JOBSHEET

**PRAKTIKUM BASIS DATA**

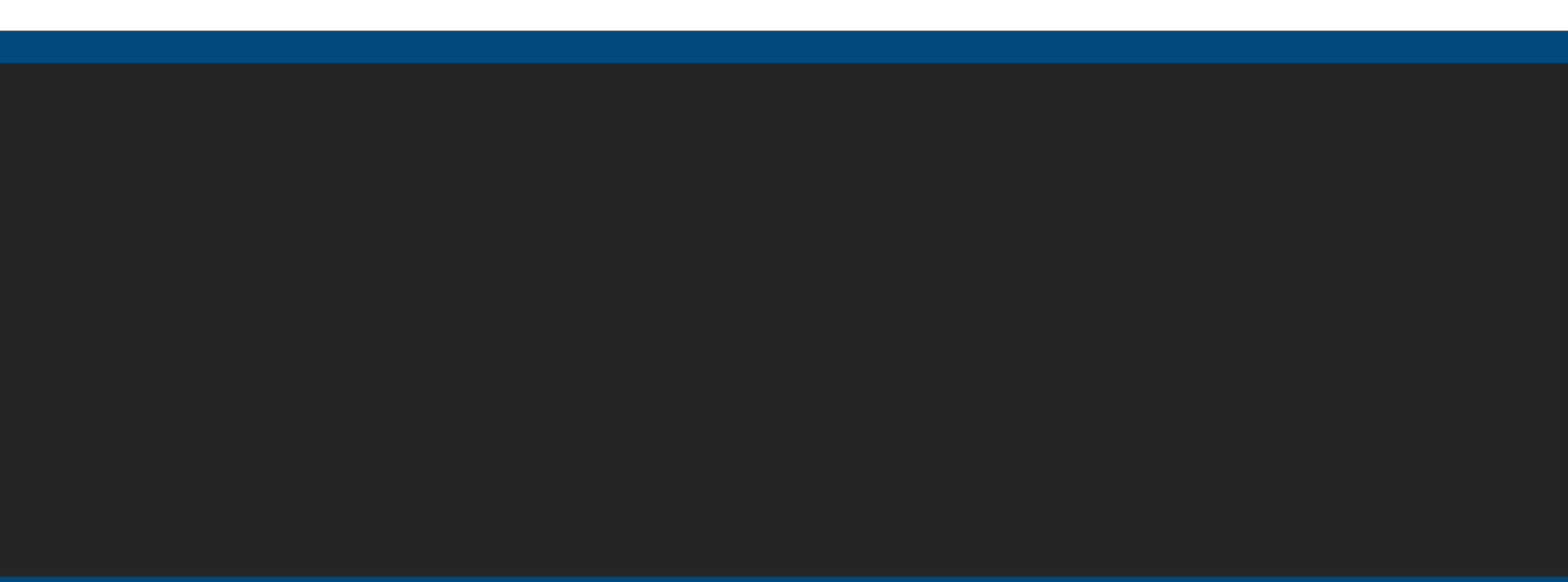
**Jurusan Teknologi Informasi**

**POLITEKNIK NEGERI MALANG**

**PERTEMUAN 12**

**MYSQL – DATA RETRIEVAL LANGUAGE**

**Team Teaching:**



Dwi Puspitasari, S.Kom., M.Kom.

Yan Watequlis Syaifudin, ST., MMT., PhD. Annisa Puspa Kirana, S. Kom, M.Kom Yoppy Yunhasnawa, S.ST., M.Sc.

Candra Bella VistaRetno Damayanti, S.Pd., M.T. **1**

|  |  |
| --- | --- |
|  | Jurusan Teknologi Informasi Politeknik Negeri Malang **Jobsheet-12: MySQL – Data Retrieval Language (DRL) Mata Kuliah Basis Data**  Pengampu: Tim Ajar Basis Data  NAME : Fitri cahyaniati(11)  *Mei 2024* |

# Topik

Data Retrieval Language (DRL) pada DBMS MySQL

# Tujuan

Mahasiswa diharapkan dapat memahami penggunaan dan dapat membuat SQL statement SELECT dengan berbagai klausa.

