

Modul 04 Database



Modul BASIS DATA

Pelatihan Jabatan Fungsional
Analisis Data Ilmiah



Fungsi Layanan Pengembangan Kompetensi Kedinasan
Direktorat Pengembangan Kompetensi
Deputi Sumber Daya Manusia Ilmu Pengetahuan dan Teknologi
2022

Materi Pokok

Mata pelatihan ini terdiri dari 5 (lima) Materi Pokok, yaitu:

1. Pengantar basis data;
 - a. Konsep Basis Data, *Relational Database*, *SQL* dan *NoSQL*
 - b. *Query Standar* dan *transaction handling*
 - c. Pengaturan *Permission* dan *Security*
 - d. *Backup* dan *Restore Database*
2. Struktur data;
 - a. Desain Database
 - b. Pengertian Struktur Data
 - c. Jenis Struktur Data
 - d. Struktur Data pada Basis Data
3. Standardisasi data;
 - a. Konsep Standardisasi Data
 - b. Standar Data
 - c. Teknik Standardisasi Data pada Basis Data
4. Normalisasi database;
 - a. Konsep Normalisasi
 - b. Normalisasi Bentuk Normal pertama (1NF)
 - c. Konsep mengenai kunci-kuncipada tabel *database* – Super, Primary, Candidate, Foreign, Key, dll
 - d. Normalisasi Bentuk Normal kedua (2NF)
 - e. Normalisasi Bentuk Normal ketiga (3NF)
 - f. Normalisasi Boyce-Codd Normal Form (BCNF)
 - g. Normalisasi Bentuk Normal keempat (4NF)
5. *Data warehouse*.
 - a. Definisi Gudang Data
 - b. Komponen Gudang Data
 - c. Perancangan Gudang Data
 - d. Dimensional Modelling
 - e. Operasi OLAP

Quiz!



4.1

Database vs. Data Warehouse: Mengubah Data Menjadi Keputusan

Memahami perbedaan mendasar untuk keunggulan kompetitif



Fondasi Data - Apa Itu Database?



Definisi

Kumpulan data terstruktur yang disimpan secara elektronik untuk akses cepat dan aman



Fokus Utama

OLTP – Online Transaction Processing untuk pencatatan transaksi harian secara real-time



Contoh Praktis

Database pelanggan toko, inventaris barang, data karyawan, dan sistem penjualan

Analogi Database: Buku Kas Harian



Perumpamaan Sederhana

Database seperti buku kas yang mencatat setiap pemasukan dan pengeluaran secara rinci setiap hari

- Cepat untuk menambah transaksi baru
- Efisien mengubah atau menghapus data tunggal
- Kurang ideal untuk analisis tren jangka panjang



Database: Jantung Operasional Bisnis

Sistem yang menjalankan kegiatan harian Anda dengan efisiensi maksimal

Melampaui Transaksi - Kekuatan Data Warehouse

Definisi

Kumpulan data terintegrasi, berorientasi subjek, dan bersifat time-variant (berbasis waktu) untuk analisis mendalam

Fokus Utama

OLAP – Online Analytical Processing untuk mendukung pengambilan keputusan strategis dan business intelligence

