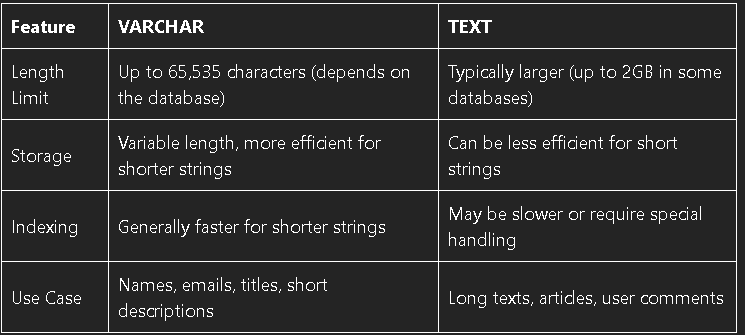
27/10/2024



26/10/2024

=>>***Check the list of databases*** : SHOW DATABASES

25/10/2024

SELECT COUNT(DISTINCT Country) FROM Customers;

=> Return the count number

25/10/2024

* ***Intro :*** MYSQL :
  + A relational database management system.(hệ thống quản lý cơ sở dữ liệu quan hệ)
  + Open-source. (Mã nguồn mở)
  + Free
  + Ideal for both small and large applications
  + Very fast, reliable, scalable, and easy to use.
  + Cross-platform.( đa nền tảng )
  + Compliant (follow ) with the ANSI SQL Standard(tiêu chuẩn).
  + To build a website that shows data from a database, need:
    - An RDBMS database program ( like MySQL)
    - A server-side scripting language, like PHP
    - To use SQL to get the data you want
    - To use HTML / CSS to style the page.
  + MySQL RDBMS ( Relational Database Management System):
    - A program used to maintain a relational database.
    - The basis(nền tảng) for all modern database systems such as MySQL, Microsoft SQL Server, Oracle, and Microsoft Access.
  + Database Table :
    - A table is a collection of related(liên quan) data entries, and it consists (bao gồm) of columns and rows.
    - A column holds specific information about every record(bản ghi) in the table.
    - A record (or row) is each individual entry that exists in a table.



* + A Relational Database:
    - Define database relationships in the form of tables. The tables are related to each other - based on data common to each.

=> The relationship between the “ Customers” and “ Orders” is the CustomerID.

* **MySQL SQL** : is used to insert, search, update and delete database records.
  + **How to Use SQL** :
    - SELECT \* FROM Customers;

=> *select all the records in the “Customers” table.*

\*\*\*\*\* ***SQL keywords are NOT case sensitive* -> select = SELECT**

* + **Semicolon after SQL Statements** : (Dấu chấm phẩy)
    - The standard way to separate each SQL statement in database systems that allow more than one SQL statement to be executed in the same call to the server.
    - Use a semicolon at the end of each SQL statement.
  + **Some of the Most Important SQL Commands** :
    - SELECT - extract (trích xuất) data from a database
    - UPDATE - update data in a database
    - DELETE -delete data from a database
    - INSERT INTO - insert new data into a database
    - CREATE DATABASE - create a new database
    - ALTER DATABASE - modify(=change) a database
    - CREATE DATABASE - create a table
    - ALTER TABLE - modify a table
    - DROP TABLE - delete a table
    - CREATE INDEX - create an index (search key)
    - DROP INDEX - delete an index
  + **MySQL SELECT Statement** :
    - SELECT statement is used to select data from a database.

\* *Result-set*  (tập kết quả): the data returned is stored (lưu trữ) in a result table

* + - SELECT *column1, column2, …*.

From *table\_name*;

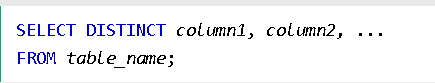
* SELECT \* FROM *table\_name*;
* **Demo Database** :



* **SELECT Columns Example** :
  + SELECT CustomerName, City, Country FROM Customers;



* **The MySQL SELECT DISTINCT Statement** :
  + Used to return only distinct (different) values.

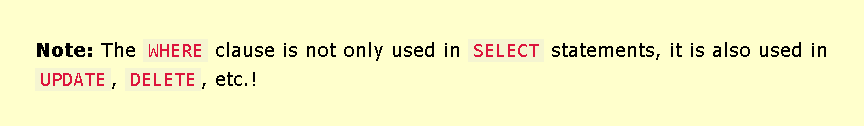


=> Will return 1 value (no same values)

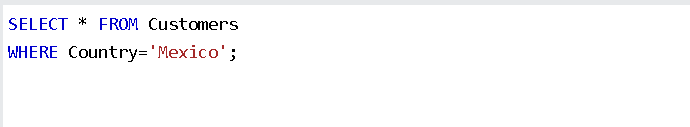
* **MySQL WHERE Clause** :
  + WHERE clause is used to filter (lọc) records.
  + Extract (trích xuất) only those records that fulfill (đáp ứng) a specified condition.
  + SELECT *column1*, *column2, ...*

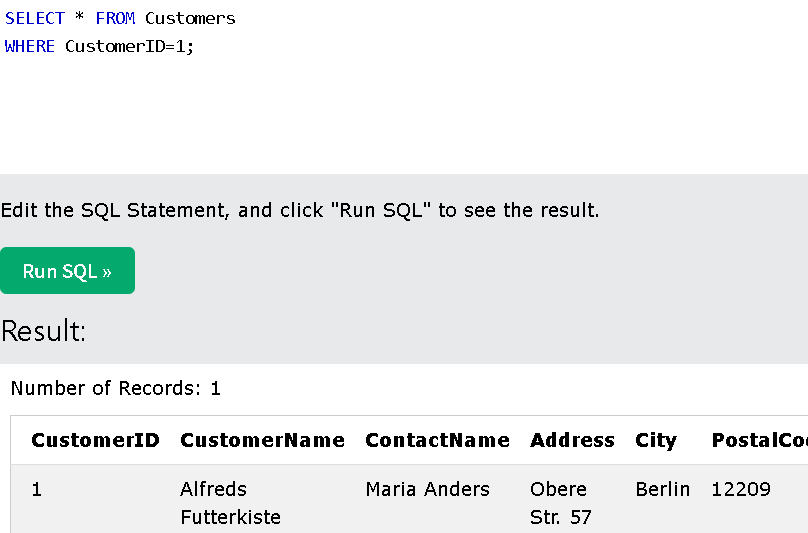
FROM *table\_name*

WHERE *condition*;



* + Example:

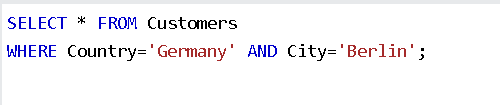


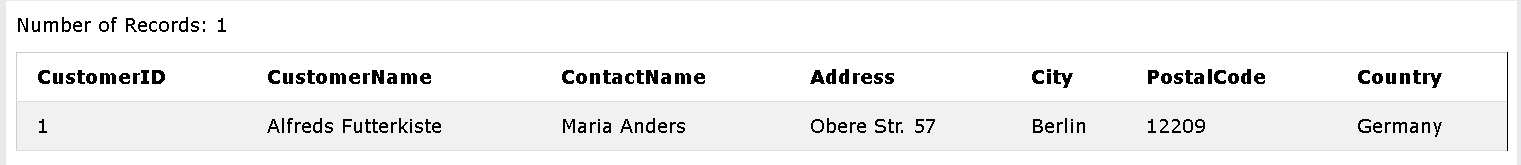


* + Operators in the WHERE Clause :
    - **‘=’ :**  Equal
    - **‘>’ :** Greater than
    - **‘<’ :** Less than
    - **‘>=’ :** Greater than or equal
    - **‘<=’ :** Less than or equal
    - **‘><’ :** Not equal. ( In some versions : **‘!=’** )
    - **‘BETWEEN’ :** Between a certain range
    - **‘LIKE’ :** Search for a pattern
    - **‘IN’ :** To specify multiple possible values for a column
* **MySQL AND, OR and NOT Operator** :
  + WHERE clause can be combined with AND, OR and NOT operators.
  + The AND and OR operator are used to filter records based on more than 1 condition:
* AND: display a record if all the conditions separated by AND are TRUE.
* OR: display a record if any of the conditions separated by OR if TRUE.
  + the NOT: display a record if the condition(s) is NOT TRUE.
    - **AND Syntax**:
      * SELECT *column1*, *column2, ...*

FROM *table\_name*

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

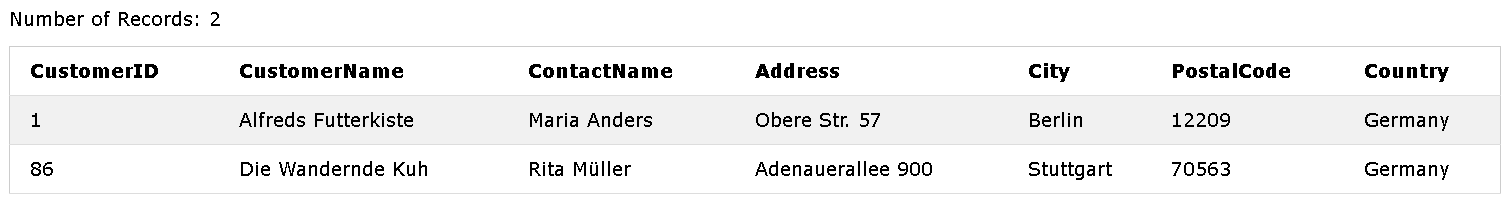
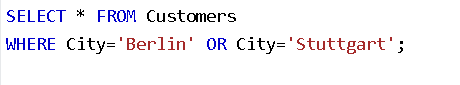




* + - **OR Syntax** :
      * SELECT *column1*, *column2, ...*

FROM *table\_name*

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

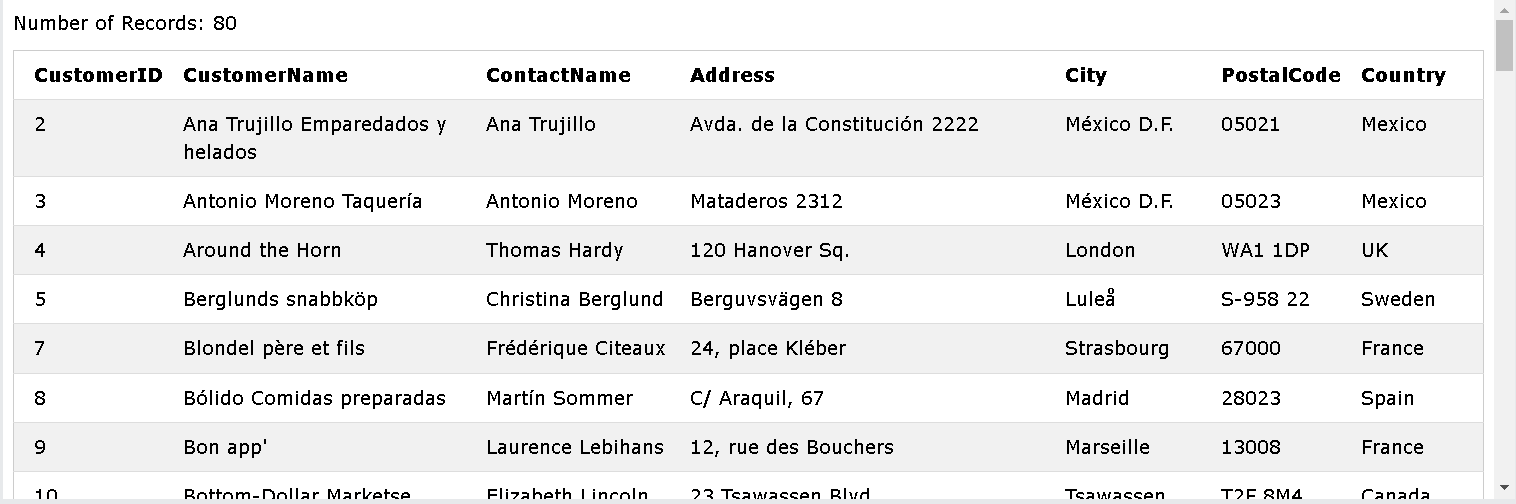


* + - **NOT Syntax** :
      * SELECT *column1*, *column2, ...*

FROM *table\_name*

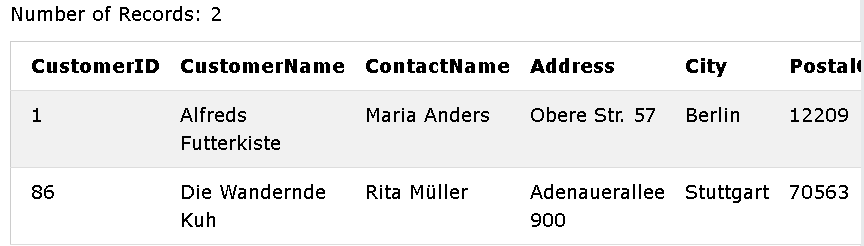
WHERE NOT *condition*;





* **Combining AND, OR and NOT** :
  + Example:
    - SELECT \* FROM Customers

WHERE Country = 'Germany' AND (City = 'Berlin' OR City = 'Stuttgart');



* SELECT \* FROM Customers

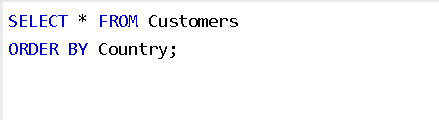
WHERE NOT Country = 'Germany' AND NOT Country = 'USA';



* **MySQL ORDER BY Keyword** : (thứ tự)
  + ORDER BY : used to sort the result-set in ascending(tăng dần) or descending(giảm dần) order.
  + ORDER BY : sort the records in ascending order by default. To sort the records in descending order, use DESC.
  + **ORDER BY Syntax** :
    - SELECT *column1*, *column2, ...*

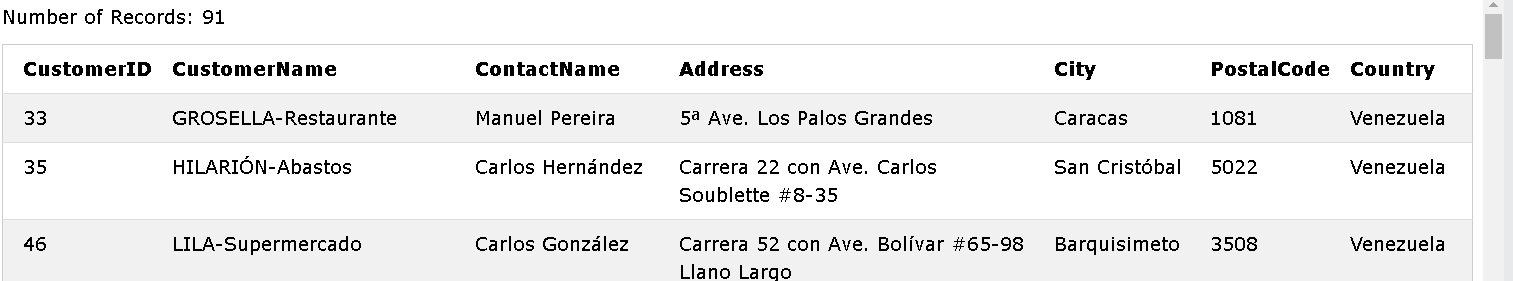
FROM *table\_name*

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



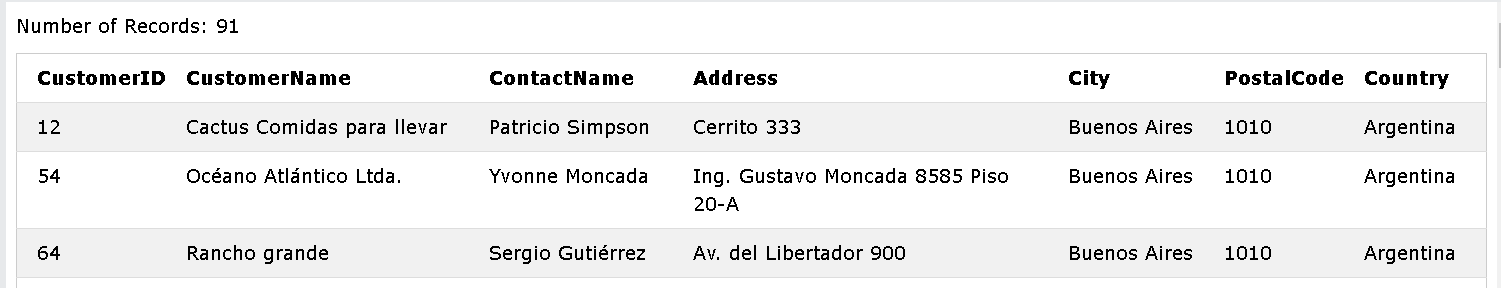
* + - Example :
      * DESC: SELECT \* FROM Customers

