PART 2 - Fundamental DE

TASK 1

1. Kapan kita harus menggunakan relational database atau nosql database?
   Pemilihan antara relational database (RDBMS) dan NoSQL database bergantung pada kebutuhan spesifik dari aplikasi atau sistem yang akan dikembangkan. Berikut adalah panduan untuk menentukan kapan harus menggunakan masing-masing jenis database:
   a. **Relational Database (RDBMS):**
      Kapan Menggunakan RDBMS:
      1) **Kebutuhan ACID Compliance**:
         Jika aplikasi memerlukan transaksi yang konsisten, isolasi, dan dapat diandalkan (Atomicity, Consistency, Isolation, Durability).
      2) **Data Terstruktur dengan Skema Tetap**:
         Jika data memiliki skema yang jelas dan tetap (misalnya, tabel dengan kolom tetap dan tipe data tertentu).
      3) **Hubungan Antar Data**:
         Jika ada banyak hubungan antar data (seperti one-to-one, one-to-many, many-to-many) dan perlu diatur dengan kunci asing (foreign keys).
      1) **Analitik dan Pelaporan**:
         Jika membutuhkan query yang kompleks, seperti join antar tabel, agregasi data, dan fungsi analitik lainnya.
      2) **Kepastian Data**:
         Ketika memerlukan validasi data yang kuat dan integritas referensial untuk memastikan bahwa data selalu akurat dan konsisten.

      Contoh Penggunaan RDBMS:
      - Sistem perbankan dan keuangan
      - Aplikasi manajemen inventaris
      - Sistem manajemen pelanggan (CRM)
      - Sistem ERP

   b. **NoSQL Database:**
      Kapan Menggunakan NoSQL:
      1) **Data Tidak Terstruktur atau Semi-Terstruktur**:
         Jika data tidak memiliki skema tetap atau sering berubah (misalnya, dokumen JSON, XML).
      2) **Skalabilitas Horizontal**:
         Jika aplikasi membutuhkan kemampuan untuk skalabilitas horizontal yang tinggi, seperti yang dibutuhkan oleh aplikasi yang menangani volume besar data atau lalu lintas tinggi (misalnya, aplikasi media sosial).
      3) **Kebutuhan Latensi Rendah**:
         Jika membutuhkan akses data yang sangat cepat dan latensi rendah, terutama untuk aplikasi real-time.
      4) **Model Data Fleksibel**:
         Jika aplikasi membutuhkan model data yang fleksibel yang dapat dengan mudah disesuaikan dengan perubahan dalam struktur data tanpa downtime.
      5) **Distribusi Data Global**:
         Jika data perlu didistribusikan di berbagai lokasi geografis untuk mendekatkan data ke pengguna akhir atau untuk keandalan yang lebih tinggi.

Jenis-jenis NoSQL Database dan Contoh Penggunaan:

a. **Document Store** (misalnya, MongoDB, CouchDB):
   - Menyimpan data dalam format dokumen (seperti JSON atau BSON).
   - **Contoh Penggunaan**: Aplikasi konten manajemen, katalog produk, blog, dan aplikasi dengan data yang kompleks dan beraneka ragam.

b. **Key-Value Store** (misalnya, Redis, DynamoDB):
   - Menyimpan data sebagai pasangan kunci-nilai.
   - **Contoh Penggunaan**: Cache, session store, aplikasi yang membutuhkan akses data dengan latensi sangat rendah.

c. **Column Family Store** (misalnya, Cassandra, HBase):
   - Menyimpan data dalam kolom, bukan baris.
   - **Contoh Penggunaan**: Aplikasi analitik big data, log pencatatan, dan aplikasi yang membutuhkan penulisan dan pembacaan data yang cepat dalam jumlah besar.

d. **Graph Database** (misalnya, Neo4j, OrientDB):
   - Menyimpan data dalam format graf yang terdiri dari node, edges, dan properties.
   - **Contoh Penggunaan**: Aplikasi jaringan sosial, manajemen identitas dan akses, rekomendasi produk.

Gunakan RDBMS jika aplikasi membutuhkan transaksi yang kuat, integritas data yang tinggi, skema tetap, dan kemampuan untuk melakukan query yang kompleks.

Gunakan NoSQL jika aplikasi membutuhkan fleksibilitas skema, skalabilitas tinggi, latensi rendah, dan data yang tidak terstruktur atau semi-terstruktur.

