

1. membuat database company_indrawan.

STRUKTUR "

SQL

```
CREATE TABLE pegawai (  
-> NIP INT PRIMARY KEY,  
-> NDep VARCHAR(255) NOT NULL,  
-> NBik VARCHAR(255),  
-> JK ENUM('L', 'P') NOT NULL,  
-> Alamat TEXT NOT NULL,  
-> Telp VARCHAR(255) NOT NULL,  
Jabatan ENUM('Manager', 'Sales', 'Staff'),  
    Gaji BIGINT NOT NULL,  
    NoCab VARCHAR(255) NOT NULL  
);
```

PENJELASAN :

1. NIP (Nomor Induk Pegawai):

- Tipe data INT dipilih karena NIP biasanya berupa angka yang dapat direpresentasikan dengan tipe data integer.
- PRIMARY KEY menunjukkan bahwa kolom NIP adalah kunci utama (primary key) yang unik untuk setiap data pegawai.

2. NDep (Nama Departemen) dan NBik (Nama Belakang):

- Tipe data VARCHAR(255) dipilih karena nama departemen dan nama belakang dapat berupa string yang panjangnya bervariasi, namun tidak melebihi 255 karakter.
- NOT NULL menunjukkan bahwa kolom NDep dan NBik harus diisi, karena informasi tersebut dianggap penting untuk data pegawai.

3. JK (Jenis Kelamin):

- Tipe data ENUM('L', 'P') dipilih karena jenis kelamin hanya memiliki dua kemungkinan, yaitu 'Laki-laki' dan 'Perempuan'.
- NOT NULL menunjukkan bahwa kolom JK harus diisi, karena informasi jenis kelamin dianggap penting untuk data pegawai.

4. Alamat:

- Tipe data TEXT dipilih karena alamat dapat berisi string yang lebih panjang daripada VARCHAR(255)
- NOT NULL menunjukkan bahwa kolom Alamat harus diisi, karena informasi tersebut dianggap penting untuk data pegawai.

5. Telp (Telepon):

- Tipe data VARCHAR(255) dipilih karena nomor telepon dapat berupa string yang panjangnya bervariasi, namun tidak melebihi 255 karakter.
- NOT NULL menunjukkan bahwa kolom Telp harus diisi, karena informasi kontak pegawai dianggap penting.

6. Jabatan:

- Tipe data ENUM('Manager', 'sales', 'Staff') dipilih karena jabatan pegawai hanya memiliki tiga kemungkinan.
- Tidak ada atribut NOT NULL atau NULL, artinya kolom Jabatan boleh diisi atau dibiarkan kosong.

7. Gaji:

- Tipe data BIGINT dipilih karena gaji pegawai dapat berupa angka yang cukup besar.
- NOT NULL menunjukkan bahwa kolom Gaji harus diisi, karena informasi gaji dianggap penting untuk data pegawai.

8. NoCab (Nomor Cabang):

- Tipe data VARCHAR(255) dipilih karena nomor cabang dapat berupa string yang panjangnya bervariasi, namun tidak melebihi 255 karakter.
- NOT NULL menunjukkan bahwa kolom NoCab harus diisi, karena informasi cabang dianggap penting untuk data pegawai.

Secara umum, penggunaan tipe data dan atribut NOT NULL atau NULL pada tabel pegawai ini dimaksudkan untuk menjaga integritas dan konsistensi data, serta memastikan bahwa informasi yang dianggap penting untuk data pegawai selalu tersedia.

HASIL :

```
MariaDB [company_indrawan]> desc pegawai
-> ;
```

Field	Type	Null	Key	Default	Extra
NIP	int(11)	NO	PRI	NULL	
NDep	varchar(255)	NO		NULL	
NBLk	varchar(255)	YES		NULL	
JK	enum('L', 'P')	NO		NULL	
Alamat	text	NO		NULL	
Telp	varchar(15)	NO		NULL	
Jabatan	enum('Manager', 'Sales', 'Staff')	YES		NULL	
Gaji	bigint(20)	NO		NULL	
NoCab	varchar(10)	NO		NULL	

9 rows in set (0.05 sec)

2. membuat table pegawai .

STRUKTUR "

SQL

```
INSERT INTO pegawai (NIP, NDep, NBlk, JK, Alamat, Telp, Jabatan, Gaji, NoCab)
VALUES
-> (10107, 'Emya', 'Salsalina', 'P', 'JL. Suci 78 Bandung', '022-555768',
'Manager', 5250000, 'C101'),
-> (10246, 'Dian', 'Anggraini', 'P', 'JL. Mawar 5 Semarang', '024-555102',
'Supervisor', 2750000, 'C103'),
-> (10324, 'Martin', 'Susanto', 'L', 'JL. Bima 51 Jakarta', '021-555785', 'Staff',
1750000, 'C102'),
-> (10252, 'Antoni', 'Irawan', 'L', 'JL. A. Yani 15 Jakarta', '021-555888',
'Manager', 5750000, 'C102'),
-> (10176, 'Diah', 'Wahyuni', 'P', 'JL. Maluku 56 Bandung', '022-555934',
'Supervisor', 2500000, 'C101'),
-> (10314, 'Ayu', 'Rahmadani', 'P', 'JL. Malaka 342 Jakarta', '021-555098',
'Supervisor', 1950000, 'C102'),
-> (10307, 'Erik', 'Adrian', 'L', 'JL. Manggis 5 Semarang', '024-555236',
'Manager', 6250000, 'C103'),
-> (10415, 'Susan', 'Sumantri', 'P', 'JL. Pahlawan 24 Surabaya', '031-555120', '',
2650000, 'C104'),
-> (10407, 'Rio', 'Gunawan', 'L', 'JL. Melati 356 Surabaya', '031-555231', 'Staff',
1725000, 'C104');
```