ORDER BY Country DESC;



* + - * Several Columns: SELECT \* FROM Customers

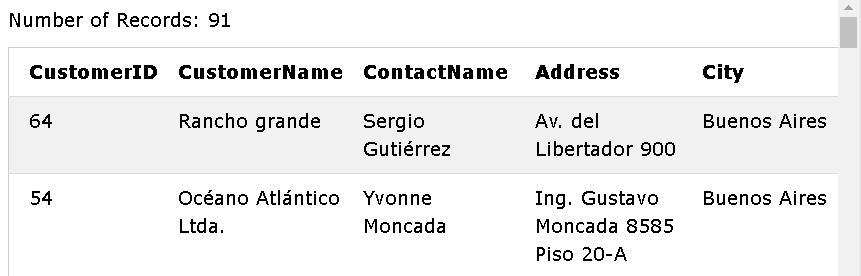
ORDER BY Country, CustomerName;



* + - * Several Columns 2:

SELECT \* FROM Customers

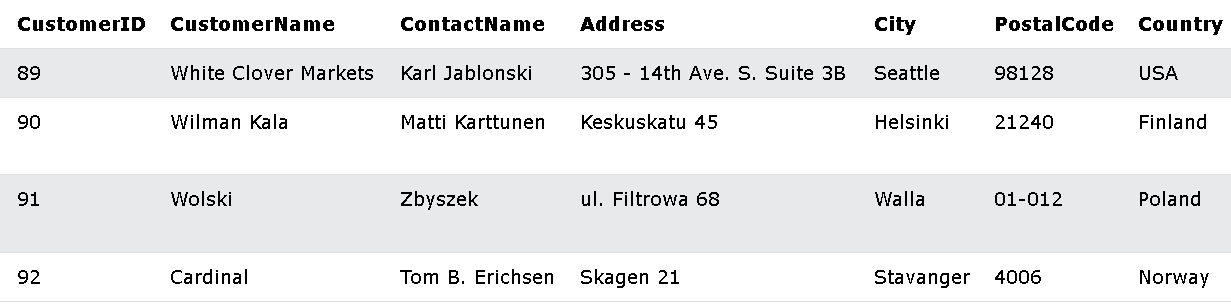
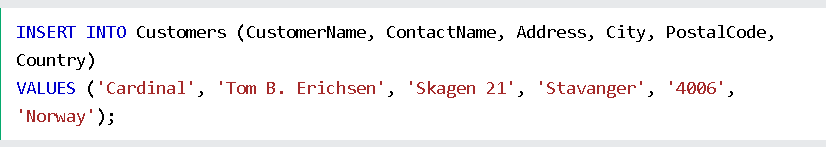
ORDER BY Country ACS, CustomerName DESC;



* **MySQL INSERT INTO Statement** :
  + INSERT INTO : used to insert new records in a table.
  + **INSERT INTO Syntax** :
    - Specify both the column names and the values to be inserted:

INSERT INTO *table\_name* (*column1*, *column2*, *column3*, ...)

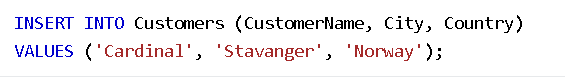
VALUES (*value1*, *value2*, *value3*, ...);

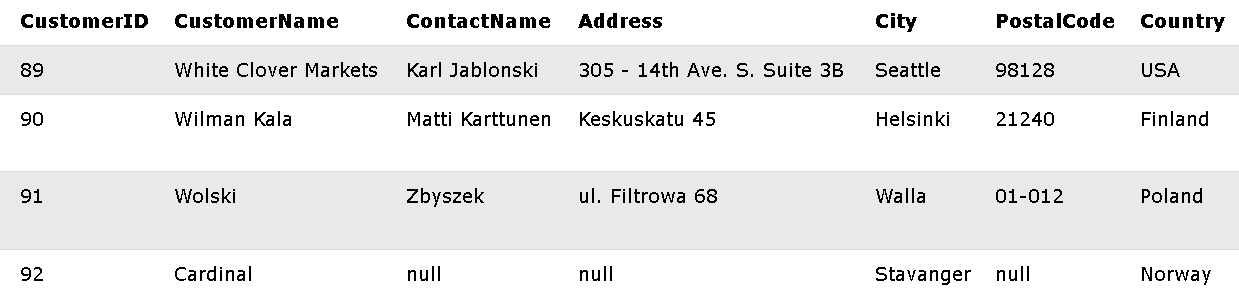


* + - Add values for all the columns of the table but no need to specify the column names in the SQL query but make sure the order of the values is in the same order as the columns in the table.:

INSERT INTO *table\_name*

VALUES (*value1*, *value2*, ….)





* **MySQL NULL Values** :
  + A Field ( trường )with NULL value is a field with no value.
  + If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a NULL value.

=> *A field with a NULL that has been left blank during record creation.(A NULL value ≠ A zero value // A field that contains spaces.)*

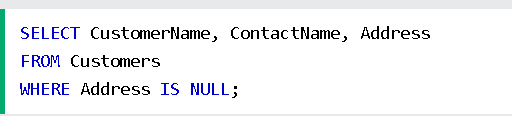
* + **Test for NULL Values** :

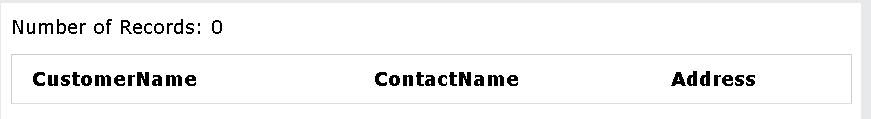


* + - We will have to use “ IS NULL “ and “ IS NOT NULL “ to test.
    - **IS NULL Syntax** :
      * SELECT *column\_names*

FROM *table\_name*

WHERE *column\_name* IS NULL;

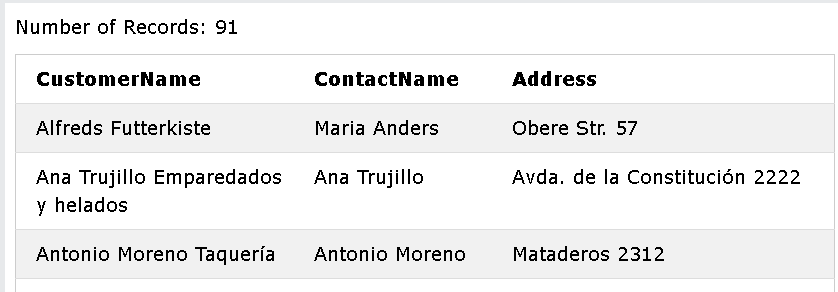
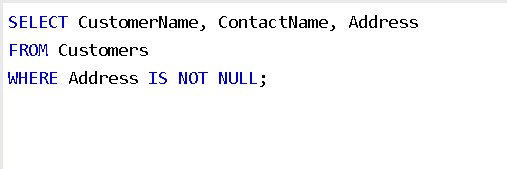




* + - **IS NOT NULL Syntax** :
      * SELECT *column\_names*

FROM *table\_name*

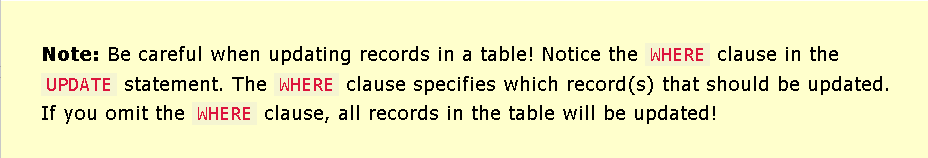
WHERE *column\_name* IS NOT NULL;

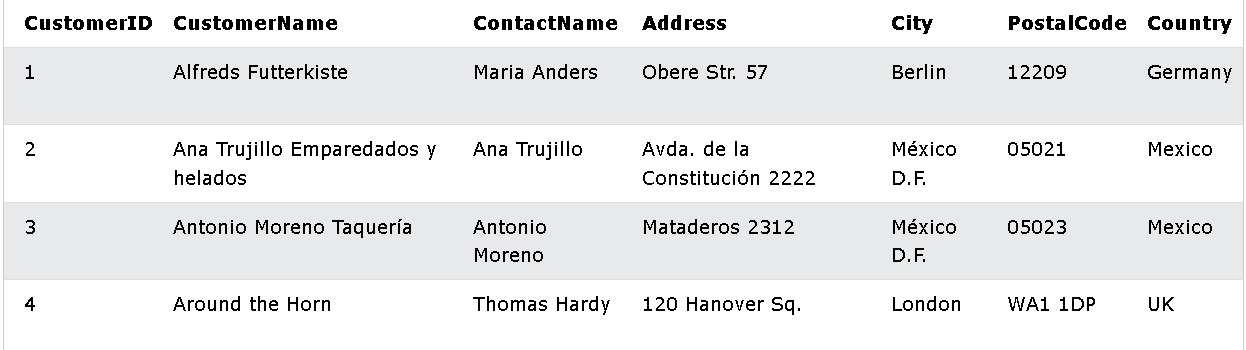


* **MySQL UPDATE Statement** :
  + UPDATE : used to modify the existing records in a table.
  + **UPDATE Syntax** :
    - UPDATE *table\_name*

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

WHERE *condition*;





* + - UPDATE a table: (update the first customer with a new contact person and a new city)

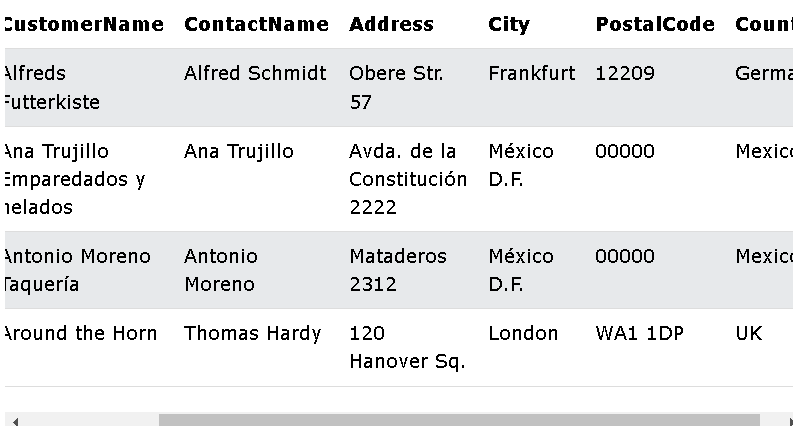


* + - UPDATE Multiple Records :
      * WHERE clause determines how many records will be updated.
      * UPDATE Customers

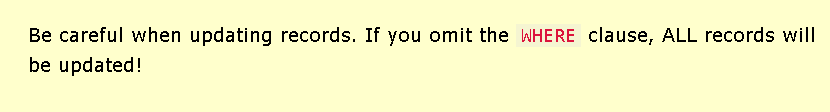
SET PostalCode = 00000

WHERE Country = 'Mexico';

-> update the postalCode to 00000 for all records where country is “MEXICO”



* + - UPDATE Warning :

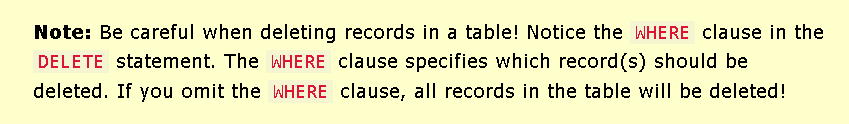


* + - * UPDATE Customers

SET PostalCode = 00000;

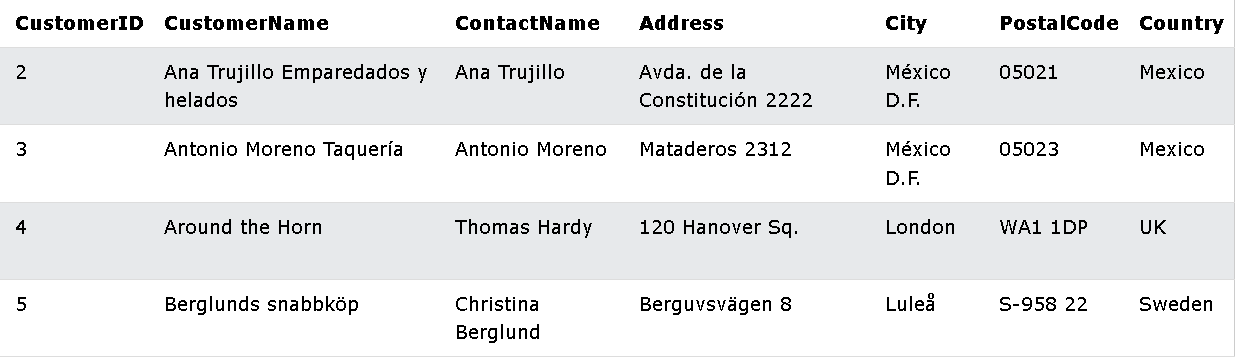


* **MySQL DELETE Statement** :
  + DELETE : used to delete existing records in a table.
  + **DELETE Syntax** :
    - DELETE FROM *table\_name* WHERE *condition*;





* + - Example :
      * DELETE FROM Customers WHERE CustomerName = ‘Alfreds Futterkiste’



* + **DELETE All Records** :
    - DELETE FROM *table\_name*;
* **MySQL LIMIT Clause** :
  + LIMIT : used to specify the number of records to return.
  + LIMIT : useful on large tables with thousands of records. Returning a large number of records can impact performance.
  + **LIMIT Syntax** :
    - SELECT *column\_name(s)*

FROM *table\_name*

WHERE *condition*

LIMIT *number (*OFFSET *the\_beginning)*;



* + - Example :
      * SELECT \* FROM Customers

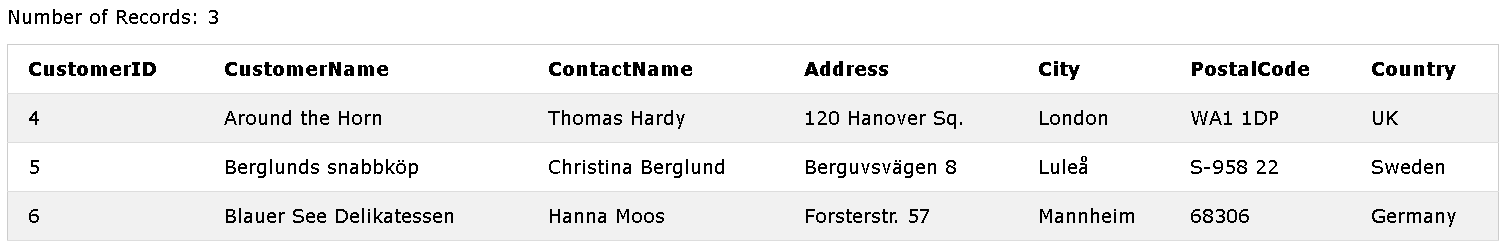
LIMIT 3;



=> From 1-3

* + - * If we want to specify the beginning :-> OFFSET
        + SELECT \* FROM Customers

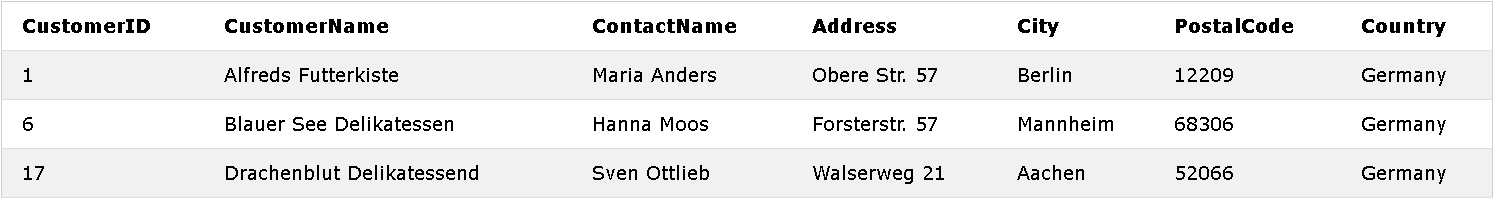
LIMIT 3 OFFSET 3;



* + **ADD a WHERE Clause** :
    - Select the first three records from the “ Customers “ table, where the country is “Germany”
      * SELECT \* FROM Customers

WHERE Country='Germany'

LIMIT 3;



* **MySQL MIN() and MAX() Function** :
  + MIN() : function returns the smallest value of the selected column.
  + MAX(): function returns the largest value of the selected column.
  + **MIN() Syntax** :
    - SELECT MIN(*column\_name*)

FROM *table\_name*

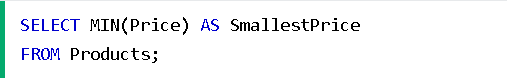
WHERE *condition*;

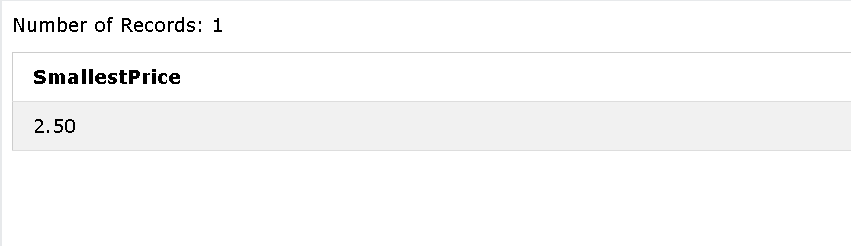
* + **MAX() Syntax** :
    - SELECT MAX(*column\_name*)

FROM *table\_name*

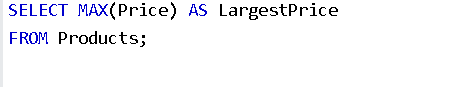
WHERE *condition*;

* + - Example :
      * MIN():





* + - * MAX():



* **MySQL COUNT(), AVG() and SUM() Functions** :
  + **COUNT() Syntax** : returns the number of rows that matches a specified criterion (tiêu chí).
    - SELECT COUNT(*column\_name*)

FROM *table\_name*

WHERE *condition*;

* + **AVG() Syntax** : returns the average value of a numeric (number) column.
    - SELECT AVG (*column\_name*)

FROM *table\_name*

WHERE *condition*;

* + **SUM() Syntax** :
    - SELECT SUM(*column\_name*)

FROM *table\_name*

WHERE *condition*;

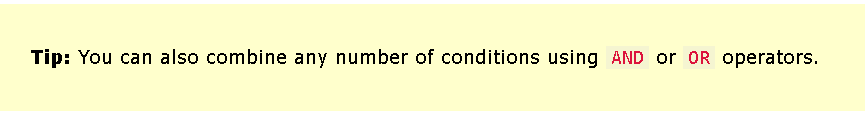
* **MySQL LIKE Operator** :
  + LIKE : used in a WHERE clause to search for a specified pattern( mẫu ) in a column.
  + 2 wildcards(ký tự) often used in conjunction(kết hợp) with LIKE :
    - The percent sign(**%**) = zero, one, or multiple characters
    - The underscore sign (**\_**) = one, single character.

