INTRO TO SQL

Muhammad Saipul Rohman

ABOUT ME

- Data Engineer in PT. Xapiens Teknologi Indonesia
- IT Trainer, Ex-Lecturer
- More than 10 years experience in IT
- More than 3 years experience in Data Engineer
- LinkedIn : https://www.linkedin.com/in/muhammad-saipul-r/

AGENDA

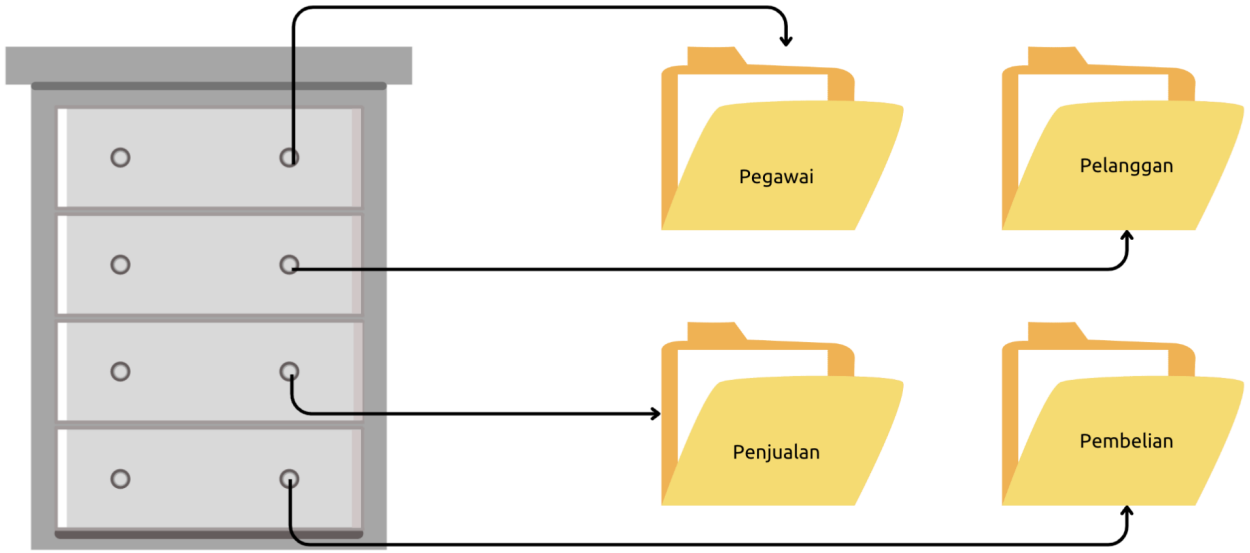
- Intro Database
- Intro Database Management System
- Data Definition Language
- Data Manipulation Language
- Data Query Language
- Hands on materi DDL, DML dan DQL

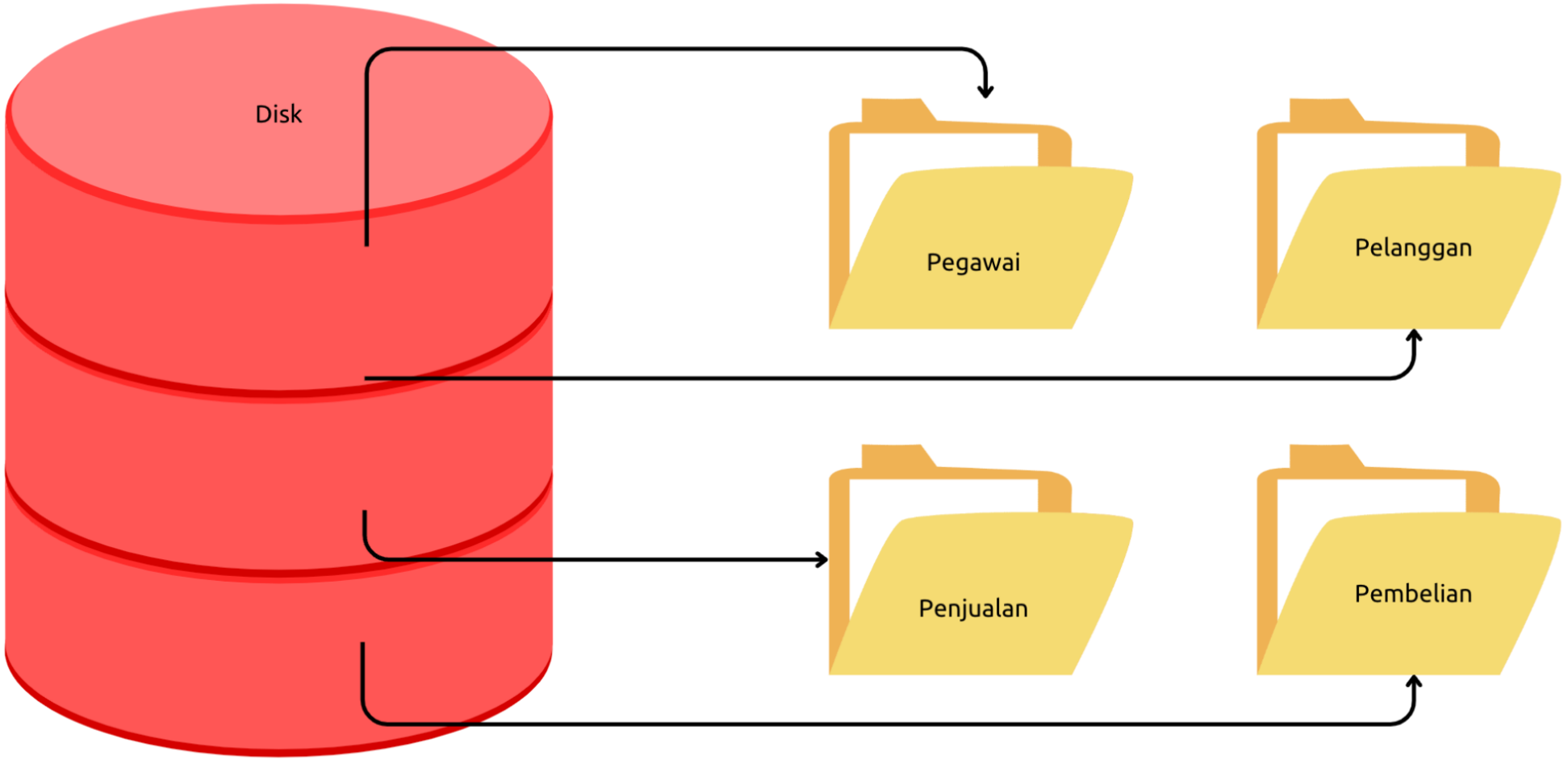
Notes : Materi ini mulai dari syntax DDL sampai DQL menggunakan syntax yang support di PostgreSQL