Integrasi

Menggabungkan data dari berbagai sumber operasional untuk gambaran holistik

Analogi Data Warehouse: Laporan Keuangan Tahunan

Pandangan Strategis

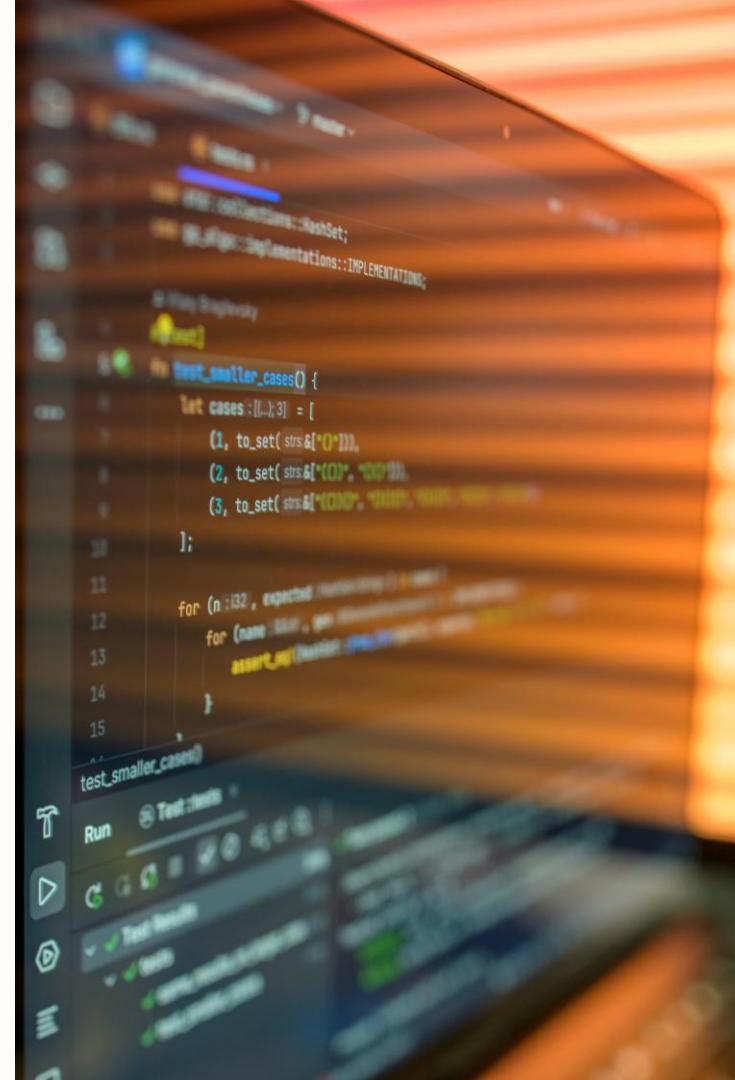
Seperti laporan keuangan tahunan yang merangkum semua transaksi untuk analisis kinerja secara komprehensif

- Menyediakan pandangan historis dan tren
- Mendukung strategi masa depan
- Data read-only untuk integritas analisis



Data Warehouse: Kompas Bisnis Anda

Memandu Anda menuju pertumbuhan dan inovasi dengan insight berbasis data



Perbedaan Kunci: Database vs. Data Warehouse

Fokus

Database: Transaksi operasional

Data Warehouse: Analisis strategis

Sumber Data

Database: Satu sistem operasional

Data Warehouse: Banyak sistem terintegrasi

Sifat Data

Database: Volatil dan dapat diubah

Data Warehouse: Non-volatil dan tetap

Tujuan

Database: Operasional harian

Data Warehouse: Keputusan strategis

Proses ETL: Jembatan Antar Dunia Data



Extract

Mengambil data dari berbagai sumber database operasional secara efisien



Transform

Membersihkan, menstandarisasi, dan mengintegrasikan data untuk konsistensi



Load

Memuat data yang sudah bersih ke dalam Data Warehouse untuk analisis

Proses krusial ini memastikan kualitas dan integritas data di Data Warehouse

Fitur 

Database

Data Warehouse

Jenis

Operasional (OLTP)

Analitis (OLAP)

Fokus

Transaksi harian, *update*

Analisis, pelaporan tren

Data

Terkini, detail

Historis, terintegrasi

Skema

Dinormalisasi (3NF)

Denormalisasi (Star/Snowflake)

Waktu Respon

Sangat cepat (milidetik)