# Pendahuluan

DRL atau DSL (Data Selection Language) adalah set perintah yang dgunakan untuk mengambil data dari server basis data. DRL terdiri dari klausa yang dapat dikembangkan lebih lanjut, antara lain:

1. **FROM** : Memilih nama sebuah tabel
2. **WHERE** : Menunjukkan secara spesifik suatu row/baris yang akan dicari
3. **GROUP BY** : Menyusung data ke dalam grup
4. **HAVING :** Memilih diantara kelompok-kelompok data yang didefinisikan klausa GROUP BY
5. **ORDER BY** : Secara spesifik digunakan untuk menentukan suatu baris pada kolom tertentu
6. **AS** : Memberikan nama alias sementara untuk tabel atau kolom

DRL lebih diterapkan dalam beberapa hal dalam beberapa hal seperti :

# QUERY

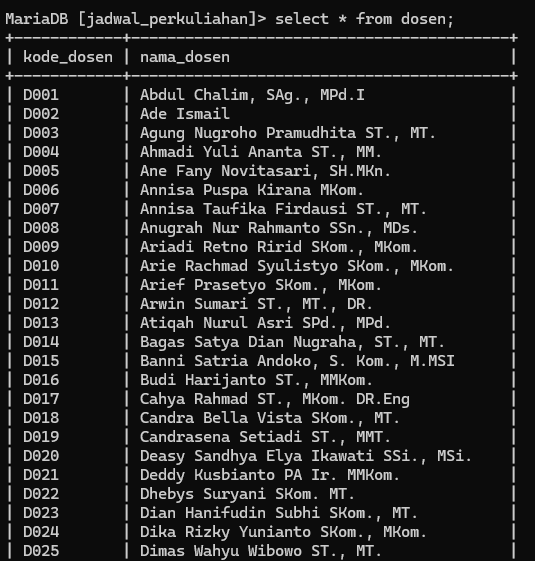
1. **SUB-QUERY**

# FUNGSI AGREGASI

**QUERY**

Query merupakan operasi yang melibatkan satu atau lebih tabel untuk melakukan retrival data. Pengambilan data dilakukan dengan beberapa klausa berikut :

1. SELECT \* untuk memilih semua kolom Format

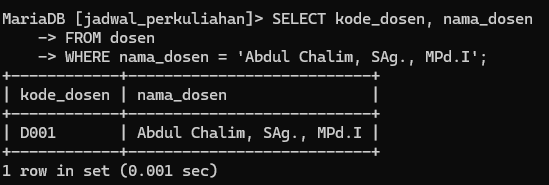
SELECT \* FROM *table\_name*

1. SELECT dengan WHERE untuk menampilkan baris dengan suatu kondisi Format

SELECT *column1*, *column2,...*

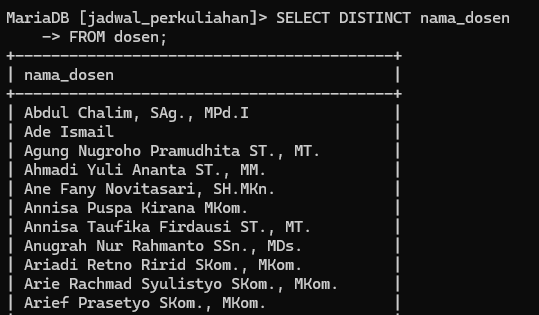
FROM *table\_name*

WHERE *condition*;



1. SELECT dengan DISTINCT untumenampilkan data dengan eliminasi data yang sama (duplicate) Format

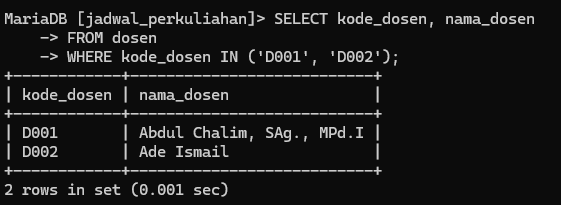
SELECT DISTINCT *column* FROM *table\_name*;



