FUNDAMENTAL OF DE

PART 1 - Fundamental DE

TASK 1

1. Apa peran utama seorang Data Engineer dalam ekosistem data? Bagaimana peran ini berbeda dari Data Scientist dan Data Analyst?
2. berikan beberapa contoh peran dari seorang Data Engineer yang mungkin bersinggungan atau bahkan sama dengan peran Data Scientist dan Data Analyst !
3. Jelaskan langkah-langkah proses ETL dan ELT yang berperan dalam pekerjaan seorang data engineer !

PART 2 - Fundamental DE

TASK 1

1. Kapan kita harus menggunakan relational database atau nosql database ?
2. Apa perbedaan antara database, data lake, data warehouse, dan data mart ?
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

JAWABAN

1. Peran Utama Seorang Data Engineer dalam Ekosistem Data

Peran utama seorang Data Engineer adalah untuk membangun, mengelola, dan mengoptimalkan infrastruktur yang memungkinkan pengumpulan, penyimpanan, dan analisis data. Mereka fokus pada aspek teknis yang memastikan data tersedia dan dapat digunakan oleh Data Scientist dan Data Analyst. Data Engineer bertanggung jawab atas pembuatan pipeline data yang efisien, serta pengolahan dan pemeliharaan data dalam skala besar.

Perbedaan dengan Data Scientist dan Data Analyst:

- **Data Scientist**: Berfokus pada analisis data yang lebih kompleks dan pengembangan model machine learning. Mereka menggunakan statistik, algoritma pembelajaran mesin, dan teknik analitik lainnya untuk mengidentifikasi tren dan membuat prediksi dari data.
- **Data Analyst**: Lebih berfokus pada analisis data yang ada untuk menghasilkan laporan dan visualisasi yang membantu dalam pengambilan keputusan bisnis. Mereka sering menggunakan alat BI (Business Intelligence) untuk mengubah data mentah menjadi informasi yang dapat digunakan.

2. Contoh Peran Data Engineer yang Bersinggungan dengan Data Scientist dan Data Analyst

- **Pembersihan Data (Data Cleaning)**: Data Engineer mungkin terlibat dalam membersihkan data sebelum disimpan dalam database, sementara Data Scientist dan Data Analyst menggunakan data bersih tersebut untuk analisis lebih lanjut.
- **Data Integration**: Data Engineer mengintegrasikan data dari berbagai sumber, yang kemudian digunakan oleh Data Scientist dan Data Analyst untuk analisis dan pelaporan.
- **Pembangunan Pipeline Data**: Data Engineer membuat pipeline otomatis untuk ekstraksi, transformasi, dan pemuatan (ETL) data, yang sangat penting bagi Data Scientist dan Data Analyst untuk mendapatkan data yang siap dianalisis.
- **Pemantauan Kualitas Data**: Data Engineer bertanggung jawab untuk memastikan kualitas data yang tinggi, yang merupakan dasar penting bagi analisis yang dilakukan oleh Data Scientist dan Data Analyst.

3. Langkah-langkah Proses ETL dan ELT dalam Pekerjaan Seorang Data Engineer

Proses ETL (Extract, Transform, Load):

1. **Extract**: Mengambil data dari berbagai sumber, seperti database relasional, API, atau file CSV.
2. **Transform**: Mengubah data menjadi format yang sesuai dan lebih berguna, termasuk pembersihan data, normalisasi, agregasi, dan pembuatan indeks.
3. **Load**: Memuat data yang telah diubah ke dalam sistem penyimpanan tujuan seperti data warehouse atau data lake.

Proses ELT (Extract, Load, Transform):

1. **Extract**: Mengambil data dari berbagai sumber.
2. **Load**: Memuat data mentah langsung ke dalam sistem penyimpanan tujuan.
3. **Transform**: Melakukan transformasi data di dalam sistem penyimpanan itu sendiri, sering kali menggunakan kemampuan komputasi yang disediakan oleh data warehouse atau data lake tersebut.

Perbedaan utama antara ETL dan ELT terletak pada urutan transformasi data: dalam ETL, transformasi terjadi sebelum data dimuat ke sistem penyimpanan, sementara dalam ELT, transformasi terjadi setelah data dimuat ke dalam sistem penyimpanan. Kedua proses ini esensial dalam memastikan bahwa data yang digunakan oleh Data Scientist dan Data Analyst adalah data yang berkualitas dan siap untuk dianalisis.

