



LEARNING PROGRESS REVIEW WEEK 2

AI-POWERED DATA SCIENCE

**SKILLS OF A DATA SCIENTST | BASIC SQL |
ADVANCED SQL**



**TEAM 4
DATA EXPLORERS**

DATA EXPLORERS TEAM MEMBERS

- Steven Rusly
- Muhammad Syauqi Luvaniesky
- Alfian Rosyid
- Raihan Maulana Nilmada
- Edvan Tazul Arifin
- Pebry Zauhary

DATA EXPLORERS TEAM MEMBERS



Steven
Rusly



Muhammad
Syauqi
Luvaniesky



Alfian
Rosyid



Raihan
Maulana
Nilmada



Edvan Tazul
Arifin



Pebry
Zauhary



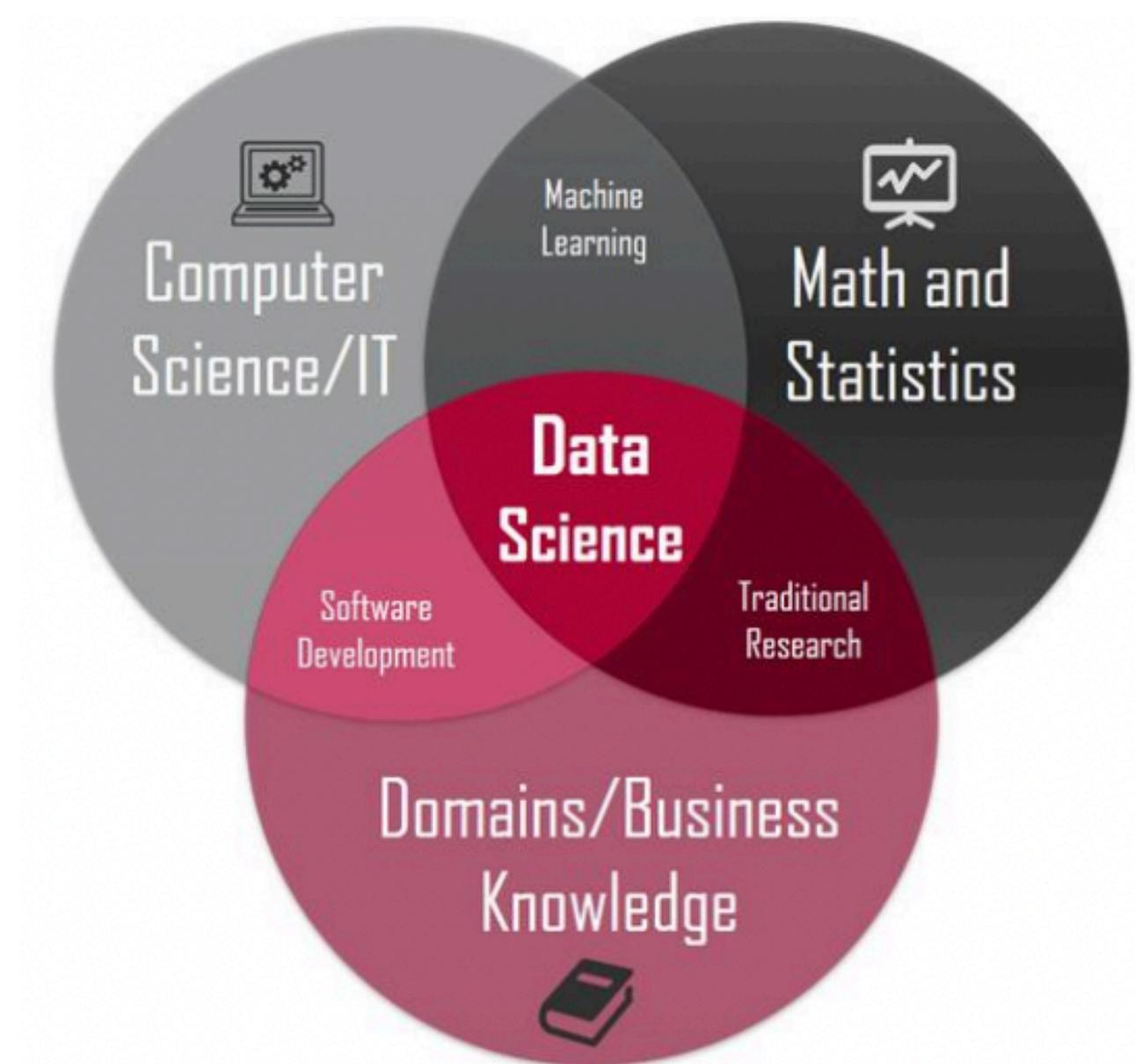
SESI 4

DATA SCIENCE MINDSET

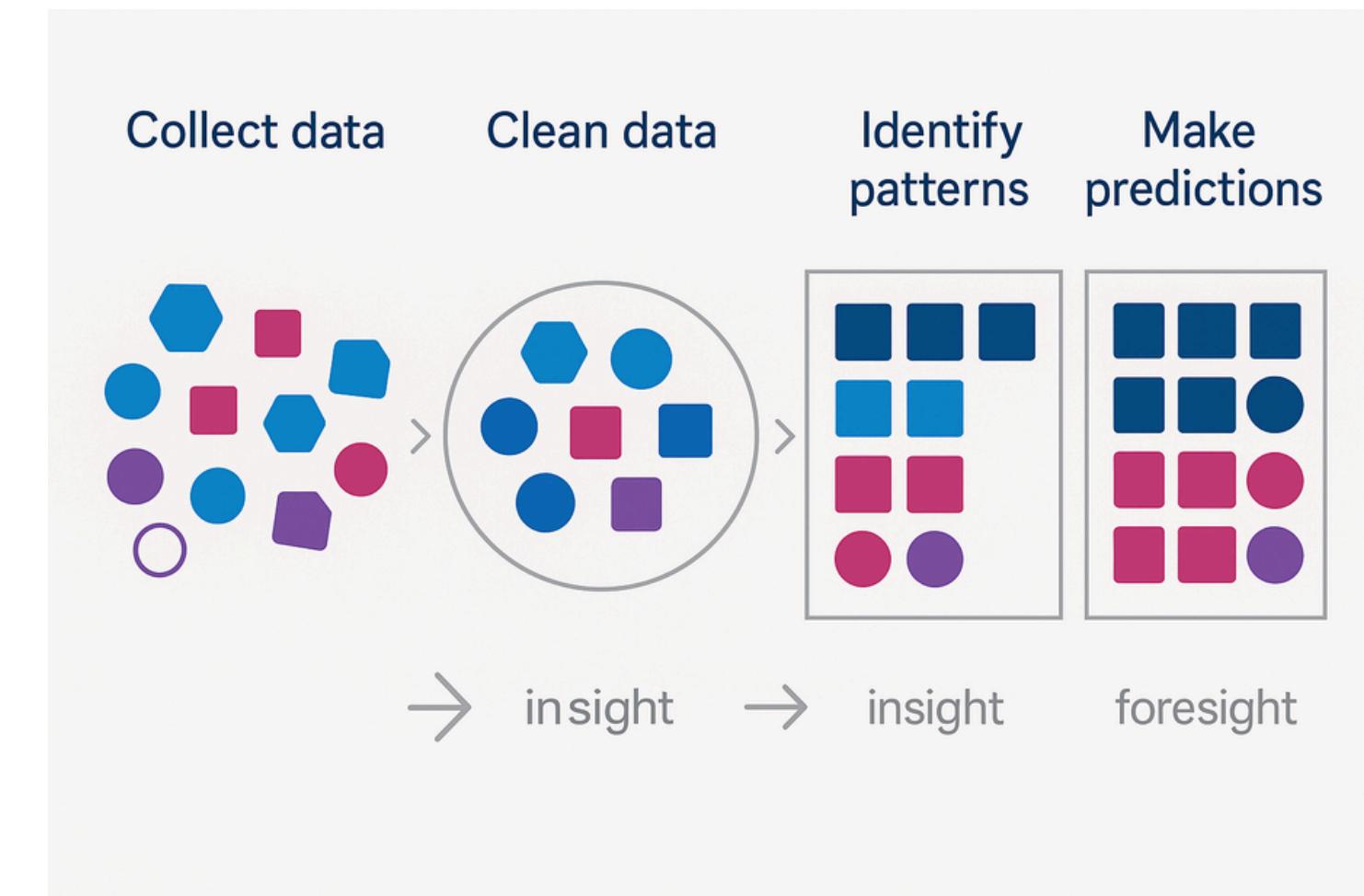
PENGENALAN DATA SCIENCE

PENGOLAHAN DATA YANG DILAKUKAN OLEH DATA SCIENTIST DIDASARKAN PADA KOMBINASI BIDANG KEILMUAN:

1. MATEMATIKA ATAU STATISTIKA,
2. COMPUTER SCIENCE, DAN
3. DOMAIN KNOWLEDGE ATAU EXPERTISE.



INSIGHT



beberapa cara yang dapat dijadikan bantuan menemukan insight diantaranya:

- 1. Melakukan Analisa Informasi Deskriptif**
- 2. Melakukan Analisa Hubungan Informasi Diagnostik**
- 3. Melakukan Teknik Statistika Deskriptif dan Uji Statistika**

MEMBANGUN ALGORITMA PREDIKSI