INTRO DATABASE

APA ITU DATABASE ?

- Database adalah tempat penyimpanan data yang terstruktur
- Database bisa diibaratkan sebagai lemari arsip
- Tujuan nya agar mudah mencari data




Disk
Pegawai
Pelanggan
Penjualan
Pembelian

OPERASI DATABASE

Operasi Database meliputi Membuat, Membaca, Mengubah dan Menghapus data atau disebut juga dengan CRUD(Create,Read,Update,Delete)

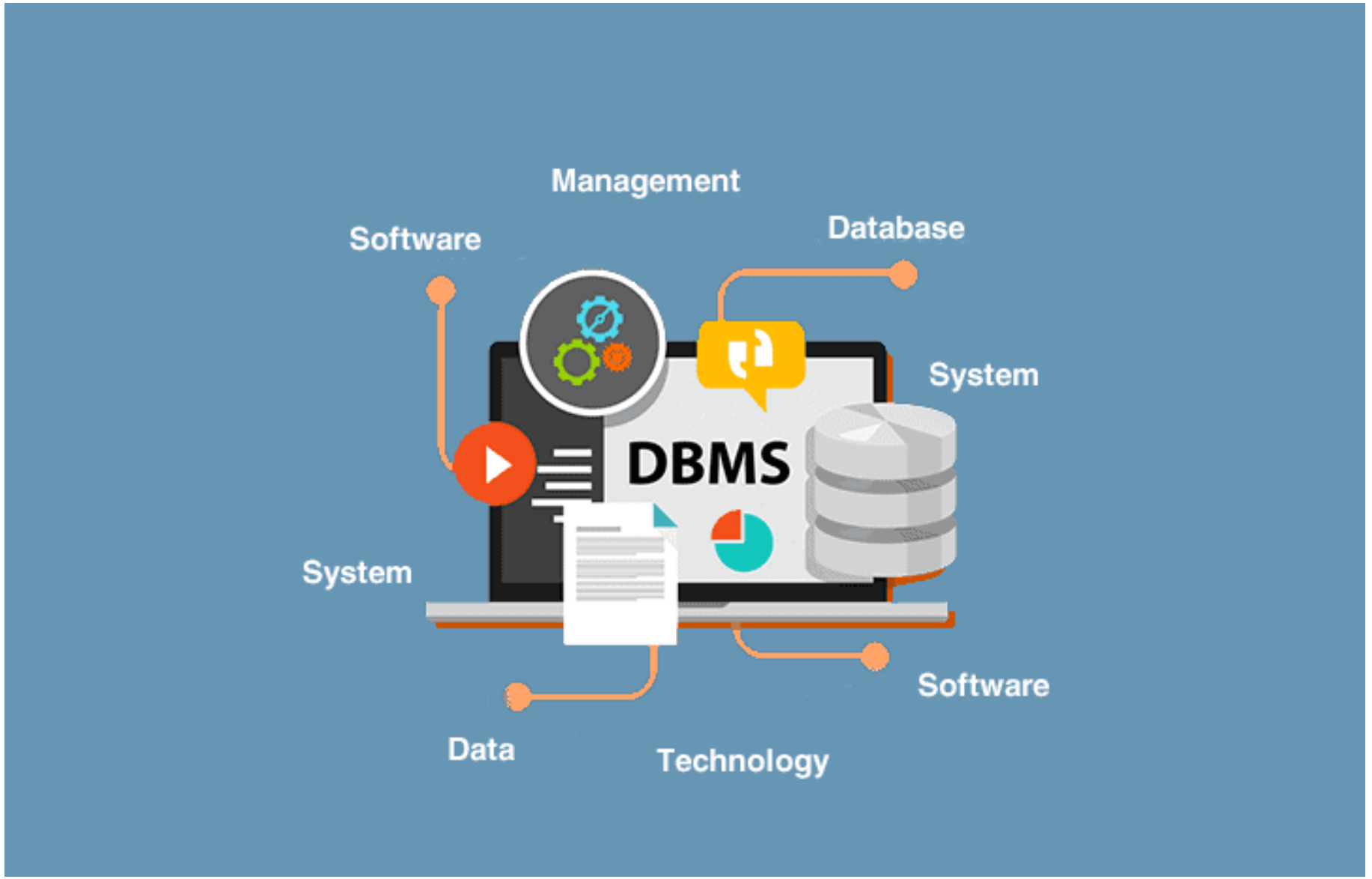
CONTOH PENERAPAN DATABASE

- Bank, dalam mengelola data nasabah, tabungan, transaksi dll
- TokoOnline, dalam mengelola data pelanggan, penjualan, barang, pembelian dll

