. Jelaskan DML?
37. Berbagai operasi digunakan dalam aljabar relasional. 39.
38. Berbagai operasi digunakan dalam kalkulus relasional.
39. Apa saja fitur dari operasi gabungan?
40. . Apa saja jenis data yang digunakan dalam SQL?
41. Apakah SQL bahasa non prosedural?
42. Mengapa tupel digunakan dalam model relasional?
43. Apa dua nama atribut lainnya?
44. Apa fungsi dari perintah update dan delete?
45. Apa yang Anda maksud dengan kunci komposit?
46. Apa yang Anda maksud dengan entitas dependen dan independen? 48.
47. Apa yang dimaksud dengan himpunan entitas?
48. Jelaskan entitas reguler?
49. Apa yang Anda maksud dengan model hierarkis?
50. Apa yang dimaksud dengan kunci?
51. Tulis jenis kunci?
52. Apa yang Anda maksud dengan batasan integritas?
53. Apa yang Anda maksud dengan DML prosedural?
54. Apa itu pemrosesan kueri?
55. Apa fungsi parser dalam pemrosesan query?
56. Apa itu organisasi file?
57. Apa itu file langsung?
58. Apa yang Anda maksud dengan hashing?
59. Apa yang dimaksud dengan pemrosesan transaksi?
60. Apa yang Anda maksud dengan B-Tree?
61. Jelaskan secara singkat pemrosesan file?
62. Jelaskan secara singkat pemrosesan Database?
63. 64. Sebutkan semua model yang tersedia untuk sistem database?
64. Tuliskan dua perbedaan antara DBMS dan RDBMS?
65. Apa itu Normalisasi?
66. Apa yang dimaksud dengan kamus data?
67. Tuliskan empat komponen DBMS?
68. Tuliskan empat komponen lingkungan DBMS?
69. Tuliskan bentuk lengkap alat CASE?

70. Apa itu DOMAIN?
71. Apa itu nilai domain?
72. Tulis berbagai jenis kendala?
73. Apa perbedaan antara tipe data numerik dan float di SQL?
74. Apa itu optimasi Ouery?
75. Apa dua fungsi manajer DB?
76. Berbagai jenis pengguna?
77. Di mana kita menyimpan struktur data dalam DBMS?
78. Di mana kita harus mengubah data ketika terjadi perubahan pada data?
79. Jelaskan secara singkat model server klien?
80. Apa yang Anda maksud dengan batasan nol?
81. Apa itu metadata?
82. Apa unit dasar diagram E-R?
83. Apa yang Anda maksud dengan atribut?
84. Apa yang Anda maksud dengan SQL?
85. Jelaskan DBMS terdistribusi?
86. Jelaskan secara singkat programmer aplikasi.
87. Apa yang dimaksud dengan operator pembanding?
88. Berbagai jenis kueri pemilihan?
89. Dua perbandingan antara model Jaringan dan Relasional?
90. Bagaimana tampilan berbeda dari tabel?
91. Jelaskan pendekatan berorientasi objek?
92. Mengapa query drop digunakan?
93. Bentuk lengkap BCNF?
94. Apa perbedaan antara 3NF dan NCNF?
95. Apa yang dimaksud dengan ketergantungan fungsional?
96. Apa perlunya normalisasi?
97. Apa yang dimaksud dengan operasi pemetaan?
98. Apa perbedaan antara bahasa query prosedural dan non prosedural?
99. Apa yang dimaksud dengan himpunan derajat hubungan?

PERTANYAAN JENIS PENDEK

1. Apa yang Anda maksud dengan kardinalitas? Apa saja jenis kardinalitas yang berbeda?
2. Tentukan: (a) DDL, (b) DML
3. Apa itu kunci utama?
4. Apa saja macam-macam tipe Data di SQL?
5. Apa itu relasi? Tentukan model data relasional.
6. Apa yang dimaksud dengan SQL? Apa ciri-ciri SQL?
7. Apa peran Administrator Database?
8. Apa yang dimaksud dengan Sistem Manajemen Database dan Database?
9. Apa masalah dengan sistem pemrosesan file tradisional?
10. Apa yang dimaksud dengan pengolahan data?