Cepat untuk kueri kompleks

RDBMS vs. Columnar: OLTP vs. OLAP

ml4devs.com/datastores



OLTP



Rows are stored together

Word-aligned heterogeneous values take more space

Fast row reads and updates

High ACID transaction throughput

GBs of data, scale vertically

Suitable for typical CRUD applications

Logical Table

Date	Product	Sale
D1	P1	S1
D2	P2	S2
D3	P3	S3
D4	P4	S4

RDBMS: Row-Oriented Datastore

OLAP



Columns are stored together

Type aware encoding aids compression

Fast column summarizations

High aggregation query throughput

TBs of data, scale horizontally

Suitable for analytics and machine learning

© Satish Chandra Gupta



CC BY-NC-ND 4.0 International License

creativecommons.org/licenses/by-nc-nd/4.0/

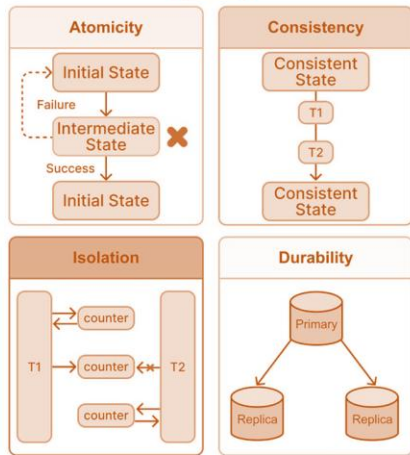
scgupta.me

twitter.com/scgupta

linkedin.com/in/scgupta

SQL vs NoSQL: Choosing the Right Database

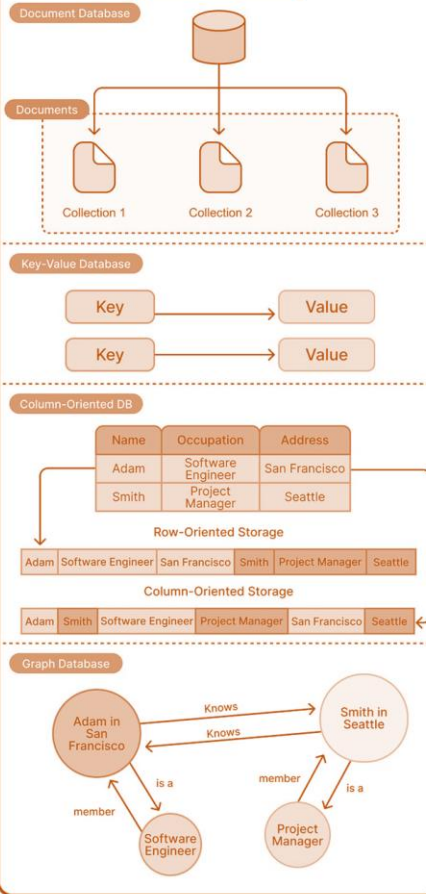
ACID



BASE

- ✓ **Basically Available (BA)**
System is available most of the time even in the face of failures
- ✓ **Soft State (S)**
State of the system may change over time even without external input
- ✓ **Eventually Consistent (E)**
System will eventually become consistent once changes are propagated to all nodes

NoSQL Databases





Kesimpulan: Data yang Tepat, Keputusan yang Cerdas

Database

Tulang punggung operasional harian yang menjalankan transaksi secara efisien

Data Warehouse

Mesin analisis untuk pertumbuhan strategis dan insight berbasis data

Sinergi

Memahami perbedaan dan integrasi keduanya adalah kunci sukses di era data-driven

4.2

Memahami Normalisasi Database: Dari 1NF hingga 5NF

Normalisasi adalah proses sistematis untuk merancang database yang efisien dan bebas dari redundansi. Dalam presentasi ini, kita akan menjelajahi lima bentuk normal (1NF, 2NF, 3NF, 4NF, dan 5NF) serta memahami bagaimana masing-masing tingkat membantu menciptakan struktur database yang optimal.



Quiz!



Mengapa Normalisasi Penting?

Tujuan Utama

Mengurangi redundansi data dan meningkatkan integritas data dalam sistem database

Integritas Data

Memastikan data tetap akurat dan konsisten di seluruh database

Efisiensi

Mengoptimalkan penggunaan ruang penyimpanan dan kinerja query

Menghindari Anomali

1

Insertion Anomaly

Tidak bisa menambah data baru tanpa menyertakan data terkait yang mungkin belum tersedia. Misalnya, tidak bisa menambahkan karyawan baru tanpa menentukan departemen terlebih dahulu.

2

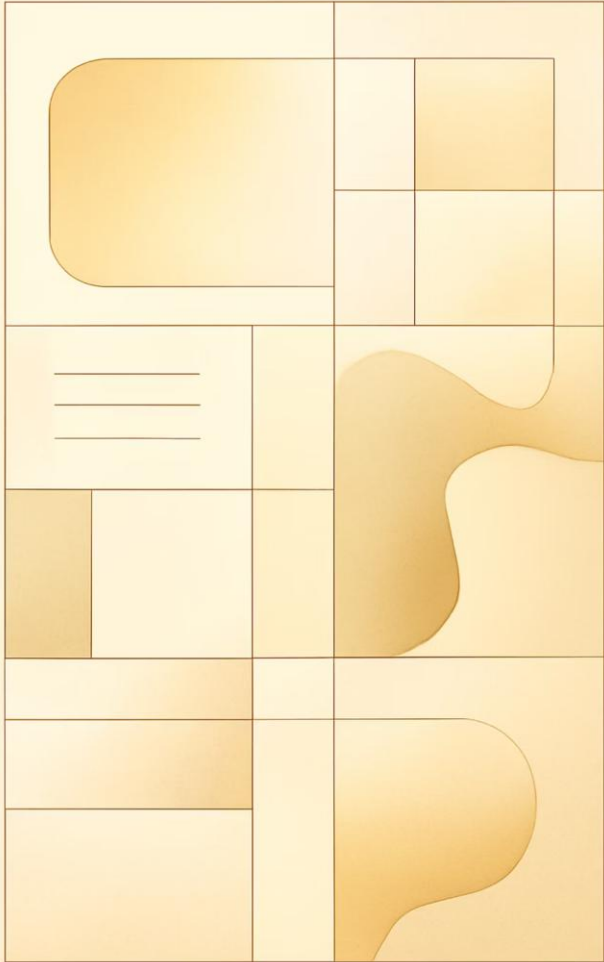
Deletion Anomaly

Menghapus satu data malah menghapus informasi penting lainnya. Contoh: menghapus pesanan bisa menghapus informasi pelanggan secara tidak sengaja.