PENJELASAN :

- *INSERT INTO pegawai:*
 - Menunjukkan bahwa Anda akan menambahkan data ke tabel bernama pegawai.
- *(NIP, NDep, NBlk, JK, Alamat, Telp, Jabatan, Gaji, NoCab):*
 - Ini adalah daftar kolom dalam tabel pegawai yang akan diisi dengan data. Kolom-kolom ini adalah:
 - NIP (Nomor Induk Pegawai)
 - NDep (Nama Depan)
 - NBlk (Nama Belakang)
 - JK (Jenis Kelamin)
 - Alamat (Alamat)
 - Telp (Telepon)
 - Jabatan (Jabatan)
 - Gaji (Gaji)
 - NoCab (Nomor Cabang)
- *VALUES:*

- Menunjukkan data yang akan dimasukkan ke dalam tabel. Data untuk setiap baris harus sesuai dengan urutan kolom yang disebutkan sebelumnya.
- *Data yang Dimasukkan:*
 - Baris pertama:
 - NIP: 10107
 - NDep: 'Emya'
 - NBlk: 'Salsalina'
 - JK: 'P' (Perempuan)
 - Alamat: 'JL. Suci 78 Bandung'
 - Telp: '022-555768'
 - Jabatan: 'Manager'
 - Gaji: 5250000
 - NoCab: 'C101'
 - Baris kedua dan seterusnya mengikuti pola yang sama, dengan data yang berbeda.
- *Catatan:*
 - Pada baris 10415 untuk Susan Sumantri, kolom Jabatan tidak diisi ("), yang bisa menyebabkan masalah jika kolom Jabatan adalah ENUM dan tidak termasuk nilai kosong. Pastikan kolom Jabatan memiliki nilai yang valid.
 - Jika ada nilai kosong (") dalam kolom Jabatan, Anda mungkin ingin memperbarui baris ini dengan jabatan yang sesuai, seperti 'Staff', atau menyesuaikan schema tabel untuk mengizinkan nilai kosong jika perlu.

HASIL :

```
MariaDB [company_indrawan]> select * from pegawai;
```

NIP	NDep	NBlk	JK	Alamat	Telp	Jabatan	Gaji	NoCab
10107	Emya	Salsalina	P	JL. Suci 78 Bandung	022-555768	Manager	5250000	C181
10176	Diah	Wahyuni	P	JL. Maluku 56 Bandung	022-555934	Sales	2500000	C101
10246	Dian	Anggraini	P	JL. Mawar 5 Semarang	024-555102	Sales	2750000	C103
10252	Antoni	Irawan	L	JL. A. Yani 15 Jakarta	021-555888	Manager	5758000	C102
10307	Erik	Semarang	L	JL. Manggis 5	024-555236	Manager	6250000	C103
10314	Ayu	Rahmadani	P	JL. Malaka 342 Jakarta	021-555098	Sales	1950000	C102
10324	Martin	Susanto	L	JL. Bima 51 Jakarta	021-555785	Staff	1750000	C102
10407	Rio	Gunawan	L	JL. Melati 356 Surabaya	031-555231	Staff	1725000	C104
10415	Susan	Sumantri	P	JL. Pahlawan 24 Surabaya	031-555120		2650000	C104

9 rows in set (0.00 sec)

select table

jumlahpegawai and jumlahjabatan

SINTAKS

```
select COUNT(NIP) AS Jumlahpegawai, COUNT(Jabatan) AS JumlahJabatan FROM pegawai;
```

PENJELASAN

1. **SELECT:** Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.
2. **COUNT(NIP) AS Jumlahpegawai:**
 - COUNT(NIP) adalah fungsi agregat SQL yang menghitung jumlah baris dalam kolom "NIP" (Nomor Induk Pegawai). Fungsi COUNT menghitung jumlah baris yang memiliki nilai non-NULL dalam kolom yang diberikan.
 - AS Jumlahpegawai adalah penggunaan alias (alias) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "Jumlahpegawai".
3. **COUNT(Jabatan) AS JumlahJabatan:**
 - COUNT(Jabatan) adalah fungsi agregat SQL yang menghitung jumlah baris dalam kolom "Jabatan". Fungsi COUNT menghitung jumlah baris yang memiliki nilai non-NULL dalam kolom yang diberikan.
 - AS JumlahJabatan adalah penggunaan alias (alias) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "JumlahJabatan".
4. **FROM pegawai:** Ini adalah klausa SQL yang menentukan tabel dari mana data akan diambil, yaitu tabel "pegawai".
Jadi, pernyataan SQL tersebut akan menghasilkan dua kolom:
"Jumlahpegawai" yang berisi jumlah baris (jumlah pegawai) dalam tabel "pegawai".
"JumlahJabatan" yang berisi jumlah baris (jumlah jabatan) dalam tabel "pegawai".
5. "JumlahJabatan" yang berisi jumlah baris (jumlah jabatan) dalam tabel "pegawai".