1. SELECT dengan IN untuk menampilkan data yang spesifik Format

SELECT *column\_name(s)*

FROM *table\_name*

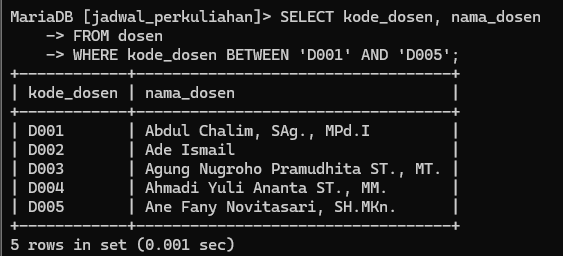
WHERE *column\_name* IN (*value1*, *value2*, ...);

1. SELECT dengan BETWEEN untuk menampilkan data pada jarak (range) tertentu

Format

SELECT *column\_name(s)*

FROM *table\_name*

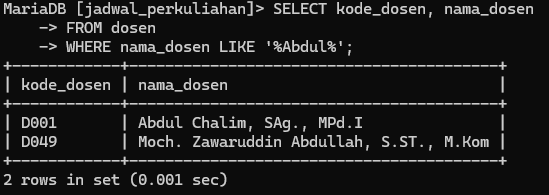
**WHERE *column\_name* BETWEEN *value1* AND *value2;*

1. SELECT dengan LIKE untuk menampilkan data yang memiliki kemiripan dengan keyword yang diinginkan

Format

SELECT *column1,column2,...*

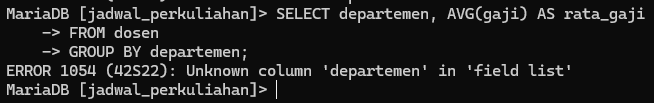
FROM *table\_name*

WHERE *columnN* LIKE *pattern*;

1. SELECT dengan GROUP BY untuk menampilkan susunan data dalam bentuk grup Format

SELECT *column1*, *column2,... condition*

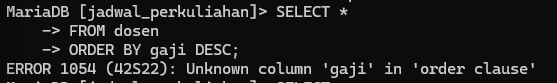
FROM *table\_name*

GROUP BY *column1, column2, ...*;

1. SELECT dengan ORDER BY untuk menampilkan baris secara spesifik dan terurut maju atau mundur Format

SELECT *column1*, *column2, ...*

FROM *table\_name*

ORDER BY *column1, column2, ...* ASC|DESC;

1. SELECT dengan AND, OR and NOT untuk menampilkan data dengan kondisi dan atau atau tidak Format

AND

SELECT *column1*, *column2, ...*

FROM *table\_name*

WHERE *condition1* AND *condition2* AND *condition3 ...*;

OR

SELECT *column1*, *column2, ...*

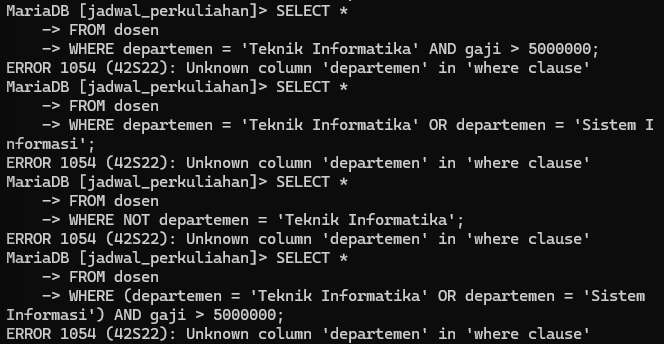
FROM *table\_name*

WHERE *condition1* OR *condition2* OR *condition3 ...*;

NOT

SELECT *column1*, *column2, ...*

FROM *table\_name*

WHERE NOT *condition*;