Tantangan besar bagi Data Scientist tentunya membangun sebuah algoritma prediksi berdasarkan hubungan keterkaitan tiap informasi didalam data yang sebelumnya telah dikumpulkan serta digabungkan. Data Scientist dalam membangun algoritma prediksi dapat didasarkan dalam menerapkan penggunaan teori dan aturan konsep:

1. classical statistic,
2. machine learning, atau
3. human-centered modelling.

DATA FUNDAMENTALS



DATA FUNDAMENTALS

1. **Volume**: Volume mengacu pada skala data yang sangat besar. Tantangan dalam Volume adalah pembuatan infrastruktur penyimpanan dan pemrosesan data yang mampu menangani skala besar secara efisien.
2. **Velocity**: Velocity merujuk pada kecepatan data yang dihasilkan, diproses, dan dianalisis. Tantangan dalam Velocity adalah pembuatan sistem yang mampu menangkap dan memproses data secara cepat.

DATA FUNDAMENTALS

3. **Variety:** Variety menunjukkan keragaman jenis data yang dihasilkan, mulai dari terstruktur sampai yang tidak terstruktur.
4. **Veracity:** Veracity menggambarkan tingkat kepercayaan terhadap data atau keakuratan data. Tantangan dalam Veracity adalah penyaringan, pembersihan dan validasi data.