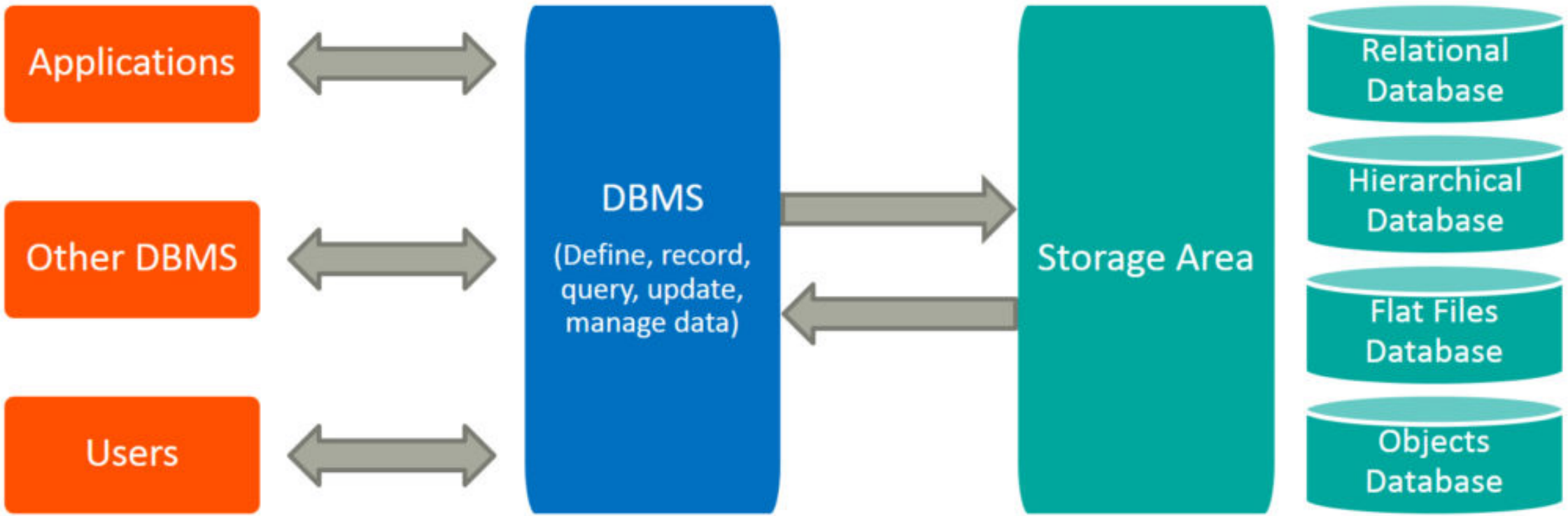
INTRO DBMS

APA ITU DBMS ?

- DBMS atau Database Management System adalah software yang mengelola data.
- Tanpa DBMS, untuk mengelola data seperti data produk, data penjualan dst, kita harus simpan dalam bentuk file misal dalam bentuk Excel.
- DBMS biasanya di install di Komputer server dan untuk mengakses DBMS nya, kita perlu menginstall software DMBS client nya seperti DBeaver(untuk semua DBMS), PgAdmin(Hanya PostgreSQL), SQLWorkBench(Hanya MySQL).
- Contoh DBMS popular yaitu SQL Server, MySQL, PostgreSQL, Oracle, DB2, dst