1. SELECT dengan UNION, INTERSECT dan EXCEPT untuk menampikan data dengan operasi himpunan yang melibatkan lebih dari satu tabel

Format

UNION

*Tanpa duplikasi*

SELECT *column\_name(s)* FROM *table1*

UNION

SELECT *column\_name(s)* FROM *table2*;

*Dengan duplikasi*

SELECT *column\_name(s)* FROM *table1*

UNION ALL

SELECT *column\_name(s)* FROM *table2*;

INTERSECT

*Tanpa duplikasi*

SELECT *column\_name(s)* FROM *table1*

INTERSECT

SELECT *column\_name(s)* FROM *table2*;

*Dengan duplikasi*

SELECT *column\_name(s)* FROM *table1*

INTERSECT ALL

SELECT *column\_name(s)* FROM *table2*;

EXCEPT

*Tanpa duplikasi*

SELECT *column\_name(s)* FROM *table1*

EXCEPT

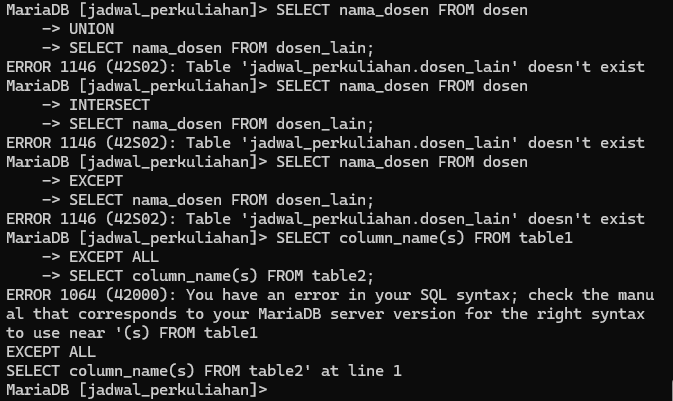
SELECT *column\_name(s)* FROM *table2*;

*Dengan duplikasi*

SELECT *column\_name(s)* FROM *table1*

EXCEPT ALL

SELECT *column\_name(s)* FROM *table2*;



# SUB-QUERY

Sub-query adalah adanya query di dalam query lain. Sub-query juga disebut dengan perintah SELECT bersarang (nested SELECT). Retrival data dalam sub-query dilakukan oleh SELECT untuk menemukan data pada SELECT utama. Klausa GROUP BY dan HAVING juga dapat digunakan untuk retrival data pasa sub- query. Sub-query biasanya digunakan untuk test keanggotaan himpunan atau perbandingan himpunan dan kardinalitas.

Format

*Untuk test keanggotaan* SELECT *column\_name(s)* FROM *table\_name*

WHERE *column\_name* IN (*SELECT STATEMENT*); *atau*

SELECT *column\_name(s)*

FROM *table\_name*

GROUP BY *column1, column2, ...*;

HAVING *column\_name* IN (*SELECT STATEMENT*);

*Untuk perbandingan himpunan*

SELECT *column\_name(s)* FROM *table\_name operator*

(*SELECT STATEMENT*);

# FUNGSI AGREGASI

Fungsi agregasi akan melakukan perhitungan kepada sekumpulan nilai dan menghasilkan suatu nilai tunggal.

Jenis:

1. **AVG** untuk menghitung rata-rata Format

SELECT AVG(*column\_name*) FROM *table\_name*

WHERE *condition*;

1. **MAX** untuk mencari nilai tertinggi Format

SELECT MAX(*column\_name*) FROM *table\_name*

WHERE *condition*;

1. **MIN** untuk mencari nilai terendah Format

SELECT MIN(*column\_name*) FROM *table\_name*

WHERE *condition*;

1. **SUM** untuk menjumlahkan sekumpulan nilai Format

SELECT SUM(*column\_name*) FROM *table\_name*

WHERE *condition*;

