Pengaksesan Basis Data

SQL [2]

Fak. Teknik Prodi Teknik Informatika
Universitas Pasundan

Caca E. Supriana, S.Si., MT.
caca.e.supriana@unpas.ac.id
caca-e-supriana.blogspot.com

SQL Statements

- **SELECT** → memilih satu, beberapa atau semua field dari satu atau lebih tabel
- **FROM** → memilih tabel yang field nya dipilih dalam perintah select di atas
- [**WHERE**] → merupakan syarat dari field, tabel atau relasi antar tabel
- [**GROUP BY**] → mengelompokan field mengacu pada field tertentu

MySQL Query Commands

1. *Use Database*
2. *Create Table*
3. *Insert Query*
4. *Alter Query*
5. *Update Query*
6. *Functions : Min, Max, Avg, Sum, Sqrt, Date, String*
7. *Delete Query*
8. *Drop : Database, Table*
9. *And / Or*
10. *Clause : In, Between, Like, Count, Group by*

Use Database & Table

- Syntax :

```
CREATE TABLE table_name (column_name column_type);
```

- Example :

NOT NULL is being used because we do not want this field to be **NULL**. **AUTO_INCREMENT** to go ahead and add the next available number to the id field. **PRIMARY KEY** is used to define a column as primary key.

```
root@host# mysql -u root -p
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> CREATE TABLE tutorials_tbl(
   -> tutorial_id INT NOT NULL AUTO_INCREMENT,
   -> tutorial_title VARCHAR(100) NOT NULL,
   -> tutorial_author VARCHAR(40) NOT NULL,
   -> submission_date DATE,
   -> PRIMARY KEY ( tutorial_id )
   -> );
Query OK, 0 rows affected (0.16 sec)
mysql>
```

Insert Query

- Syntax :

```
INSERT INTO table_name ( field1, field2,...fieldN )
                       VALUES
                       ( value1, value2,...valueN );
```

- Example :

To **insert string data types**, it is required to keep all the values into double or single quote, for example: "**value**".

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> INSERT INTO tutorials_tbl
    ->(tutorial_title, tutorial_author, submission_date)
    ->VALUES
    ->("Learn PHP", "John Poul", NOW());
Query OK, 1 row affected (0.01 sec)
mysql> INSERT INTO tutorials_tbl
    ->(tutorial_title, tutorial_author, submission_date)
    ->VALUES
    ->("Learn MySQL", "Abdul S", NOW());
Query OK, 1 row affected (0.01 sec)
mysql> INSERT INTO tutorials_tbl
    ->(tutorial_title, tutorial_author, submission_date)
    ->VALUES
    ->("JAVA Tutorial", "Sanjay", '2007-05-06');
```

Alter Query

DROP an existing column i from above MySQL table. **DROP** will not work if the column is the only one left in the table. To add a column, use **ADD** and specify the column definition.

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> create table testalter_tbl
    -> (
    -> i INT,
    -> c CHAR(1)
    -> );
Query OK, 0 rows affected (0.05 sec)
mysql> SHOW COLUMNS FROM testalter_tbl;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| i     | int(11) | YES  |     | NULL    |       |
| c     | char(1) | YES  |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> ALTER TABLE testalter_tbl  DROP i;
```

```
mysql> ALTER TABLE testalter_tbl ADD i INT;
```

```
mysql> SHOW COLUMNS FROM testalter_tbl;
+-------+---------+------+-----+---------+-------+
| Field | Type    | Null | Key | Default | Extra |
+-------+---------+------+-----+---------+-------+
| c     | char(1) | YES  |     | NULL    |       |
| i     | int(11) | YES  |     | NULL    |       |
+-------+---------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

Update Query

- Syntax :

```
UPDATE table_name SET field1=new-value1, field2=new-value2
[WHERE Clause]
```

- Example :

- Update one or more fields altogether.
- Specify any condition using WHERE clause.
- Update values in a single table at a time.

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> UPDATE tutorials_tbl
    -> SET tutorial_title='Learning JAVA'
    -> WHERE tutorial_id=3;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

Functions : Min

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

MIN aggregate function allows us to select the lowest (**minimum**) value for a certain column.