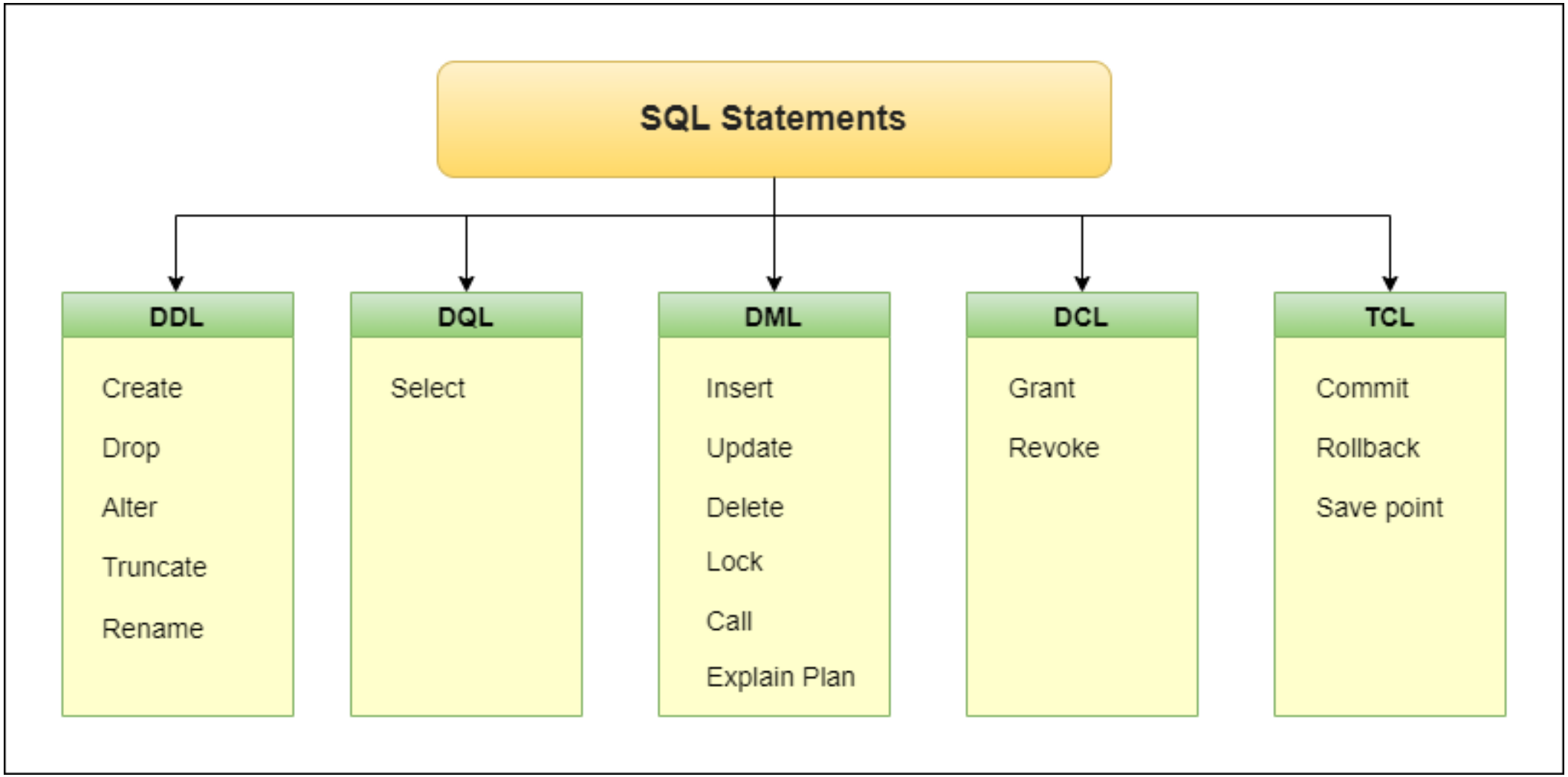

Management
Software
Database
System
DBMS
System
Software
Data
Technology

Database Management System



BAHASA DATABASE

- Data Definition Language (DDL)
- Data Manipulation Language(DML)
- Data Query Language (DQL)
- Data Control Language (DCL) ← tidak di bahas di workshop ini
- Transaction Control Language (TCL) ← tidak di bahas di workshop ini


SQL Statements
DDL
Create
Drop
Alter
Truncate
Rename
DQL
Select
DML
Insert
Update
Delete
Lock
Call
Explain Plan
DCL
Grant
Revoke
TCL
Commit
Rollback
Save point

DATA DEFINITION LANGUAGE

CREATE DATABASE

CREATE adalah perintah untuk membuat database dan objek nya(table, view, index, function, dst)

Syntax :

CREATE DATABASE namadatabase

Contoh :

CREATE DATABASE mydb

Bisa juga dengan menggunakan syntax CREATE DATABASE IF NOT EXISTS ← MySQL Only

Untuk mencoba di postgres bisa refer ke link ini

CREATE TABLE

Syntax

```
CREATE TABLE table_name (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);
```

Example :

```
CREATE TABLE Student (
    StudendId int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    Mark int
);
```

CREATE TABLE

Bisa juga dengan menggunakan syntax

- CREATE TABLE IF NOT EXISTS nama_table

Example :

```
CREATE TABLE IF NOT EXISTS Student (
    StudendId int,
    LastName varchar(255),
    FirstName varchar(255),
    Address varchar(255),
    Mark int
);
```

- CREATE OR REPLACE ← hanya support view table

DROP DATABASE

DROP adalah perintah untuk menghapus database dan objeknya(table, view, function, dst)

Syntaks :

DROP DATABASE [IF EXISTS] database_name

Untuk mencegah error jika tidak ada database maka gunakan syntax IF EXISTS

Example :

DROP DATABASE testDB;

DROP DATABSE IF EXISTS testDB;

DROP TABLE

Syntaks :

DROP TABLE [IF EXISTS] table_name [CASCADE | RESTRICT]

- Gunakan IF EXISTS untuk menghindari error jika tidak ada table.
- Opsi CASCADE digunakan untuk menghapus table dan depedensi di table lain.
- OPSI RESTRICT sudah default digunakan dengan tujuan tidak bisa menghapus table jika ada depedensi di table lain

Example :

```
DROP TABLE IF EXISTS Shippers;
```

```
DROP TABLE IF EXISTS Shippers CASCADE;
```

ALTER TABLE

Adalah perintah untuk memodifikasi table misalnya menambah kolom baru, mengubah tipe data kolom, mengganti nama kolom atau menghapus kolom. Alter table juga digunakan untuk menambah dan menghapus constraint(Primary Key, Foreign Key, Unique Key, dst).

- ALTER TABLE ADD COLUMN
- ALTER TABLE DROP COLUMN
- ALTER TABLE RENAME COLUMN
- ALTER TABLE ALTER DATATYPE

ALTER TABLE ADD COLUMN

Perintah untuk menambah kolom di table.