3

Update Anomaly

Perubahan data di satu tempat tidak konsisten dengan tempat lain karena redundansi. Misalnya, mengubah alamat pelanggan harus dilakukan di banyak tempat.



Bentuk Normal Pertama (1NF): Nilai Atomik

Aturan-Aturan 1NF

01

Nilai Atomik

Setiap kolom harus berisi nilai atomik (tidak ada daftar atau grup berulang dalam satu sel)

02

Tipe Data Konsisten

Setiap kolom harus memiliki tipe data yang konsisten di seluruh baris

03

Primary Key Unik

Setiap tabel harus memiliki primary key yang unik untuk mengidentifikasi setiap baris

04

Tidak Ada Urutan

Urutan baris tidak boleh menyampaikan informasi semantik

Contoh Pelanggaran dan Solusi

✗ Pelanggaran

Kolom "Telepon" berisi beberapa nomor dalam satu sel seperti: 0812-3456-7890, 0878-1234-5678

✓ Solusi

Pisahkan nilai berulang ke tabel terpisah dengan relasi ke entitas utama, atau buat kolom terpisah untuk setiap jenis telepon

Bentuk Normal Kedua (2NF): Ketergantungan Penuh



Prasyarat 1NF

Tabel harus sudah memenuhi bentuk normal pertama



Primary Key Gabungan

Masalah muncul terutama pada tabel dengan primary key gabungan



Ketergantungan Penuh

Semua kolom non-primary key harus bergantung pada seluruh primary key

Contoh Kasus



Pelanggaran 2NF

Tabel Pesanan dengan primary key gabungan
(ID_Pesanan, ID_Produk):

- Nama Produk → bergantung hanya pada ID_Produk
- Jumlah → bergantung pada kedua ID
- Harga Satuan → bergantung pada ID_Produk

✓ Solusi Normalisasi

Pecah menjadi dua tabel:

- Tabel Pesanan: ID_Pesanan, ID_Produk, Jumlah
- Tabel Produk: ID_Produk, Nama Produk, Harga Satuan

Bentuk Normal Ketiga (3NF): Ketergantungan Transitif



Prasyarat 2NF

Tabel harus sudah dalam bentuk normal kedua



Hapus Transitif

Tidak boleh ada ketergantungan transitif antara kolom non-primary key



Ketergantungan Langsung

Setiap kolom non-primary key harus bergantung langsung pada primary key

Ilustrasi Ketergantungan Transitif

$A \rightarrow B$

$B \rightarrow C$

Pindah C

Bentuk Normal Keempat (4NF): Ketergantungan Multivalued



Prasyarat 3NF

Tabel harus sudah memenuhi bentuk normal ketiga



Identifikasi Multivalued

Temukan ketergantungan multivalued independen dalam tabel



Pecah Tabel

Pisahkan setiap ketergantungan ke tabel terpisah

Contoh Kasus 4NF



Pelanggaran 4NF

Tabel Karyawan yang menyimpan:

- ID_Karyawan (primary key)
- Mata Kuliah (banyak nilai)
- Buku Rekomendasi (banyak nilai)

Karyawan bisa mengambil banyak mata kuliah DAN merekomendasikan banyak buku secara independen

✓ Solusi 4NF

Tiga tabel terpisah:

- Tabel Karyawan: ID_Karyawan, Nama
- Tabel Kursus: ID_Karyawan, Mata_Kuliah
- Tabel Rekomendasi: ID_Karyawan, Buku

Setiap relasi many-to-many diatur dengan tabel join

Bentuk Normal Kelima (5NF): Ketergantungan Join

Prasyarat 4NF

Tabel harus sudah dalam bentuk normal keempat

Join Dependency

Menghilangkan redundansi yang hanya dapat dihilangkan dengan melakukan join pada tabel-tabel yang terlibat

Normalisasi Tertinggi

Bentuk normal paling ketat dan jarang diterapkan dalam praktik industri

Memahami Join Dependency

5NF mengatasi kasus di mana informasi dapat direkonstruksi dari gabungan (join) beberapa tabel, tetapi tidak dapat dipisahkan lebih lanjut tanpa kehilangan informasi. Ini terjadi pada skenario dengan tiga atau lebih atribut yang memiliki ketergantungan kompleks.

Catatan Praktis: Sebagian besar sistem database modern jarang menerapkan 5NF karena kompleksitasnya yang tinggi dan manfaat marginal dibandingkan dengan biaya implementasi.