5. **Value:** Value adalah manfaat atau nilai yang didapat dari data. Tantangan dalam Value adalah pengubahan data menjadi informasi yang dapat digunakan.

DATA ANALYSIS TOOLS

PROGRAMMING LANGUAGES

PYTHON

1. Python adalah bahasa pemrograman serbaguna yang banyak digunakan dalam analisis data karena library seperti Pandas, NumPy, dan SciPy untuk manipulasi data.
2. Python juga mendukung visualisasi dengan Matplotlib dan Seaborn, serta memiliki library machine learning seperti Scikit-learn, TensorFlow, dan PyTorch.
3. Use case: Exploratory Data Analysis (EDA), visualisasi, analitik prediktif

DATA ANALYSIS TOOLS

PROGRAMMING LANGUAGES

R

1. R adalah bahasa pemrograman yang dirancang khusus untuk statistik dan analisis data. R memiliki pustaka seperti ggplot2 untuk visualisasi dan dplyr untuk manipulasi data.
2. Kelebihan: Sangat kuat untuk analisis statistik dan pembuatan grafik fleksibel.
3. Use case: Analisis statistik yang kompleks, pembuatan grafik publikasi, dan visualisasi data.

DATA ANALYSIS TOOLS

VISUALIZATION TOOLS

Matplotlib

Library Python untuk membuat grafik 2D.

PowerBI

Alat analitik dari Microsoft untuk membuat laporan.