***=> Can be used in combinations. both***

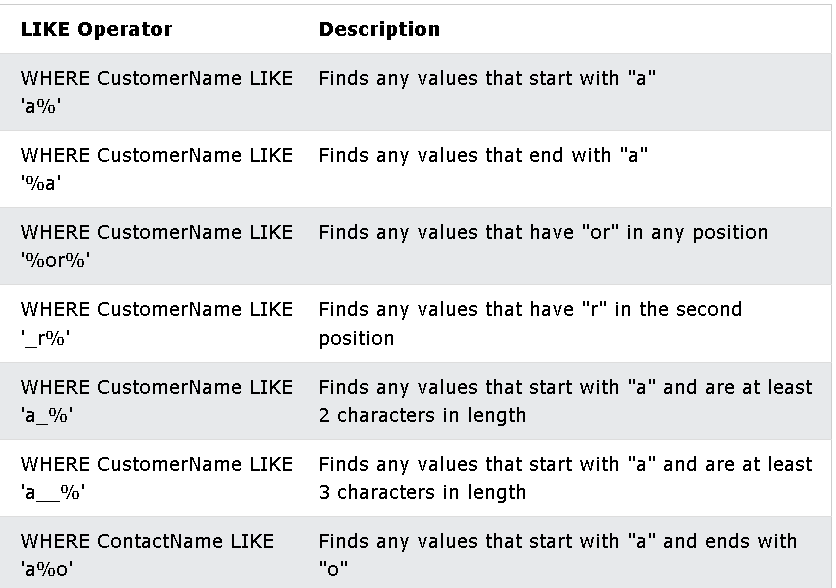
* + **LIKE Syntax** :
    - SELECT *column1*, *column2*,...

FROM *table\_name*

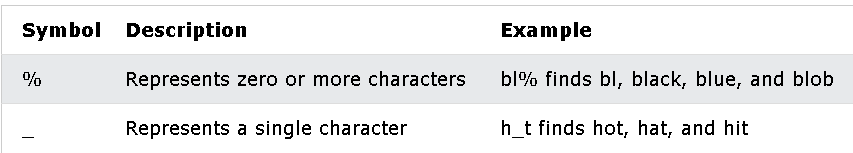
WHERE *column* (NOT)LIKE *pattern*;



* + - Example :



* **MySQL Wildcards** :
  + **Wildcard character** : used to substitute(thay thế) one or more characters in a string.
    - WIldcard characters : used with the LIKE operator. The LIKE: used in a WHERE clause to search for a specified pattern in a column.
    - Wildcard characters :





* **MySQL IN Operator** :
  + The IN : specify multiple values in a WHERE clause.
  + The IN : a shorthand for multiple OR conditions.
  + **IN Syntax** :
    - SELECT *column\_name(s)*

FROM *table\_name*

WHERE *column\_name* (NOT) IN (*value1*, *value2*, ...);

* + - SELECT *column\_name(s)*

FROM *table\_name*

WHERE *column\_name* (NOT) IN (*SELECT STATEMENT*);



* **The MySQL BETWEEN Operator** :
  + The BETWEEN : select values within a given range(phạm vi nhất định).
  + The BETWEEN : inclusive : begin and end values are included.
  + **BETWEEN Syntax** :
    - SELECT *column\_name(s)*

FROM *table\_name*

WHERE *column\_name* (NOT)BETWEEN *value1* AND *value2*

**(**AND *column\_name* (NOT) IN (*value1*, *value2*,...);**)**

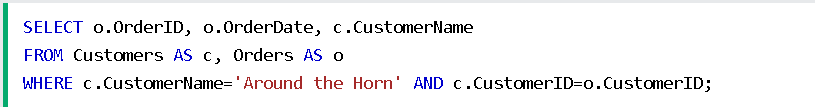
**(**ORDER BY *column\_name***)**

* **MySQL Aliases** : (bí danh)
  + Aliases: used to give a table, or a column in a table, a temporary name.
  + Aliases: often used to make column names more readable (có thể đọc đc).
  + Alias: only exists for a duration(khoảng thời gian)of that query.
  + Alias: created with AS keyword.
  + **Alias Column Syntax** :
    - A Column:
      * SELECT column\_name AS alias\_name

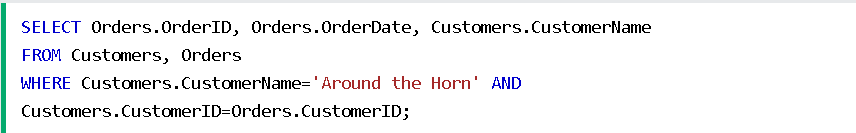
FROM table\_name;

* + - A Table :
      * SELECT *column\_name(s)*

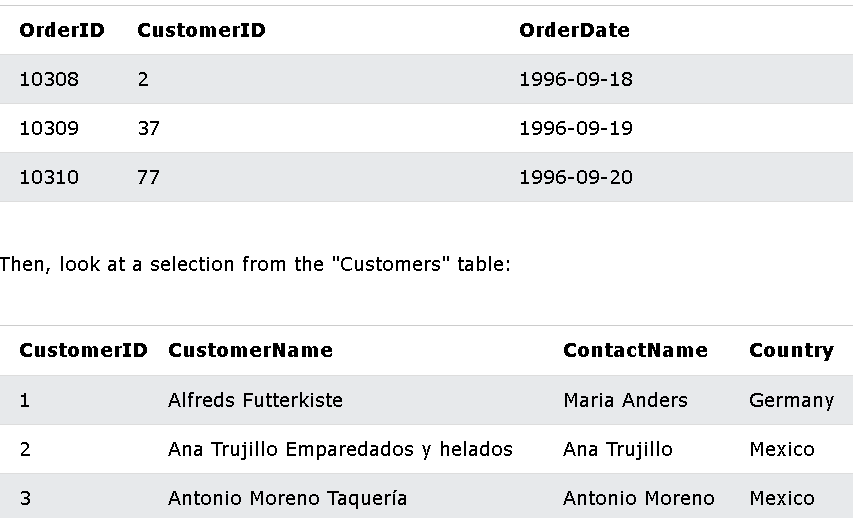
FROM *table\_name* AS *alias\_name*;



**=**



* **MySQL Joins** :
  + JOIN : used to combine rows from 2 or more tables(kết hợp hàng từ 2 bảng trở lên) , based on a related column between them.



* + Both tables have “CustomerID”

=>>>

* + SELECT *o.OrderID, c.CustomerName, o.OrderDate*

FROM *Orders* AS *o*

INNER JOIN *Customers* AS *c* ON *o.CustomerID=c.CustomerID;*

* + **Supported Types of Joins MySQL** :
    - INNER JOIN : return records have matching values in both tables.
    - LEFT JOIN : return all records from the left table and matched records from the right table.
    - RIGHT JOIN : return all records from the right table and matched records from the left table.
    - CROSS JOIN : return all records from both tables.

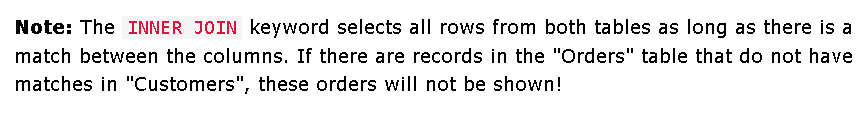
   

* **MySQL INNER JOIN Keyword** : (can be joined more than 2 tables)
  + INNER JOIN : select records that have matching values in both tables.
    - SELECT *column\_name(s)*

FROM *table1*

INNER JOIN *table2*

ON *table1.column\_name* = *table2.column\_name*;

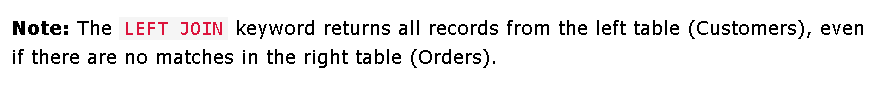


* **MySQL LEFT JOIN Keyword** :
  + LEFT JOIN : return all records from the left table(1) and the matching records from the right table(2).
    - SELECT *column\_name(s)*

FROM *table1*

LEFT JOIN *table2*

ON *table1.column\_name = table2.column\_name*;

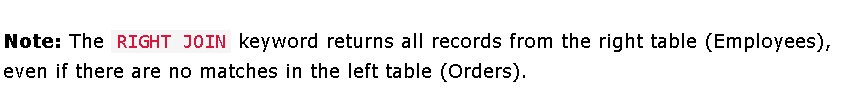


* **MySQL RIGHT JOIN Keyword** :
  + RIGHT JOIN : return all records from the right table(2) and the matching records from the left table(1).
    - SELECT *column\_name(s)*

FROM *table1*

RIGHT JOIN *table2*

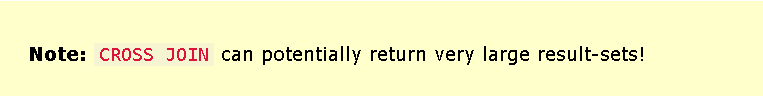
ON *table1.column\_name = table2.column\_name;*



* **MySQL CROSS JOIN Keyword** :
  + CROSS JOIN : return all records from both (or more) tables.
    - SELECT *column\_name(s)*

FROM *table1*

CROSS JOIN *table2;*



**\*\*\*\* *CROSS JOIN :*** *return all matching records from all tables whether the other table matches or not.This will return all of the records of all tables.*

* **MySQL Self Join** :
  + Self join : a regular join, ***the table is joined with itself.***
  + SELECT *A.CustomerName* AS *CustomerName1, B.CustomerName* AS *CustomerName2, A.City*

FROM *Customers A, Customers B*

WHERE *A.CustomerID <> B.CustomerID* (so sánh 2 id khác nhau của 2 table)

AND *A.City = B.City*

ORDER BY *A.City;*

* **MySQL UNION Operator** :
  + UNION : used to combine the result-set of two or more SELECT statements.
  + **UNION Syntax** : -> select only distinct ( riêng biệt ) values by default.
    - SELECT *column\_name(s)* FROM *table1*

UNION

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

WHERE *condition*

ORDER BY ….;

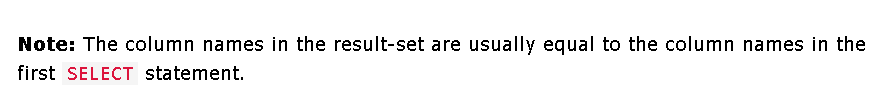
* + **UNION ALL Syntax** : allow duplicate (trùng lặp) values
    - SELECT *column\_name(s)* FROM *table1*

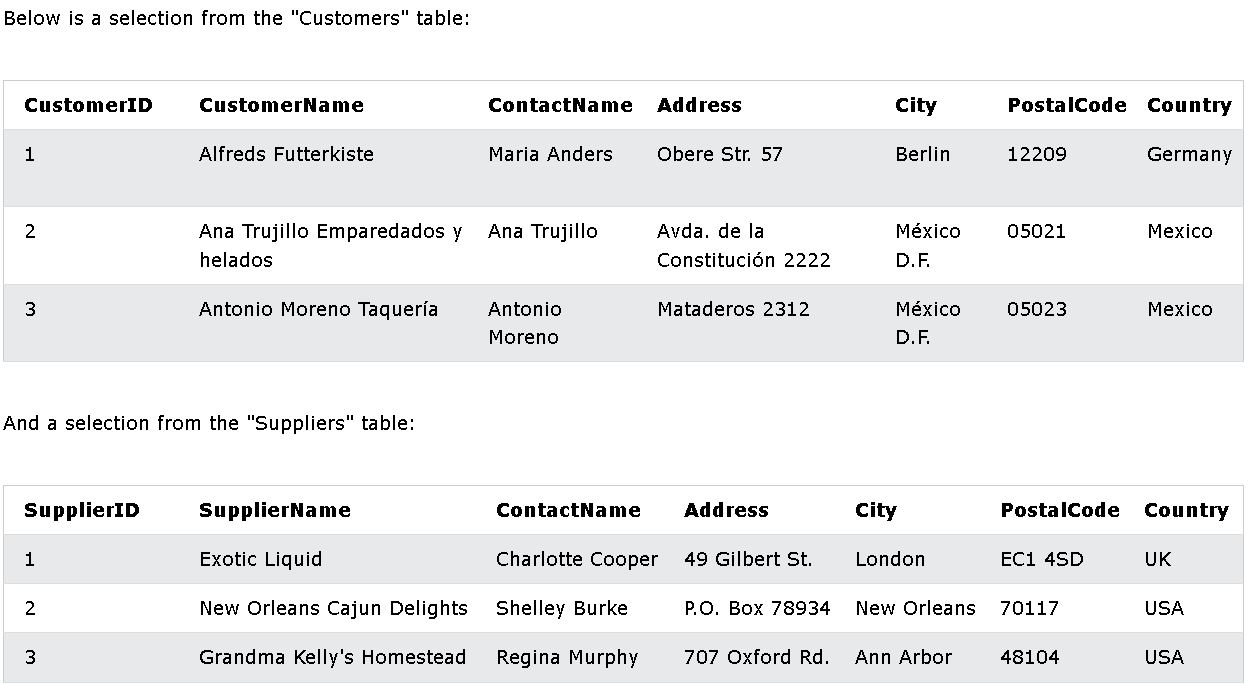
UNION ALL

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

WHERE *condition*

ORDER BY ….;



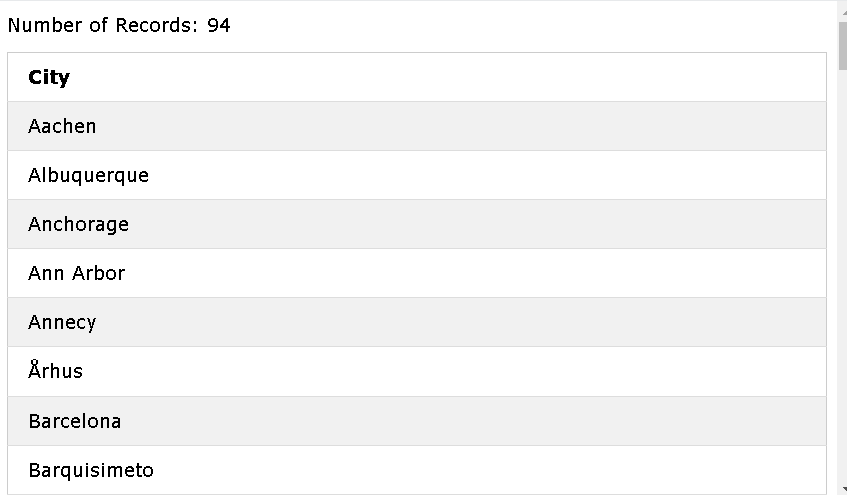


* + - Example :
      * *UNION* :
        + SELECT City FROM Customers

UNION

SELECT City FROM Suppliers

ORDER BY City;



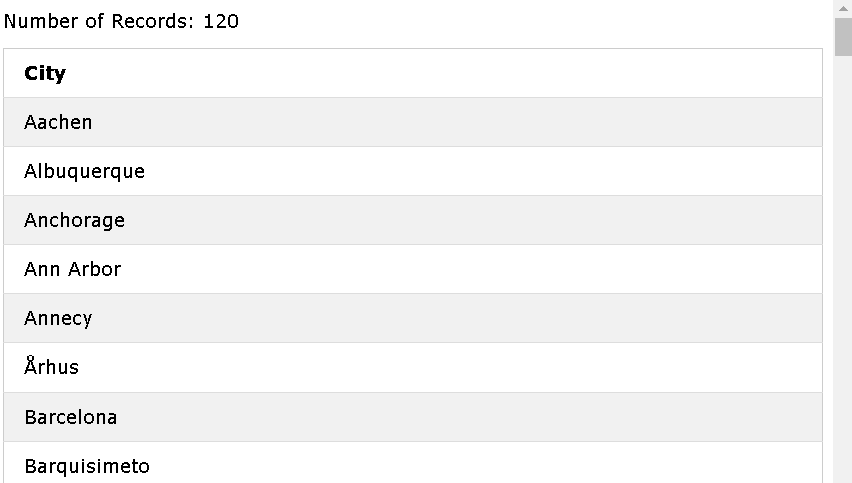
=> Nhóm Records đc SELECT vào cùng 1 cột của 2 tables, và xếp theo thứ tự : ORDER. Những Records bị lặp sẽ chỉ được hiển thị 1 lần.