Boyce-Codd Normal Form (BCNF)

Definisi

Bentuk normal yang lebih ketat dari 3NF

Aturan Utama

Untuk setiap dependensi fungsional $X \rightarrow Y$,
X harus menjadi superkey

Keunggulan

Menghilangkan semua anomali yang mungkin tersisa

Perbedaan dengan 3NF

3NF

Mengizinkan dependensi fungsional $X \rightarrow Y$ jika
Y adalah bagian dari candidate key (prime
attribute)

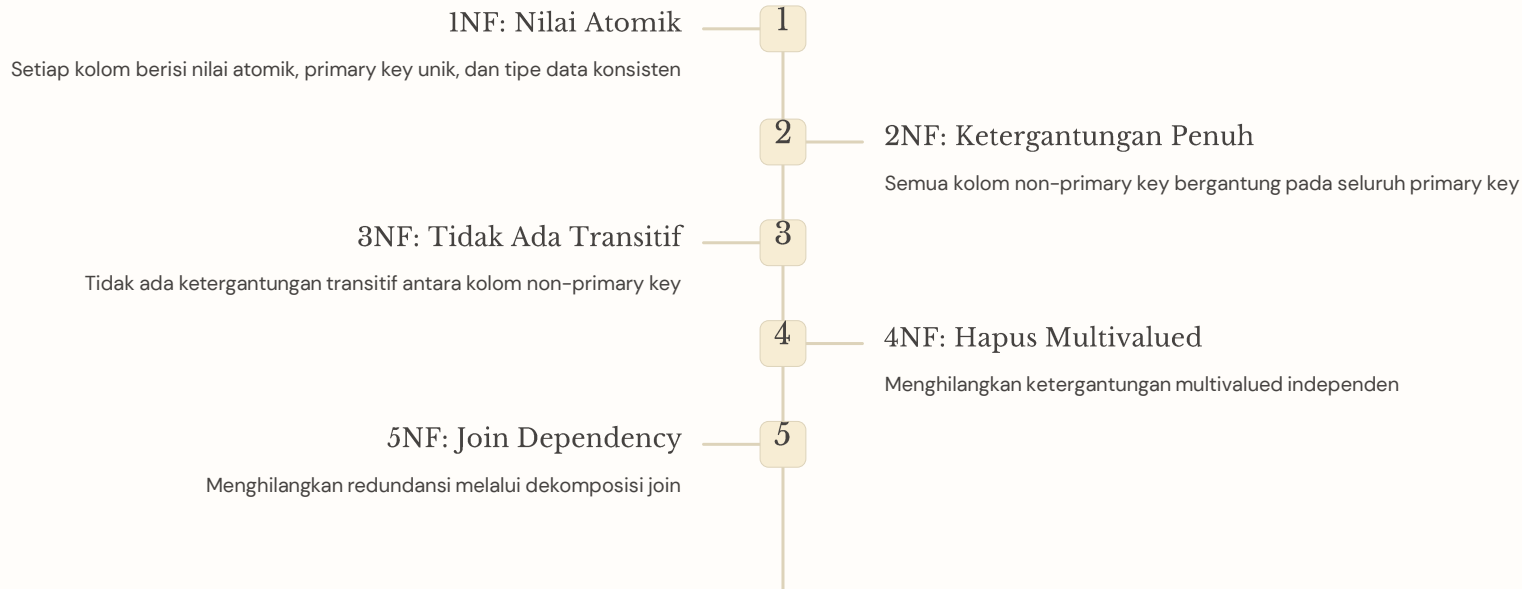
BCNF

Lebih ketat – X harus menjadi superkey tanpa
pengecualian



Penting: Kebanyakan desain database yang mencapai 3NF sudah cukup memadai dan bebas dari anomali penambahan, pengubahan, dan penghapusan. BCNF memastikan tidak ada anomali yang tersisa, tetapi jarang diperlukan dalam praktik.

Ringkasan: Perjalanan Normalisasi



Tujuan Akhir Normalisasi



Database Efisien

Struktur yang optimal dengan redundansi minimal



Bebas Anomali

Tidak ada masalah insertion, deletion, atau update



Integritas Data

Konsistensi dan akurasi data terjaga



Mudah Pemeliharaan

Struktur yang mudah dipahami dan dikelola

Penerapan dalam Praktik

Fokus Utama: 3NF atau BCNF Umumnya, mencapai 3NF atau BCNF sudah sangat baik untuk sebagian besar aplikasi bisnis dan sistem produksi	Manfaat Nyata Memastikan integritas data, mengurangi biaya penyimpanan, dan mempermudah pemeliharaan sistem database	Pertimbangan Trade-off Kadang perlu denormalisasi untuk optimasi performa query, tetapi harus dilakukan dengan hati-hati
--	---	---

Kapan Menggunakan Tingkat Normalisasi Lebih Tinggi?