1. **COUNT()** untuk menunjukkan jumlah baris pada kolom yang diinginkan Format

SELECT COUNT(*column\_name*) FROM *table\_name*

WHERE *condition*;

# Praktikum – Bagian 1: Membuat Database untuk Percobaan

|  |  |
| --- | --- |
| **Langka h** | **Keterangan** |
| 1 | Studi kasus yang digunakan sama dengan jobsheet DML dengan skema/model relasional/EER diagram dari database berikut. |
|  |  |
| 2 | Skema tersebut adalah skema database pada sebuah sistem informasi penjadwalan di jurusan Teknologi Informasi. Sesuai dengan percobaan pada jobsheet DML. Hapuslah tabel mahasiswa pada database jadwal\_perkuliahan dengan menggunakan perintah **DROP TABLE**, sehingga |

|  |  |
| --- | --- |
|  | menghasilkan *record* keseluruhan tabel sebagai berikut (*record* tidak harus sama persis dengan tampilan ini): |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
| 3 | Setelah selesai membuat database diatas, lanjutkan ke **Praktikum – Bagian 2**. |

**Praktikum – Bagian 2: Percobaan SELECT QUERY**

|  |  |
| --- | --- |
| **Langka h** | **Keterangan** |
| 1 | Untuk menampilkan baris dengan suatu kondisi tertentu, digunakan statement **SELECT** dengan **WHERE**. Eksekusi SQL berikut untuk menampilkan (record) pada tabel **Ruang** yang memiliki kolom nama\_ruang = LKJ1. |
|  |  |
|  |  |
| 2 | Untuk menampilkan data dengan eliminasi data yang sama (duplicate), digunakan statement **SELECT** dengan **DISTINCT**. Eksekusi SQL berikut untuk menampilkan kolom kode\_hari dari tabel **jadwal** yang bernilai tidak sama. |
|  |  |

|  |  |
| --- | --- |
|  |  |
| 3 | Untuk menampilkan data yang spesifik digunakan statement **SELECT** dengan **IN**. Eksekusi SQL berikut untuk menampilkan kolom kode\_ruang, nama\_ruang dan deskripsi\_ruang dari tabel **ruang** yang memiliki nama\_ruang RT01 atau RT10. |
|  |  |
|  |  |
| 4 | Untuk menampilkan data pada jarak (range) tertentu digunakan statement **SELECT** dengan **BETWEEN**. Eksekusi SQL berikut untuk menampilkan kolom kode\_ruang dan nama\_ruang dan deskripsi\_ruang dari tabel **ruang** yang memiliki kode\_ruang antara 0501 dan 0508. |
|  |  |
|  |  |
| 5 | Untuk menampilkan data yang memiliki kemiripan dengan keyword yang diinginkan digunakan **SELECT** dengan **LIKE**. Eksekusi SQL berikut untuk menampilkan kolom kode\_dosen, dan nama\_dosen, pada tabel **dosen** yang memiliki nama dengan huruf awal ‘E’. |

|  |  |
| --- | --- |
|  |  |
|  |  |
| 6 | Untuk menampilkan susunan data dalam bentuk grup, digunakan **SELECT** dengan **GROUP BY**. Eksekusi SQL berikut untuk menampilkan kolom kode\_dosen, kode\_mk, kode\_ruang dan kode\_hari pada tabel **jadwal** yang dikelompokkan berdasarkan kode\_hari. |
|  |  |
|  |  |
| 7 | Untuk menampilkan baris secara spesifik dan terurut maju atau mundur, digunakan **SELECT** dengan **ORDER BY**. Eksekusi SQL berikut untuk menampilkan kolom kode\_jp dan jp\_mulai pada tabel **jp** dari jam yang paling mulai hingga berakhir. |
|  |  |

|  |  |
| --- | --- |
|  |  |
| 8 | Untuk menampilkan data dengan kondisi dan atau atau tidak, digunakan **SELECT** dengan **AND**, **OR** and **NOT**. Eksekusi SQL berikut untuk menampilkan semua kolom pada tabel **jadwal** dengan kode\_hari = ‘001’ dan jp\_mulai = 1. |
|  |  |
|  |  |

