Name - Aparna Bharti

Batch - Data Engineering batch-1

**TOPIC - SQL COMMANDS**

**SQL\_DAY.Docs**

**Create table Customer**

The CREATE TABLE is a DDL statement which is used to create tables in the database.The table gets created as soon as the CREATE TABLE script is executed and is ready to hold the data onwards.The user must have the CREATE TABLE system privilege to create the table in its own schema.But to create a table in any user's schema, user must have CREATE ANY TABLE schema.

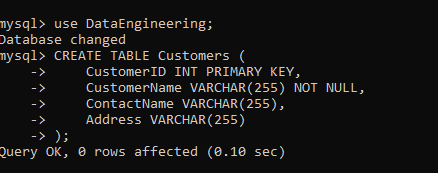
**Syntax -**

CREATE TABLE [IF NOT EXISTS] table\_name( column\_definition1, column\_definition2,

........,

table\_constraints

);



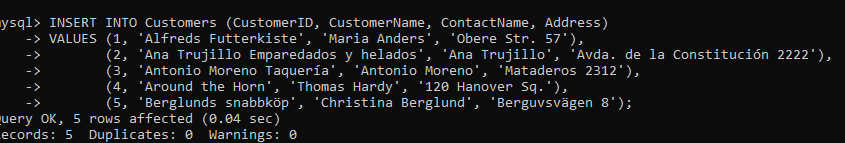
**Insert values**

MySQL INSERT statement is used to store or add data in MySQL table within the database. We can perform insertion of records in two ways using a single query in MySQL:

1. Insert record in a single row
2. Insert record in multiple rows

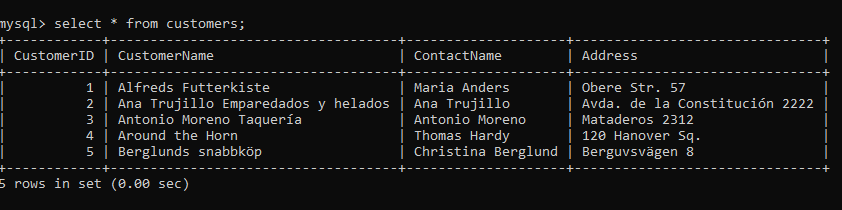
**Syntax -**

INSERT INTO table\_name ( field1, field2,...fieldN ) VALUES ( value1, value2,...valueN );

****

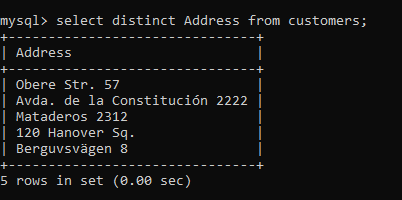
**Select all columns**

If you want to return all columns, without specifying every column name, you can use the SELECT \* syntax:

****

**Select DISTINCT column\_name**

The SELECT DISTINCT statement is used to return only distinct (different) values.

****

**Update Record**

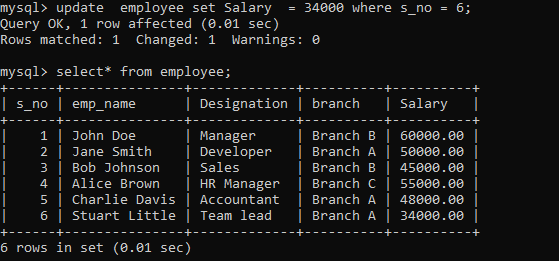
The UPDATE statement is used to modify the existing records in a table.

### **Syntax**

UPDATE *table\_name*

SET *column1* = *value1*, *column2* = *value2*, ...

WHERE *condition*;

****

**Alter Record**

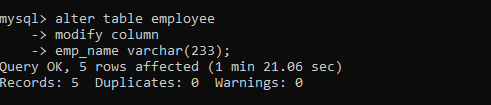
The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

**Syntax**

ALTER TABLE *table\_name*

ADD *column\_name datatype*;

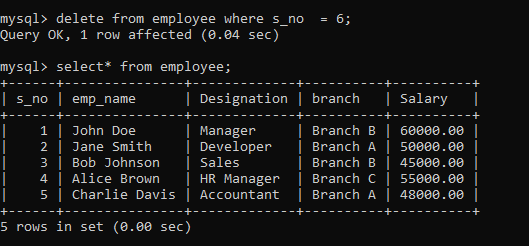


**Delete record**

The DELETE statement is used to delete existing records in a table.