* + - * *UNION ALL* :
        + SELECT City FROM Customers

UNION ALL

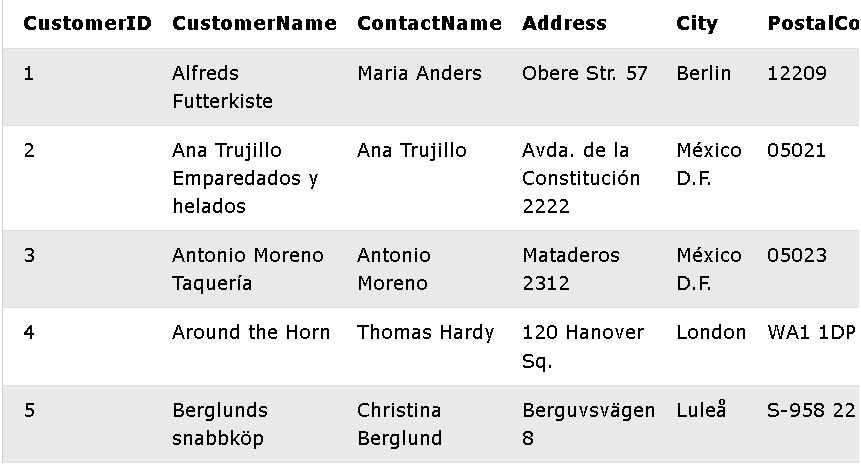
SELECT City FROM Suppliers

ORDER BY City;



=> Vừa nhóm vừa liệt kê dù có trùng hay đã bị lặp.

* **MySQL GROUP BY Statement** :
  + GROUP BY : groups rows that have the same values into summary rows, like “ find the number of customers in each country”
  + GROUP BY : used with aggregate(tổng hợp) functions ( COUNT(), MAX(), MIN(), SUM(), AVG() ) to group the result-set by one or more columns.
  + **GROUP BY Syntax** :



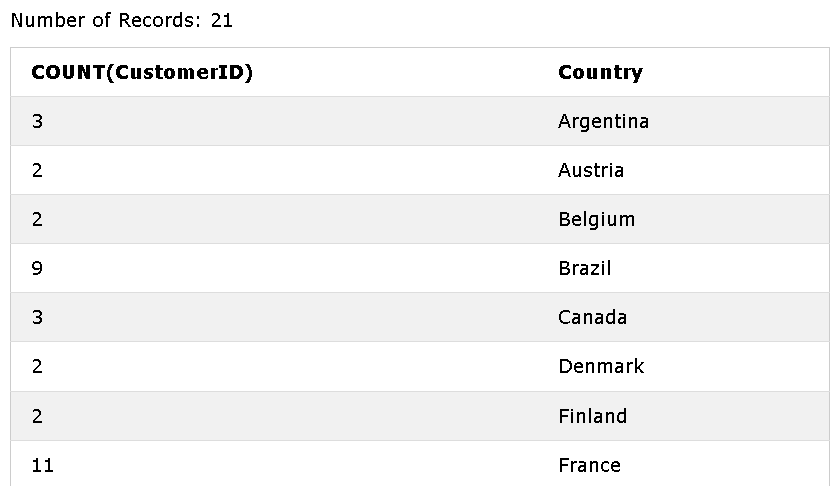
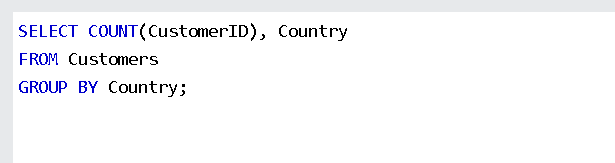
* + - SELECT *column\_name(s)*

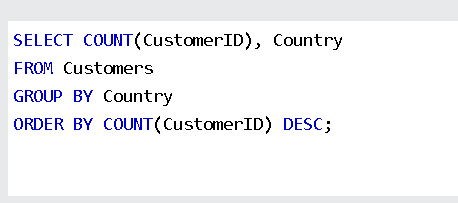
FROM *table\_name*

WHERE *condition*

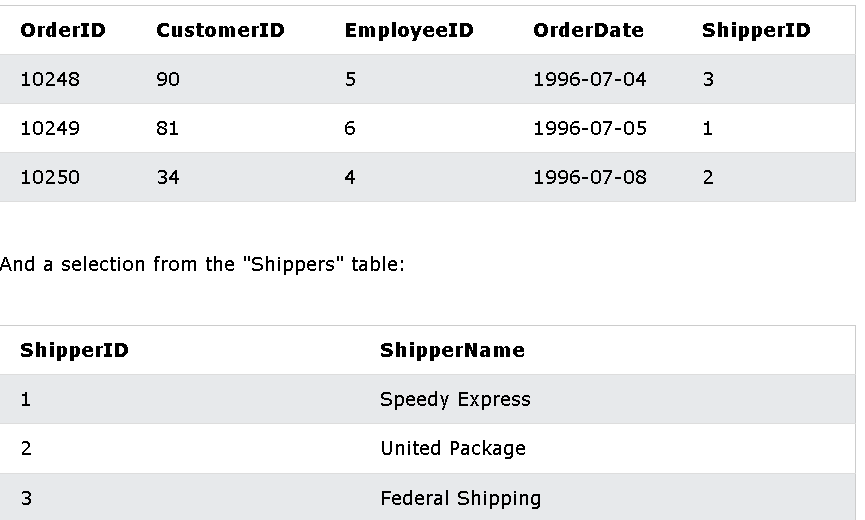
GROUP BY *column\_name(s)*

GROUP BY  *column\_name(s);*





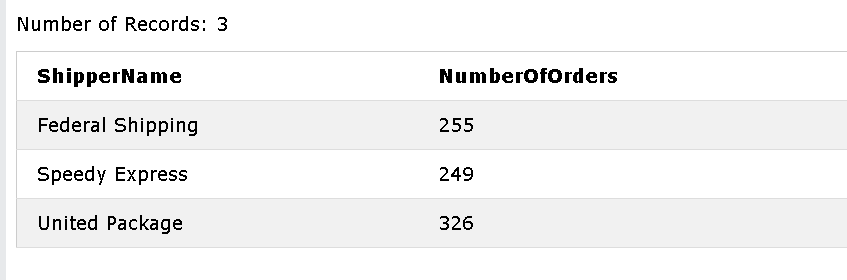
* + - *GROUP BY With JOIN* :



* + - * SELECT *s.ShipperName,* COUNT(*o.OrderID*) AS *NumberOfOrders* FROM *Orders AS o*

LEFT JOIN *Shippers* AS s ON *o.ShipperID* = *s.ShipperID*

GROUP BY *ShipperName*;



* **MySQL HAVING Clause** :
  + HAVING : added to SQL because WHERE can not be used with aggregate functions.
  + **HAVING Syntax** :
    - SELECT *column\_name(s)*

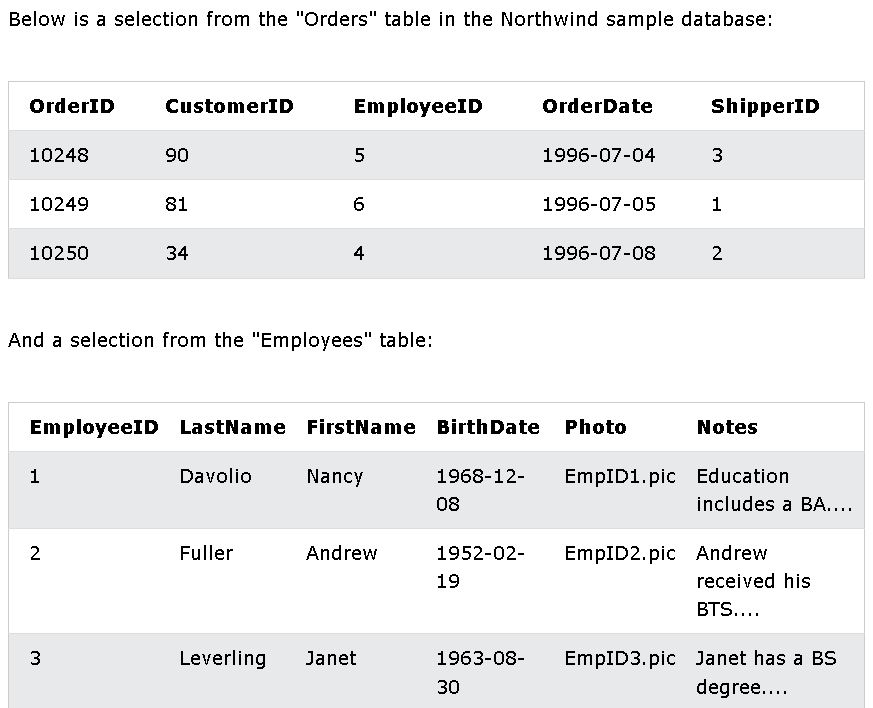
FROM *table\_name(s)*

WHERE *condition*

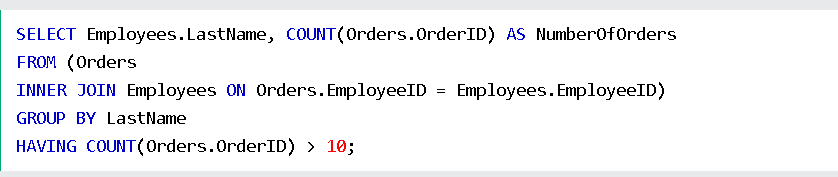
GROUP BY *column\_name(s)*

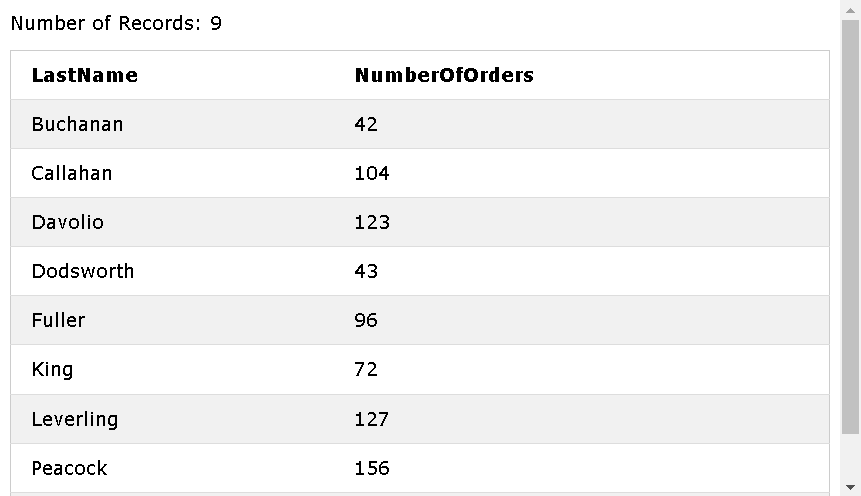
HAVING *condition*

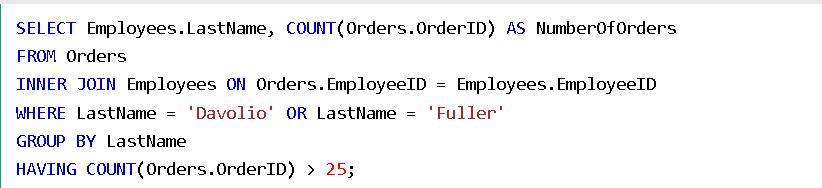
ORDER BY *column\_name(s);*

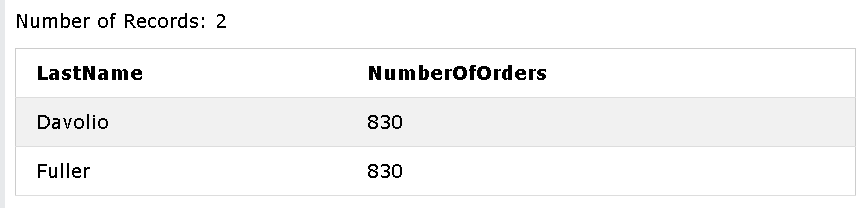


* + - Example :









* **MySQL EXISTS Operator** :
  + EXISTS : used to test for the existence of any record in a subquery ( truy vấn con ).
  + EXISTS : return TRUE if the subquery returns one or more records .
  + **EXISTS Syntax** :
    - SELECT *column\_name(s)*

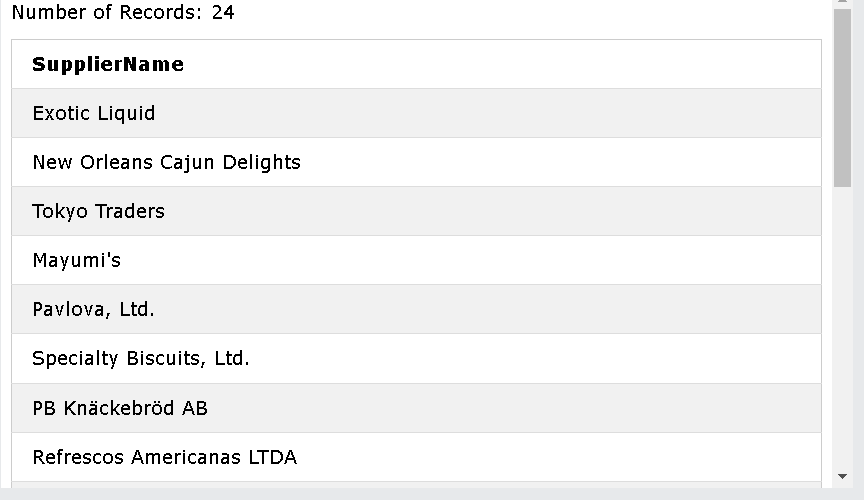
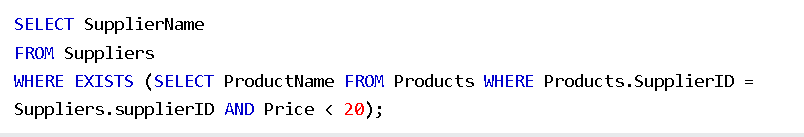
FROM *table\_name*

WHERE EXISTS

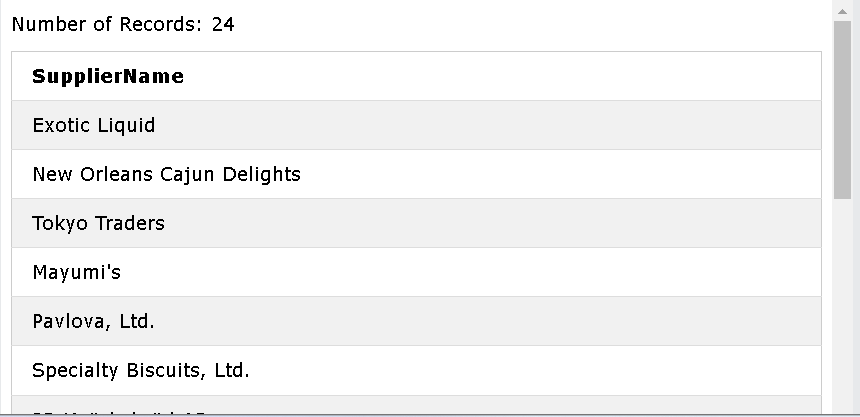
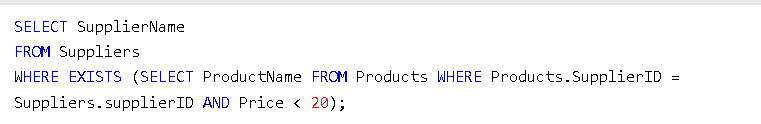
*(*SELECT *colum\_name* FROM *table\_name* WHERE *condition);*