11. Apa yang dimaksud dengan entitas, atribut, himpunan entitas, dan relasi?
12. Bagaimana model data E-R berguna?
13. Tentukan atribut. Apa yang dimaksud dengan atribut kunci?
14. Tentukan entitas subtipe dan supertipe?
15. Berikan contoh relasi berikut :
    a. Many-to-One
    b. One-to-One
    c. One-to-Many
    d. Many-to-Many
16. Definisikan foreign key? Bagaimana perannya dalam operasi join?
17. Apa yang dimaksud dengan Operasi Pemetaan?
18. Apa yang dimaksud dengan redundansi? Bagaimana ini bisa dihindari?
19. Apa yang dimaksud dengan Normalisasi? Mengapa ini berguna?
20. Apa perbedaan antara DML Prosedural dan DML Non-Prosedural?
21. Apa yang Anda maksud dengan contoh dan skema? Jelaskan perbedaan antara ini.
22. Apa saja berbagai komponen dari sistem database?
23. Apa peran dari tiga tingkat Abstraksi Data?
24. Bagaimana hubungan banyak-ke-banyak dipetakan ke dalam tabel?
25. Apa yang dimaksud dengan kunci? Jelaskan perbedaan antara kunci primer dan kunci kandidat.
26. Apa perbedaan antara himpunan entitas kuat dan himpunan entitas set?
27. Berikan pernyataan SQL yang membuat tabel STUDENT yang terdiri dari field berikut.

Nama	CHAR (40)
Kelas	CHAR (6)
Nilai	NOMOR (4)
Peringkat	CHAR (8)

28. Apa itu relasi? Apa perbedaan antara tabel dan atribut.
29. Jika Rl adalah relasi dengan 5 baris dan R_2 adalah relasi dengan 3 baris, berapa banyak baris hasil perkalian kartesius dari R_l dan R_2?
30. Subdivisi SQL mana yang digunakan untuk meletakkan nilai dalam tabel dan yang mana untuk membuat tabel
31. Mengapa Data Control Language (DCL) digunakan? Menjelaskan.
32. Jelaskan jenis hubungan berikut ini :
    (iii) Kartu Pelajar dan KTP
    (iv) Nasabah dan Bank
    (v) No Pelajar dan Rolling
    (vi) Nasabah dan Mobil
33. Bedakan antara perintah SQL DROP TABLE dan DROP VIEW.
34. Apakah Kamus Data merupakan bagian penting dari DBMS. Mengapa?
35. Apa yang dimaksud dengan istilah Ouery Processing? Apa saja berbagai langkah yang terlibat dalam proses ini?
36. Apa perbedaan antara WHERE dan HAVING CLAUSE?

37. Apa itu organisasi file? Jelaskan Sequential-file dan direct-Files?
38. Bedakan antara bentuk Normal Pertama dan Bentuk Normal Kedua.
39. Apa itu ketergantungan multinilai? Batasan seperti apa yang ditentukan?
40. Diskusikan berbagai jenis operasi gabungan? Mengapa bergabung ini diperlukan.
41. Sebutkan operasi aljabar relasional dan tujuan masing-masing.
42. Apa perbedaan antara kalkulus relasional tupel dan kalkulus relasi domain?
43. SQL disebut sebagai non-procedurallap.guage. Menjelaskan?
44. Apa yang Anda maksud dengan atribut? Menjelaskan macam-macam atribut.
45. Definisikan istilah-istilah berikut:
    a. Tupel
    b. Domain
    c. Relasi
    d. Entitas
    e. Entitas Reguler
46. Apa perbedaan antara operasi pilih dan proyek? Berikan contoh.
47. Menjelaskan konsep metadata.
48. Apa perlunya Normalisasi? Defme bentuk Normal Ketiga.
49. Jelaskan istilah DBMS Terdistribusi dan DBMS Client-Server.
50. Apa yang Anda maksud dengan Hashing?
51. Apa itu integritas?
52. Apa itu Diagram ER? Apa saja simbol yang digunakan di dalamnya? Jelaskan dengan sebuah contoh.
53. Apa perlunya normalisasi? Jelaskan tiga langkah pertama yang terlibat dalam normalisasi.
54. Buat daftar semua aturan Codd.
55. Apa yang Anda maksud dengan abstraksi database? Ada berapa jenis?
56. Perbedaan antara pendekatan berorientasi file dan database.
57. Perbedaan antara sistem Database dan sistem basis Pengetahuan.
58. Menjelaskan arsitektur client server secara detail.
59. Apa arsitektur databasenya. Jelaskan dengan diagram
60. Apa itu DBA, apa fungsi DBA?
61. Gambarlah diagram ER untuk sistem perbankan.
62. Apa itu pemrosesan kueri. Jelaskan berbagai langkah yang terlibat di dalamnya.
63. Apa itu manajemen Database.
64. Apa itu bahasa query terstruktur? Bagaimana DDL dan DML berbeda? Menjelaskan.
65. Apa sistem filenya. Jelaskan file sekuensial dan file langsung.
66. Apa kelemahan dari sistem file. Jelaskan secara rinci.
67. Apa redundansi data? Bagaimana cara menghilangkan redundansi data? Jelaskan itu.