HASIL :

```
MariaDB [company_indrawan]> select COUNT(NIP) AS Jumlahpegawai, COUNT(Jabatan) AS JumlahJabatan FROM pegawai;
+-----+-----+
| Jumlahpegawai | JumlahJabatan |
+-----+-----+
|          9   |          9   |
+-----+-----+
1 row in set (0.02 sec)
```

jumlah pegawai

sintaks

```
SELECT COUNT(NIP) AS Jumlahpegawai
-> FROM pegawai
```

```
-> where NoCab ='C102';
```

PENJELASAN

1. **SELECT COUNT(NIP) AS Jumlahpegawai:**

- **SELECT:** Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.
- **COUNT(NIP):** Ini adalah fungsi agregat SQL yang menghitung jumlah baris dalam kolom "NIP" (Nomor Induk Pegawai). Fungsi COUNT menghitung jumlah baris yang memiliki nilai non-NULL dalam kolom yang diberikan.
- **AS Jumlahpegawai:** Ini adalah penggunaan alias (alias) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "Jumlahpegawai".

2. **FROM pegawai:**

- **FROM:** Ini adalah klausa SQL yang menentukan tabel dari mana data akan diambil, yaitu tabel "pegawai".

3. **WHERE NoCab ='C102':**

- **WHERE:** Ini adalah klausa SQL yang digunakan untuk memfilter baris berdasarkan kondisi tertentu.

- **NoCab =C102:** Ini adalah kondisi yang akan digunakan untuk memfilter baris di mana nilai kolom "NoCab" (Nomor Cabang) sama dengan 'C102'.

Jadi, pernyataan SQL tersebut akan menghasilkan satu kolom:

- "Jumlahpegawai" yang berisi jumlah baris (jumlah pegawai) dalam tabel "pegawai" di mana nilai "NoCab" (Nomor Cabang) sama dengan 'C102'.

HASIL:

```
MariaDB [company_indrawan]> SELECT COUNT(NIP) AS Jumlahpegawai
-> FROM pegawai
-> where NoCab ='C102';
+-----+
| Jumlahpegawai |
+-----+
|              3 |
+-----+
1 row in set (0.01 sec)
```

NoCab jumlahpegawai

sintaks

```
SELECT NoCab, COUNT(NIP) AS Jumlahpegawai  
-> FROM Pegawai  
-> GROUP BY NoCab;
```

PENJELASAN

1. SELECT NoCab, COUNT(NIP) AS Jumlahpegawai:

- SELECT: Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.
- NoCab: Ini adalah nama kolom yang akan diambil dari tabel. Kolom ini mungkin berisi nomor cabang.
- COUNT(NIP): Ini adalah fungsi agregat SQL yang menghitung jumlah baris dalam kolom "NIP" (Nomor Induk Pegawai). Fungsi COUNT menghitung jumlah baris yang memiliki nilai non-NULL dalam kolom yang diberikan.
- AS Jumlahpegawai: Ini adalah penggunaan alias (alias) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "Jumlahpegawai".

2. FROM Pegawai:

- FROM: Ini adalah klausa SQL yang menentukan tabel dari mana data akan diambil, yaitu tabel "Pegawai".

3. GROUP BY NoCab:

- GROUP BY: Ini adalah klausa SQL yang digunakan untuk mengelompokkan baris berdasarkan satu atau lebih kolom.
- NoCab: Ini adalah kolom yang akan digunakan untuk mengelompokkan baris. Dalam hal ini, baris akan dikelompokkan berdasarkan nilai kolom "NoCab" (Nomor Cabang).
Jadi, pernyataan SQL tersebut akan menghasilkan dua kolom:

- "NoCab" yang berisi nomor cabang
- "Jumlahpegawai" yang berisi jumlah pegawai untuk setiap nomor cabang

HASIL:

```
MariaDB [company_indrawan]> SELECT NoCab, COUNT(NIP) AS Jumlahpegawai
-> FROM Pegawai
-> GROUP BY NoCab;
```

NoCab	Jumlahpegawai
C101	1
C102	3
C103	2
C104	2
C181	1

```
5 rows in set (0.03 sec)
```

NoCab jumlah pegawai having sintaks

```
Select Nocab, Count(NIP) AS Jumlahpegawai
-> FROM pegawai
-> GROUP BY NoCab HAVING COUNT(NIP) >= 3;
```

PENJELASAN

1. SELECT NoCab, COUNT(NIP) AS Jumlahpegawai:

- **SELECT:** Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.
- **NoCab:** Ini adalah nama kolom yang akan diambil dari tabel. Kolom ini mungkin berisi nomor cabang.
- **COUNT(NIP):** Ini adalah fungsi agregat SQL yang menghitung jumlah baris dalam kolom "NIP" (Nomor Induk Pegawai). Fungsi COUNT menghitung jumlah baris yang memiliki nilai non-NULL dalam kolom yang diberikan.
- **AS Jumlahpegawai:** Ini adalah penggunaan alias (alias) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "Jumlahpegawai".

