**--SELECT**

\*=semua atribut

SELECT \* FROM Customers

**--DISTINCT**

SELECT DISTINCT lastname FROM employees;

SELECT COUNT(DISTINCT Country) FROM Customers;

**--ORDER BY**

ASC=mengurutkan terkecil->terbesar (DEFAULT)

DESC=mengurutkan terbesar->terkecil

SELECT \* FROM Products ORDER BY Price ASC;

lebih dari 2

SELECT \* FROM Customers

ORDER BY Country ASC, CustomerName DESC;

**--LIKE && Wildcard**

|  |  |
| --- | --- |
| **Simbol** | Deskripsi |
| % | Mewakili nol atau lebih karakter. |
| \_ | Mewakili satu karakter. |
| [] | Mewakili satu karakter yang termasuk dalam kumpulan karakter yang ditentukan. |
| ^ | Mewakili satu karakter yang tidak termasuk dalam kumpulan karakter yang ditentukan. (NEGASI) |
| - | Mewakili satu karakter dalam rentang yang ditentukan. |

- CustomerName dimulai dengan huruf "a".

SElECT \* FROM Customers WHERE CustomerName LIKE “a%”;

-memilih huruf a hingga f

SELECT \* FROM Customers WHERE CustomerName LIKE “[a-f]%”;

-memilih huruf tertentu

SELECT \* FROM Customers WHERE CustomerName LIKE “[axyAXY]%”;

-min Panjang 3 char dan diawali huruf a

SELECT \* FROM Customers WHERE CustomerName LIKE “a\_\_%”;

**--NOT**

-WHERE NOT

SELECT \* FROM Customers WHERE NOT Country=”Spain”;

**-NOT LIKE**

SELECT \* FROM Customers Where CustomerName NOT LIKE “A%”;

**-NOT BETWEEN**

SELECT \* FROM Customers

WHERE CustomerID NOT BETWEEN 10 AND 60;

**-NOT IN**

SELECT \* FROM Customers WHERE City NOT IN (“Paris”,”London”);

**--INSERT**

-Berdasarkan spesifik kolom

INSERT INTO Customers (CustomerName, City, Country)

VALUES ('Cardinal', 'Stavanger', 'Norway');

**--UPDATE**

UPDATE Customers SET ContactName = “Alfred Schmidt”, City=”Frankfurt” WHERE CustomerID=1;

**--DELETE**

-menghapus semua baris

DELETE FROM employees;

-menghapus dari kode tertentu

DELETE FROM employees WHERE officeCode=4;

-menghapus dengan clausa LIMIT (menghapus 2)

DELETE FROM Customers LIMIT 2;

**--Fungsi Agregat**

Terdiri MAX,MIN,AVG,SUM,COUNT

**--GROUP BY**

SELECT status, COUNT(\*)

FROM orders

GROUP BY status;

**--HAVING**

menerapkan kondisi yang menjadi filter ke **setiap grup baris**, sedangkan klausa WHERE menerapkan kondisi filter ke **setiap baris**.

SELECT ordernumber,

SUM(quantityOrdered) AS itemsCount,

SUM(priceeach\*quantityOrdered) AS total

FROM orderdetails GROUP BY ordernumber

HAVING total>1000;

**--JOIN**





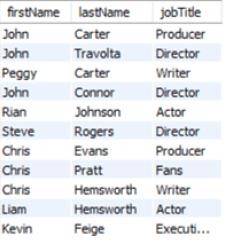
**-INNER jOIN**

SELECT em.firstName,em.lastName,jo.jobTitle

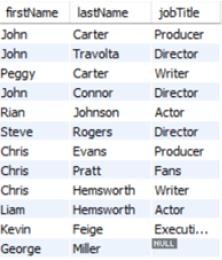
FROM employees as em

INNER JOIN jobs as jo

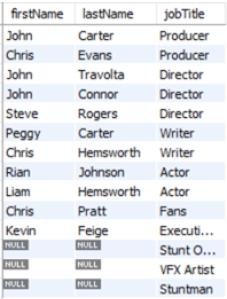
ON em.jobID=jo.jobID;

****

**-LEFT JOIN**



**-RIGHT JOIN**



**-FULL OUTER JOIN**

SELECT em.firstName. em.lastName, jo.jobTitle

FROM employees AS em

LEFT JOIN jobs as jo

ON em.jobID=jo.jobID

UNION

SELECT em.firstName,em.lastName,jo.jobTitle

FROM employees AS em

RIGHT JOIN jobs AS jo

ON em.jobID=jo.jobID



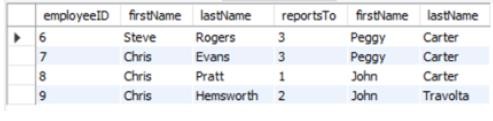
**-SELF JOIN**

SELECT em1.employeeID,em1.firstName,em1.lastName,

Em1.reportsTo,em2.firstName,em2.lastName

FROM employees as em1,employees as em2

WHERE em1.reportsTo=em2.employeeID;