PERTANYAAN JENIS PANJANG

1. Apa saja macam-macam komponen Sistem Database? Jelaskan secara rinci.
2. Apa yang Anda maksud dengan model data? Jelaskan jaringan, model hierarki dan relasional secara rinci.
3. Jelaskan macam-macam tingkat abstraksi data dalam sistem Database?
4. Apa yang dimaksud dengan Database? Apa tujuan dari sistem database? Menjelaskan.
5. Apa yang Anda maksud dengan DBMS? Jelaskan fungsinya.
6. Jelaskan arsitektur DBMS dan kelebihannya? Sebutkan dua kelemahan utama DBMS?
7. Apa itu DBA? Apa tanggung jawab utama dari DBA dan desainer database?
8. Apa masalah dengan sistem pemrosesan file tradisional? Bagaimana mereka dihapus dalam sistem database? Jelaskan
9. Apa yang dimaksud dengan Entity-Relationship Diagram? Jelaskan
10. Jelaskan berbagai istilah model E-R dan bagaimana mereka diwakili dalam model E-R?
11. Apa yang dimaksud dengan istilah hubungan antar entitas? Jelaskan berbagai jenis hubungan yang dapat ada dengan contoh.
12. Jelaskan konsep entitas dependen? Berikan contoh.
13. Apa yang dimaksud dengan pemetaan kardinalitas? Menjelaskan macam-macam kardinalitas
14. Apa perbedaan partisipasi total dan partisipasi parsial? Menjelaskan.
15. Apa perbedaan antara atribut tunggal dan multinilai?
16. Jelaskan konsep kendala partisipasi?
17. Diskusikan berbagai operasi pembaruan pada relasi dan jenis batasan integritas yang harus diperiksa untuk setiap operasi pembaruan?
18. Diskusikan berbagai jenis operasi gabungan? Mengapa bergabung ini membutuhkan?
19. Apa yang dimaksud dengan normalisasi? Menjelaskan.
20. Apa yang dimaksud dengan BCNF? Mengapa digunakan dan apa bedanya dengan 3 NF?
21. Jelaskan arsitektur tiga tingkat DBMS? Juga jelaskan pentingnya dalam lingkungan database.
22. Mendiskusikan konsep bahasa database dan interface.
23. Berikan berbagai kelebihan dan kekurangan model jaringan. Apa bedanya dengan model relasional?
24. Apa itu hubungan? Apa saja macam-macam hubungan? Jelaskan dengan contoh.
25. Jelaskan Aturan Codd secara rinci.
26. Apa yang Anda maksud dengan RDBMS? Apa karakteristiknya?
27. Jelaskan integritas Entitas dan integritas Referensial secara rinci.
28. Apa perbedaan antara DBMS dan RDBMS? Manakah dari mereka yang lebih cocok?
29. Apa itu aljabar relasional? Diskusikan berbagai operasi aljabar relasional.
30. Jelaskan berbagai jenis kalkulus relasional secara rinci.
31. Apa itu kalkulus relasional? Membedakan aljabar relasional dan kalkulus relasional.
32. Apa yang Anda maksud dengan nilai Null? Jelaskan dengan contoh yang sesuai.