2. FROM pegawai:

- **FROM:** Ini adalah klausa SQL yang menentukan tabel dari mana data akan diambil, yaitu tabel "pegawai".

3. GROUP BY NoCab:

- **GROUP BY:** Ini adalah klausa SQL yang digunakan untuk mengelompokkan baris berdasarkan satu atau lebih kolom.

- NoCab: Ini adalah kolom yang akan digunakan untuk mengelompokkan baris. Dalam hal ini, baris akan dikelompokkan berdasarkan nilai kolom "NoCab" (Nomor Cabang).

4. HAVING COUNT(NIP) >= 3:

- HAVING: Ini adalah klausa SQL yang digunakan untuk memfilter kelompok berdasarkan kondisi tertentu, setelah pengelompokan dilakukan.
- COUNT(NIP) >= 3: Ini adalah kondisi yang akan digunakan untuk memfilter kelompok di mana jumlah baris (jumlah pegawai) dalam setiap grup (cabang) lebih besar atau sama dengan 3.

Jadi, pernyataan SQL tersebut akan menghasilkan dua kolom:

- "NoCab" yang berisi nomor cabang
- "Jumlahpegawai" yang berisi jumlah pegawai untuk setiap nomor cabang yang memiliki jumlah pegawai lebih besar atau sama dengan 3.

HASIL:

```
MariaDB [company_indrawan]> Select Nocab, Count(NIP) AS Jumlahpegawai
-> FROM pegawai
-> GROUP BY NoCab HAVING COUNT(NIP)>= 3;
```

Nocab	Jumlahpegawai
C102	3

1 row in set (0.01 sec)

total_gaji

sintaks

```
SELECT SUM(Gaji) AS Total_GAJI
-> FROM pegawai;
```

PENJELASAN

1. SELECT:

- Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.

2. SUM(Gaji):

- SUM: Ini adalah fungsi agregat SQL yang menghitung jumlah total dari nilai-nilai dalam kolom yang diberikan.
- Gaji: Ini adalah nama kolom yang akan dijumlahkan, yaitu kolom "Gaji".

3. AS:

- Ini adalah kata kunci SQL yang digunakan untuk memberikan alias (nama baru) pada kolom hasil.

4. Total_GAJI:

- Ini adalah alias (nama baru) yang diberikan pada kolom yang berisi jumlah total gaji.

5. FROM:

- Ini adalah kata kunci SQL yang menentukan tabel dari mana data akan diambil.

6. pegawai:

- Ini adalah nama tabel dari mana data akan diambil.

Jadi, keseluruhan pernyataan SQL tersebut akan mengambil data dari tabel "pegawai", menghitung jumlah total gaji dari semua pegawai, dan menampilkannya dengan nama kolom "Total_GAJI".

HASIL:

```
MariaDB [company_indrawan]> SELECT SUM(Gaji) AS Total_GAJI
-> FROM pegawai;
+-----+
| Total_GAJI |
+-----+
| 30583000 |
+-----+
1 row in set (0.00 sec)
```

gaji manager

sintaks

```
SELECT SUM(Gaji) AS Gaji_Manager
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
```

PENJELASAN

1. SELECT SUM(Gaji) AS Gaji_Manager:

- SELECT: Ini adalah kata kunci SQL yang digunakan untuk memulai pernyataan SQL untuk mengambil data dari tabel.
 - SUM(Gaji): Ini adalah fungsi agregat SQL yang menghitung jumlah total dari nilai-nilai dalam kolom "Gaji".
 - AS Gaji_Manager: Ini adalah penggunaan alias (nama baru) untuk memberikan nama kolom yang lebih mudah dipahami, yaitu "Gaji_Manager".
2. FROM pegawai:
- FROM: Ini adalah klausa SQL yang menentukan tabel dari mana data akan diambil, yaitu tabel "pegawai".
3. WHERE Jabatan = 'Manager':
- WHERE: Ini adalah klausa SQL yang digunakan untuk memfilter data berdasarkan kondisi tertentu.
 - Jabatan = 'Manager': Ini adalah kondisi filter yang akan memilih hanya data di mana nilai kolom "Jabatan" adalah 'Manager'.
- Jadi, pernyataan SQL tersebut akan:
4. Mengambil data dari tabel "pegawai".
5. Menghitung jumlah total gaji dari semua pegawai yang memiliki jabatan "Manager".
6. Menampilkan hasil jumlah total gaji dengan nama kolom "Gaji_Manager".