* + - Example :







* **MySQL ANY and ALL Operators** :
  + ANY and ALL : allow to perform a comparison between a single column value and a range(phạm vi) of other values.
  + ANY : return a boolean value as a result.
    - return TRUE if ANY of the subquery values meet the condition.
    - => Condition will be true if the operation is true for any of the values in the range.
    - **ANY Syntax** :
      * SELECT *column\_name(s)*

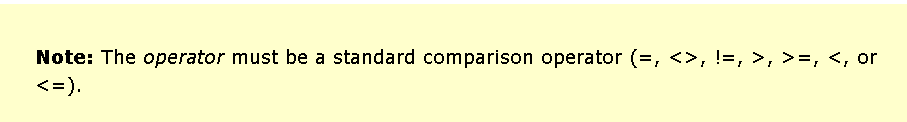
FROM *table\_name*

WHERE *column\_name operator* ANY

*(*SELECT *column\_name*

FROM  *table\_name*

WHERE *condition);*



* + ALL : return a boolean value as a result.
    - return TRUE if ALL of the subquery values meet the condition.
    - used with SELECT, WHERE and HAVING statements.
    - **ALL Syntax With SELECT** :
      * SELECT ALL *column\_name(s)*

FROM *table\_name*

WHERE *condition;*

* + - **ALL Syntax With WHERE or HAVING** :
      * SELECT *column\_name(s)*

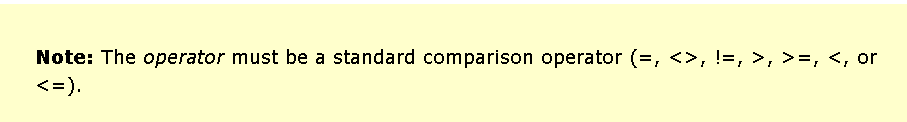
FROM *table\_name*

WHERE *column\_name operator* ALL

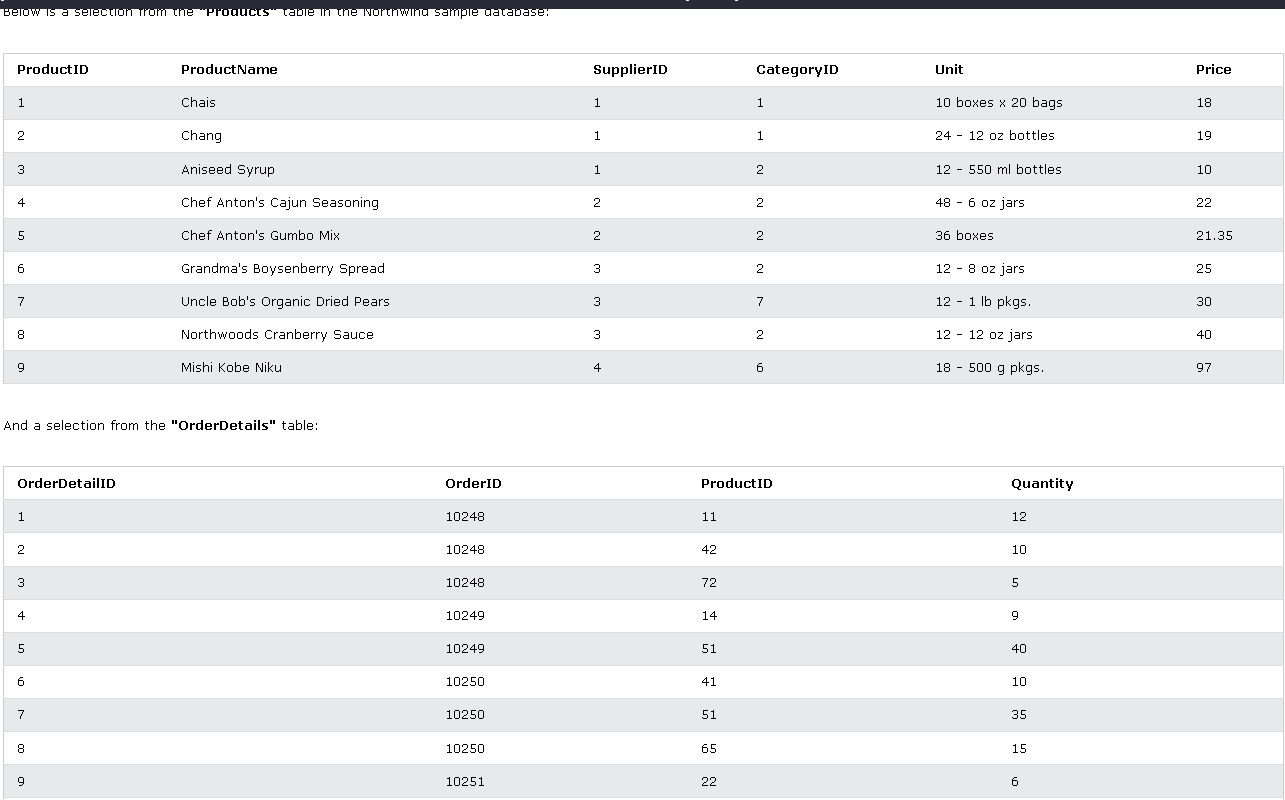
*(*SELECT *column\_name*

FROM *table\_name*

WHERE *condition);*



* + - Example :



* + - * *ANY* :
        + SELECT ProductName

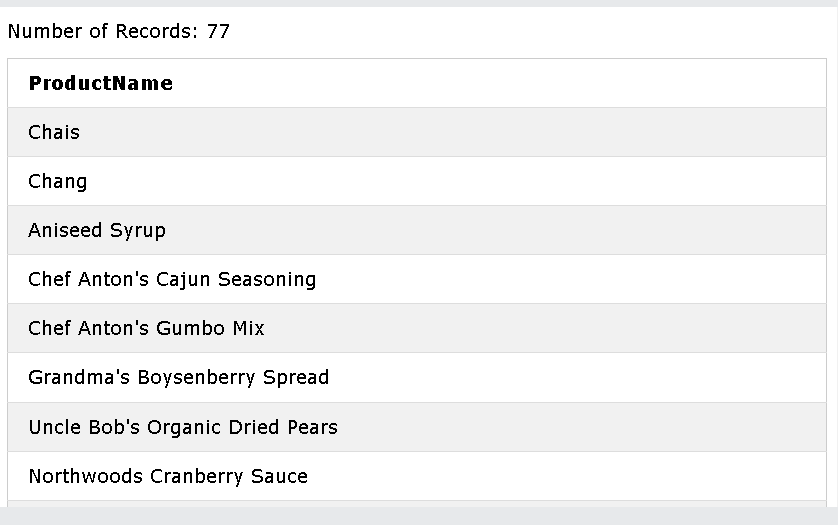
FROM Products

WHERE ProductID = ANY

(SELECT ProductID

FROM OrderDetails

WHERE Quantity < 15);



* + - * *ALL* :
        + SELECT ProductName

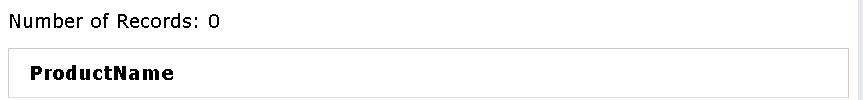
FROM Products

WHERE ProductID = ALL

(SELECT ProductID

FROM OrderDetails

WHERE Quantity = 10);



=> Muốn có giá trị trả về thì ở Command “ WHERE ProductID = ALL “ ALL phải là 1 giá trị cụ thể , chứ không phải là 1 tập hợp.

***Ví dụ*** : WHERE ProductID = ALL { 1 }

***NOT*** : WHERE ProductID = ALL { 2, 3, … }

* **MySQL INSERT INTO SELECT Statement** :
  + INSERT INTO SELECT : copy data from 1 table and insert it into another table.
  + INSERT INTO SELECT : require the data types in sources and target table matches.

***\*\*\*\**** : The existing records in the target table are unaffected.

* + **INSERT INTO SELECT Syntax** :
    - INSERT INTO *table2*

SELECT *\** FROM *table1*

WHERE *condition;*

=> Copy all columns from 1 table -> another table

* + - INSERT INTO *table2 ( column1,...)*

SELECT *column1, ….*

FROM *table1*

WHERE *condition;*

=> Copy only some columns from 1 table -> another table

* + Example :
    - INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country)

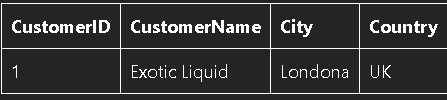
SELECT SupplierName, ContactName, Address, City, PostalCode, Country FROM Suppliers;



* + - INSERT INTO Customers ( CustomerName, City, Country)

SELECT SupplierName, City, Country FROM Suppliers

WHERE Country = ‘Germany’



* **MySQL CASE Statement** :
  + CASE : go through conditions and return a value then the first condition is met( like if-then-else) -> Once a condition is true, it will stop reading and return the result. If no conditions are true, it returns the value in the ELSE clause.
  + If no ELSE -> return NULL
  + **CASE Syntax** :
    - CASE

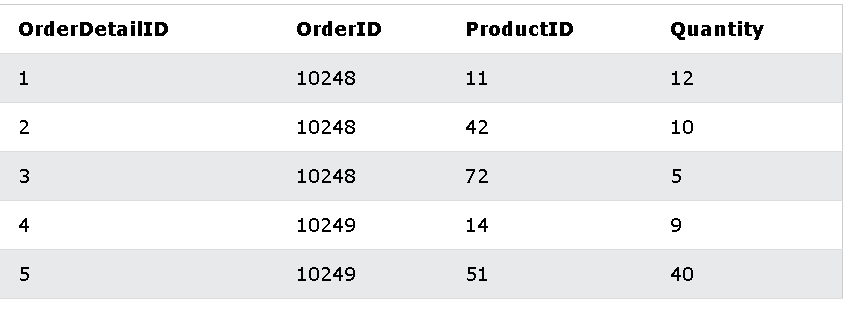
WHEN *condition1* THEN *result1*

WHEN *conditionN* THEN *resultN*

ELSE *result*

END*;*

* + - Example :



* + - * SELECT OrderID, Quantity,

CASE

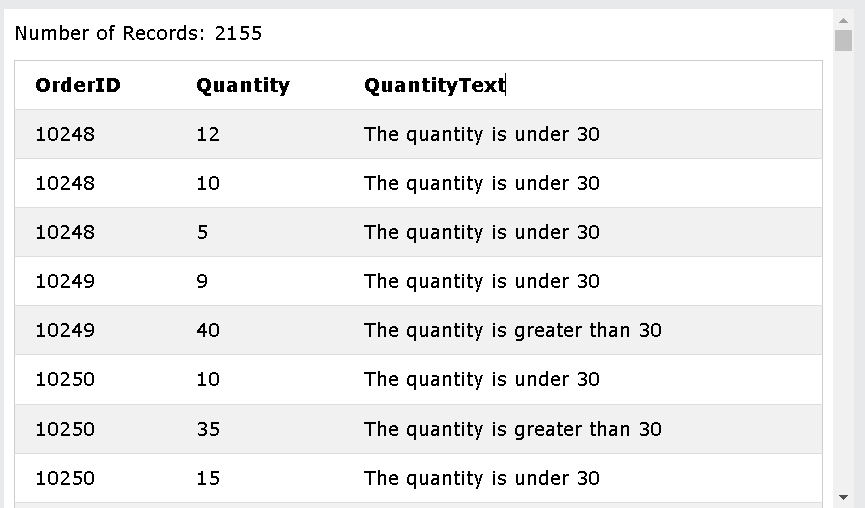
WHEN Quantity > 30 THEN 'The quantity is greater than 30'

WHEN Quantity = 30 THEN 'The quantity is 30'

ELSE 'The quantity is under 30'

END AS Quantity Text

FROM OrderDetails;



* + - * SELECT CustomerName, City, Country

FROM Customers

ORDER BY

(CASE

WHEN City IS NULL THEN Country

ELSE City

END);



* **MySQL NULL Functions** :



* + Statement :
    - SELECT ProductName, UnitPrice \* (UnitsInStock + UnitsOnOrder)

FROM Products;



* + - **MySQL IFNULL() Function** :
      * IFNULL() : return an alternative(thay the) value if an expression is NULL.
      * SELECT ProductName, UnitPrice \* ( UnitsInStock + IFNULL(UnitsOnOrder, 0))

FROM Products;



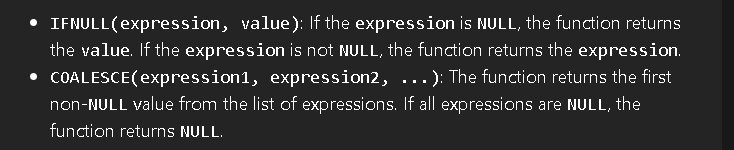
\*\*\*\* : IFNULL(column, số thay thế nếu NULL)

* + - **MySQL COALESCE() Function** :
      * SELECT ProductName, UnitPrice \* ( UnitsInStock + COALESCE(UnitsOnOrder, 0))

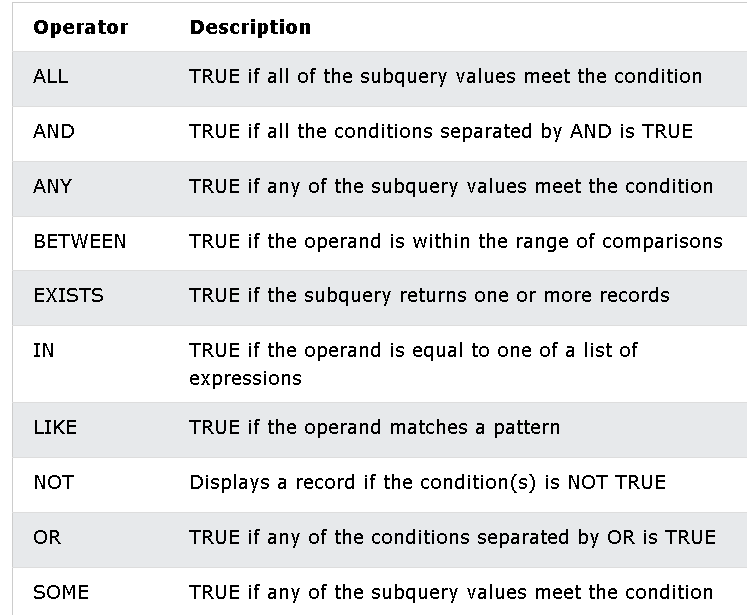
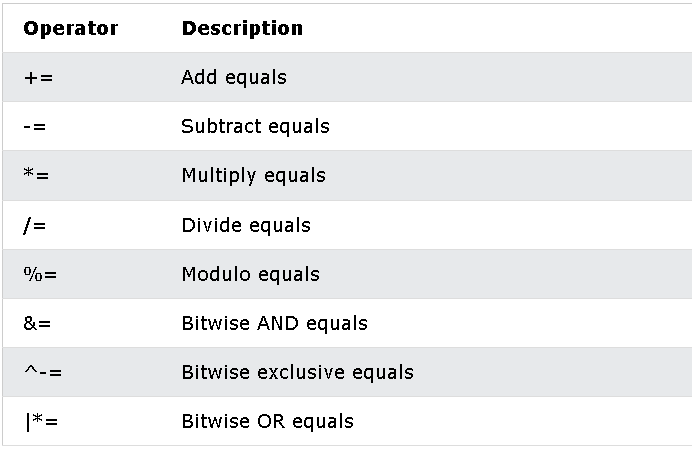
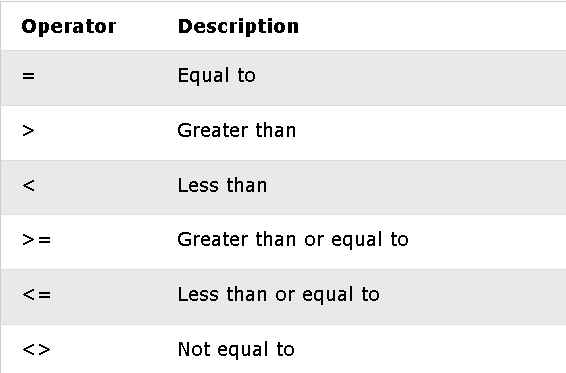
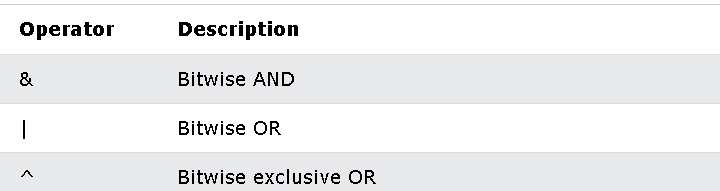
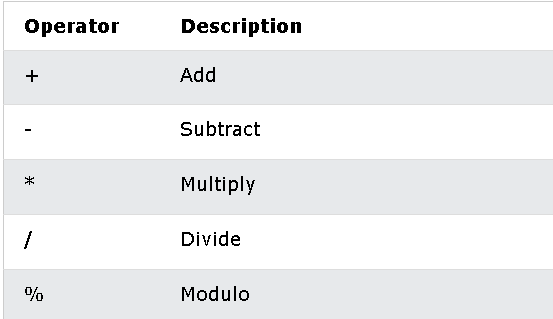
FROM Products;