|  |  |
| --- | --- |
| 9 | Untuk menampilkan data dari kolom yang terlibat dalam dua tabel dapat digunakan **SELECT** dengan **UNION**. **UNION** secara otomatis akan menghilangkan duplikasi. Eksekusi SQL berikut untuk menampilkan kolom kode\_hari yang ada di tabel **hari** atau **jadwal**. |
|  |  |
|  |  |
| 10 | Untuk menampilkan data dari kolom yang terlibat dalam dua tabel dapat digunakan **SELECT** dengan **UNION ALL**. **UNION** ALL akan menampilkan duplikasi data. Eksekusi SQL berikut untuk menampilkan kolom nik yang yang ada di tabel **penugasan** atau **departemen**. |
|  |  |
|  | Silakan Capture hasil output dari query di atas |
| 11 | Setelah berhasil mengeksekusi SQL tersebut, lanjutkan ke **Praktikum - Bagian 3**. |

# Praktikum - Bagian 3: Percobaan SELECT Sub-Query

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Untuk menampilkan data (test keanggotaan sub-query) yang berasal dari pemilihan tampilan data lain digunakan tambahan statement **IN**. Eksekusi SQL berikut untuk menampilkan kolom kode\_dosen, kode\_mk, kode\_hari, jp\_mulai dan jp\_selesai pada tabel **jadwal** dimana jp\_selesai adalah jp\_selesai yang paling lama pada tabel **jadwal**. |
|  |  |

|  |  |
| --- | --- |
|  |  |
| 2 | Untuk menampilkan data (perbandingan himpunan sub-query) yang berasal dari pemilihan tampilan data lain digunakan tambahan operator. Eksekusi SQL berikut untuk menampilkan kode\_dosen, kode\_mk, kode\_hari, jp\_mulai dan jp\_selesai pada tabel **jadwal** berdasarkan semua record kolom jp\_selesai harus bernilai kurang dari jp\_selesai yang dimiliki oleh jp\_selesai = 6 dari tabel **jadwal**. |
|  |  |

|  |  |
| --- | --- |
|  | Subquery mengembalikan lebih dari 1 baris, artinya subquery Anda mengembalikan lebih dari satu nilai untuk jp\_selesai. Untuk menggunakan operator perbandingan seperti <, >, <=, atau >=, subquery harus mengembalikan tepat satu nilai. |
| 3 | Setelah berhasil mengeksekusi SQL tersebut, lanjutkan ke **Praktikum - Bagian 4**. |

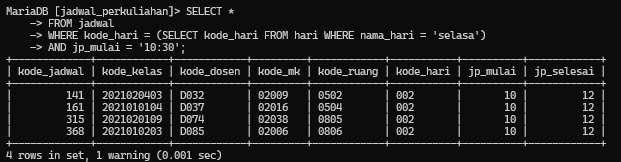
**Praktikum - Bagian 4: Percobaan FUNGSI AGREGASI**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Untuk menghitung rata-rata, digunakan tambahan statement **AVG.** Eksekusi SQL berikut untuk menampilkan rata-rata durasi perkuliahan dalam satu jadwal perkuliahan sesuai kolom jp\_mulai, jp\_selesai dari tabel **jadwal**. |
|  |  |
|  |  |
| 2 | Untuk menghitung nilai tertinggi, digunakan tambahan statement **MAX.** Eksekusi SQL berikut untuk menampilkan dimulainya jam perkuliahan yang paling siang sesuai kolom jp\_mulai dari tabel **jadwal**. |
|  |  |