HASIL:

```
MariaDB [company_indrawan]> SELECT SUM(Gaji) AS Gaji_Manager
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
+-----+
| Gaji_Manager |
+-----+
|      17258000 |
+-----+
1 row in set (0.00 sec)
```

NoCab Total_gaji

sintaks

```
SELECT NoCab, SUM(Gaji) AS Total_Gaji
-> FROM pegawai
-> GROUP BY NoCab;
```

PENJELASAN

1. SELECT NoCab, SUM(Gaji) AS Total_Gaji
 - SELECT: Ini adalah kata kunci untuk menentukan kolom-kolom yang ingin ditampilkan dalam hasil query.
 - NoCab: Kolom yang berisi nomor cabang tempat seorang pegawai bekerja.
 - SUM(Gaji): Fungsi agregat yang akan menjumlahkan seluruh nilai pada kolom Gaji.
 - AS Total_Gaji: Memberikan alias atau nama baru untuk hasil penjumlahan gaji, yaitu "Total_Gaji".
2. FROM pegawai
 - FROM: Kata kunci yang menentukan tabel tempat data berasal.
 - pegawai: Nama tabel yang berisi data-data pegawai.
3. GROUP BY NoCab
 - GROUP BY: Kata kunci yang akan mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab.
 - Jadi, query ini akan menghitung total gaji untuk setiap nomor cabang yang ada pada tabel pegawai.

Secara keseluruhan, query ini akan menghasilkan sebuah tabel dengan dua kolom:
4. NoCab: Menampilkan nomor cabang
5. Total_Gaji: Mena

HASIL:

```
MariaDB [company_indrawan]> SELECT NoCab, SUM(Gaji) AS Total_Gaji
-> FROM pegawai
-> GROUP BY NoCab;
```

NoCab	Total_Gaji
C101	2500000
C102	9458000
C103	9000000
C104	4375000
C181	5250000

5 rows in set (0.01 sec)

Total gaji nocab having SUM
sintaks

```
SELECT Nocab, SUM(gaji) AS Total_gaji
-> FROM pegawai
-> GROUP BY NoCab HAVING SUM(Gaji)>= 8000000;
```

PENJELASAN

1. SELECT Nocab, SUM(Gaji) AS Total_Gaji
 - SELECT: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
 - NoCab: Kolom yang berisi nomor cabang tempat pegawai bekerja.
 - SUM(Gaji): Fungsi agregat yang akan menjumlahkan seluruh nilai pada kolom Gaji.
 - AS Total_Gaji: Memberikan alias "Total_Gaji" untuk hasil penjumlahan gaji.
2. FROM pegawai
 - FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".
3. GROUP BY NoCab
 - GROUP BY: Mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab.
4. HAVING SUM(Gaji) >= 8000000

- HAVING: Menambahkan kondisi filter pada hasil pengelompokkan.
- SUM(Gaji) >= 8000000: Hanya menampilkan kelompok-kelompok (cabang) yang memiliki total gaji lebih besar atau sama dengan 8.000.000.

Jadi, query ini akan menghasilkan tabel yang berisi:

5. NoCab: Nomor cabang
6. Total_Gaji: Total gaji untuk

HASIL:

```
MariaDB [company_indrawan]> SELECT Nocab, SUM(gaji) AS Total_gaji
-> FROM pegawai
-> GROUP BY NoCab HAVING SUM(Gaji)>= 8000000;
+-----+-----+
| Nocab | Total_gaji |
+-----+-----+
| C102  | 9458000    |
| C103  | 9000000    |
+-----+-----+
2 rows in set (0.00 sec)
```

select AVG(gaji)

sintaks

```
select AVG(gaji) AS Rata_rata -> FROM pegawai;
```

PENJELASAN

1. SELECT AVG(Gaji) AS Rata_rata

- SELECT: Menentukan kolom yang akan ditampilkan dalam hasil query.
- AVG(Gaji): Fungsi agregat yang akan menghitung rata-rata (average) dari nilai-nilai pada kolom Gaji.
- AS Rata_rata: Memberikan alias "Rata_rata" untuk hasil perhitungan rata-rata gaji.

2. FROM pegawai

- FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".

Jadi, query ini akan menghasilkan sebuah tabel dengan satu baris dan satu kolom, yang berisi nilai rata-rata gaji dari seluruh pegawai yang terdapat dalam tabel "pegawai".

HASIL:

```
MariaDB [company_indrawan]> select AVG(gaji) AS Rata_rata
-> FROM pegawai;
+-----+
| Rata_rata |
+-----+
| 3398111.1111 |
+-----+
1 row in set (0.01 sec)
```

select gaji maneger

sintaks

```
SELECT AVG(gaji) AS GajiRataMgr
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
```

PENJELASAN

1. SELECT AVG(Gaji) AS GajiRataMgr

- SELECT: Menentukan kolom yang akan ditampilkan dalam hasil query.
- AVG(Gaji): Fungsi agregat yang akan menghitung rata-rata (average) dari nilai-nilai pada kolom Gaji.
- AS GajiRataMgr: Memberikan alias "GajiRataMgr" untuk hasil perhitungan rata-rata gaji.

2. FROM pegawai

- FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".

3. WHERE Jabatan = 'Manager'

- WHERE: Menambahkan kondisi filter pada baris-baris data.
- Jabatan = 'Manager': Hanya akan memilih baris-baris data di mana nilai pada kolom Jabatan adalah 'Manager'.