**Syntax**

DELETE FROM *table\_name* WHERE *condition*;



**Count no of employee has salary same (there should at least be one employee has same salary as other)**

The GROUP BY statement groups rows that have the same values into summary rows, like "find the number of customers in each country".

The GROUP BY statement is often used with aggregate functions (COUNT(), MAX(), MIN(), SUM(), AVG()) to group the result-set by one or more columns.

### **Syntax**

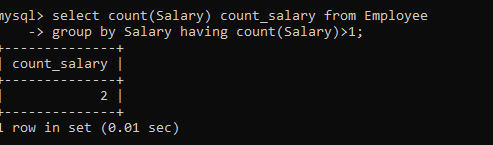
SELECT *column\_name(s)*

FROM *table\_name*

WHERE *condition*

GROUP BY *column\_name(s)*

ORDER BY *column\_name(s);*

****

**Count no of employee has salary same**

The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

### **Syntax**

SELECT *column\_name(s)*

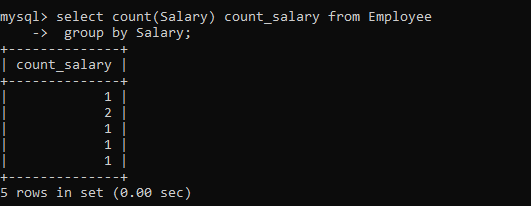
FROM *table\_name*

WHERE *condition*

GROUP BY *column\_name(s)*

HAVING *condition*

ORDER BY *column\_name(s);*

****

**Commit and Rollback**

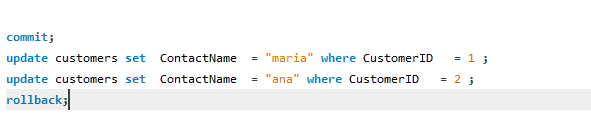
COMMIT in SQL is a transaction control language that is used to permanently save the changes done in the transaction in tables/databases. The database cannot regain its previous state after its execution of a commit.

ROLLBACK in SQL is a transactional control language that is used to undo the transactions that have not been saved in the database. The command has only been used to undo changes since the last COMMIT.

**Syntax**

Commit;

Rollback;

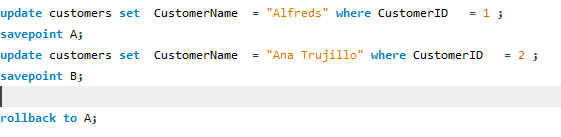
****

**Savepoint**

A save point is a logical rollback point within a transaction. When you set a save point, whenever an error occurs past a save point, you can undo the events you have done up to the save point using the rollback.

**Syntax**

SAVEPOINT identifier

****

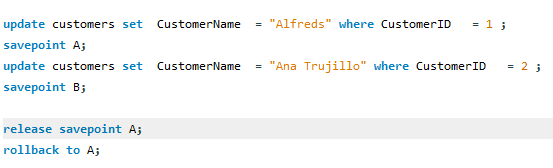
**Release Savepoint**

The RELEASE SAVEPOINT command is used to remove an existing SAVEPOINT.

The syntax for a RELEASE SAVEPOINT command is as follows.

**Syntax**

Release Savepoint A

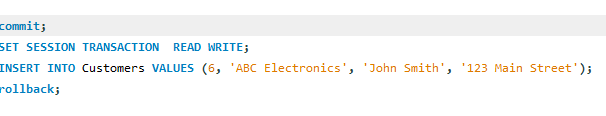
****

**Set Transaction**

The SET TRANSACTION command can be used to initiate a database transaction. This command is used to specify characteristics for the transaction that follows. For example, you can specify a transaction to be read only or read and write.

**Syntax**

Set transaction read only;



**Set operators in Mysql**

SET operators are special type of operators which are used to combine the result of two queries.

Operators covered under SET operators are:

1. **UNION**
2. **UNION ALL**
3. **INTERSECT**
4. **MINUS**

**Union**

The UNION operator is used to combine the result-set of two or more SELECT statements.