Syntaks :

```
ALTER TABLE table_name
ADD COLUMN column_name datatype constraint;
```

Example :

```
ALTER TABLE Persons
ADD COLUMN DateOfBirth date;
```

```
ALTER TABLE Persons
ADD COLUMN Address VARCHAR NOT NULL;
```

ALTER TABLE DROP COLUMN

Perintah untuk menghapus kolom di table

Syntaks:

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

Example:

```
ALTER TABLE links
DROP COLUMN active;
```

ALTER TABLE RENAME COLUMN

Perintah untuk mengganti nama kolom di table

Syntaks :

```
ALTER TABLE table_name
RENAME COLUMN column_name
TO new_column_name;
```

Example :

```
ALTER TABLE links
RENAME COLUMN title TO link_title;
```

ALTER TABLE ALTER COLUMN

Perintah untuk mengganti tipe data kolom di table

Syntaks :

```
ALTER TABLE table_name
ALTER COLUMN column_name TYPE new_data_type;
```

Example :

```
ALTER TABLE assets
ALTER COLUMN name TYPE VARCHAR;
```

TRUNCATE TABLE

Untuk menghapus semua data di table, bisa menggunakan perintah DELETE. Tetapi jika data di table sudah banyak menggunakan perintah DELETE tidak efisien. Untuk case ini bisa menggunakan perintah TRUNCATE.

Syntaks :

```
TRUNCATE TABLE table_name;
```

Untuk menghapus lebih dari satu table menggunakan syntax :

```
TRUNCATE TABLE
    table_name1,
    table_name2,
    ...;
```

DATA MANIPULATION LANGUAGE

INSERT

Perintah untuk menambahkan data ke table

Syntaks dasar :

```
INSERT INTO table_name(column1, column2, ...)
VALUES (value1, value2, ...);
```

Untuk menampilkan data yang di tambahkan ke table bisa menggunakan syntax :

```
INSERT INTO table_name(column1, column2, ...)
VALUES (value1, value2, ...)
RETURNING *;
```

Example :

```
INSERT INTO links (url, name)
VALUES('https://www.postgresqltutorial.com','PostgreSQL Tutorial');
```

UPDATE

Perintah untuk mengubah data di table

Syntaks:

```
UPDATE table_name
SET column1 = value1,
    column2 = value2,
    ...
WHERE condition;
```

Example :

```
UPDATE courses
SET published_date = '2020-08-01'
WHERE course_id = 3;
```

DELETE

Untuk menghapus data di table

Syntaks :

- Untuk menghapus semua data di table

```
DELETE FROM table_name
```

- Untuk menghapus data menggunakan filter

```
DELETE FROM table_name
WHERE condition;
```

- Untuk menghapus data dan menampilkan data yang dihapus ke layer

```
DELETE FROM table_name
WHERE condition
RETURNING *;
```

DATA QUERY LANGUAGE

SELECT DATA

- Perintah untuk menampilkan data di table.
- SELECT bisa digunakan untuk mengambil semua kolom atau sebagian kolom

Syntaks :

SELECT * FROM table_name;

SELECT column_name FROM table_name;

WHERE CLAUSE

- Digunakan bersamaan dengan SELECT untuk memfilter pada saat melakukan pencarian data dari table

Syntaks

```
SELECT * FROM table_name
WHERE column_name = 'xx';
```

ORDER BY CLAUSE

- Digunakan untuk mengurutkan data Ketika kita menggunakan perintah SQL SELECT.
- ORDER BY clause digunakan untuk mengurutkan data berdasarkan kolom yang dipilih dan jenis urutan (ASC atau DESC)

```
SELECT * FROM employee
ORDER BY join_date DESC
```

LIMIT CLAUSE

- Digunakan untuk membatasi pengambilan jumlah data saat menggunakan SQL SELECT.
- Selain membatasi jumlah data, kita juga bisa meng-skip sejumlah data yang tidak ingin kita lihat.
- LIMIT biasanya digunakan saat melakukan paging di aplikasi kita dengan kombinasi OFFSET

```
SELECT * FROM products
WHERE price > 0
LIMIT 10;
```

SELECT DISTINCT DATA

Untuk menampilkan data yang unik, kita bisa menggunakan SELECT dengan tambahan DISTINCT sebelum nama kolom nya.

```
SELECT DISTINCT age
FROM people;
```

AGGREGATE FUNCTION

- Salah satu fungsi SQL untuk menampilkan data rata-rata, terkecil, terbesar, jumlah data atau total data.
- Fungsi aggregrate yang common yaitu COUNT, MAX, MIN, SUM, AVG

```sql
SELECT COUNT(id) AS "Total Product" FROM products;

SELECT AVG(price) AS "Rata-Rata Harga" FROM products;

SELECT MAX(price) AS "Harga Termahal" FROM products;

SELECT MIN(price) AS "Harga Termurah" FROM products;
```

GROUP BY

Digunakan bersamaan dengan fungsi aggregrate untuk grouping berdasarkan kriteria tertentu.

Contoh Ketika ingin grouping data category dari table products

```
SELECT category,
       COUNT(id) as "Total Product"
FROM products
GROUP BY category;
```

HAVING CLAUSE

HAVING CLAUSE digunakan untuk filter data yang sudah digrouping atau di gunakan fungsi aggregate.