✓ Gunakan 3NF/BCNF - Sistem transaksi umum - Aplikasi bisnis standar - Keperluan integritas data tinggi - CRUD operations dominan	⚠ Pertimbangkan 4NF/5NF - Data warehouse dan BI - Kasus multivalued kompleks - Keperluan normalisasi maksimal - Sudah menguasai 3NF dengan baik
---	---

Langkah-Langkah Praktis

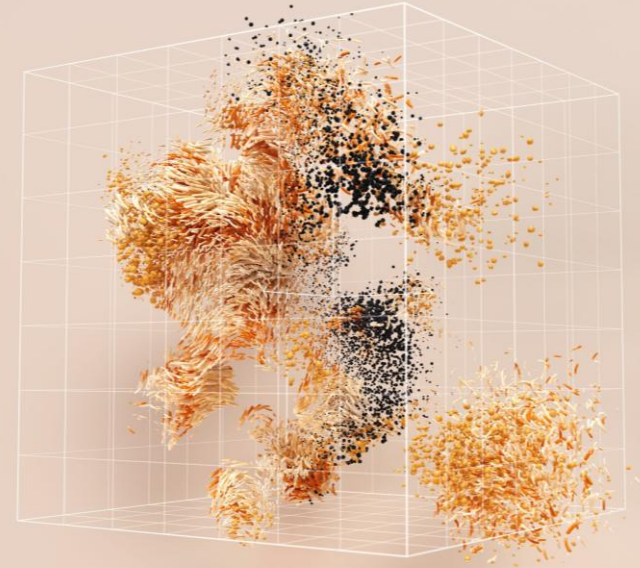
01	02	03
Identifikasi Entity Tentukan entitas dan atribut yang perlu disimpan	Tentukan Primary Key Setiap tabel harus memiliki primary key yang unik	Periksa Ketergantungan Identifikasi semua functional dependencies
04	05	
Normalisasi Bertahap Terapkan 1NF, 2NF, 3NF secara berurutan	Verifikasi dan Uji Periksa integritas dan uji query yang umum	

Quiz!



4.3

SQL vs NoSQL: Memilih Database yang Tepat



Pendahuluan: Fondasi Data Aplikasi Anda



Keputusan Krusial

Memilih database yang tepat adalah keputusan krusial dalam pengembangan aplikasi modern.



Dua Opsi Populer

Dua opsi populer: SQL (relasional) dan NoSQL (non-relasional) dengan karakteristik unik.



Pertimbangan Matang

Keduanya memiliki keunggulan dan kelemahan, pemilihan bergantung pada kebutuhan spesifik proyek.

Memahami Model Data Relasional (SQL)



Tabel Terstruktur

Data diorganisir dalam **tabel** dengan **baris** dan **kolom** yang jelas.

Skema Tetap

Menggunakan **skema tetap** yang terdefinisi dengan baik dan konsisten.

Relasi Kuat

Menekankan **relasi antar tabel** melalui kunci (primary key, foreign key).

SQL Standar

Bahasa kueri standar: **Structured Query Language (SQL)**.

Contoh: MySQL, PostgreSQL, Oracle, SQLite

Menjelajahi Model Data Non-Relasional (NoSQL)



Key-Value Stores

Penyimpanan pasangan kunci-nilai sederhana. **Contoh:** Redis, DynamoDB



Wide-Column Stores

Struktur kolom fleksibel dengan skalabilitas tinggi. **Contoh:** Cassandra, HBase



Document Databases

Data disimpan dalam format dokumen seperti JSON. **Contoh:** MongoDB, CouchDB



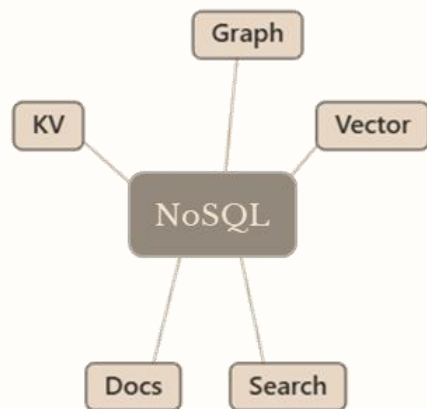
Graph Databases

Menangani data dengan relasi kompleks. **Contoh:** Neo4j, Amazon Neptune



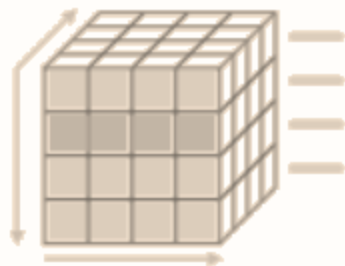
Fleksibilitas Tinggi

Fleksibel, tidak terikat skema tetap, memungkinkan perubahan struktur data dengan mudah.