33. Mengapa diperlukan normalisasi? Apa kekurangannya?
34. Diskusikan berbagai bentuk normal dalam normalisasi dengan contoh yang sesuai.
35. Definisikan istilah anomali. Jelaskan BCNF secara rinci
36. Mengapa kontrol konkurensi diperlukan?
37. Apa itu Deadlock? Bagaimana bisa terjadi Deadlock? menjelaskan.
38. Jelaskan secara singkat satu algoritma pencegahan Deadlock.
39. Bagaimana jika time tamping digunakan? Jelaskan secara singkat
40. Apa itu penguncian dua fase dan bagaimana menjamin serializability?
41. Diskusikan mekanisme kontrol konkurensi secara rinci menggunakan contoh yang sesuai.
42. Bedakan antara penguncian dua fase dan penguncian dua fase yang ketat.
43. Bagaimana cara menghindari deadlock saat menggunakan 2pL?
44. Bagaimana kunci berbagi dan kunci eksklusif berbeda? Menjelaskan.
45. Bagaimana grafik prioritas dapat digunakan untuk mendeteksi Deadlock?
46. Apa itu log sistem? Apa tujuan dari sistem log in pemulihan sistem?
47. Apa yang Anda pahami tentang database terdistribusi? Sebutkan berbagai kelebihan dan kekurangan sistem manajemen Database terdistribusi.
48. Menjelaskan arsitektur database Client-Server secara detail.
49. Apa perbedaan utama antara sistem paralel dan sistem terdistribusi? Menjelaskan.
50. Diskusikan konsep Ouery Processing. Apa itu pengurai? Mengapa digunakan?
51. Apa itu optimasi Ouery? Apa saja teknik berbeda yang digunakan di dalamnya.

BANK PERTANYAAN DBMS

1. Tuliskan perbedaan antara Sistem Database Vs Sistem Berkas
2. Jelaskan Tampilan Data.
3. Definisikan istilah-istilah berikut:
    a. Contoh
    b. Skema
    c. Skema logis
    d. Skema fisik
4. Jelaskan Secara Singkat tentang Model Data:
    a. Model ER
    b. Model Relasional
    c. Model Berorientasi Objek
5. Menjelaskan Bahasa Database
6. Menjelaskan Pengguna dan Administrator Database
7. Menjelaskan 'Manajemen Transaksi.
8. Jelaskan Secara Singkat Struktur Sistem Database
9. Jelaskan Himpunan Entitas dan Himpunan Relasi.
10. Menjelaskan Atribut dan jenis-jenisnya beserta contohnya.
11. Jelaskan Constraints dan jenisnya dengan contoh
12. Jelaskan Pemetaan Kardinalitas dengan contoh.

13. Tentukan kunci dan jenisnya dengan contoh.
14. Menjelaskan struktur logika keseluruhan Diagram E-R.
15. Jelaskan Simbol yang digunakan dalam notasi E-R.
16. Mendefinisikan UML dan simbol yang digunakan dalam notasi UML.
17. Jelaskan Aljabar Relasional dan jelaskan semua operasi Fundamentalnya dengan contoh.
18. Menjelaskan Operasi Komposisi dalam Aljabar Relasional.
19. Jelaskan Definisi Formal Aljabar Relasional.
20. Jelaskan Tampilan dengan contoh.
21. Menjelaskan Kalkulus Relasional Tuple dengan semua operasinya. Jelaskan Perbedaan Kalkulus Relasional dengan Aljabar Relasional.
22. Jelaskan Kalkulus Relasional Domain beserta contohnya. Dan nyatakan perbedaan antara ini dan kalkulus relasional Thple.
23. Apa itu Join dan Jenisnya dengan Contoh query.
24. Apa yang disematkan-SQL dengan contoh.
25. Jelaskan Penegasan dengan contoh.
26. Jelaskan Cursor dan jenis-jenisnya beserta contohnya.
27. Jelaskan Keamanan dan Pelanggarannya beserta contohnya.
28. Tentukan 2 NF dengan' contoh.
29. Tentukan batasan Domain?
30. Jelaskan kendala Integritas Referensial dengan contoh?
31. Apakah Modifikasi Database Melanggar integritas Referensial?
32. Definisikan Asersi dengan contoh?
33. Jelaskan Pemicu dengan contoh
34. Jelaskan Pemicu di SQL? Dan jenis-jenisnya beserta contohnya.
35. Jelaskan kapan tidak boleh menggunakan Pemicu?
36. Jelaskan Pelanggaran Keamanan dengan contoh.
37. Jelaskan Otorisasi?
38. Jelaskan Pemberian Hak Istimewa?
39. Jejak audit baik-baik saja dengan contoh.
40. Jelaskan Otorisasi di SQL?
41. Jelaskan Bentuk Normal Pertama beserta contohnya? (*)
42. Jelaskan ketergantungan fungsional dan ketergantungan fungsional penuh? (*)
43. Jelaskan Penguraian dan Sifat-sifatnya? (*)
44. Jelaskan bentuk normal Boyce-codd beserta contohnya? (*)
45. Jelaskan bentuk normal ketiga beserta contohnya? (*)
46. Jelaskan Normal keempat beserta contohnya? (*)
47. Sebutkan perbedaan antara bentuk normal Boyce-codd dan bentuk normal ketiga?
48. Jelaskan Atribut Asing?
49. Jelaskan integritas referensial, modifikasi DB dengan kaskade penghapusan dan pembaruan
50. Jelaskan Pemicu dan jenisnya dengan contoh.