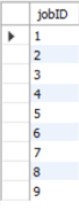
**-UNION**

SELECT jobID FROM employees

WHERE jobID IS NOT NULL

UNION

SELECT jobID FROM jobs;

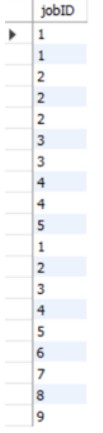


**--UNION ALL**

SELECTjobID FROM employees WHERE jobID IS NOT NULL AND jobID<6

UNION ALL

SELECT jobID FROM jobs;



**--SUBQUERY**

Table jobs

****

Table employees



**-ANY**

SELECT jobTitle,basicSalary

FROM jobs WHERE jobID=ANY

(SELECT jobID FROM employees WHERE salary>10000000);

**-IN**

**IN** sama dengan **=ANY**

SELECT jobTitle, basicSalary

FROM jobs WHERE jobID IN

(SELECT jobID FROM employees WHERE salary>10000000);

(OUTPUT SAMA DENGAN ANY)

**-NOT IN**

SELECT jobTitle, basicSalary

FROM jobs WHERE jobID NOT IN

(SELECT jobID FROM employees WHERE salary>10000000);

**-SOME**

SELECT jobTitle, basicSalary FROM jobs WHERE jobID <> SOME/ANY

(SELECT jobID FROM employees WHERE salary>10000000)

**-ALL**

SELECT \* FROM jobs WHERE jobID>ALL

(SELECT jobID FroM employees WHERE salary>10000000)

**-EXISTS**

SELECT firstName,lastName,jobID,salary FROM employees WHERE EXISTS

(SELECT jobID FROM jobs WHERE employees.jobID=jobs.jobID AND employees.salary<1.1\*jobs.basicSalary);

**--TRIGGER**

Menggunakan

DELIMITER $$

END $$

DELIMITER;

**--Contoh-Contoh Lainnya**

**-Mendapatkan umur**

SELECT TIMESTAMPDIFF(YEAR, tanggal\_lahir, CURDATE()) AS umur FROM pelanggan;

**-Mendapatkan umur di atas rata-rata**

SELECT \* FROM pelanggan

WHERE TIMESTAMPDIFF(YEAR, tanggal\_lahir, CURDATE()) > (SELECT AVG(TIMESTAMPDIFF(YEAR, tanggal\_lahir, CURDATE())) AS rata\_umur

FROM pelanggan)

**-Menggabungkan nama depan dan belakang**

SELECT CONCAT(nama\_depan, ' ', nama\_belakang) AS nama\_lengkap FROM pelanggan;

**Pembahasan UK2**

Dokter

|  |  |  |  |
| --- | --- | --- | --- |
| Kode\_Dokter | Nama\_Dokter | SPE | Jam\_Jaga |
| RZ | Dr. Rafifah Azzahra | ANK | 07.00 |
| RD | Dr. Raisha Dwi Arinda | GG | 13.00 |
| SD | Dr.  Shafa Dian Cantika | SPD | 19.00 |
| TF | Dr.  Teddy Farhan | GG | 07.00 |
| AA | Dr. Austin Arya Cetta | SPD | 13.00 |

 Pasien

|  |  |  |
| --- | --- | --- |
| NRM | Nama\_Pasien | Alamat |
| 0100055 | Bambang Cahyana | Sleman |
| 0400033 | Doni Kiswanto | Bantul |
| 0300011 | Budi Raharjo | Purworejo |

Spesialis

|  |  |
| --- | --- |
| SPE | Spesialis |
| ANK | Anak |
| GG | Gigi |
| SPD | Penyakit Dalam |

Kejadian\_UGD

|  |  |  |  |
| --- | --- | --- | --- |
| NRM | Tgl\_Kejadian | Keterangan | Kode\_Dokter |
| 0100055 | 12 Juni 2019 | Kejang-kejang | AA |
| 0400033 | 12 Juni 2019 | Kecelakaan | AA |
| 0300011 | 12 Juni 2019 | Kecelakaan | RZ |
| 0200022 | 13 Juni 2019 | Batuk darah | RZ |
| 0000011 | 13 Juni 2019 | Kecelakaan | RD |