Google Data Studio/Looker Studio

Alat visualisasi data berbasis drag-and-drop untuk membuat dashboard tanpa perlu coding

DATA ANALYSIS TOOLS

MACHINE LEARNING PLATFORM

Scikit Learn

- Python yang digunakan untuk Machine Learning
- Pembuatan model prediktif seperti regresi, klasifikasi, clustering

Keahlian Data Scientist

Keahlian Data Scientist:

- 1. Data Drilling**
- 2. Diagnostic**
- 3. Data Protection**

Data Drilling

Data Drilling:

Proses menelusuri data untuk mengungkap informasi terperinci. Informasi ini digunakan untuk memahami alasan dari sebuah tren atau anomali.

Tujuan Data Driling:

1. Mendukung analisis penyebab utama dengan memecah metrik yang kompleks
2. Meningkatkan pengetahuan dengan memberikan informasi yang lebih mendalam
3. Memungkinkan pelaporan interaktif tempat pengguna dapat menjelajahi data secara dinamis

Konsep Data Drilling

Drill-Down vs Drill-Up:

- Drill-Down: Bergerak dari ringkasan ke detail
- Drill-Up: Mengagregasi data ke tingkat yang lebih tinggi

Hierarchical Structures:

- Umum ditemukan dalam OLAP dan dashboard BI
- Data Drilling berdasarkan waktu atau geografi

Interactive Dashboards:

- Alat seperti Tableau, Power BI, dan Looker memungkinkan grafik dan bagan untuk drilling
- Mengklik elemen untuk tampilan yang lebih mendalam
- Filter dan Slicer untuk melihat data secara seksama dan interaktif

DIAGNOSTIC

1. Exploratory Data Analysis (EDA):

- Menggunakan ringkasan statistik, visualisasi, dan plot untuk mengeksplorasi data.
- Identifikasi outliers, skewness, dan missing values.

2. Correlation Analysis:

- Mengukur hubungan linier antara variabel.
- Membantu mengidentifikasi faktor-faktor penyebab yang potensial.

3. Segmentation & Comparison:

- Bagi data menjadi beberapa kelompok (misalnya berdasarkan jenis pelanggan, waktu, atau wilayah geografis).
- Bandingkan metrik antar segmen untuk melihat perbedaan.

4. Time Series Diagnostics:

- Analisis tren, musiman, dan titik perubahan dari waktu ke waktu.
- Alat bantu: rata-rata bergerak, dekomposisi, dan deteksi anomali.

Data Protection

Data Protection:

Perlindungan data pribadi, sensitif, atau penting dari akses yang tidak sah, penyalahgunaan, kehilangan, atau kerusakan. Hal ini memastikan privasi dan keamanan

Tujuan Data Protection:

1. Melindungi kepercayaan pengguna dan reputasi perusahaan
2. Mencegah pelanggaran data, penipuan, dan kerugian finansial
3. Memastikan kepatuhan terhadap standar hukum dan industri

Data Protection

Prinsip Data Protection:

1. Confidentiality: Hanya orang berwenang yang dapat mengakses data
2. Integrity: Data harus akurat dan tidak diubah tanpa izin
3. Availability: Data harus dapat diakses oleh mereka yang membutuhkannya saat mereka membutuhkannya
4. Accountability: Organisasi harus menunjukkan penanganan data yang bertanggung jawab

Standar Hukum Data Protection:

1. General Data Protection Regulation (GDPR) - EU
2. California Consumer Privacy Act (CCPA)
3. Health Insurance Portability and Accountability Act (HIPAA) - USA
4. ISO/IEC 27001

SQL vs NoSQL

SQL

SQL (Structured Query Language)

SQL adalah bahasa untuk mengelola database relasional, dimana data disimpan dalam tabel dengan format terstruktur.

Contoh SQL: MySQL, PostgreSQL, SQLite

Use Case:

- Data Terstruktur: Data pelanggan atau transaksi penjualan
- Sistem Transaksional: Aplikasi yang membutuhkan konsistensi data tinggi

NoSQL

NoSQL (Non-Structured Query Language)

NoSQL adalah database yang menyimpan data tidak terstruktur atau semi-terstruktur, dengan fleksibilitas skema yang tinggi.