Jadi, query ini akan menghasilkan sebuah tabel dengan satu baris dan satu kolom, yang berisi nilai rata-rata gaji dari seluruh pegawai yang memiliki jabatan sebagai Manager.

HASIL:

```
MariaDB [company_indrawan]> SELECT AVG(gaji) AS GajiRataMgr
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
+-----+
| GajiRataMgr |
+-----+
| 5752666.6667 |
+-----+
1 row in set (0.00 sec)
```

avg ratagaji

sintaks

```
SELECT NoCab,AVG(Gaji) AS Ratagaji
-> FROM pegawai
-> GROUP BY NoCab;
```

PENJELASAN

1. SELECT NoCab, AVG(Gaji) AS RataGaji

- SELECT: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
- NoCab: Kolom yang berisi nomor cabang tempat pegawai bekerja.
- AVG(Gaji): Fungsi agregat yang akan menghitung rata-rata (average) dari nilai-nilai pada kolom Gaji.
- AS RataGaji: Memberikan alias "RataGaji" untuk hasil perhitungan rata-rata gaji.

2. FROM pegawai

- FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".

3. GROUP BY NoCab

- GROUP BY: Mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab. Jadi, query ini akan menghasilkan sebuah tabel dengan dua kolom:

4. NoCab: Nomor cabang

5. RataGaji: Rata-rata gaji untuk masing-masing cabang

HASIL:

```
MariaDB [company_indrawan]> SELECT NoCab,AVG(Gaji) AS Ratagaji
-> FROM pegawai
-> GROUP BY NoCab;
+-----+-----+
| NoCab | Ratagaji |
+-----+-----+
| C101  | 2500000.0000 |
| C102  | 3152666.6667 |
| C103  | 4500000.0000 |
| C104  | 2187500.0000 |
| C181  | 5250000.0000 |
+-----+-----+
5 rows in set (0.01 sec)
```

avg ratagaji no cab-no cab

sintaks

```
SELECT Nocab, AVG(gaji) AS Ratagaji
-> FROM pegawai
-> GROUP BY NoCab HAVING NoCab ='C101' OR NoCAB = 'C102';
```

PENJELASAN

1. SELECT NoCab, AVG(Gaji) AS RataGaji

- SELECT: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
- NoCab: Kolom yang berisi nomor cabang tempat pegawai bekerja.
- AVG(Gaji): Fungsi agregat yang akan menghitung rata-rata (average) dari nilai-nilai pada kolom Gaji.
- AS RataGaji: Memberikan alias "RataGaji" untuk hasil perhitungan rata-rata gaji.

2. FROM pegawai

- FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".

3. GROUP BY NoCab

- GROUP BY: Mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab.

4. HAVING NoCab = 'C101' OR NoCab = 'C102'

- HAVING: Menambahkan kondisi filter pada hasil pengelompokkan.
- NoCab = 'C101' OR NoCab = 'C102': Hanya akan menampilkan kelompok-kelompok (cabang) yang memiliki nomor cabang 'C101' atau 'C102'.

Jadi, query ini akan menghasilkan sebuah tabel dengan dua kolom:

5. NoCab: Nomor cabang (hanya 'C101' dan 'C102')

6. RataGaji: Rata-rata gaji

HASIL:

```
MariaDB [company_indrawan]> SELECT Nocab, AVG(gaji) AS Ratagaji
-> FROM pegawai
-> GROUP BY NoCab HAVING NoCab ='C101' OR NoCAB = 'C102';
```

Nocab	Ratagaji
C101	2500000.0000
C102	3152666.6667

2 rows in set (0.01 sec)

SELECT MAX Gajiterkecil AS GajiTerbesar

sintaks

```
SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai;
```

PENJELASAN

mari saya jelaskan query SQL ini:

1. `SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil`
 - `SELECT`: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
 - `MAX(Gaji)`: Fungsi agregat yang akan menghitung nilai gaji terbesar.
 - `MIN(Gaji)`: Fungsi agregat yang akan menghitung nilai gaji terkecil.
 - `AS GajiTerbesar`: Memberikan alias "GajiTerbesar" untuk hasil perhitungan nilai gaji terbesar.
 - `AS GajiTerkecil`: Memberikan alias "GajiTerkecil" untuk hasil perhitungan nilai gaji terkecil.
2. `FROM pegawai`
 - `FROM`: Menentukan tabel sumber data, yaitu tabel "pegawai".Jadi, query ini akan menghasilkan sebuah tabel dengan dua baris dan dua kolom:
3. Baris pertama: Nilai gaji terbesar di antara semua pegawai.
4. Baris kedua: Nilai gaji terkecil di antara semua pegawai.