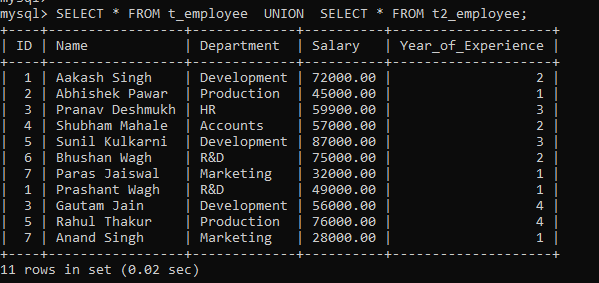
* Every SELECT statement within UNION must have the same number of columns
* The columns must also have similar data types
* The columns in every SELECT statement must also be in the same order

**Syntax**

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

UNION

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



**Union All**

MySQL UNION ALL operator is a union query command which syndicates multiple SELECT statements' results into a single result row. Like, the MySQL UNION operator, it is also a useful command in MySQL database to combine more than two of the output set provided by using SELECT queries

**Syntax**

SELECT expression1, expression2, ... expression\_n

FROM tables

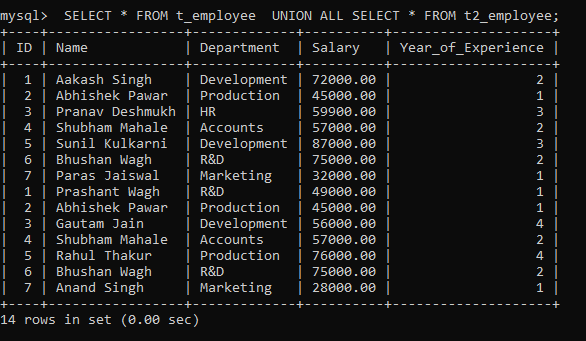
[WHERE conditions]

UNION ALL

SELECT expression1, expression2, ... expression\_n

FROM tables

[WHERE conditions];



**Intersect**

The INTERSECT operator is a kind of SET operation in SQL that includes UNION, UNION ALL, MINUS, and INTERSECT. The INTERSECT operator returns the distinct (common) elements in two sets or common records from two or more tables. In other words, it compares the result obtained by two queries and produces unique rows, which are the result returned by both queries.

**The number and order of columns in both queries has to be the same**. The data types of corresponding columns from both the select queries must be compatible with each other.

**Syntax**

SELECT column\_lists FROM table\_name WHERE condition

INTERSECT

SELECT column\_lists FROM table\_name WHERE condition;



**Minus**

The MINUS operator is a kind of SET operation in SQL which also includes INTERSECT, UNION, and UNION ALL. The MINUS operator returns the unique element from the first table/set, which is not found in the second table/set

**Syntax**

SELECT column\_lists FROM table\_name WHERE (condition)

MINUS

SELECT column\_lists FROM table\_name WHERE (condition);



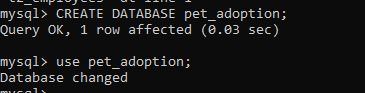
**SQL\_D.Docs**

**Set Up Dog Shelter Database: CREATE DATABASE**

The CREATE DATABASE statement is used to create a new SQL database.

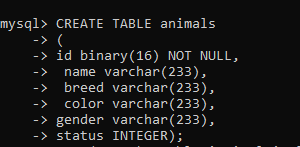
**Syntax**

CREATE DATABASE *databasename*;

****

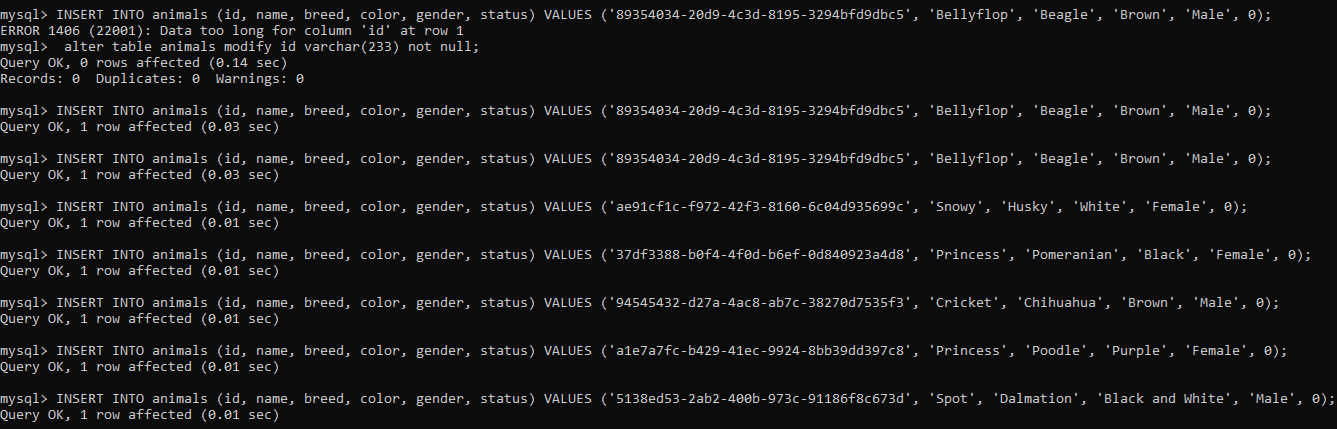
**A Table for Animals:CREATE TABLE & UUID**