Pemilihan antara RDBMS dan NoSQL juga dapat dipengaruhi oleh faktor lain seperti tim pengembang, alat yang sudah ada, dan persyaratan bisnis. Dalam beberapa kasus, kombinasi dari kedua jenis database (polygot persistence) mungkin menjadi solusi terbaik untuk memenuhi berbagai kebutuhan aplikasi.

2. Apa perbedaan antara database, data lake, data warehouse, dan data mart?

a. Database

Definisi: Sebuah sistem yang dirancang untuk menyimpan, mengelola, dan mengakses data secara efisien. Database umumnya digunakan untuk operasi sehari-hari dalam aplikasi bisnis dan dapat berupa RDBMS atau NoSQL.

Ciri-ciri:
- **Tujuan**: Mendukung operasi transaksi dan aplikasi bisnis sehari-hari.
- **Skema**: Biasanya memiliki skema tetap (terutama RDBMS).
- **Jenis Data**: Terstruktur (dalam tabel dengan kolom dan baris) atau semi-terstruktur (misalnya, JSON dalam NoSQL).
- **Contoh**: MySQL, PostgreSQL, MongoDB, Redis.

Penggunaan:
- Sistem manajemen pelanggan (CRM)
- Sistem inventaris
- Aplikasi web dan mobile

b. **Data Lake**

Definisi: Penyimpanan terpusat yang dapat menampung data dalam berbagai format, baik terstruktur maupun tidak terstruktur. Data lake dirancang untuk menyimpan data dalam skala besar dengan biaya rendah.

Ciri-ciri:

- **Tujuan**: Menyimpan data mentah dalam jumlah besar untuk analitik mendalam dan machine learning.
- **Skema**: Skema-on-read, artinya skema diterapkan saat data dibaca, bukan saat data disimpan.
- **Jenis Data**: Terstruktur, semi-terstruktur, dan tidak terstruktur (misalnya, log file, gambar, video, sensor data).
- **Contoh**: Amazon S3, Hadoop HDFS, Azure Data Lake**.**

Penggunaan:

- Data mentah untuk analitik big data
- Machine learning dan AI
- Penyimpanan data historis yang besar

a. **Data Warehouse**

Definisi: Sistem penyimpanan data yang dioptimalkan untuk query dan analitik. Data warehouse mengkonsolidasikan data dari berbagai sumber untuk analisis dan pelaporan.

Ciri-ciri:

- **Tujuan**: Mendukung analitik bisnis dan pelaporan dengan kinerja tinggi.
- **Skema**: Skema tetap (skema-on-write), artinya data diatur sesuai skema saat disimpan.
- **Jenis Data**: Terstruktur, biasanya diubah dan dimuat dari berbagai sumber operasional.
- **Contoh**: Amazon Redshift, Google BigQuery, Snowflake.

Penggunaan:

- Pelaporan bisnis dan analitik
- Analisis data historis
- Pengambilan keputusan berbasis data

b. **Data Mart**

Definisi: Subset dari data warehouse yang difokuskan pada satu area bisnis atau departemen tertentu. Data mart dirancang untuk memenuhi kebutuhan analitik yang spesifik dan lebih terfokus.

Ciri-ciri:

- **Tujuan**: Menyediakan data yang relevan untuk analitik dan pelaporan spesifik departemen atau tim tertentu.
- **Skema**: Skema tetap, sesuai kebutuhan spesifik pengguna.
- **Jenis Data**: Terstruktur, sering diambil dari data warehouse.
- **Contoh**: Bagian dari data warehouse atau bisa diimplementasikan menggunakan sistem database tertentu**.**

Penggunaan:

- Analitik pemasaran
- Pelaporan keuangan
- Analitik penjualan

Perbandingan

a. **Database**:
   - Fokus pada operasi sehari-hari dan aplikasi transaksi.
   - Menyimpan data terstruktur dan semi-terstruktur dengan skema tetap.
b. **Data Lake**:
   - Menyimpan data mentah dalam berbagai format (terstruktur, semi-terstruktur, tidak terstruktur).
   - Fleksibel dengan skema-on-read dan cocok untuk analitik big data dan machine learning.
c. **Data Warehouse**:
   - Dioptimalkan untuk analitik dan pelaporan dengan skema tetap.
   - Menyimpan data terstruktur yang dikonsolidasi dari berbagai sumber.
d. **Data Mart**:
   - Subset dari data warehouse, difokuskan pada area bisnis atau departemen tertentu.
   - Mendukung analitik dan pelaporan spesifik dengan skema tetap.