HASIL:

```
MariaDB [company_indrawan]> SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai;
```

GajiTerbesar	GajiTerkecil
6250000	1725000

```
1 row in set (0.04 sec)
```

SELECT MAX Gajiterkecil AS GajiTerbesar manager sintaks

```
SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
```

PENJELASAN

1. `SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil`

- SELECT: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
- MAX(Gaji): Fungsi agregat yang akan menghitung nilai gaji terbesar.
- MIN(Gaji): Fungsi agregat yang akan menghitung nilai gaji terkecil.
- AS GajiTerbesar: Memberikan alias "GajiTerbesar" untuk hasil perhitungan nilai gaji terbesar.
- AS GajiTerkecil: Memberikan alias "GajiTerkecil" untuk hasil perhitungan nilai gaji terkecil.

2. FROM pegawai

- FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".

3. WHERE Jabatan = 'Manager'

- WHERE: Menambahkan kondisi filter pada baris-baris data.
- Jabatan = 'Manager': Hanya akan memilih baris-baris data di mana nilai pada kolom Jabatan adalah 'Manager'.

Jadi, query ini akan menghasilkan sebuah tabel dengan dua baris dan dua kolom:

- Baris pertama: Nilai gaji terbesar di antara seluruh pegawai yang memiliki jabatan sebagai Manager.
- Baris kedua: Nilai gaji terkecil di antara seluruh pegawai yang memiliki jabatan sebagai Manager.

HASIL:

```
MariaDB [company_indrawan]> SELECT MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai
-> WHERE Jabatan = 'Manager';
```

GajiTerbesar	GajiTerkecil
6250000	5250000

```
1 row in set (0.01 sec)
```

SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji)

sintaks

```
SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai
-> GROUP BY NoCab;
```

PENJELASAN

1. **SELECT NoCab, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil:**
 - SELECT berfungsi untuk menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
 - NoCab adalah kolom yang berisi nomor cabang tempat pegawai bekerja.
 - MAX(Gaji) adalah fungsi agregat yang akan menghitung nilai gaji terbesar untuk masing-masing cabang.
 - MIN(Gaji) adalah fungsi agregat yang akan menghitung nilai gaji terkecil untuk masing-masing cabang.
 - AS GajiTerbesar dan AS GajiTerkecil memberikan alias untuk hasil perhitungan nilai gaji terbesar dan terkecil.
2. **FROM pegawai:**
 - FROM menentukan tabel sumber data, yaitu tabel "pegawai".
3. **GROUP BY NoCab:**
 - GROUP BY mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab. Dengan query ini, hasilnya akan berupa tabel yang berisi:
4. **Kolom NoCab:** Nomor cabang tempat pegawai bekerja.
5. **Kolom GajiTerbesar:** Nilai gaji terbesar untuk masing-masing cabang.
6. **Kolom GajiTerkecil:** Nilai gaji terkecil untuk masing-masing cabang.

HASIL:

```
MariaDB [company_indrawan]> SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
-> FROM pegawai
-> GROUP BY NoCab;
```

NoCAB	GajiTerbesar	GajiTerkecil
C101	2500000	2500000
C102	5758000	1750000
C103	6250000	2750000
C104	2650000	1725000
C181	5250000	5250000

```
5 rows in set (0.03 sec)
```

SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji)
(3)

sintaks

```
SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil  
-> FROM pegawai  
-> GROUP BY NoCab;
```

PENJELASAN

1. SELECT NoCab, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil
 - SELECT: Menentukan kolom-kolom yang akan ditampilkan dalam hasil query.
 - NoCab: Kolom yang berisi nomor cabang tempat pegawai bekerja.
 - MAX(Gaji): Fungsi agregat yang akan menghitung nilai gaji terbesar untuk masing-masing cabang.
 - MIN(Gaji): Fungsi agregat yang akan menghitung nilai gaji terkecil untuk masing-masing cabang.
 - AS GajiTerbesar: Memberikan alias "GajiTerbesar" untuk hasil perhitungan nilai gaji terbesar.
 - AS GajiTerkecil: Memberikan alias "GajiTerkecil" untuk hasil perhitungan nilai gaji terkecil.
2. FROM pegawai
 - FROM: Menentukan tabel sumber data, yaitu tabel "pegawai".
3. GROUP BY NoCab
 - GROUP BY: Mengelompokkan baris-baris data berdasarkan nilai pada kolom NoCab. Jadi, query ini akan menghasilkan sebuah tabel dengan tiga kolom:
4. NoCab: Nomor cabang tempat pegawai bekerja.
5. GajiTerbesar: Nilai gaji terbesar untuk masing-masing cabang.
6. GajiTerkecil: Nilai gaji terkecil untuk masing-masing cabang

HASIL:

```
MariaDB [company_indrawan]> SELECT NoCAB, MAX(Gaji) AS GajiTerbesar, MIN(Gaji) AS GajiTerkecil  
-> FROM pegawai  
-> GROUP BY NoCab HAVING SUM(Gaji)>= 3;
```

NoCAB	GajiTerbesar	GajiTerkecil
C101	2500000	2500000
C102	5758000	1750000
C103	6250000	2750000
C104	2650000	1725000
C181	5250000	5250000

5 rows in set (0.02 sec)

SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji

sintaks

```
SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji,  
-> AVG(Gaji) AS RataGaji, MAX(Gaji) AS GajiMaks, Min(gaji) AS Gajmin  
-> FROM pegawai;
```