MySQL UUID or Universally Unique Identifier is a 128-bit value that is unique across the globe and is generated using an algorithm. The UUID value is a hexadecimal string of 32 characters, separated by hyphens in a specific format 8-4-4-4-12.

****

****

**Add Dogs to Database: INSERT**

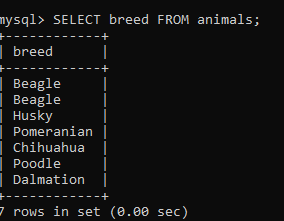
****

**Retrieve List of Dogs: SELECT \* FROM**

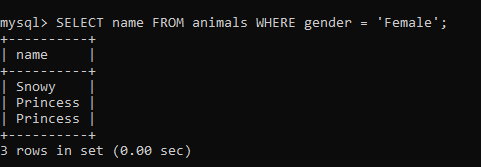
SELECT \* FROM animals;



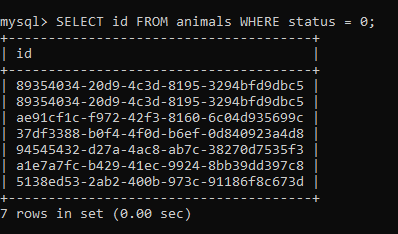
SELECT breed FROM animals;



SELECT name FROM animals WHERE gender = 'Female';

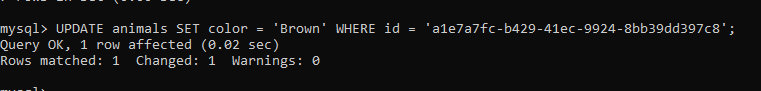


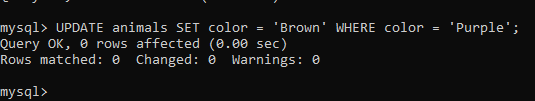
SELECT id FROM animals WHERE status = 0;



**Update Dogs’ Information: UPDATE & DELETE FROM**

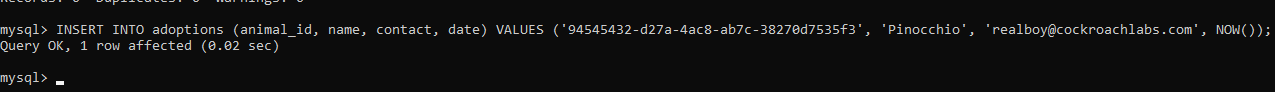
UPDATE animals SET color = 'Brown' WHERE id = 'a1e7a7fc-b429-41ec-9924-8bb39dd397c8';



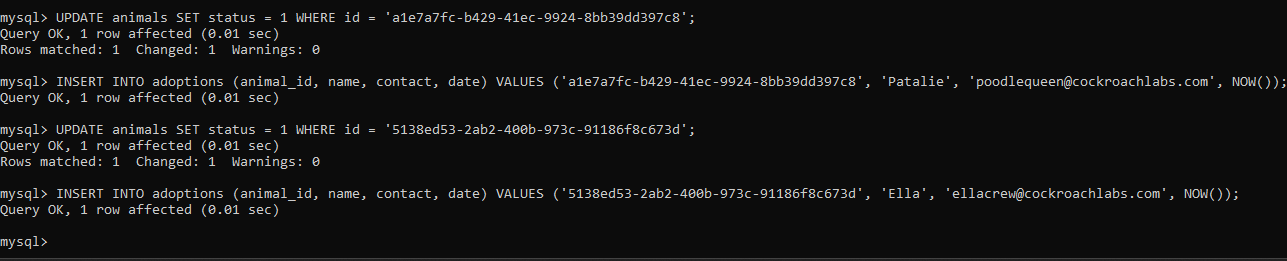
****

**Our First Shelter Adoption: UPDATE & INSERT**

INSERT INTO adoptions (animal\_id, name, contact, date) VALUES ('94545432-d27a-4ac8-ab7c-38270d7535f3', 'Pinocchio', 'realboy@cockroachlabs.com', NOW());