--Data NRM pasien, nama pasien, kode dokter, keterangan kejadian ‘kecelakaan’ pada ‘12 Juni 2019’

SELECT Pasien.NRM, Pasien.Nama\_Pasien, Kejadian\_UGD.Kode\_Dokter, Kejadian\_UGD.Keterangan

FROM Pasien

JOIN Kejadian\_UGD ON Pasien.NRM = Kejadian\_UGD.NRM

WHERE Kejadian\_UGD.Keterangan = 'Kecelakaan' AND Kejadian\_UGD.Tgl\_Kejadian = '2019-06-12';

-- Data nama pasien dan nama dokter untuk kejadian ‘kecelakaan’ pada ’12 Juni 2019’

Per

SELECT Pasien.Nama\_Pasien, Dokter.Nama\_Dokter

FROM Kejadian\_UGD

JOIN Pasien ON Kejadian\_UGD.NRM = Pasien.NRM

JOIN Dokter ON Kejadian\_UGD.Kode\_Dokter = Dokter.Kode\_Dokter

WHERE Kejadian\_UGD.Keterangan = 'Kecelakaan' AND Kejadian\_UGD.Tgl\_Kejadian = '12 Juni 2019';

- menampilkan nama pelanggan berikut : Martin, Mariam, Masitah, Masruri, Marwah, Mabrur.

namapel like "Ma%"

**--KASUS KOST**

CREATE TABLE penghuni (

idPenghuni INT PRIMARY KEY AUTO\_INCREMENT,

namaDepan VARCHAR(255),

namaBelakang VARCHAR(255),

jalan VARCHAR(255),

RT INT,

RW INT,

kelurahan VARCHAR(255),

kecamatan VARCHAR(255),

kota VARCHAR(255),

provinsi VARCHAR(255),

kodePos INT,

tanggalLahir DATE);

CREATE TABLE penghuni\_email (

idPenghuni INT,

email VARCHAR(255),

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE CASCADE);

CREATE TABLE penghuni\_noTelp (

idPenghuni INT,

noTelp VARCHAR(255),

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE CASCADE);

CREATE TABLE menyewa (

idPenghuni INT,

tglMulaiSewa DATE,

tglAkhirSewa DATE,

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE SET NULL);

CREATE TABLE tipeKamar (

idTipeKamar INT PRIMARY KEY AUTO\_INCREMENT,

tipeKamar VARCHAR(255),

hargaKamar DECIMAL(10,2),

luasKamar DECIMAL(10,2));

CREATE TABLE kamar (

noKamar INT PRIMARY KEY AUTO\_INCREMENT,

idTipeKamar INT,

idPenghuni INT,

FOREIGN KEY (idTipeKamar) REFERENCES tipeKamar(idTipeKamar) ON DELETE SET NULL,

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE SET NULL);

CREATE TABLE transaksi (

idPenghuni INT,

noKamar INT,

idTipeKamar INT,

idTransaksi INT PRIMARY KEY AUTO\_INCREMENT,

jenisTransaksi ENUM('sewa kamar', 'air', 'listrik', 'denda'),

jumlahBiaya DECIMAL(10,2),

keterangan VARCHAR(255),

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE SET NULL,

FOREIGN KEY (noKamar) REFERENCES kamar(noKamar) ON DELETE SET NULL,

FOREIGN KEY (idTipeKamar) REFERENCES tipeKamar(idTipeKamar));

CREATE TABLE membayar (

idPenghuni INT,

noKamar INT,

idTipeKamar INT,

idTransaksi INT PRIMARY KEY AUTO\_INCREMENT,

tglTransaksi DATE,

metodePembayaran ENUM('tunai', 'non-tunai'),

FOREIGN KEY (idPenghuni) REFERENCES penghuni(idPenghuni) ON DELETE SET NULL,

FOREIGN KEY (noKamar) REFERENCES kamar(noKamar) ON DELETE SET NULL,

FOREIGN KEY (idTipeKamar) REFERENCES tipeKamar(idTipeKamar) ON DELETE SET NULL,

FOREIGN KEY (idTransaksi) REFERENCES transaksi(idTransaksi));

**-SUBQUERY**

**Mengambil daftar penghuni yang memiliki email:**

**SELECT namaDepan, namaBelakang**

FROM penghuni WHERE idPenghuni IN (SELECT idPenghuni FROM penghuni\_email);

**Mengambil daftar penghuni yang memiliki kamar dengan tipe kamar tertentu:**

SELECT namaDepan, namaBelakang

FROM penghuni WHERE idPenghuni IN (SELECT idPenghuni FROM kamar WHERE idTipeKamar = 1);