PENJELASAN

1. SELECT COUNT(NIP) AS Jumlahpegawai:
 - Fungsi COUNT(NIP) menghitung jumlah baris/record pada tabel "pegawai".
 - Hasil hitungan ini diberikan alias "Jumlahpegawai".
2. SUM(Gaji) AS Totalgaji:
 - Fungsi SUM(Gaji) menjumlahkan seluruh nilai gaji pada tabel "pegawai".
 - Hasil penjumlahan ini diberikan alias "Totalgaji".
3. AVG(Gaji) AS RataGaji:
 - Fungsi AVG(Gaji) menghitung rata-rata nilai gaji pada tabel "pegawai".
 - Hasil perhitungan rata-rata ini diberikan alias "RataGaji".
4. MAX(Gaji) AS GajiMaks:
 - Fungsi MAX(Gaji) mencari nilai gaji terbesar pada tabel "pegawai".
 - Hasil nilai terbesar ini diberikan alias "GajiMaks".
5. MIN(Gaji) AS GajiMin:
 - Fungsi MIN(Gaji) mencari nilai gaji terkecil pada tabel "pegawai".
 - Hasil nilai terkecil ini diberikan alias "GajiMin".
6. FROM pegawai:
 - Menentukan tabel sumber data, yaitu tabel "pegawai".Jadi, query ini akan menghasilkan sebuah tabel dengan 5 kolom:
7. Jumlahpegawai: Jumlah total pegawai
8. Totalgaji: Total seluruh gaji pegawai
9. RataGaji: Rata-rata gaji pegawai
10. GajiMaks: Gaji terbesar
11. GajiMin: Gaji terkecil

HASIL:

```
MariaDB [company_indrawan]> SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji,
-> AVG(Gaji) AS RataGaji, MAX(Gaji) AS GajiMaks, Min(gaji) AS Gajmin
-> FROM pegawai;
```

Jumlahpegawai	Totalgaji	RataGaji	GajiMaks	Gajmin
9	30583000	3398111.1111	6250000	1725000

```
1 row in set (0.02 sec)
```

SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji, NoCab HAVING SUM(Gaji) <= 2600000

sintaks

```
AS Jumlahpegawai, SUM(Gaji) AS Totalgaji,
-> AVG(Gaji) AS RataGaji, MAX(Gaji) AS GajiMaks, Min(gaji) AS Gajmin
-> FROM pegawai
-> WHERE Jabatan = 'Staf' OR Jabatan = 'Sales'
-> GROUP BY NoCab HAVING SUM(Gaji) <= 2600000 ;
```

PENJELASAN

1. SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji, AVG(Gaji) AS RataGaji, MAX(Gaji) AS GajiMaks, MIN(Gaji) AS GajiMin:
 - Ini sama dengan penjelasan sebelumnya, menggunakan fungsi agregat untuk menghitung jumlah pegawai, total gaji, rata-rata gaji, gaji maksimum, dan gaji minimum.
2. FROM pegawai:
 - Mengambil data dari tabel "pegawai".
3. WHERE Jabatan = 'Staf' OR Jabatan = 'Sales':
 - Memfilter data hanya untuk pegawai yang memiliki jabatan "Staf" atau "Sales".
4. GROUP BY NoCab:
 - Mengelompokkan hasil perhitungan berdasarkan nomor cabang (NoCab).
5. HAVING SUM(Gaji) <= 2600000:
 - Memfilter kelompok-kelompok data berdasarkan total gaji, hanya menampilkan kelompok yang memiliki total gaji <= Rp 2.600.000.

Jadi, query ini akan menghasilkan tabel yang berisi:
6. Jumlahpegawai: Jumlah pegawai berjabatan Staf atau Sales di setiap cabang.
7. Totalgaji: Total gaji pegawai berjabatan Staf atau Sales di setiap cabang yang total gajinya

<= Rp 2.600.000.

8. RataGaji: Rata-rata gaji pegawai berjabatan Staf atau Sales di setiap cabang yang total gajinya <= Rp 2.600.000.
9. GajiMaks: Gaji maksimum pegawai berjabatan Staf atau Sales di setiap cabang yang total gajinya <= Rp 2.600.000.
10. GajiMin: Gaji minimum pegawai berjabatan Staf atau Sales di setiap cabang yang total gajinya <= Rp 2.600.000.

HASIL:

```
MariaDB [company_indrawan]> SELECT COUNT(NIP) AS Jumlahpegawai, SUM(Gaji) AS Totalgaji,  
-> AVG(Gaji) AS RataGaji, MAX(Gaji) AS GajiMaks, Min(gaji) AS Gajmin  
-> FROM pegawai  
-> WHERE Jabatan = 'Staf' OR Jabatan ='Sales'  
-> GROUP BY NoCab HAVING SUM(Gaji) <= 2600000 ;
```

Jumlahpegawai	Totalgaji	RataGaji	GajiMaks	Gajmin
1	2500000	2500000.0000	2500000	2500000
1	1950000	1950000.0000	1950000	1950000

2 rows in set (0.02 sec)