Contoh Penggunaan:

a. **Database**:
   - CRM untuk manajemen data pelanggan.
   - Sistem inventaris untuk mengelola stok barang.
b. **Data Lake**:
   - Penyimpanan data mentah dari sensor IoT untuk analitik.
   - Penyimpanan data log web server untuk analisis tren pengguna.
c. **Data Warehouse**:
   - Analitik penjualan yang mengkonsolidasikan data dari berbagai sistem POS.
   - Pelaporan keuangan yang mengumpulkan data dari berbagai departemen.
d. **Data Mart**:
   - Analitik pemasaran untuk kampanye iklan yang menggunakan subset data dari data warehouse.
   - Pelaporan kinerja tim penjualan dengan data yang relevan dari data warehouse.

Dengan memahami perbedaan ini, organisasi dapat memilih solusi penyimpanan data yang paling sesuai dengan kebutuhan spesifik mereka, baik itu untuk operasi transaksi, analitik big data, pelaporan bisnis, atau analitik khusus departemen.

3. Jelaskan apa itu normalisasi database, dan normalisasikan tabel dibawah!

employee_id	employee_name	job_code	job	city_code	city_name	province_code	province_name
1	John Smith	101	Software Engineer	201	New York	301	New York
2	Alice Johnson	102	Data Analyst	202	Los Angeles	302	California
3	Bob Davis	103	Data Engineer	203	Chicago	303	Illinois
4	Emily Wilson	101	Software Engineer	204	Houston	304	Texas
5	Michael Lee	102	Data Analyst	205	Miami	305	Florida
6	Sarah Brown	103	Data Engineer	206	Boston	306	Massachusetts
7	James Clark	101	Software Engineer	207	San Fransisco	307	California
8	Laura Taylor	102	Data Analyst	208	Seattle	308	Washington
9	Daniel White	103	Data Engineer	209	Denver	309	Colorado
10	Olivia Martin	101	Software Engineer	210	Atlanta	310	Georgia

Normalisasi adalah proses pengorganisasian data dalam database untuk mengurangi redundansi dan meningkatkan integritas data, dengan memecah table menjadi table-table baru lainnya.

Tabel asli:

employee_id	employee_name	job_code	job	city_code	city_name	province_code	province_name
1	John Smith	101	Software Engineer	201	New York	301	New York
2	Alice Johnson	102	Data Analyst	202	Los Angeles	302	California
3	Bob Davis	103	Data Engineer	203	Chicago	303	Illinois
4	Emily Wilson	101	Software Engineer	204	Houston	304	Texas
5	Michael Lee	102	Data Analyst	205	Miami	305	Florida
6	Sarah Brown	103	Data Engineer	206	Boston	306	Massachusetts
7	James Clark	101	Software Engineer	207	San Fransisco	307	California
8	Laura Taylor	102	Data Analyst	208	Seattle	308	Washington
9	Daniel White	103	Data Engineer	209	Denver	309	Colorado
10	Olivia Martin	101	Software Engineer	210	Atlanta	310	Georgia

Normalisasi Table

Table employees:

employee_id	employee_name	job_code	city_code
1	John Smith	101	201
2	Alice Johnson	102	202
3	Bob Davis	103	203
4	Emily Wilson	101	204
5	Michael Lee	102	205
6	Sarah Brown	103	206
7	James Clark	101	207
8	Laura Taylor	102	208
9	Daniel White	103	209
10	Olivia Martin	101	210

Table jobs:

job_code	job
101	Software Engineer
102	Data Analyst
103	Data Engineer

Table locations:

city_code	city_name	province_code
201	New York	301
202	Los Angeles	302
203	Chicago	303
204	Houston	304
205	Miami	305
206	Boston	306
207	San Francisco	307
208	Seattle	308
209	Denver	309
210	Atlanta	310

Table provinces:

province_code	province_name
301	New York
302	California
303	Illinois
304	Texas
305	Florida
306	Massachusetts
307	California
308	Washington
309	Colorado
310	Georgia

Dengan tabel-tabel ini, data telah dinormalisasi.