Contoh :

```
SELECT customer_id, SUM (amount)
FROM payment
GROUP BY customer_id;
HAVING SUM(amount) > 50
```

	customer_id smallint	sum numeric
1	184	80.80
2	87	137.72
3	477	106.79
4	273	130.72
5	550	151.69
6	51	123.70
7	394	77.80
8	272	65.87
9	70	75.83
10	190	102.75
11	350	63.79

JOIN

- JOIN digunakan untuk melakukan query ke lebih dari satu table
- Tetapi perlu di ingat jika terlalu banyak melakukan JOIN maka akan mengakibatkan proses query semakin lambat dan berat.
- Idealnya Ketika melakukan JOIN jangan lebih dari 5 tabel

```sql
SELECT *
FROM wishlist
         JOIN products ON products.id = wishlist.id_product;

SELECT products.id, products.name, wishlist.description
FROM wishlist
         JOIN products ON products.id = wishlist.id_product;
```

JOIN MULTIPLE TABLE

```sql
SELECT customer.email, products.id, products.name, wishlist.description
FROM wishlist
         JOIN products ON wishlist.id_product = products.id
         JOIN customer ON wishlist.id_customer = customer.id;
```

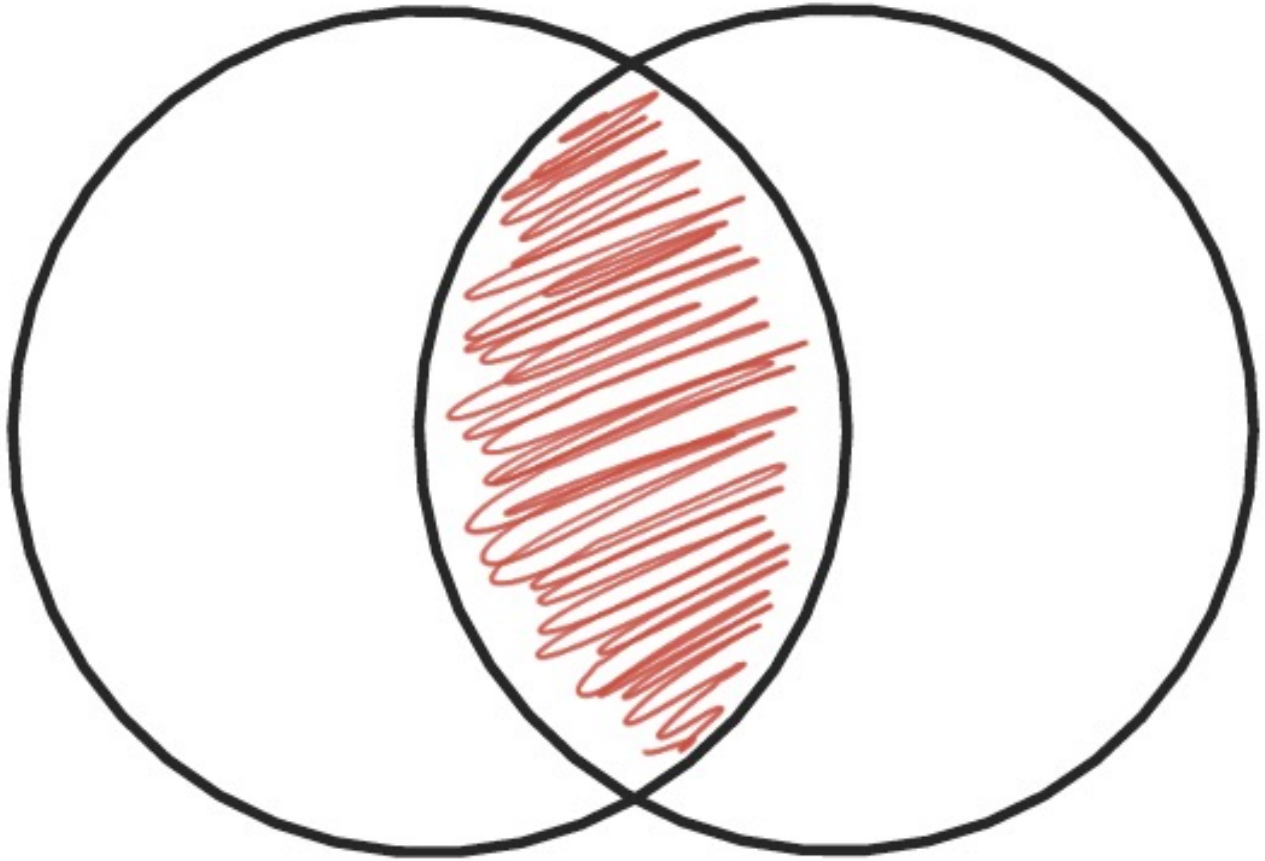
JENIS-JENIS JOIN

- INNER JOIN
- LEFT JOIN
- RIGHT JOIN
- FULL JOIN

INNER JOIN

- INNER JOIN adalah salah satu teknik JOIN yang menampilkan data yang ada di table pertama dan kedua
- Jika ada data di table pertama yang tidak memiliki relasi di table kedua atau sebaliknya, maka hasil INNER JOIN tidak akan ditampilkan.
- INNER JOIN adalah default dari JOIN di PostgreSQL