Contoh SQL: MongoDB, Cassandra, Couchbase

Use Case:

- Data Tidak Terstruktur: log, dokumen, JSON
- Sistem Skalabilitas Tinggi: Aplikasi yang membutuhkan skala besar

SESI 5

BASIC SQL

SCOPE OF SQL

SQL (Structure Query Language) merupakan Standar bahasa pengolahan data.

Jenis Perintah SQL :

- a. Data Definition Language (DDL)
- b. Data Manipulation Language (DML)
- c. Data Control Language (DCL)

SCOPE OF SQL

Data Definition Langauge

- Umum nya digunakan untuk mendefinisikan struktur data dalam database

Command dalam kategori DDL

1. CREATE: Membuat table
2. ALTER: Mengubah table
3. RENAME: mengubah nama table
4. DROP: menghapus table
5. SHOW: menampilkan table

SCOPE OF SQL

Data Manipulation Language

- Kumpulan Command dalam SQL yang berfungsi untuk memanipulasi data di dalam table di sebuah database

Command dalam kategori DML

1. INSERT: memasukan data baru ke dalam table
2. SELECT: Menampilkan data tertentu dari satu table atau lebih
3. UPDATE: Memperbarui data dalam suatu table
4. DELETE: Menghapus data dari table

SCOPE OF SQL

Data Control Language

- Umumnya digunakan untuk mengatur siapa yang punya akses untuk database tertentu

Command dalam kategori DCL

1. Grant: Memberi akses
2. Revoke: Menghapus akses

CONTOH APLIKASI BASIC SQL

Pembuatan table baru

```
CREATE TABLE users (
    username VARCHAR(50),
    email VARCHAR(100),
);
```

Insert data ke table

```
INSERT INTO users (username, email)
VALUES
    ('mirai', 'mirai@example.com'),
    ('sora', 'sora@example.com'),
    ('rei', 'rei@example.com');
```

CONTOH APLIKASI BASIC SQL

Update command

```
UPDATE users  
SET email =  
'mirai_new@example.com'  
WHERE username = 'mirai';
```

Alter table add column

```
ALTER TABLE users  
ADD COLUMN phone_number  
VARCHAR(20);
```



SESI 6

ADVANCED SQL

ADVANCED SQL

SQL tingkat lanjut merupakan perluasan dari kemampuan SQL dasar yang melibatkan teknik dan fitur yang lebih kompleks untuk mengelola, memanipulasi, dan menganalisis data. Jika SQL dasar berfokus pada kueri sederhana seperti SELECT, INSERT, UPDATE, dan DELETE, SQL tingkat lanjut mencakup konsep yang lebih mendalam untuk menangani skenario data yang lebih rumit.

SELECT DISTINCT

Perintah SQL untuk mengambil nilai unik saja. Jika ada nilai yang sama atau duplikat dalam tabel, SELECT DISTINCT hanya akan menampilkan nilai satu kali dalam hasil query.

```
SELECT DISTINCT column1, column2, ...  
FROM table_name;
```

```
1 SELECT DISTINCT job_title  
2 FROM discovrio_employees
```

job_title
Marketing Director
Web Developer
Manager
Project Manager
Sales Associate
Financial Analyst
Accountant
Data Analyst
HR Specialist
Sales Executive
Marketing Manager

WHERE

Perintah SQL untuk memfilter data berdasarkan kondisi tertentu. Dengan perintah SELECT, hasil query hanya akan menampilkan data yang memenuhi syarat pada kondisi atau kriteria tertentu dalam query.

SELECT column1, column2, ...

FROM table_name

WHERE condition;

```
1 SELECT first_name, hire_date, job_title  
2 FROM discovrio_employees  
3 WHERE job_title = 'Data Analyst'
```

first_name	hire_date	job_title
Charles	2023-07-08	Data Analyst
Samantha	2023-03-01	Data Analyst
Justin	2023-07-17	Data Analyst
Amanda	2024-07-05	Data Analyst

STRING FUNCTION

SQL functions to manipulate string data type.