* + - * ***The differences of IFNULL and COALESCE*** *:*

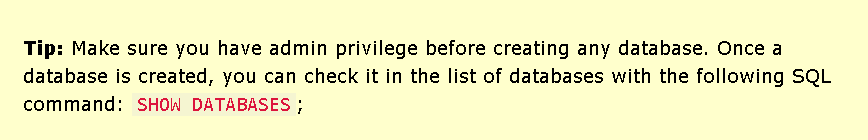


* **MySQL Comments** :
  + Comments: used to explain sections of SQL statements, or to prevent execution of SQL statements.
  + **Single Line Comments** : –
  + **Multi-Line Comments** : /\* and end with \*/
* **MySQL Operators** :

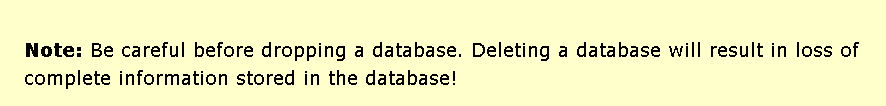


26/10/2024

* **MySQL CREATE DATABASE Statement** :
  + CREATE DATABASE : used to create a new SQL database.
    - **Syntax** :
      * CREATE DATABASE *databasename;*
      * Example :
        + CREATE DATABASE testDB;



* + - * ***Check the list of databases*** : SHOW DATABASES
* **MySQL DROP DATABASE Statement** :
  + DROP DATABASE : used to drop an existing SQL database.
    - **Syntax** :
      * DROP DATABASE *databasename*;



* + - * Example :
        + DROP DATABASE testDB;
* **MySQL CREATE TABLE Statement** :
  + CREATE TABLE : used to create a new table in a database.
    - **Syntax** :
      * CREATE TABLE *table\_name (*

*column1 datatype,*

*column2 datatype,*

*column3 datatype,*

*….*

*);*

* + - * + *The column parameters ( thông số ) specify the names of the columns of the table.*
        + *The datatype parameter : specify the type of data the column can hold ( varchar, integer, …)*

* + - Example :
      * CREATE TABLE Person (

PersonID int,

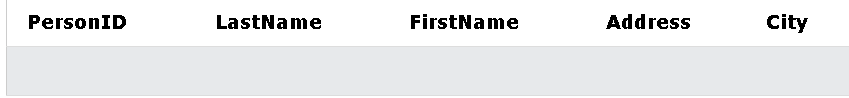
LastName varchar(225),

FirstName varchar(225),

Address varchar(225),

City varchar(225)

);



=>>Tip: The empty "Persons" table can now be filled with data with the SQL [INSERT INTO](https://www.w3schools.com/MySQL/mysql_insert.asp) statement.

* + **Create Table Using Another Table** : ( Copy an existing table)
    - The new table gets the same column definitions. -> All columns or specific columns can be selected.
    - If you create a new table using an existing table, the new table will be filled with the existing values from the old table.
    - **Syntax** :
      * CREATE TABLE *new\_table\_name* AS

SELECT *column1, column2,...*

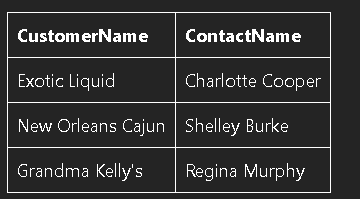
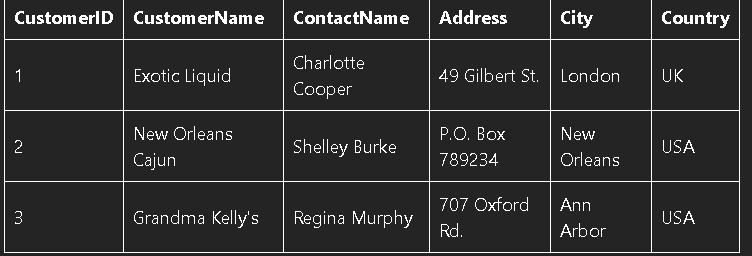
FROM *existing\_table\_name*

WHERE *…;*

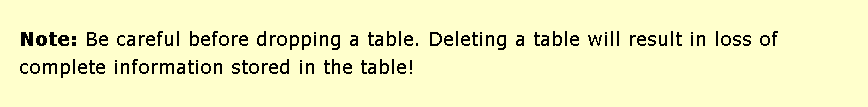
CREATE TABLE TestTable AS

SELECT customername, contactname

FROM customers;



* **MySQL DROP TABLE Statement** :
  + DROP TABLE : used to drop an existing table in a database.
  + **Syntax** :
    - DROP TABLE *table\_name;*



* + **MySQL TRUNCATE TABLE** :
    - TRUNCATE TABLE : used to delete the data inside a table, but not the table itself.
    - **Syntax** :
      * TRUNCATE TABLE *table\_name;*

=> Remove all the values of that table.

* **MySQL ALTER TABLE Statement** :
  + ALTER TABLE : used to add, delete or modify columns in an existing table.
  + ALTER TABLE : used to add and drop various constraints(ràng buộc khác nhau) on an existing table.
  + **ALTER TABLE - ADD Column** :
    - *ALTER TABLE table\_name*

*ADD column\_name datatype;*

* + - * Example :
        + ALTER TABLE Customers

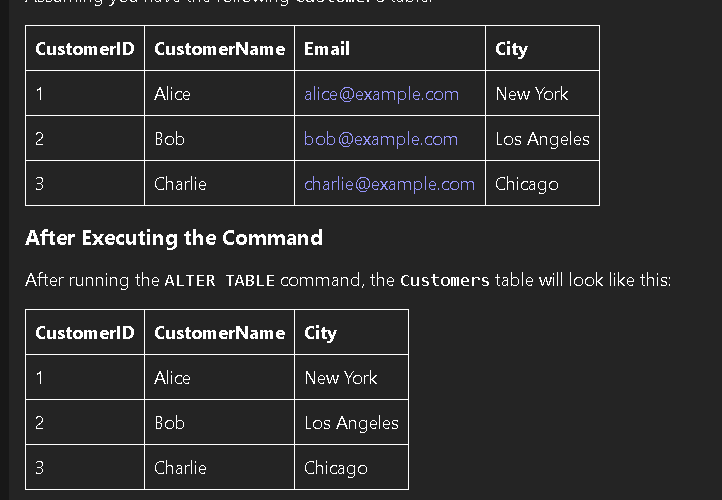
ADD Email varchar(255);

* + **ALTER TABLE - DROP COLUMN** :
    - *ALTER TABLE table\_name*

*DROP COLUMN column\_name*;

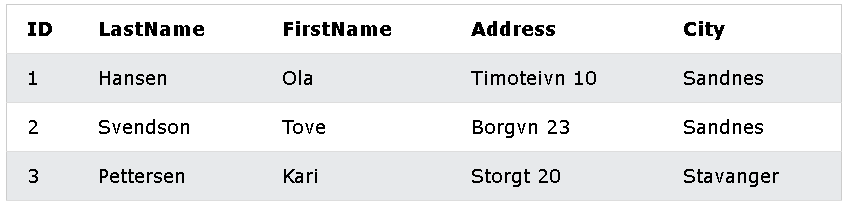
* + - * Example :
        + ALTER TABLE Customers

DROP COLUMN Email;

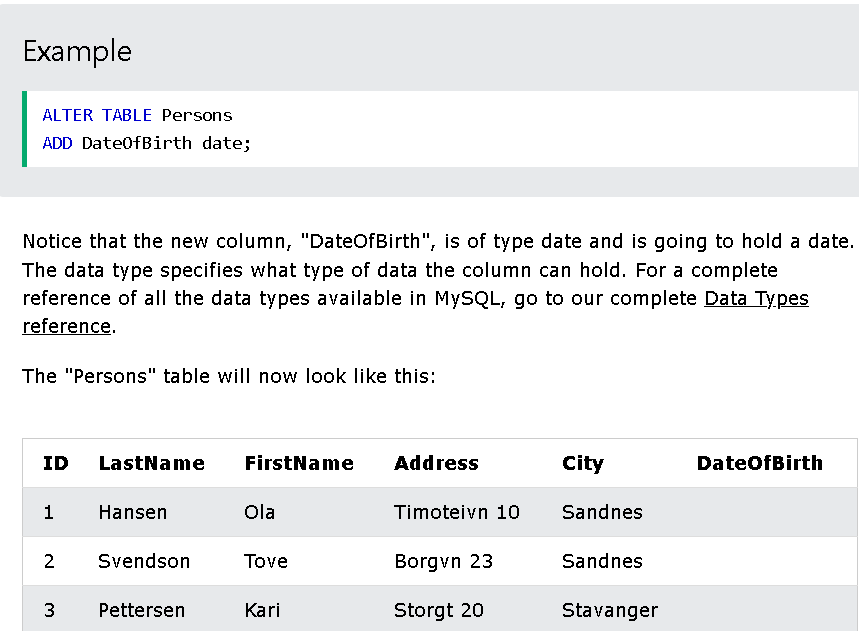


* + **ALTER TABLE - MODIFY COLUMN** :
    - ALTER TABLE *table\_name*

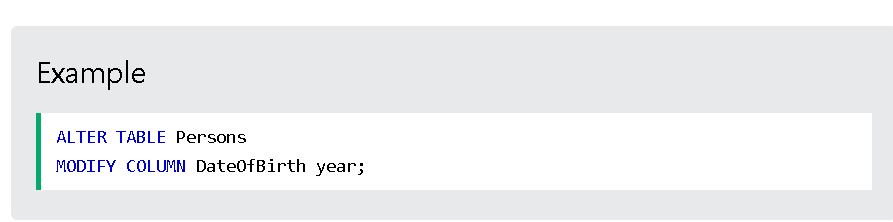
MODIFY COLUMN *column\_name datatype;*



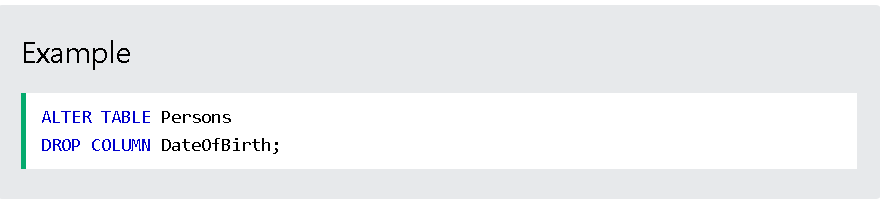
* + Example :

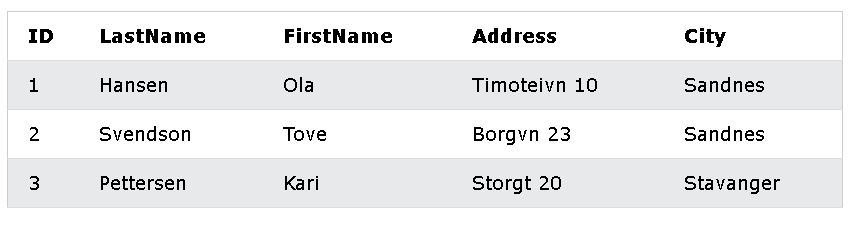


=> ADD



=> MODIFY

 => DROP



* **MySQL Constraints** :
  + Constraints : used to specify rules for data in a table.
    - Can be specified when the table is created with the CREATE TABLE.
    - Can be specified after the table is created with ALTER TABLE.
  + **Create Constraints** :
    - **Syntax** :
      * CREATE TABLE *table\_name (*

*column1 datatype* constraint*,*

*….*

*);*

* + **MySQL Constraints** :
    - SQL constraints : used to specify the rules for data in a table.
    - Used to limit the type of data that can go into a table.
      * -> Ensure the accuracy(công bằng) and reliability(tin cậy) of the data in the table.
      * -> If there is any violation(vi phạm) between the constraint(ràng buộc) and the data action, the action is aborted.(hủy bỏ)
    - Can be column or table level.
      * Column level : apply to a column.
      * Table level : apply to the whole table.
    - Constraints :
      * NOT NULL : Ensure a column cannot have a NULL value
      * UNIQUE : Ensure all values in a column are different
      * PRIMARY KEY : Combination of NOT NULL and UNIQUE.
        + Uniquely identifies(xác định duy nhất) each row in a table.
      * FOREIGN KEY : Prevent actions would destroy links between tables.
      * CHECK : Ensure the values in a column satisfies(thỏa mãn) a specific condition.
      * DEFAULT : Set a default value for a column id no value is specified
      * CREATE INDEX : Use to create and retrieve data from a database very quickly.
* **MySQL NOT NULL Constraint** :
  + By default, a column can hold NULL values.
  + NOT NULL : enforces ( buộc ) a column to NOT accept NULL values.
    - -> Enforces (thực thi ) a field (1 trường) always contain a value => Cannot insert a new record or update a record without adding a value.
      * **Syntax** :
        + …. *column\_name datatype* NOT NULL*;* ….

* **MySQL UNIQUE Constraints** :
  + UNIQUE : unsure all values in a column are different.
  + Both UNIQUE and PRIMARY KEY : provide a guarantee ( đảm bảo ) for uniqueness ( tính duy nhất ) for a column or set of columns.
  + PRIMARY KEY : automatically have a UNIQUE constraint.

***\*\*\*\* : Can have many UNIQUE constraints per table, but only 1 PRIMARY KEY constraint per table.***

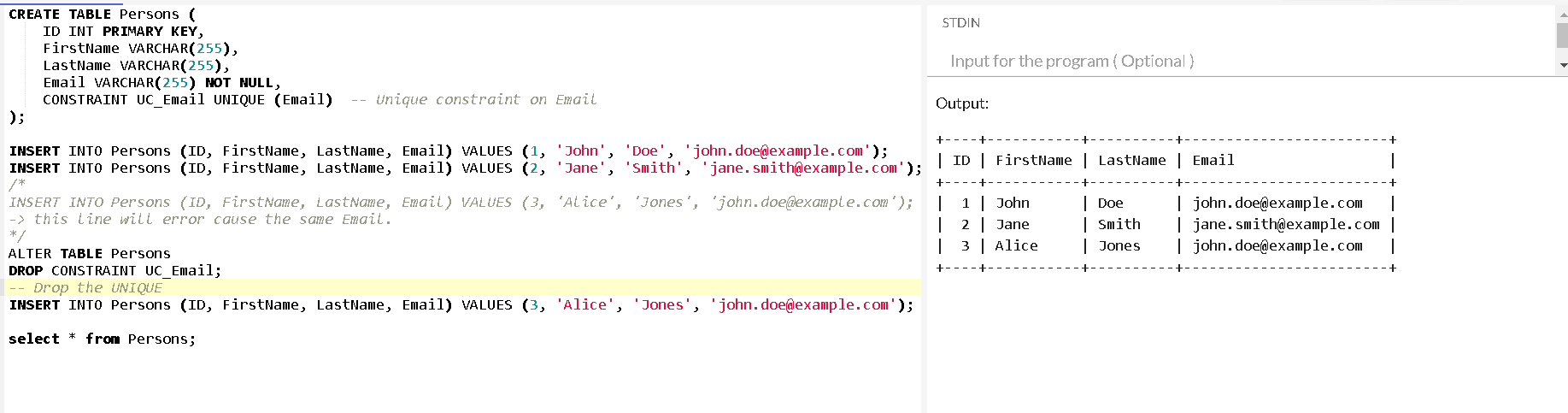
* + - **Syntax** :
      * ***On CREATE TABLE*** :
        + CREATE TABLE *table\_name (*

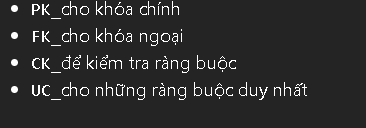
*name\_column1 datatype (*NOTNULL*),*

*…..,*

CONSTRAINT *UC\_constraint\_name* UNIQUE *(column\_name,...)*

*);*

**

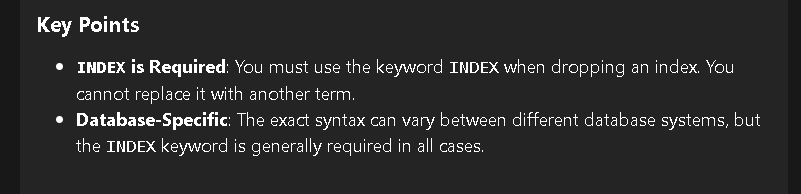


\* constraint\_type : UNIQUE , PRIMARY KEY ,...