```
mysql> SELECT MIN(daily_typing_pages)
    -> FROM employee_tbl;
+--------------------------+
| MIN(daily_typing_pages)  |
+--------------------------+
|                      100 |
+--------------------------+
1 row in set (0.00 sec)
```

Functions : Max

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

MAX aggregate function allows us to select the highest (**maximum**) value for a certain column

```
mysql> SELECT MAX(daily_typing_pages)
    -> FROM employee_tbl;
+--------------------------+
| MAX(daily_typing_pages)  |
+--------------------------+
|                      350 |
+--------------------------+
1 row in set (0.00 sec)
```

Functions : Avg

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

AVG aggregate function selects the **average** value for certain table column.

```
mysql> SELECT AVG(daily_typing_pages)
    -> FROM employee_tbl;
+-------------------------+
| AVG(daily_typing_pages) |
+-------------------------+
|                230.0000 |
+-------------------------+
1 row in set (0.03 sec)
```

Functions : Sum

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

SUM aggregate function allows selecting the **total** for a **numeric** column.

```
mysql> SELECT SUM(daily_typing_pages)
    -> FROM employee_tbl;
+--------------------------+
| SUM(daily_typing_pages)  |
+--------------------------+
|                     1610 |
+--------------------------+
1 row in set (0.00 sec)
```

Functions : Sqrt

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Used to generate a **square root** of a given number.

```
mysql> SELECT name, SQRT(daily_typing_pages)
    -> FROM employee_tbl;
+------+--------------------------+
| name | SQRT(daily_typing_pages) |
+------+--------------------------+
| John |                15.811388 |
| Ram  |                14.832397 |
| Jack |                13.038405 |
| Jack |                10.000000 |
| Jill |                14.832397 |
| Zara |                17.320508 |
| Zara |                18.708287 |
+------+--------------------------+
7 rows in set (0.00 sec)
```

Functions : Date

Example :
%W Weekday name (Sunday..Saturday)
%Y Year, numeric, four digits
%M Month name (January..December)

```
mysql> SELECT DATE_FORMAT('1997-10-04 22:23:00', '%W %M %Y');
+-----------------------------------------------------------+
| DATE_FORMAT('1997-10-04 22:23:00', '%W %M %Y')            |
+-----------------------------------------------------------+
| Saturday October 1997                                     |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT DAYNAME('1998-02-05');
+-----------------------------------------------------------+
| DAYNAME('1998-02-05')                                     |
+-----------------------------------------------------------+
| Thursday                                                  |
+-----------------------------------------------------------+
1 row in set (0.00 sec)
```

Functions : String

```
mysql> SELECT SUBSTRING('Quadratically',5);
+-------------------------------------------------------------+
| SSUBSTRING('Quadratically',5)                               |
+-------------------------------------------------------------+
| ratically                                                   |
+-------------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT SUBSTRING('foobarbar' FROM 4);
+-------------------------------------------------------------+
| SUBSTRING('foobarbar' FROM 4)                               |
+-------------------------------------------------------------+
| barbar                                                      |
+-------------------------------------------------------------+
1 row in set (0.00 sec)
```

```
mysql> SELECT UPPER('Allah-hus-samad');
+-------------------------------------------------------------+
| UPPER('Allah-hus-samad')                                    |
+-------------------------------------------------------------+
| ALLAH-HUS-SAMAD                                             |
+-------------------------------------------------------------+
1 row in set (0.00 sec)
```

Delete Query

- Syntax :

```
DELETE FROM table_name [WHERE Clause]
```

- Example :

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> DELETE FROM tutorials_tbl WHERE tutorial_id=3;
Query OK, 1 row affected (0.23 sec)
```

Drop : Database

- Syntax :

```
[root@host]# mysqladmin -u root -p drop TUTORIALS
Enter password:******
```

- Example :

```
Dropping the database is potentially a very bad thing to do.
Any data stored in the database will be destroyed.

Do you really want to drop the 'TUTORIALS' database [y/N] y
Database "TUTORIALS" dropped
```

Drop : Table

- Syntax :

```
DROP TABLE table_name ;
```

- Example :

```
root@host# mysql -u root -p
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> DROP TABLE tutorials_tbl
Query OK, 0 rows affected (0.8 sec)
mysql>
```

Or

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

```
mysql>SELECT * FROM employee_tbl
    ->WHERE daily_typing_pages= 250 OR
    ->daily_typing_pages= 220 OR  daily_typing_pages= 170;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    4 | Jill | 2007-04-06 |                220 |
+------+------+------------+--------------------+
4 rows in set (0.02 sec)
```