**Mengambil daftar penghuni yang memiliki transaksi dengan biaya lebih dari rata-rata:**

SELECT namaDepan, namaBelakang

FROM penghuni

WHERE idPenghuni IN (SELECT idPenghuni FROM transaksi WHERE jumlahBiaya > (SELECT AVG(jumlahBiaya) FROM transaksi));

**-AGREGAT**

**Menghitung total biaya transaksi untuk setiap penghuni:**

SELECT idPenghuni, SUM(jumlahBiaya) AS totalBiaya FROM transaksi GROUP BY idPenghuni;

**Menghitung rata-rata harga kamar untuk setiap tipe kamar:**

SELECT idTipeKamar, AVG(hargaKamar) AS rataHargaKamar FROM tipeKamar GROUP BY idTipeKamar;

**Menghitung jumlah penghuni yang memiliki email:**

SELECT COUNT(\*) AS jumlahPenghuniDenganEmail

FROM penghuni\_email;

**-JOIN**

**Inner Join - Mengambil daftar penghuni beserta informasi kamar yang mereka sewa:**

SELECT p.namaDepan, p.namaBelakang, k.noKamar

FROM penghuni p

INNER JOIN kamar k ON p.idPenghuni = k.idPenghuni;

**Left Outer Join - Mengambil daftar penghuni beserta informasi kamar yang mereka sewa (termasuk penghuni tanpa kamar):**

SELECT p.namaDepan, p.namaBelakang, k.noKamar

FROM penghuni p

LEFT OUTER JOIN kamar k ON p.idPenghuni = k.idPenghuni;

**Right Outer Join - Mengambil daftar kamar beserta informasi penghuninya (termasuk kamar tanpa penghuni):**

**-TRIGGER**

**Trigger untuk mengupdate tanggal terakhir pembayaran pada tabel penghuni ketika ada transaksi pembayaran baru:**

CREATE TRIGGER update\_tanggal\_pembayaran

AFTER INSERT ON membayar

FOR EACH ROW

BEGIN

UPDATE penghuni

SET tanggalLahir = NEW.tglTransaksi

WHERE idPenghuni = NEW.idPenghuni;

END;

**Trigger untuk menghapus kamar dari tabel kamar ketika penghuni terkait dihapus dari tabel penghuni:**

CREATE TRIGGER delete\_kamar

AFTER DELETE ON penghuni

FOR EACH ROW

BEGIN

DELETE FROM kamar

WHERE idPenghuni = OLD.idPenghuni;

END;

**Trigger untuk mengupdate jumlah biaya transaksi pada tabel transaksi ketika ada perubahan pada harga kamar pada tabel tipeKamar:**

CREATE TRIGGER update\_biaya\_transaksi

AFTER UPDATE ON tipeKamar

FOR EACH ROW

BEGIN

UPDATE transaksi

SET jumlahBiaya = NEW.hargaKamar

WHERE idTipeKamar = NEW.idTipeKamar;

END;

**-PROCEDURE**

Procedure untuk menghitung total biaya transaksi untuk seorang penghuni berdasarkan ID penghuni dan mengembalikan nilainya:

CREATE PROCEDURE hitung\_total\_biaya(IN penghuniID INT, OUT total DECIMAL(10,2))

BEGIN

SELECT SUM(jumlahBiaya) INTO total

FROM transaksi

WHERE idPenghuni = penghuniID;

END;

**Procedure untuk menghapus penghuni dan semua data terkaitnya (email, nomor telepon, transaksi, pembayaran, dan kamar):**

CREATE PROCEDURE hapus\_penghuni(IN penghuniID INT)

BEGIN

DELETE FROM penghuni\_email WHERE idPenghuni = penghuniID;

DELETE FROM penghuni\_noTelp WHERE idPenghuni = penghuniID;

DELETE FROM transaksi WHERE idPenghuni = penghuniID;

DELETE FROM membayar WHERE idPenghuni = penghuniID;

DELETE FROM kamar WHERE idPenghuni = penghuniID;

DELETE FROM penghuni WHERE idPenghuni = penghuniID;

END;

**Procedure untuk mengupdate harga kamar pada tabel tipeKamar berdasarkan ID tipe kamar:**

CREATE PROCEDURE update\_harga\_kamar(IN tipeKamarID INT, IN newHarga DECIMAL(10,2))

BEGIN

UPDATE tipeKamar

SET hargaKamar = newHarga

WHERE idTipeKamar = tipeKamarID;

END;