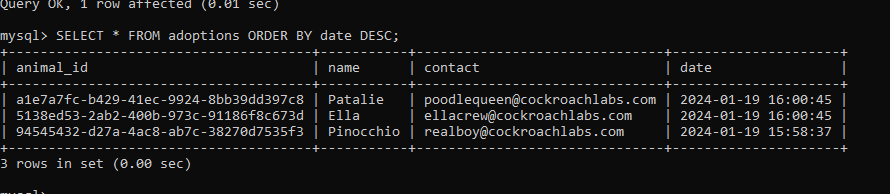


**Record Adoptions**

****

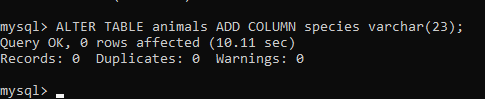
**Verify Changes: ORDER BY**

SELECT \* FROM adoptions ORDER BY date DESC;



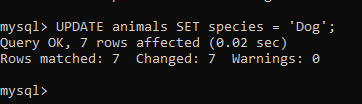
**Add a Table Column: ALTER TABLE**

ALTER TABLE animals ADD COLUMN species varchar(23);



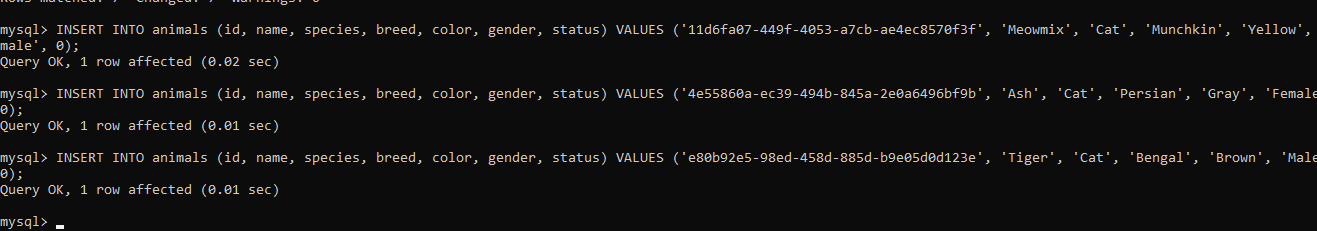
**Set All Current Animals to “Dog”: SET & UPDATE**

UPDATE animals SET species = 'Dog';

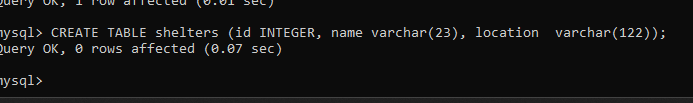


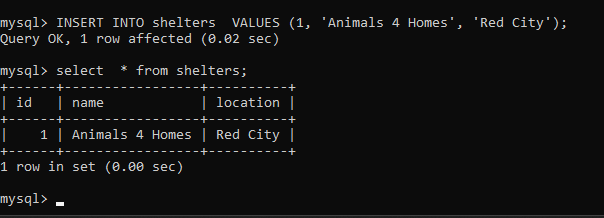
**Add All Cats to the Database**

****



**A Table for Shelters: ALTER TABLE … ADD COLUMN**

****

****

**Index**

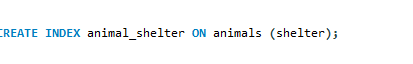
The CREATE INDEX statement is used to create indexes in tables.

Indexes are used to retrieve data from the database more quickly than otherwise. The users cannot see the indexes, they are just used to speed up searches/queries.