INNER JOIN DIAGRAM



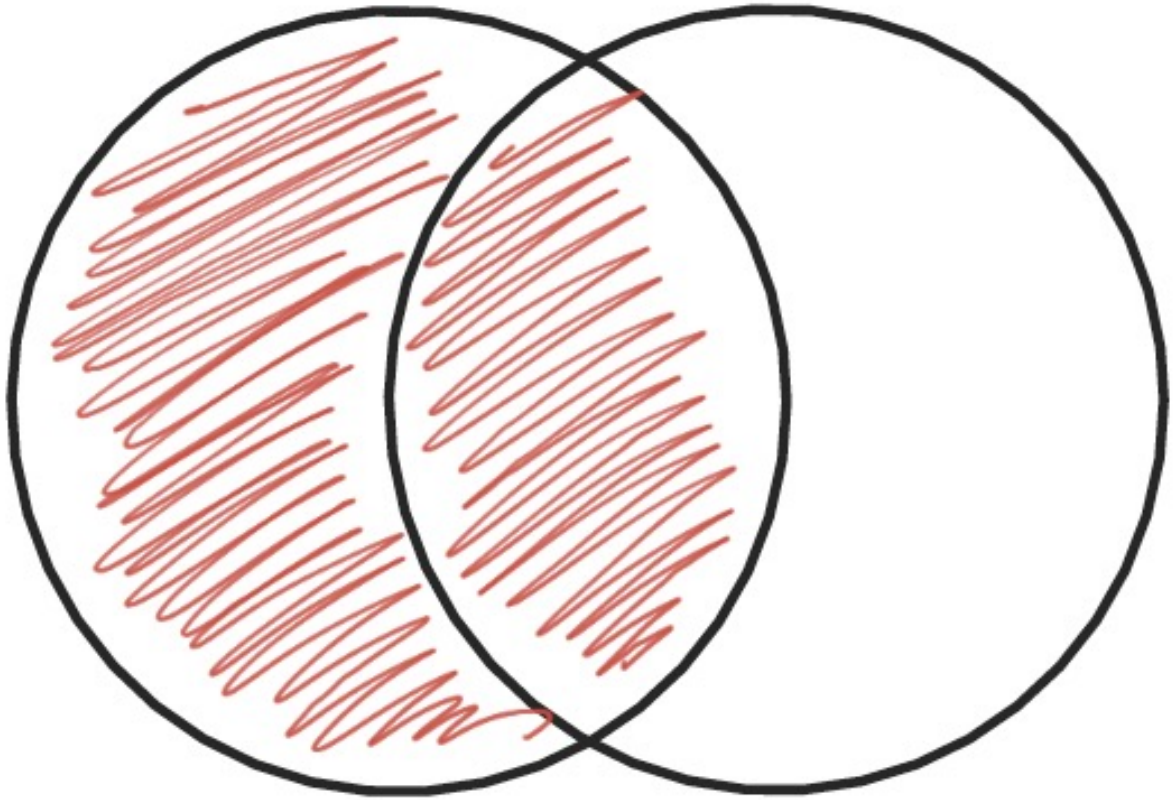
CONTOH INNER JOIN

```sql
SELECT *
FROM categories
         INNER JOIN products ON products.id_category = categories.id;
```

LEFT JOIN

- LEFT JOIN adalah salah satu teknik JOIN yang menampilkan semua data yang ada di table pertama
- Jika ada data yang tidak memiliki relasi di table kedua, maka hasilnya akan menampilkan NULL

LEFT JOIN DIAGRAM



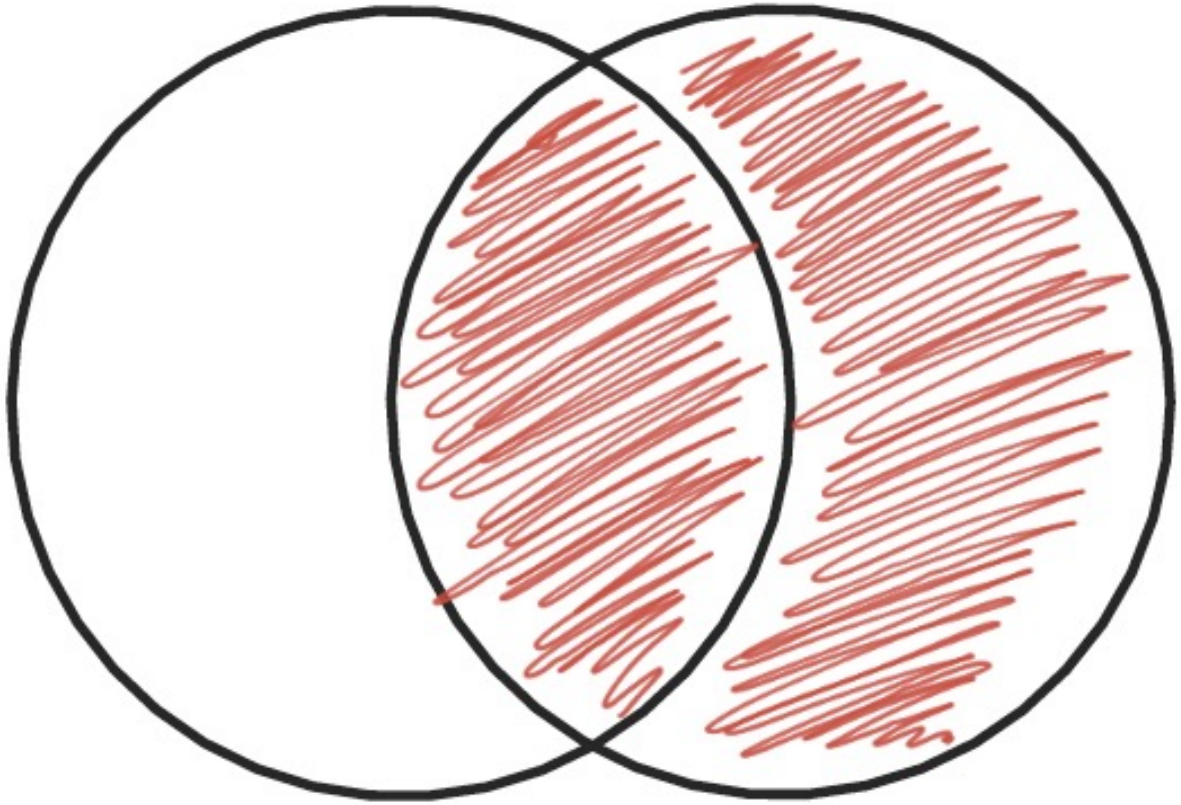
CONTOH LEFT JOIN

```sql
SELECT *
FROM categories
         LEFT JOIN products ON products.id_category = categories.id;
```