Command		Definition
1	LOWER	Mengonversi data string menjadi huruf kecil.

Syntax:

```
1 SELECT first_name, LOWER(first_name)
2 FROM discovrio_employees
3 LIMIT 3
```

first_name	lower
John	john
Michael	michael
Sarah	sarah

Command		Definition
2	UPPER	Mengonversi data string menjadi huruf besar.

Syntax:

```
1 SELECT first_name, UPPER(first_name)
2 FROM discovrio_employees
3 LIMIT 3
```

first_name	upper
John	JOHN
Michael	MICHAEL
Sarah	SARAH

STRING FUNCTION

Command		Definition
3	LENGTH	Menghitung panjang karakter string.

Command		Definition
4	CONCAT	Menggabungkan beberapa string menjadi satu.

Syntax:

```
1 SELECT first_name, LENGTH(first_name)
2 FROM discovrio_employees
3 LIMIT 3
```

first_name ▲ length ▲

John	4
------	---

Michael	7
---------	---

Sarah	5
-------	---

```
1 SELECT first_name, last_name, CONCAT(first_name, ' ', last_name)
2 FROM discovrio_employees
3 LIMIT 3
```

first_name	last_name	concat
John	Doe	John Doe
Michael	Johnson	Michael Johnson
Sarah	Martinez	Sarah Martinez

Command		Definition
4	SUBSTRING	Mengekstrak atau memotong beberapa karakter.

```
1 SELECT first_name, SUBSTRING(first_name, 1,2)
2 FROM discovrio_employees
3 LIMIT 3
```

first_name	substring
John	Jo
Michael	Mi
Sarah	Sa

AGGREGATE FUNCTIONS

- **AVG()**

Fungsi untuk mencari **rata-rata** dari data di tabel

- **COUNT()**

Fungsi untuk menghitung **berapa banyak** data di suatu tabel

- **MAX()**

Fungsi untuk mencari **nilai terbesar** dari sejumlah data di tabel.

- **MIN()**

Fungsi untuk mencari **nilai terkecil** dari sejumlah data di tabel.

- **SUM()**

Fungsi untuk mencari **total** dari sejumlah data di tabel.

AGGREGATE FUNCTIONS

GROUP BY

Fungsi untuk mengelompokkan data berdasarkan kolom tertentu.

Fungsi **GROUP BY** biasa dikombinasikan dengan fungsi *aggregate* lainnya.

Struktur dasar fungsi **GROUP BY**:

SELECT daftar_column

FROM nama_table

GROUP BY column1, column2, columnN

AGGREGATE FUNCTIONS

GROUP BY

Contoh penerapan fungsi **GROUP BY** dikombinasikan dengan *aggregate functions*:

```
SELECT kota_asal, AVG(salary) AS avg_salary  
FROM raihan_ds49  
GROUP BY kota_asal;
```

```
SELECT kota_asal, COUNT(kota_asal) AS count_ppl  
FROM table_ds49  
GROUP BY kota_asal;
```

```
SELECT kota_asal, MAX(salary) AS max_salary  
FROM table_ds49  
GROUP BY kota_asal;
```

```
SELECT kota_asal, MIN(salary) AS max_salary  
FROM table_ds49  
GROUP BY kota_asal;
```

```
SELECT kota_asal, SUM(salary) AS total_salary  
FROM table_ds49  
GROUP BY kota_asal;
```

JOIN TABLE

Fungsi untuk **menggabungkan data** dari dua table berbeda yang berelasi menjadi satu table sementara.