**Syntax**

CREATE INDEX *index\_name*

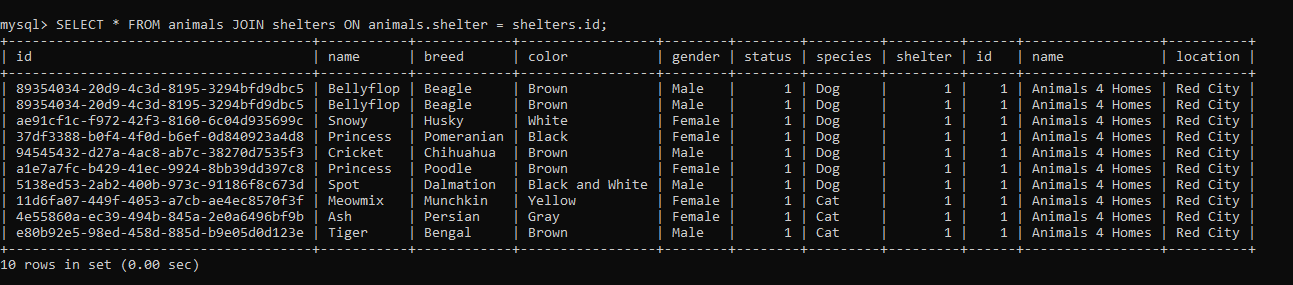
ON *table\_name* (*column1*, *column2*, ...);



**JOIN**

**SQL Join** statement is used to combine data or rows from two or more tables based on a common field between them. Different types of Joins are as follows:

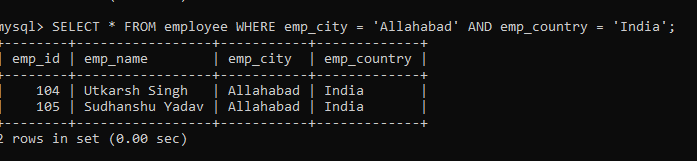
* INNER JOIN
* LEFT JOIN
* RIGHT JOIN
* FULL JOIN
* NATURAL JOIN



## **Types of Logical Operators in SQL**

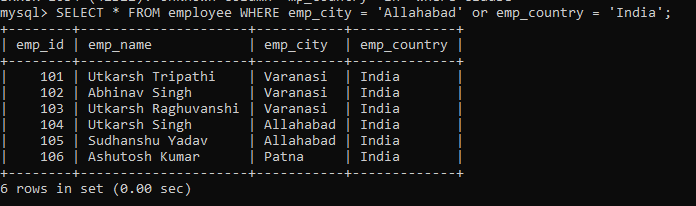
**AND**

## The AND operator displays a record if all the conditions separated by AND are TRUE



**OR**

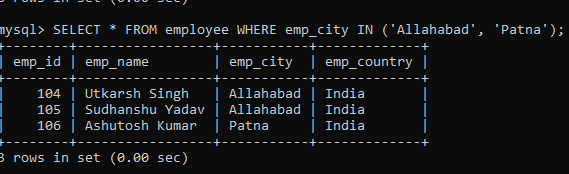
The OR operator displays a record if any of the conditions separated by OR is TRUE.



**IN**

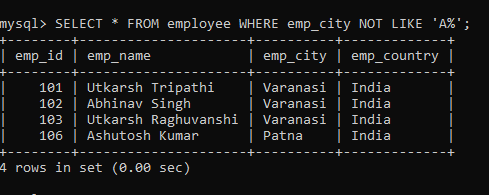
The IN operator allows you to specify multiple values in a WHERE clause.

The IN operator is a shorthand for multiple OR conditions.



**NOT**

The NOT operator displays a record if the condition(s) is NOT TRUE.



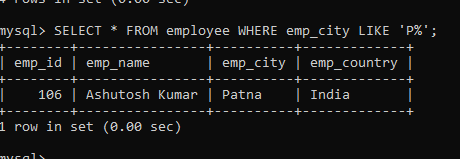
**LIKE**

The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.

There are two wildcards often used in conjunction with the LIKE operator:

* The percent sign (%) represents zero, one, or multiple characters
* The underscore sign (\_) represents one, single character

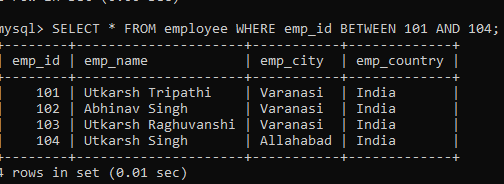
The percent sign and the underscore can also be used in combinations!



**Between**

The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

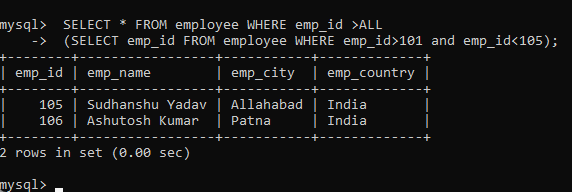
The BETWEEN operator is inclusive: begin and end values are included.



