DDL & CONSTRAINT

create table

```
CREATE TABLE orders (
        order_id SERIAL PRIMARY KEY,
        order_date DATE NOT NULL
);
        CREATE TABLE order_items (
        order_item_id SERIAL PRIMARY KEY,
        order_id INT NOT NULL,
        product_id INT NOT NULL,
        FOREIGN KEY (order_id) REFERENCES orders(order_id) ON DELETE CASCADE,
        FOREIGN KEY (product_id) REFERENCES products(product_id) ON DELETE RESTRICT
);
```

Dalam database, ada dua cara untuk menambahkan constraints: saat membuat tabel (CREATE TABLE) dan dengan mengubah tabel yang sudah ada (ALTER TABLE).

alter table

- Add column

```
ALTER TABLE Mahasiswa
Add angkatan INT;
```

- Drop column

```
ALTER TABLE Mahasiswa
drop angkatan;
```

- Rename column

```
ALTER TABLE Mahasiswa
RENAME COLUMN nama TO name;
```

- Alter column datatype

```
ALTER TABLE Mahasiswa
ALTER COLUMN jurusan TYPE varchar(2);
```

- NOT NULL: Memastikan bahwa kolom tidak boleh memiliki nilai NULL.
- UNIQUE: Memastikan bahwa semua nilai dalam kolom berbeda.
- CHECK: Memastikan nilai dalam kolom memenuhi kondisi tertentu.
- DEFAULT: Menetapkan nilai default untuk kolom jika tidak ada nilai yang ditentukan.
- CREATE INDEX: Digunakan untuk membuat dan mengambil data dari basis data dengan sangat cepat.

```
ALTER TABLE products ADD CHECK (name <> 'Bejo');
ALTER TABLE products ADD CONSTRAINT some_name UNIQUE (product_no);
ALTER TABLE products ADD FOREIGN KEY (product_group_id) REFERENCES product_groups;
ALTER TABLE example_table ADD CONSTRAINT pk_example_id PRIMARY KEY (id);
```

DML

insert

```
INSERT INTO table_name (col1, col2, col3)
VALUES (val1, val2, val3);

--ex
INSERT INTO cars (brand, model, year)
VALUES ('Ford', 'Mustang', 1964);

INSERT INTO cars (brand, model, year)
VALUES
  ('Volvo', 'p1800', 1968),
  ('BMW', 'M1', 1978),
  ('Toyota', 'Celica', 1975);
```

delete

```
DELETE FROM table_name
WHERE column_name = value;

--ex
DELETE FROM cars
WHERE brand = 'Volvo';
```

urutan hirearki

Select > From > Join > Where > Group By > Having > Order By

update

```
UPDATE table_name
SET column_name = value,
WHERE another_column = another_value;

--ex
UPDATE cars
SET color = 'red'
WHERE brand = 'Volvo';
```

group by & having

GROUP BY digunakan untuk mengelompokkan baris yang memiliki nilai yang sama dalam kolom yang ditentukan. HAVING digunakan untuk menyaring kelompok yang terbentuk oleh GROUP BY.

```
SELECT column1, COUNT(*)
FROM table
GROUP BY column1
HAVING COUNT(*) > 1;
```

order by

```
SELECT column1, column2
FROM table
ORDER BY column1 ASC, column2 DESC;
```

subquery

```
SELECT column1
FROM table
WHERE column2 IN (SELECT column2 FROM another_table WHERE condition);
```

copy table

```
CREATE TABLE employee_as AS table employees;
```

When

```
SELECT column1,
       CASE
           WHEN column2 > 100 THEN 'High'
           WHEN column2 BETWEEN 50 AND 100 THEN 'Medium'
           ELSE 'Low'
       END as category
FROM table;
```

like huruf akhir a

```
SELECT first_name
FROM employees
WHERE first_name LIKE '%a';
```

like huruf kedua o

```
SELECT last_name
FROM employees
WHERE last_name LIKE '_o%';
```

and (hrs keduanya true)

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary>=10000 AND job_like LIKE '%MAN%';
```

or (salah satu)

```
SELECT employee_id, last_name, job_id, salary
FROM employees
WHERE salary>=10000 OR job_like LIKE '%MAN%';
```

current date

```
SELECT current_date, current_timestamp,
current_time, now()
```

gabungkan 2 kolom

```
SELECT last_name, job_id, last_name || job_id as nama_employee
from employees
```

gabungkan 2 kolom

```
SELECT first_name || 'is a' || job_id as halo
from employees
```

ganti nama awal

```
SELECT first_name as nama_awal, manager_id
from employees
```

gabungkan 2 kolom (CONCAT)

```
SELECT CONCAT(first_name, ' ', last_name) AS full_name
from employees
```

distinct

```
SELECT DISTINCT DEPARTMENT_ID
from employees
```

in (menampilkan manager)

```
SELECT employee_id,last_name,manager_id
FROM employees
WHERE manager_id in (101,102)
```

fungsi waktu

```
SELECT CURRENT_DATE;  -- Mengambil tanggal saat ini
SELECT CURRENT_TIMESTAMP;  -- Mengambil tanggal dan waktu saat ini
SELECT DATE_PART('year', CURRENT_DATE);  -- Mengambil tahun dari tanggal saat ini
SELECT NOW() + INTERVAL '1 year';  -- Menambah 1 tahun ke tanggal dan waktu saat ini
```

to Char menggubah menjadi String

```
SELECT last_name,
 TO_CHAR(hire_date, 'fmDD Month YYYY') AS HIREDATE
FROM employees;
```

```
SELECT TO_CHAR(salary, 'Rp99,999.00') SALARY
FROM employees;
```

coalesce

```
SELECT last_name, salary,
coalesce (commission_pct, 0) Comm,
(salary*12) + (salary*12*coalesce(commission_pct, 0)) AN_SAL
FROM employees;
```

JOIN

LEFT JOIN mengembalikan semua baris dari tabel kiri dan baris yang cocok dari tabel kanan. Jika tidak ada kecocokan, hasilnya adalah NULL di tabel kanan.

RIGHT JOIN mengembalikan semua baris dari tabel kanan dan baris yang cocok dari tabel kiri. Jika tidak ada kecocokan, hasilnya adalah NULL di tabel kiri.

self join

```
SELECT worker.last_name emp,manager.last_name
mgr,worker.manager_id,manager.employee_id
FROM employees worker JOIN employees manager
on (worker.manager_id=manager.employee_id);
```

```
--trunscate =menghapus semua isi tabel

--create constraint
create table orders(
order_id integer PRIMARY KEY,
shipping_address text);
create table products(
product_no integer primary key,
name text,
price numeric);
create table order_item(
product_no integer references products(product_no),
order_id integer references orders,
quantity integer,
primary key(product_no,order_id);


-join
--nonequijoin
select e.last_name,e.salary,j.grade,j.lowest_sal,j.highest_sal
from employees e join job_grades j
on e.salary between j.lowest_sal and j.highest_sal;

--left outer join
select e.last_name,e.department_id,d.department_name,d.department_id
from employees e left outer join departments d
on (e.department_id=d.department_id);

select e.last_name,e.department_id,d.department_name,d.department_id
from employees e right outer join departments d
on (e.department_id=d.department_id);

select e.last_name,e.department_id,d.department_name,d.department_id
from employees e full outer join departments d
on (e.department_id=d.department_id);
```