1. Kapan Kita Harus Menggunakan Relational Database atau NoSQL Database?

Relational Database

Relational database (RDBMS) seperti MySQL, PostgreSQL, dan Oracle adalah pilihan yang tepat ketika:

- **Data Terstruktur**: Data memiliki struktur yang jelas dan dapat diatur dalam tabel dengan kolom dan baris yang jelas.
- **Transaksi ACID**: Memerlukan dukungan untuk transaksi yang mengikuti properti ACID (Atomicity, Consistency, Isolation, Durability).
- **Integritas Data**: Kebutuhan akan integritas data yang tinggi dengan penggunaan foreign keys, constraints, dan hubungan antar tabel.
- **Skema Tetap**: Data memiliki skema yang tetap dan jarang berubah.
- **Operasi CRUD**: Aplikasi memerlukan operasi Create, Read, Update, Delete yang kompleks dan konsisten.

NoSQL Database

NoSQL database seperti MongoDB, Cassandra, dan Redis adalah pilihan yang tepat ketika:

- **Data Tidak Terstruktur atau Semi-Terstruktur**: Data tidak memiliki struktur yang kaku, seperti dokumen, JSON, XML, atau data graf.
- **Skalabilitas Horizontal**: Memerlukan kemampuan untuk skala horizontal dengan mudah, seperti menambah lebih banyak server.
- **Skema Fleksibel**: Data memiliki skema yang fleksibel atau sering berubah.
- **Volume Data Besar**: Menghadapi volume data yang sangat besar dan pertumbuhan data yang cepat.
- **Kecepatan Akses**: Memerlukan kecepatan baca/tulis yang tinggi.
- **Jenis Data Khusus**: Memerlukan penyimpanan data yang khusus, seperti key-value stores, document stores, column-family stores, atau graph databases.

2. Perbedaan antara Database, Data Lake, Data Warehouse, dan Data Mart

- **Database**: Sistem manajemen yang mengorganisir, menyimpan, dan mengelola data dalam bentuk terstruktur. Digunakan untuk operasi transaksi harian dan mendukung aplikasi operasional. Contohnya termasuk RDBMS dan NoSQL databases.
- **Data Lake**: Penyimpanan besar untuk menyimpan data dalam bentuk mentah atau asli, baik terstruktur, semi-terstruktur, maupun tidak terstruktur. Digunakan untuk menyimpan data dalam jumlah besar dengan berbagai format. Contohnya termasuk Hadoop dan Amazon S3.
- **Data Warehouse**: Sistem penyimpanan data yang mengkonsolidasikan dan mengorganisir data dari berbagai sumber dalam bentuk terstruktur dan dioptimalkan untuk query analitik. Biasanya digunakan untuk analisis bisnis dan pelaporan. Contohnya termasuk Amazon Redshift dan Google BigQuery.
- **Data Mart**: Subset dari data warehouse yang fokus pada area atau departemen tertentu dalam organisasi. Digunakan untuk memberikan analitik yang lebih spesifik dan relevan kepada departemen tertentu, seperti pemasaran atau keuangan. Contohnya bisa berupa subset yang diambil dari data warehouse yang lebih besar.

3. Normalisasi Database dan Normalisasi Tabel di Atas

Normalisasi adalah proses pengorganisasian kolom dan tabel dari database relasional untuk mengurangi redundansi data dan meningkatkan integritas data. Tujuan utamanya adalah menghilangkan data yang berulang dan memastikan data tergantung secara logis pada kunci utama.

Langkah-langkah Normalisasi:

1. **1NF (First Normal Form)**:
   - Hilangkan kelompok berulang dalam tabel. Setiap kolom harus mengandung nilai atomik.
   - Setiap kolom harus mengandung jenis data yang sama.
   - Setiap kolom dalam tabel harus memiliki nilai unik.
2. **2NF (Second Normal Form)**:
   - Harus memenuhi semua kriteria 1NF.
   - Semua kolom non-kunci harus sepenuhnya tergantung pada seluruh kunci utama.
3. **3NF (Third Normal Form)**:
   - Harus memenuhi semua kriteria 2NF.
   - Semua kolom non-kunci harus tergantung hanya pada kunci utama dan tidak ada ketergantungan transitif.

Normalisasi Tabel di Atas:

employee_id	employee_name	job_code	job	city_code	city_name	province_code	province_name
1	John Smith	101	Software	201	New York	301	New York

			Engineer				
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

Langkah 1: Buat Tabel Terpisah untuk Job, City, dan Province

Tabel 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

Tabel Jobs:

job_code	job
101	Software Engineer
102	Data Analyst
103	Data Engineer

Tabel Cities:

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

Tabel 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 normalisasi, kita memastikan bahwa setiap informasi disimpan hanya satu kali, mengurangi redundansi dan menjaga integritas data.