## **ALL**

The ALL operator:

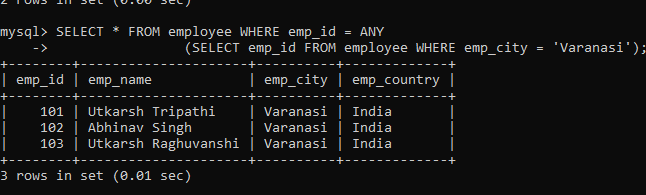
* returns a boolean value as a result
* returns TRUE if ALL of the subquery values meet the condition
* is used with SELECT, WHERE and HAVING statements



## **ANY**

The ANY operator:

* returns a boolean value as a result
* returns TRUE if ANY of the subquery values meet the condition



## 

## 

## 

## 

## 

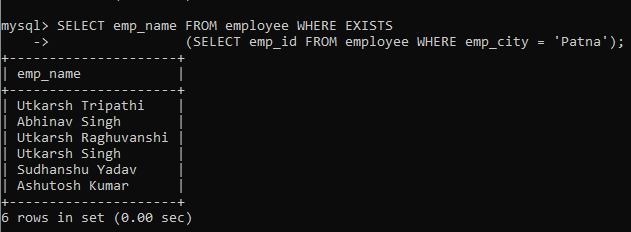
## 

## 

## **Exists**

The EXISTS operator is used to test for the existence of any record in a subquery.

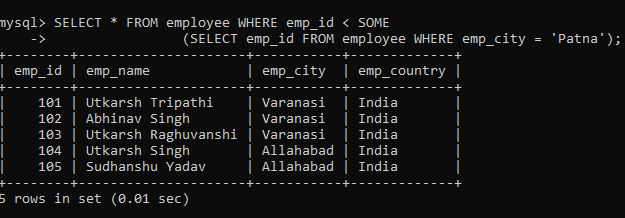
The EXISTS operator returns TRUE if the subquery returns one or more records.



## **Some**

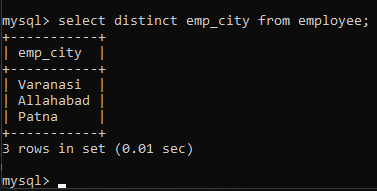
SOME operator evaluates the condition between the outer and inner tables and evaluates to true if the final result returns **any one** row. If not, then it evaluates to false.

* The SOME and ANY comparison conditions are similar to each other and are completely interchangeable.
* SOME must match at least one row in the subquery and must be preceded by comparison operators.



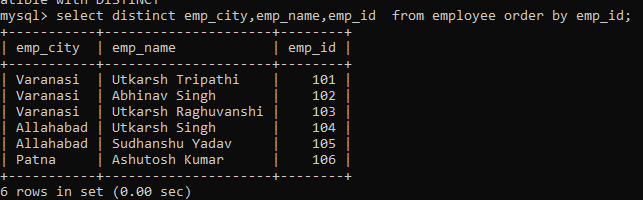
**Distinct clause**

The distinct keyword is used in conjunction with the select keyword. It is helpful when there is a need to avoid duplicate values present in any specific columns/table. When we use distinct keywords only the **unique values** are fetched.

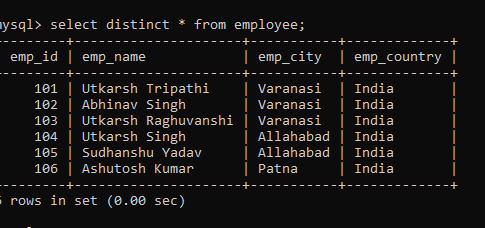


**With order by -**

Here, we will check the order by clause with a distinct clause that will filter out the data based on the order by clause.

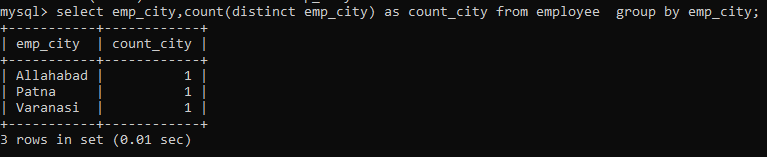


**Distinct \***

****

**With Count**

Here, we will check the COUNT() function with a DISTINCT clause, which will give the total number of students by using the COUNT() function.

****