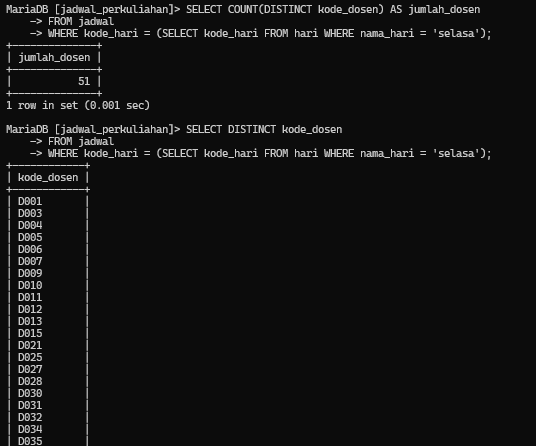
|  |  |
| --- | --- |
|  |  |
| 3 | Untuk menghitung nilai terendah, digunakan tambahan statement **MIN.** Eksekusi SQL berikut untuk menampilkan berakhirnya jam perkuliahan yang paling cepat sesuai kolom jp\_selesai dari tabel **jadwal**. |
|  |  |
|  |  |
| 4 | Untuk menghitung total jam ajar digunakan tambahan statement **SUM.** Eksekusi SQL berikut untuk menampilkan total jam ajar dosen dengan kode ‘D001’ sesuai jp\_mulai, jp\_selesai, dan kode\_dosen dari tabel jadwal. |
|  |  |
|  | Coba tampikan jadwal dosen dengan kode ‘D001’ dan pastikan apakah benar total jam ajar  dosen tersebut sesuai dengan perintah sum dengan nilai 14 pada gambar di atas. |
| 5 | Untuk menunjukkan jumlah baris pada kolom yang diinginkan digunakan COUNT(). Eksekusi SQL berikut untuk menampilkan jumlah baris pada kolom kode\_ruang dari tabel **ruang**. |
|  |  |
|  |  |
| 6 | Lanjutkan ke bagian **Tugas**! |

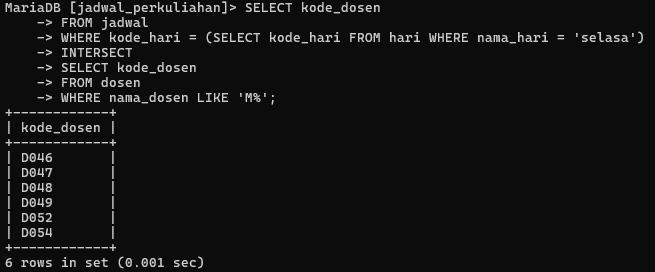
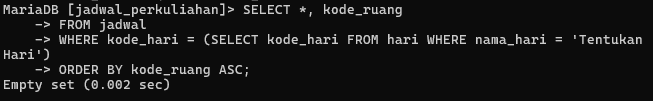
# Tugas

1. Himpunlah *screenshot* sintaksis SQL berikut hasil SELECT yang Anda dapat di komputer Anda pada bagian **Praktikum** dalam sebuah laporan dengan format **PDF**!
2. Bagaimana sintaksis untuk menampilkan semua kolom jadwal pada hari kamis yang dimulai pada pukul 08:40 ? Tunjukkan outputnya!



1. Berapakah jumlah dosen yang mengajar pada hari selasa! Tunjukkan sintaksis dan outputnya! Kemudian tampilkan list semua kode dosen yang sesuai dengan kondisi tersebut! Tunjukkan sintaksis dan outputnya!



1. Tampilkan kode dosen yang mengajar pada hari kamis dari tabel jadwal yang nama depanya di awali dengan huruf A. gunakan operasi intersect! Tunjukkan sintaksis dan output dari pencarian tersebut.
2. Tampilkan kode\_ruang tertentu pada hari tertentu dengan diurutkan berdasarkan dengan kode jam yang ter awal.

***-- Selamat Mengerjakan –***

# Daftar Pustaka

* Dwi Puspitasari, S.Kom, “**Buku Ajar Dasar Basis Data**”, *Program Studi Manajemen Informatika Politeknik Negeri Malang*, 2012.
* [http://www.tutorialride.com](http://www.tutorialride.com/)
* <https://www.w3schools.com/>