RIGHT JOIN

- RIGHT JOIN adalah salah satu teknik JOIN yang menampilkan semua data yang ada di table kedua
- Jika ada data yang tidak memiliki relasi di table pertama, maka hasilnya akan menampilkan NULL

RIGHT JOIN DIAGRAM



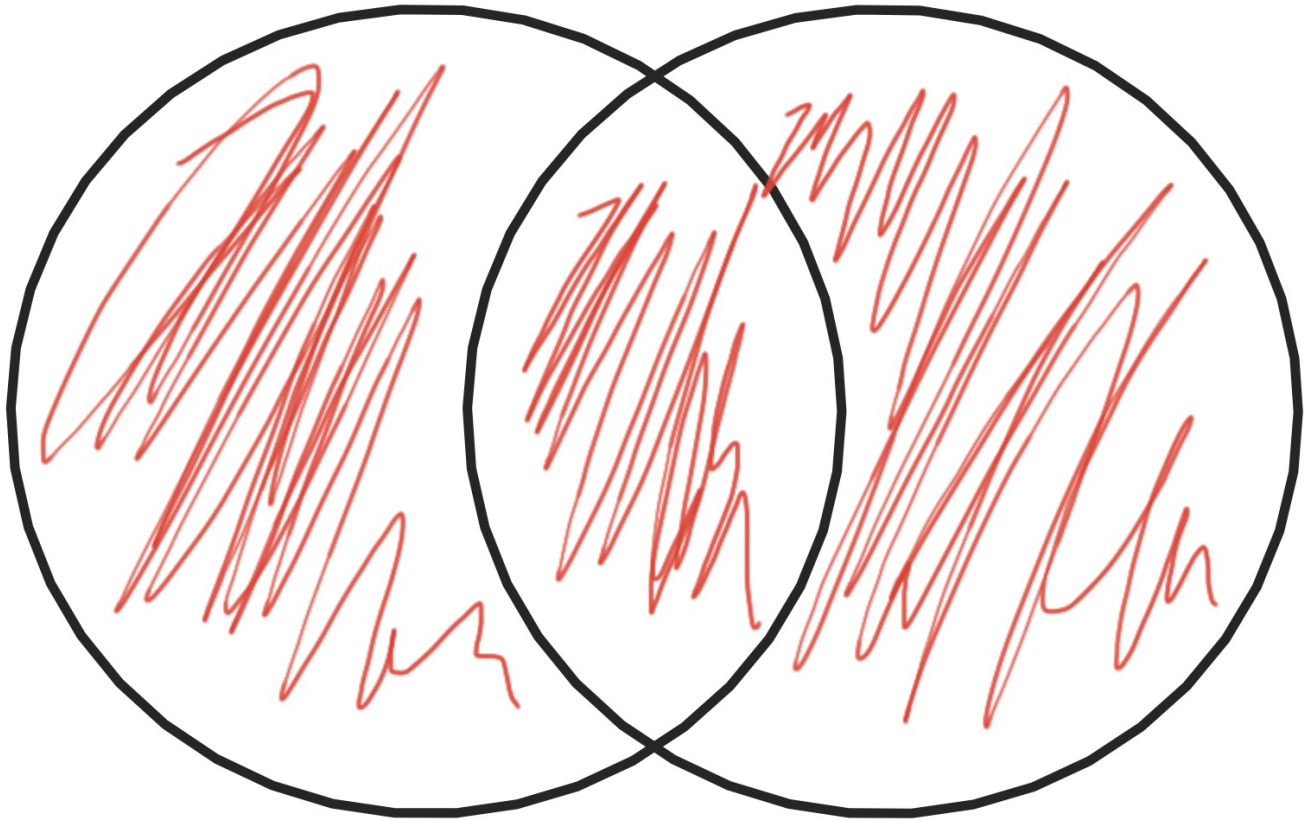
CONTOH RIGHT JOIN

```sql
SELECT *
FROM categories
         RIGHT JOIN products ON products.id_category = categories.id;
```

FULL JOIN

- FULL JOIN adalah salah satu Teknik JOIN untuk menampilkan semua data di table pertama dan kedua
- Jika tidak ada data join maka hasilnya akan berisi data NULL

FULL JOIN DIAGRAM



CONTOH FULL JOIN

```sql
SELECT *
FROM categories
         FULL JOIN products ON products.id_category = categories.id;
```

HANDS ON

Please click this link