Clause : In

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

IN clause to replace many **OR** conditions

```
mysql> SELECT * FROM employee_tbl
    -> WHERE daily_typing_pages IN ( 250, 220, 170 );
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    4 | Jill | 2007-04-06 |                220 |
+------+------+------------+--------------------+
4 rows in set (0.02 sec)
```

Clause : Between

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Fetch records with conditions daily_typing_pages more than 170 and equal and less than 300 and equal.

```
mysql> SELECT * FROM employee_tbl
    -> WHERE daily_typing_pages BETWEEN 170 AND 300;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
+------+------+------------+--------------------+
5 rows in set (0.03 sec)
```

Like

- Syntax :

```
SELECT field1, field2,...fieldN table_name1, table_name2...
WHERE field1 LIKE condition1 [AND [OR]] filed2 = 'somevalue'
```

- Example :

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> SELECT * from tutorials_tbl
    -> WHERE tutorial_author LIKE '%jay';
+-------------+----------------+-----------------+-----------------+
| tutorial_id | tutorial_title | tutorial_author | submission_date |
+-------------+----------------+-----------------+-----------------+
|           3 | JAVA Tutorial  | Sanjay          | 2007-05-21      |
+-------------+----------------+-----------------+-----------------+
1 rows in set (0.01 sec)
```

Clause : Count

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

COUNT function is the simplest function and very useful in counting the number of records, which are expected to be returned by a **SELECT** statement.

```
mysql>SELECT COUNT(*) FROM employee_tbl ;
+----------+
| COUNT(*) |
+----------+
|        7 |
+----------+
1 row in set (0.01 sec)
```

Clause : Group By

```
mysql> SELECT * FROM employee_tbl;
+------+------+------------+--------------------+
| id   | name | work_date  | daily_typing_pages |
+------+------+------------+--------------------+
|    1 | John | 2007-01-24 |                250 |
|    2 | Ram  | 2007-05-27 |                220 |
|    3 | Jack | 2007-05-06 |                170 |
|    3 | Jack | 2007-04-06 |                100 |
|    4 | Jill | 2007-04-06 |                220 |
|    5 | Zara | 2007-06-06 |                300 |
|    5 | Zara | 2007-02-06 |                350 |
+------+------+------------+--------------------+
7 rows in set (0.00 sec)
```

Use **GROUP BY** to group values from a column

```
mysql> SELECT name, COUNT(*)
    -> FROM   employee_tbl
    -> GROUP BY name;
+------+----------+
| name | COUNT(*) |
+------+----------+
| Jack |        2 |
| Jill |        1 |
| John |        1 |
| Ram  |        1 |
| Zara |        2 |
+------+----------+
5 rows in set (0.04 sec)
```

Exercise (1)

```
mysql> CREATE TABLE LATIHAN1(
    -> ID_BRG CHAR(5) NOT NULL,
    -> NAMA_BRG VARCHAR(25) NOT NULL,
    -> JUMLAH_BRG INT,
    -> TGL_KIRIM DATE,
    -> TAHUN_BERLAKU YEAR,
    -> PRIMARY KEY (ID_BRG));
Query OK, 0 rows affected (0.06 sec)

mysql> DESC LATIHAN1;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| ID_BRG        | char(5)     | NO   | PRI |         |       |
| NAMA_BRG      | varchar(25) | NO   |     |         |       |
| JUMLAH_BRG    | int(11)     | YES  |     | NULL    |       |
| TGL_KIRIM     | date        | YES  |     | NULL    |       |
| TAHUN_BERLAKU | year(4)     | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

Exercise (2)

```
mysql> INSERT INTO LATIHAN1
    -> (ID_BRG, NAMA_BRG, JUMLAH_BRG, TGL_KIRIM, TAHUN_BERLAKU)
    -> VALUES
    -> ("BRG01","PRINTER",56,"2016-07-23","2012");
Query OK, 1 row affected (0.05 sec)

mysql> SELECT * FROM LATIHAN1;
+--------+----------+------------+------------+---------------+
| ID_BRG | NAMA_BRG | JUMLAH_BRG | TGL_KIRIM  | TAHUN_BERLAKU |
+--------+----------+------------+------------+---------------+
| BRG01  | PRINTER  |         56 | 2016-07-23 |          2012 |
+--------+----------+------------+------------+---------------+
1 row in set (0.00 sec)
```