SQL

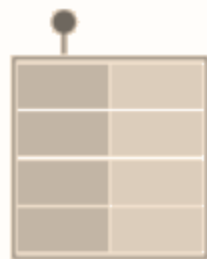
Relational Database Management Systems (RDBMS)



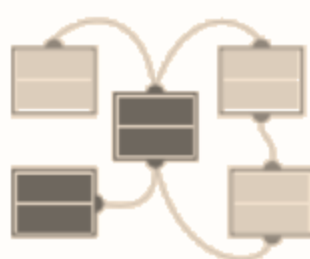
Online Analytical Processing (OLAP) Cube

NoSQL

Key-Value



Graph



Document



Column store

SQL vs NoSQL

Perbandingan Kunci: SQL vs NoSQL

Fitur	SQL (Relasional)	NoSQL (Non-Relasional)
Model Data	Tabel terstruktur, skema tetap	Fleksibel, berbagai model (dokumen, key-value, dll.)
Skalabilitas	Vertikal (upgrade hardware)	Horizontal (tambah server)
Konsistensi	ACID (Atomicity, Consistency, Isolation, Durability)	BASE (Basically Available, Soft state, Eventually consistent)
Bahasa Kueri	SQL	Bervariasi (API, kueri spesifik)
Fleksibilitas	Rendah	Tinggi
Kompleksitas	Cocok untuk data terstruktur dan relasi kompleks	Cocok untuk data tidak terstruktur/semi-terstruktur

Kapan Memilih SQL?

Integritas Data Penting

Transaksi keuangan, sistem perbankan, dan aplikasi yang memerlukan keandalan tinggi.

Struktur Stabil

Struktur data jelas dan stabil dengan relasi yang kuat antar entitas.

Kueri Kompleks

Analisis data yang melibatkan banyak join dan operasi kompleks.

Konsistensi ACID

Konsistensi data yang ketat diperlukan untuk integritas sistem.

Contoh: Aplikasi e-commerce (manajemen pesanan, inventaris), sistem ERP, sistem akuntansi

Kapan Memilih NoSQL?

Skalabilitas Tinggi

Aplikasi dengan jutaan pengguna yang memerlukan ketersediaan tinggi dan performa konsisten.

Fleksibilitas Skema

Data yang sering berubah atau tidak terstruktur memerlukan fleksibilitas dalam desain.

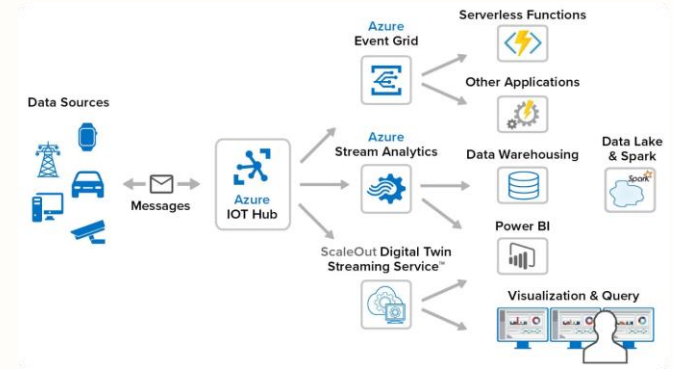
Kecepatan Real-time

Aplikasi real-time seperti game online, IoT, dan streaming data membutuhkan latensi rendah.

Volume Besar

Big data analytics dan penyimpanan data skala besar dengan kebutuhan horizontal scaling.

Contoh: Media sosial (profil pengguna, feed), IoT (data sensor), analitik real-time



Polyglot Persistence: Kombinasi Terbaik



Kombinasi Strategis

Banyak arsitektur modern menggunakan **kombinasi SQL dan NoSQL** secara bersamaan.



Beban Tepat

Masing-masing database menangani beban kerja yang paling sesuai dengan kekuatannya.



Integrasi Optimal

Contoh: E-commerce menggunakan SQL untuk transaksi dan NoSQL untuk rekomendasi produk.

Kesimpulan: Pilihlah dengan Bijak

Tidak Ada Solusi Universal

Tidak ada satu solusi yang cocok untuk semua kasus penggunaan.

Pahami Kebutuhan

Struktur data, skalabilitas, konsistensi, dan performa harus dievaluasi dengan cermat.

Pertimbangkan Kombinasi

Polyglot persistence sering kali memberikan solusi yang lebih optimal.