Ada **tiga tipe JOIN**, yaitu:

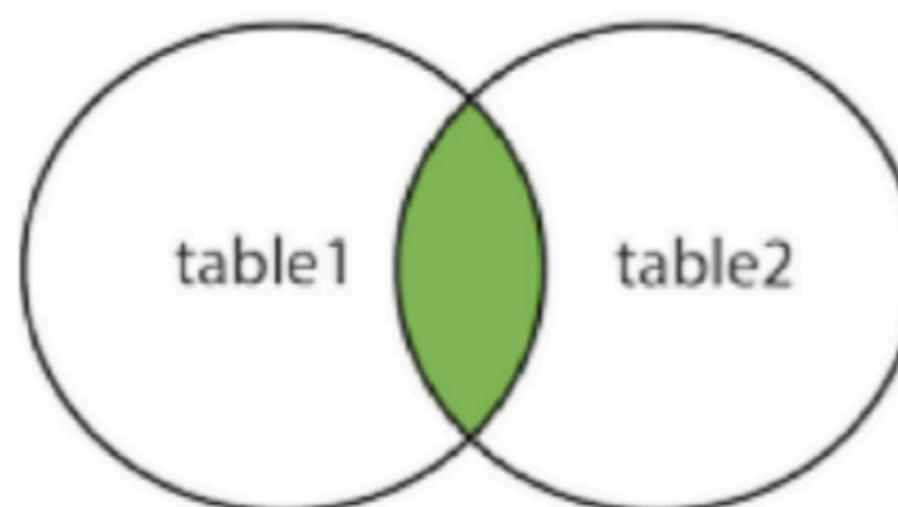
- **JOIN** atau **INNER JOIN**
- **LEFT JOIN**
- **RIGHT JOIN**

JOIN TABLE

JOIN/INNER JOIN

JOIN atau **INNER JOIN** adalah fungsi untuk **menggabungkan data berelasi** dari dua table yang **sama-sama memiliki pasangan di kedua table tersebut**.

Ilustrasi JOIN/INNER JOIN:



Contoh Query untuk JOIN/INNER JOIN:

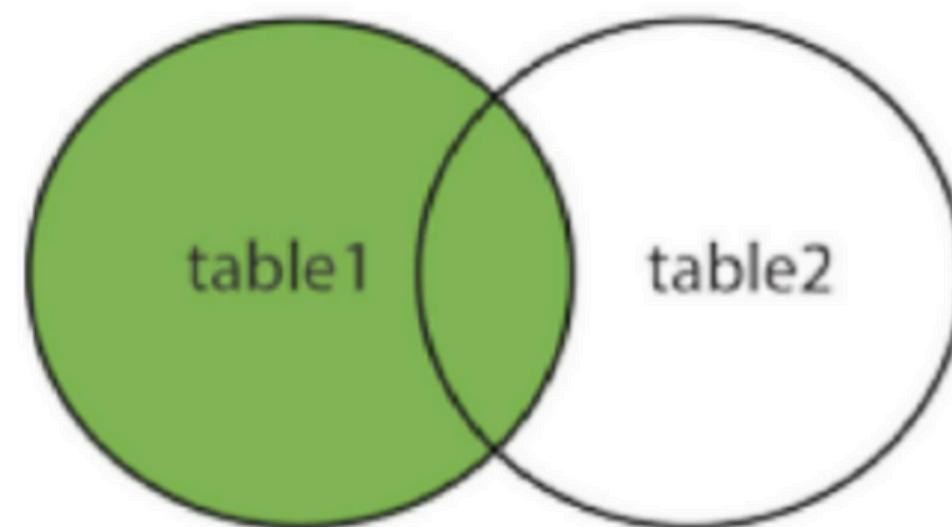
```
SELECT t1.column1, t1.column2, t2.column3, t2.column4  
FROM nama_table1 t1  
INNER JOIN nama_table2 t2 ON t1.column1 = t2.column1;
```

JOIN TABLE

LEFT JOIN

LEFT JOIN adalah fungsi untuk memunculkan **semua data dari table 1, meskipun tidak ada pasangannya di table 2**. Data table 2 yang kosong akan diberi nilai NULL.