51. Jelaskan Autorization adalah SQL dengan contohnya.
52. Jelaskan ketergantungan fungsional dan ketergantungan fungsional sepele dengan contoh.
53. Jelaskan bentuk normal keempat beserta contohnya.
54. Jelaskan Penutupan Himpunan Ketergantungan Fungsional dan Penutupan Himpunan Atribut
55. Jelaskan Penutup Kanonik dan Atribut Asing dengan contoh.
56. Jelaskan BCNF beserta contohnya dan sebutkan juga perbedaan antara bentuk 3NF ini.

DAFTAR PUSTAKA

Damayanti, K., Fardinal., (2019). The Effect of Information Technology Utilization, Management Support, Internal Control, and User Competence on Accounting Information System Quality. Schollars Bulletin, 5(12), 751-758.

DosenPendidikan. 2021. DBMS (Database Management System).

Hanifah, S., Sarpingah, S., & Putra, Y. M., (2020). The Effect of Level of Education, Accounting Knowledge, and Utilization Of Information Technology Toward Quality The Quality of MSME ' s Financial Reports.

Heripracoyo, Sulistyo. 2018. Implementasi Oracle Pada Beberapa Perusahaan Di Indonesia.

Herliansyah, Y., Nugroho, L., Ardilla, D., & Putra, Y. M., (2020). The Determinants of Micro, Small and Medium Entrepreneur (MSME) Become Customer of Islamic Banks (Religion, Religiosity, and Location of Islamic Banks). The 1st Annual Conference Economics, Business, and Social Sciences.

https://doi.org/10.4108/eai.26-3-2019.2290775

https://doi.org/10.4108/eai.3-2-2020.163573

https://sis.binus.ac.id/2018/12/12/implementasi-oracle-pada-beberapa-perusahaan-di-indonesia/. Diakses Pada 15 April 2021.

https://www.dimas-prasetyo.com/wp-content/uploads/2018/11/teori-sistem-basisdata.pd

https://www.dosenpendidikan.co.id/dbms/. Diakses Pada 15 April 2021.

Putra, Y. M. (2018). Sistem Manajemen Basis Data. Modul Kuliah Sistem Informasi Manajemen. FEB-Universitas Mercu Buana: Jakarta

Putra, Y. M. (2020). Sistem Manajemen Basis Data. Modul Kuliah Sistem Informasi Manajemen. FEB-Universitas Mercu Buana: Jakarta

Putra, Y. M., (2019). Analysis of Factors Affecting the Interests of SMEs Using Accounting Applications. Journal of Economics and Business, 2, 818-826. https://doi.org/10.31014/aior.1992.02.03.129

Zamzami, A.H., & Putra, Y. M., (2019). Intensity of Taxpayers Using E-Filing (Empirical Testing of Taxpayers in Jakarta, Bogor, Depok, Tangerang, and Bekasi). EPRA International Journal of Multidisciplinary Research (IJMR) 5(7), 154-161.