Keputusan Masa Depan

Keputusan yang tepat akan membentuk cara Anda merancang, membangun, dan menskalakan aplikasi Anda.



Quiz & Latihan!



Latihan-1: Persiapan

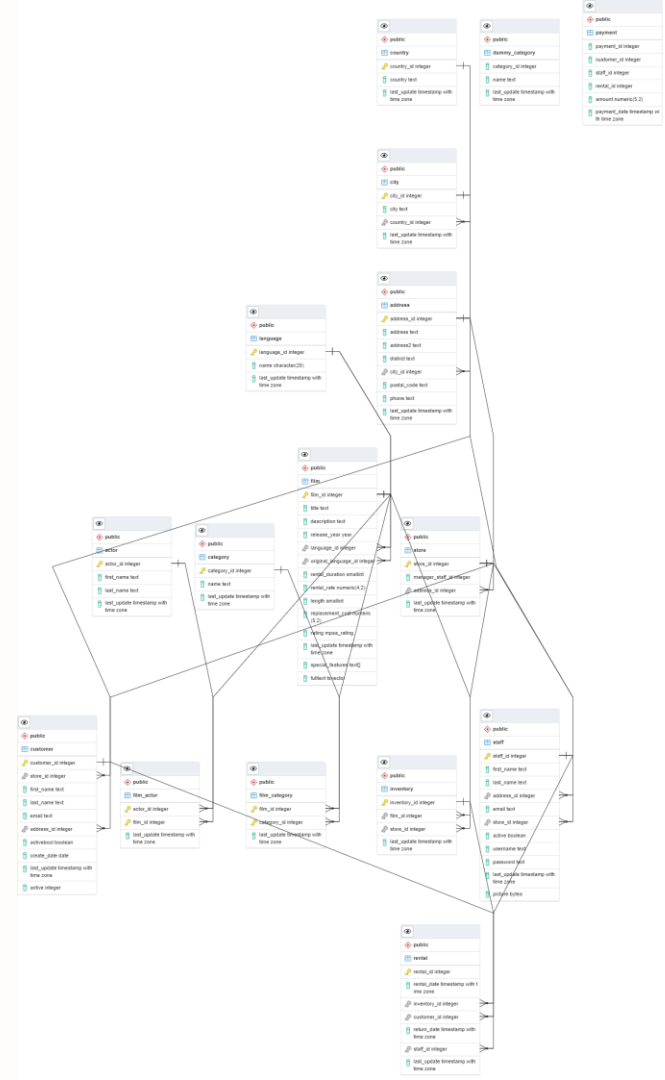
Lingkungan & Kondisi:

1. DBMS: Postgresql
2. Data: database rental video Pagila (dari Saqila Mysql)
3. Pagila database:

<https://github.com/devrimgunduz/pagila>

Persiapan:

1. Lakukan instalasi Postgresql via Docker Compose (bila belum terinstall Postgresql)
2. Lakukan docker-compose (lihat informasi di github repo tsb)
3. Load database Pagila
4. Kerjakan soal-soal di halaman berikut.



Latihan 1: Praktek Query Database

Level 1: Filtering & Sorting Dasar

- **Q1:** Buat query untuk menemukan seluruh actor yang nama keduanya dimulai dengan 'GEN'
- **Q2:** Daftar seluruh film yang Panjang durasinya lebih dari 180 menit, diurutkan dari yang paling panjang durasinya.
- **Q3:** Temukan seluruh customer yang sekarang tidak aktif (inactive).

Level 2: Joins & Agregasi

- **Q4:** Daftar seluruh kategori dan total jumlah film setiap kategori.
- **Q5:** Temukan top 10 film terpopuler berbasis frekuensi rental
- **Q6:** Tampilkan setiap ID toko (store) disertari dengan nama kota dan negara bagian.

Level 3: Advanced Analysis & Views

- **Q7:** Identifikasi top 5 genre berdasarkan total pendapatan (revenue)
- **Q8:** Hitung jumlah customer yang aktif dan tidak aktif per kota.
- **Q9:** Buat view bernama `top_five_grossing_genres` berbasis query dari Q7.

Level 4: Modifikasi Schema & Data

- **Q10:** Tambahkan satu kolom `middle_name` di table `actor`
- **Q11:** Tulis query untuk menemukan pembayaran yang dilakukan secara spesifik pada bulan April 2022 dari tabel-tabel pembayaran yang terpartisi.

Database•



Modul

BASIS DATA

Pelatihan Jabatan Fungsional
Analisis Data Ilmiah



Fungsi Layanan Pengembangan Kompetensi Kedinasan
Direktorat Pengembangan Kompetensi
Deputi Sumber Daya Manusia Ilmu Pengetahuan dan Teknologi
2022