Ilustrasi LEFT JOIN:



Contoh Query untuk LEFT JOIN:

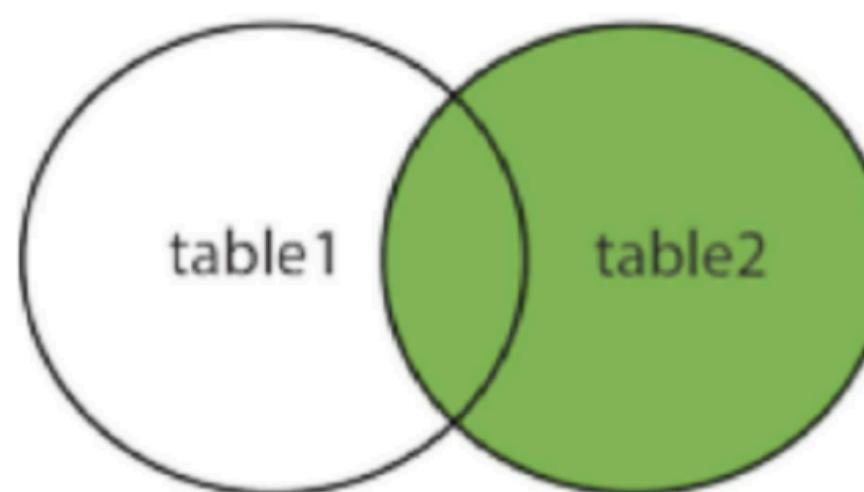
```
SELECT t1.column1, t1.column2, t2.column3, t2.column4  
FROM nama_table1 t1  
LEFT JOIN nama_table2 t2 ON t1.column1 = t2.column1;
```

JOIN TABLE

RIGHT JOIN

RIGHT JOIN adalah fungsi untuk memunculkan **semua data dari table 2, meskipun tidak ada pasangannya di table 1**. Data table 1 yang kosong akan diberi nilai NULL.

Ilustrasi RIGHT JOIN:



Contoh Query untuk RIGHT JOIN:

```
SELECT t1.column1, t1.column2, t2.column3, t2.column4  
FROM nama_table1 t1  
RIGHT JOIN nama_table2 t2 ON t1.column1 = t2.column1;
```

SUBQUERIES

Subquery, Inner Query, atau Nested Query adalah *query* yang terletak di dalam *query*.
Bertujuan untuk lebih menspesifikasi data yang ingin ditampilkan

Ada **tiga tipe Subquery**, yaitu:

- ***Subqueries in Filter***
- ***Subqueries in Table***
- ***Common Table Expression (CTE)***

SUBQUERIES

SUBQUERIES IN FILTER

Subquery yang diletakkan di dalam fungsi filter, seperti WHERE.

Contoh *subquery in filter* untuk mencari kota_asal dengan salary di atas rata-rata:

```
SELECT * FROM table_ds49 ORDER BY user_id;  
SELECT AVG(salary) as avg_salary FROM table_ds49 WHERE kota_asal = 'Jakarta';  
SELECT nama_lengkap, salary, kota_asal FROM table_ds49  
WHERE (  
    salary > (SELECT AVG(salary)  
        FROM table_ds49  
    )  
);
```

SUBQUERIES

SUBQUERIES IN TABLE

Subquery yang berupa query untuk menampilkan table lainnya.

Contoh *subquery in filter* untuk mencari rata-rata *salary* tertinggi berdasarkan kota asal:

```
SELECT MAX(avg_salary) AS max_avg_salary
FROM(
    SELECT kota_asal,
    AVG(salary) AS avg_salary
    FROM raihan_ds49
    GROUP BY kota_asal
);
```

SUBQUERIES

COMMON TABLE EXPRESSION (CTE)

CTE adalah cara untuk membuat *subquery* sementara yang bisa digunakan dalam query utama. CTE biasanya digunakan untuk membuat kode lebih mudah dibaca dan dikelola, terutama ketika ada query kompleks atau rekursif.

Contoh:

WITH salaries **AS**(

```
SELECT t1.user_id, t1.nama_lengkap, t1.salary, t2.universitas, t2.pendidikan  
FROM raihan_ds49 t1 INNER JOIN raihan_ds49_details t2  
ON t1.user_id = t2.user_id  
WHERE pendidikan = 'S1' OR pendidikan = 'S2'  
)
```

SELECT * FROM salaries;



THANK YOU