* + - * + ***On ALTER TABLE*** :

ALTER TABLE *table\_name*

DROP INDEX *index\_name;*



* **MySQL PRIMARY KEY Constraint** :
  + PRIMARY KEY : uniquely identify each record in a table.
  + PRIMARY KEY : must contain UNIQUE values, cannot contain NULL values.
  + A table can have only ONE primary key.
  + The primary key can consist of single or multiple columns.
    - * + **Syntax** :

***CREATE TABLE*** :

CREATE TABLE *table\_name (*

*name\_column1 datatype (*NOTNULL*),*

*…..,*

CONSTRAINT *PK\_constraint\_name* PRIMARY KEY *(column\_name,...)*

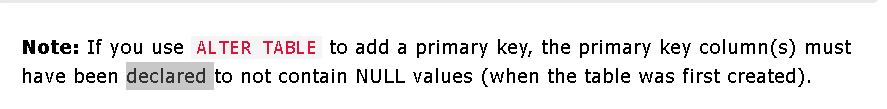
*);*

\*\*\*\* : The VALUE of the primary key can be made up of **MANY COLUMNS** (column\_name,...)

***ALTER TABLE*** :

ALTER TABLE *table\_name*

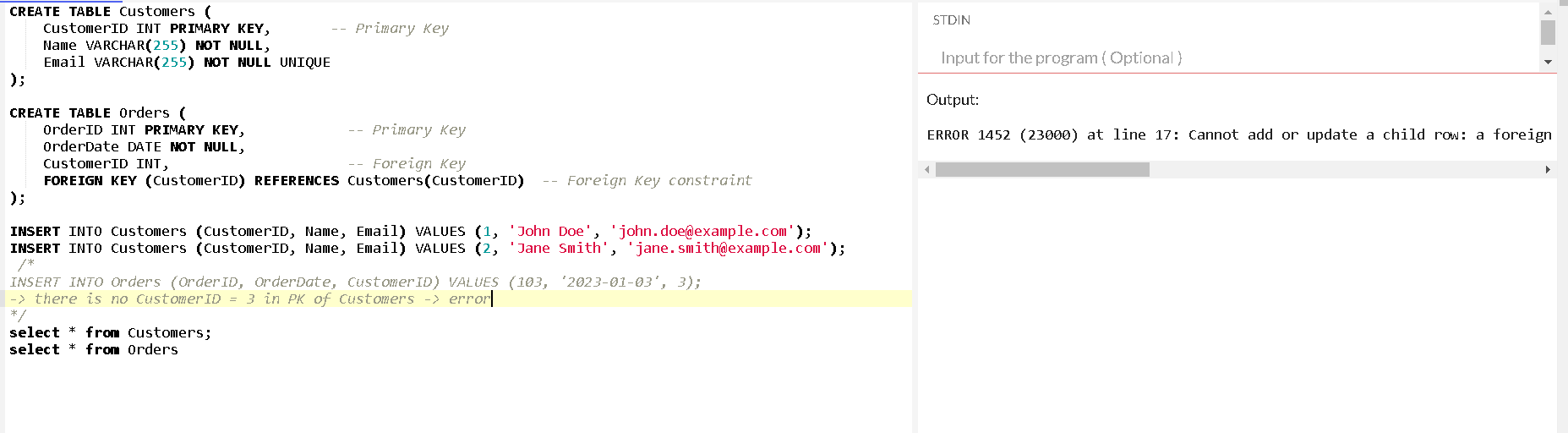
ADD CONSTRAINT *PK\_name* PRIMARY KEY *( column\_name,...);*



ALTER TABLE *table\_name*

DROP PRIMARY KEY*;*

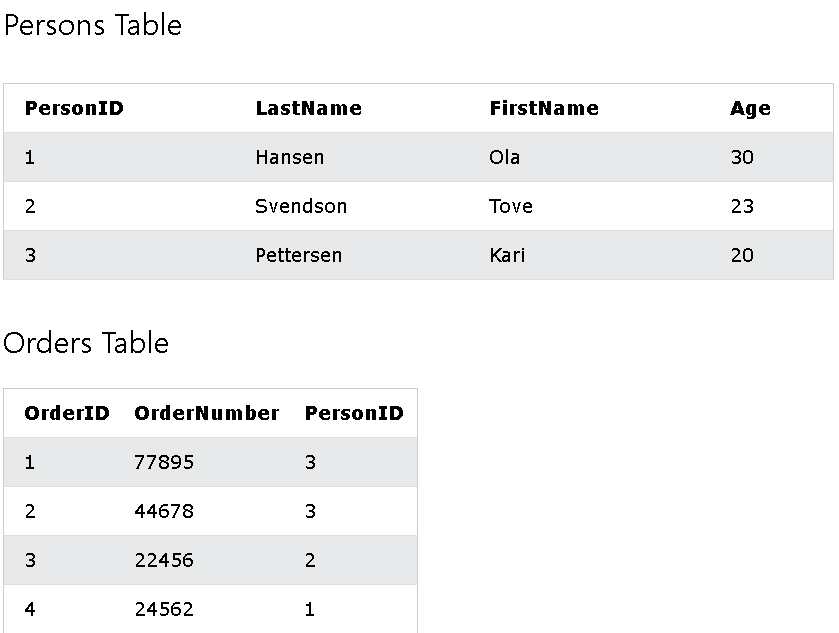
* **MySQL FOREIGN KEY Constraint** :
  + FOREIGN KEY : used to prevent actions would destroy links between tables.
  + FOREIGN KEY : a field ( collection of fields ) in 1 table -> refer to PRIMARY KEY in another table.



**\*\*\*\* : The table with the foreign key : child table // The table with the primary key : referenced or parent table.**

**=> “PersonID”** in “**Persons”** table is the PRIMARY KEY

**=> “ PersonID”** in “**Orders”** table is a FOREIGN KEY

\*\*\*\* FOREIGN KEY : prevent invalid data ( dữ liệu k hợp lệ ) from being inserted (chèn ) into the foreign key column, <-> It has to be one of the values contained in the parent table.****

* + ***On CREATE TABLE*** :
    - CREATE TABLE Orders (

OrderID int NOT NULL,

OrderNumber int NOT NULL,

PersonID int,

PRIMARY KEY ( OrderID),

CONSTRAINT FK\_PersonOrder FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)

);



* + ***On ALTER TABLE*** : ( want to create a FOREIGN KEY after creating “**Orders**” :
    - ALTER TABLE Orders

ADD CONSTRAINT FK\_PersonOrder

FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);



* + ***DROP a FOREIGN KEY Constraint*** :
    - ALTER TABLE Orders

DROP FOREIGN KEY FK\_PersonOrder;

* **MySQL CHECK Constraint** :
  + CHECK : used to limit the value range(đánh giá) can be placed (đặt) in a column.
  + Define a CHECK on a column -> Allow only certain values for this column.
  + Define a CHECK on a table -> Limit the values in certain columns based on values in other columns in the row.
  + ***On CREATE TABLE*** :
    - CREATE TABLE *table\_name* (

*column\_name datatype* (NOT NULL),

….,

CONSTRAINT *CHK\_Person* CHECK ( *condition* )

);

* + ***On ALTER TABLE*** :
    - ALTER TABLE Persons

ADD CONSTRAINT CHK\_PersonAge CHECK (Age >= 18 AND City = ‘Sandnes’);

* + ***DROP a CHECK*** :
    - ALTER TABLE Persons

DROP CHECK CHK\_PersonAge;

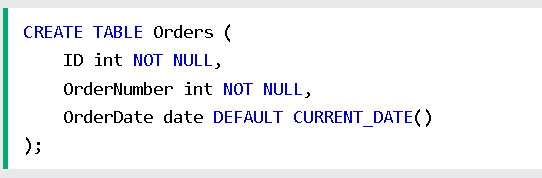
* **MySQL DEFAULT Constraint** :
  + DEFAULT : used to set a default value for a column.
  + The default value: be added to all new records, if no other value is specified.
  + ***On CREATE TABLE*** :
    - CREATE TABLE *table\_name (*

*column\_name datatype (*NOTNULL*),*

*…..,*

*column\_nameN datatype* DEFAULT  *‘value’*

*);*

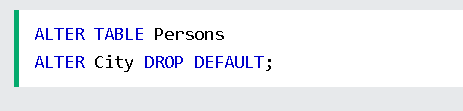
\*\*\*\* : DEFAULT : used to insert system values, by using functions like CURRENT\_DATE():   


***=> If OrderDate = NULL - > Set = Date current.***

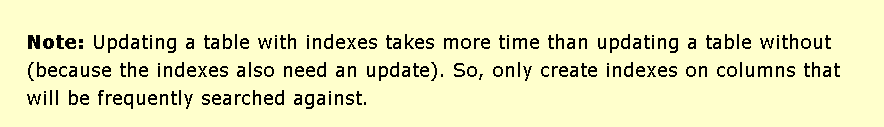
* + ***On ALTER TABLE*** :



* + ***DROP a DEFAULT*** :

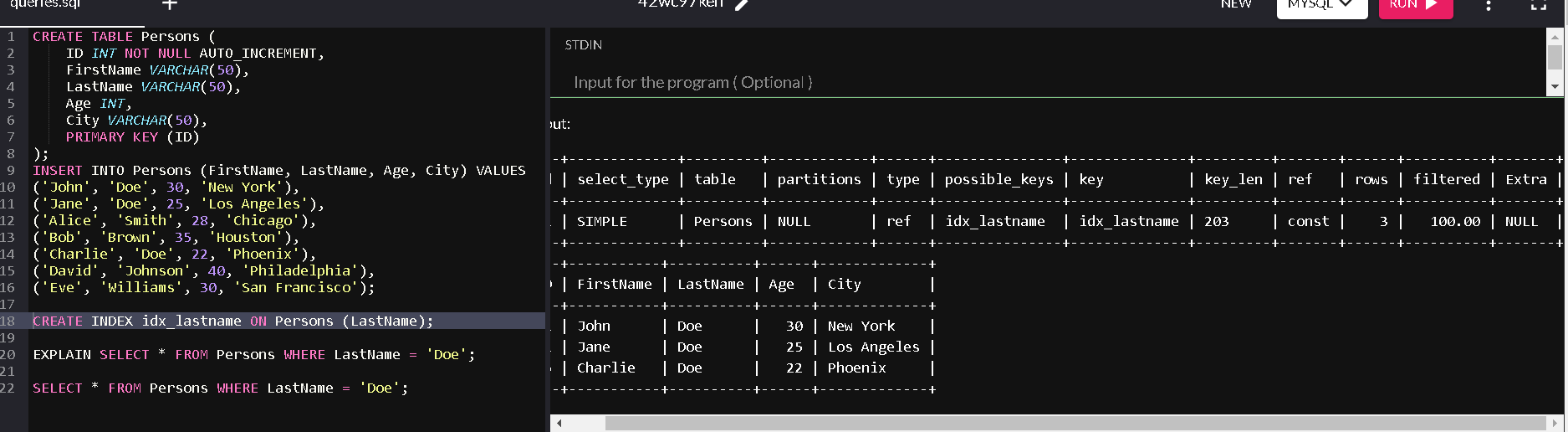


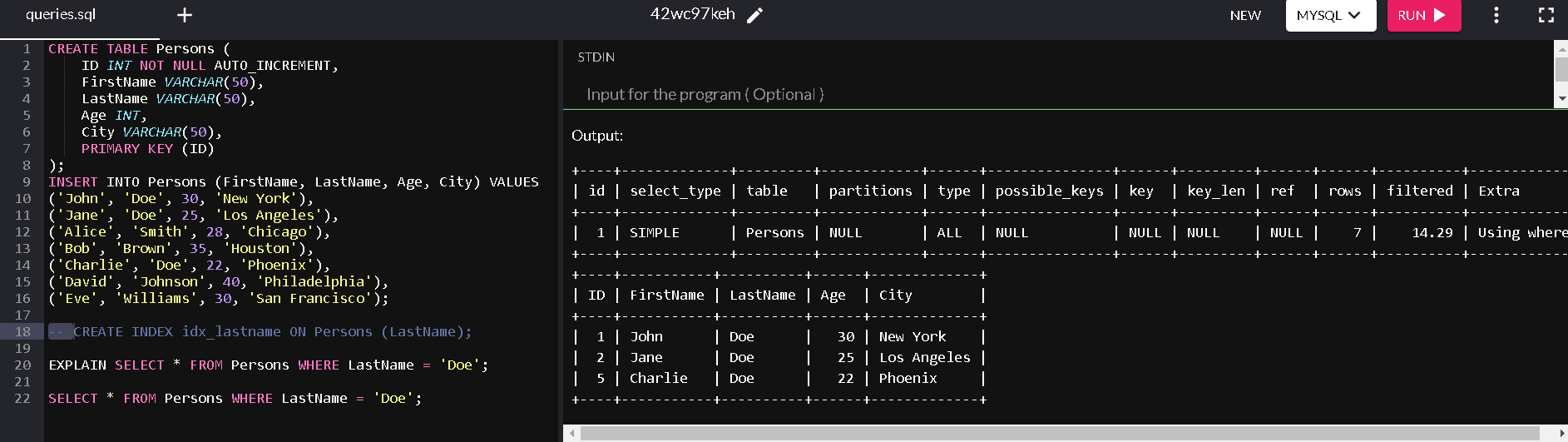
* **MySQL CREATE INDEX Statement** :
  + CREATE INDEX : used to create indexes in tables.
    - Indexes : used to retrieve ( truy xuất ) data from the database more quickly than otherwise.The users cannot see the indexes, they are just used to speed up searches/queries.



* + **CREATE INDEX Syntax :** 
    - CREATE ( UNIQUE ) INDEX *index\_name*

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

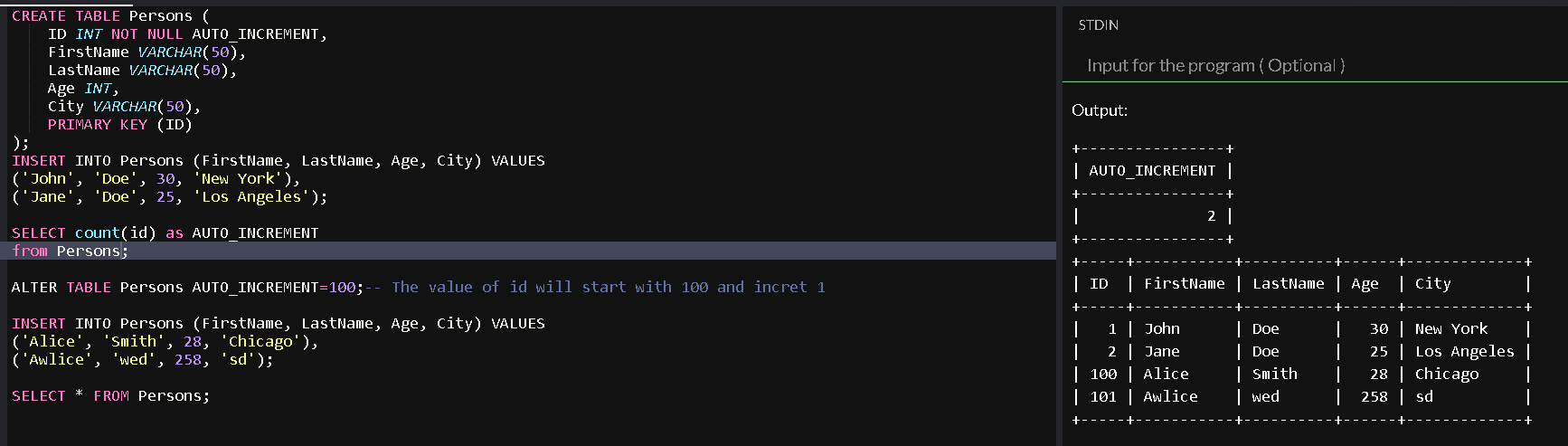
=> using create table: 

=> not using create table:

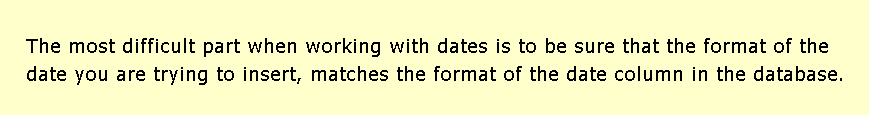
* + ***DROP INDEX Statement*** :
    - ALTER TABLE *table\_name*

DROP INDEX *index\_name*;

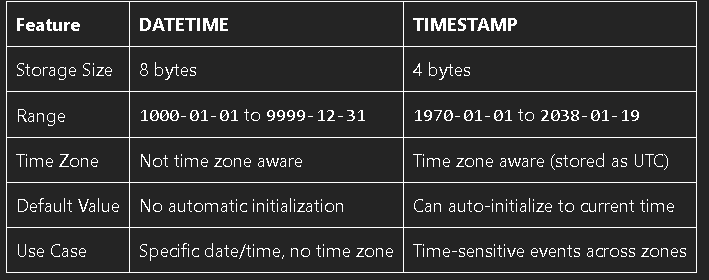
* **MySQL AUTO INCREMENT Field** : allow a unique number to be generated automatically when a new record is inserted into a table.(-> often the primary key that created automatically every time a new record inserted)
  + AUTO\_INCREMENT : perform an auto-increment feature.-> Start with 1.
  + **Syntax** :
    - …. *column\_name datatype* (NOT NULL) AUTO\_INCREMENT,  
      ….
  + Let AUTO\_INCREMENT start with another value :
    - … AUTO\_INCREMENT = value;



* **MySQL Working With Dates** :



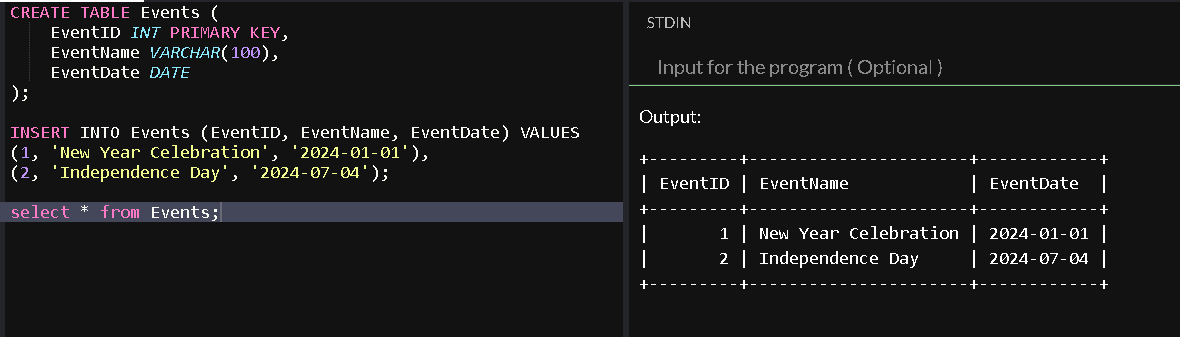
* + **MySQL Date Data Types** :
    - DATE : - format YYYY-MM-DD
    - DATETIME : - format YYYY-MM-DD HH:MI:SS (8 bytes)



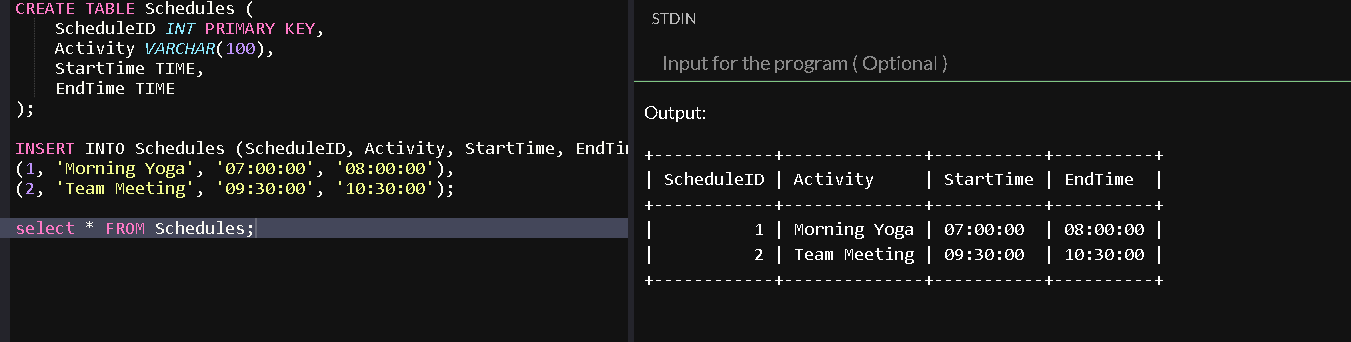
* + - TIMESTAMP : - format YYYY-MM-DD HH:MI:SS(4 bytes)
    - YEAR : - format YYYY or YY

=>> The date data type is set for a column when creating a new table in the database.

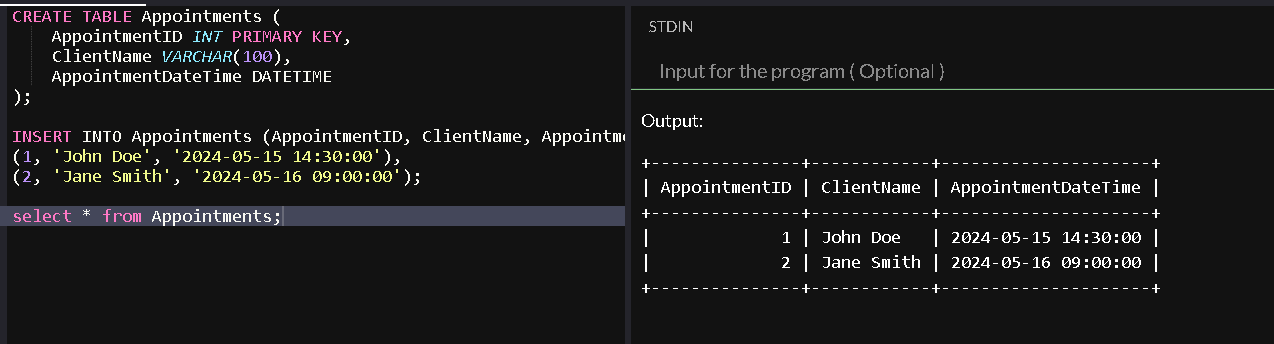
* + - Example :
      * **DATE** :



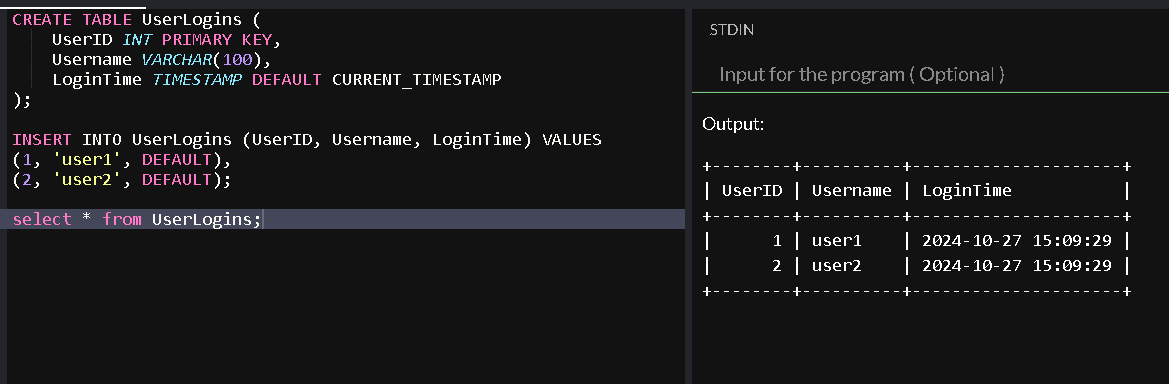
* + - * **TIME** :



* + - * **DATETIME** :



* + - * **TIMESTAMP** :



* **MySQL Views** :
  + **MySQL CREATE VIEW Statement** :
    - A view : contain rows and columns- like a real table.
    - Can add SQL statements and functions to a view and present the data as if(như thể) the data were coming from 1 single table.
    - **CREATE VIEW Syntax** :
      * CREATE VIEW *view\_name* AS

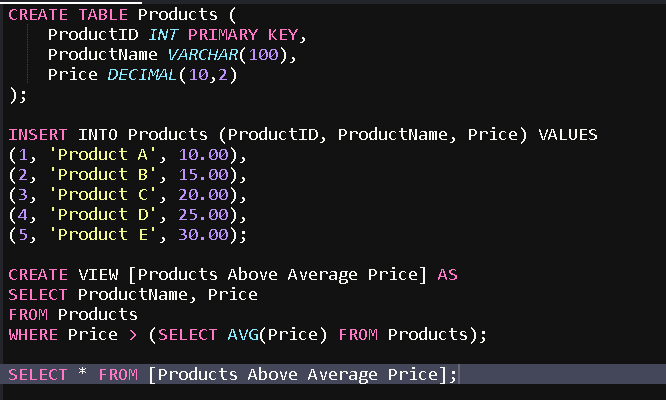
SELECT *column1, column2, …*

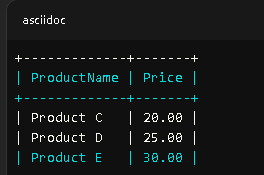
FROM *table\_name*

WHERE *condition;*

=> A view always shows up-to-date data -> The database engine recreates the view, every time a user queries it.

* + - Example :



=>

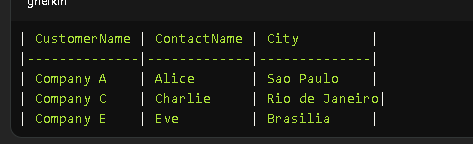
* + **MySQL Updating a View** :
    - **CREATE OR REPLACE VIEW**  :
      * CREATE OR REPLACE VIEW view\_name AS

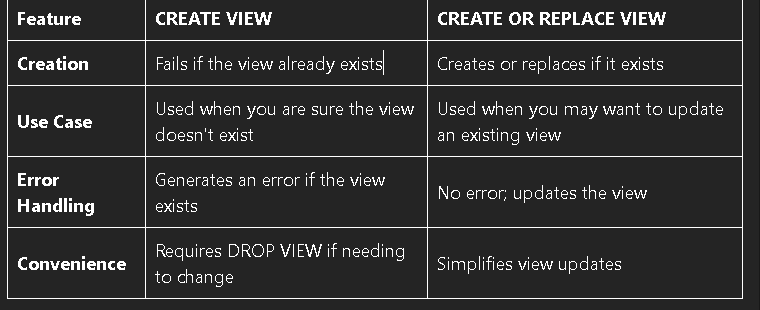
SELECT column1, column2,...

FROM table\_name

WHERE condition;

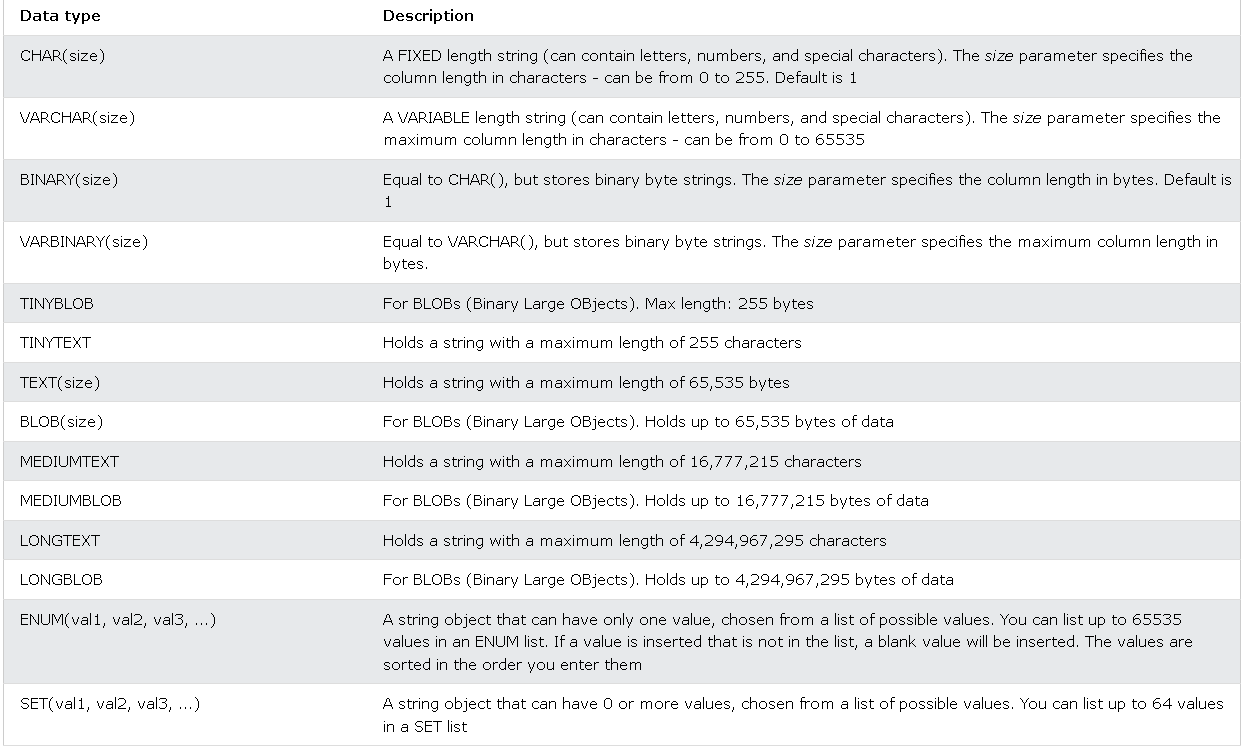


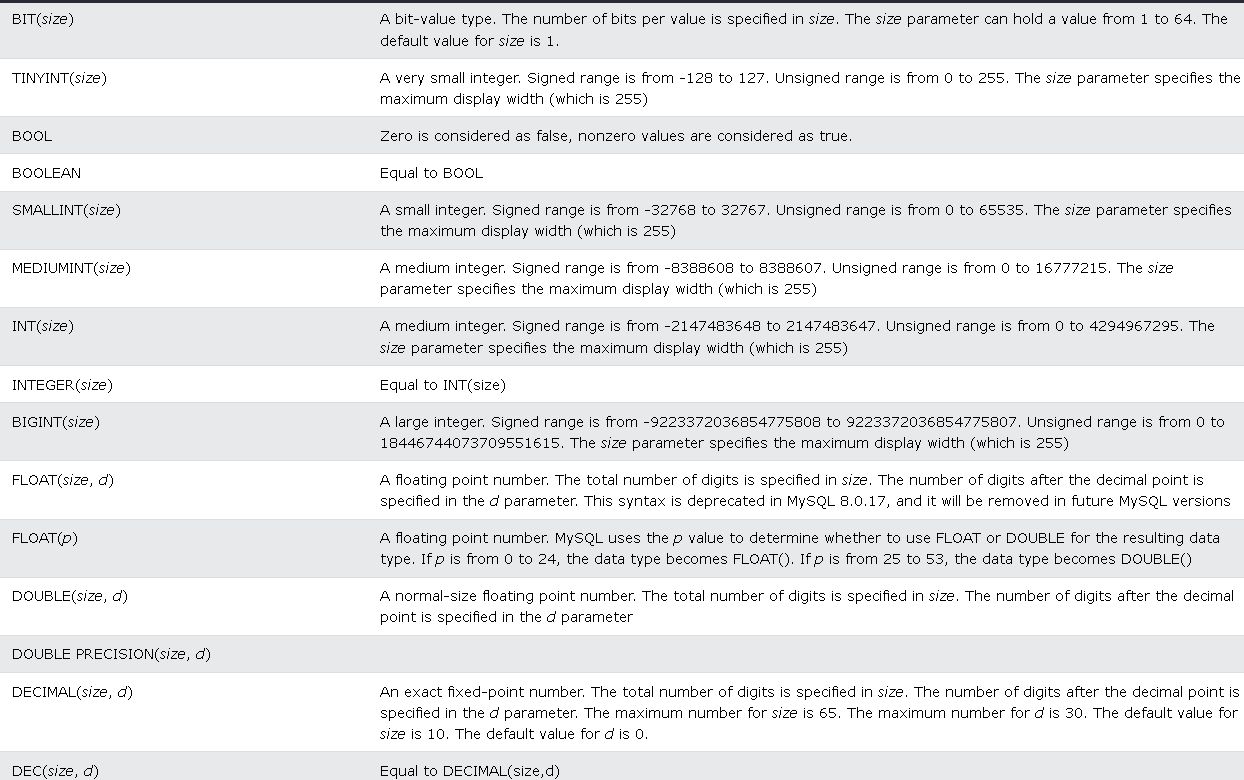
=>



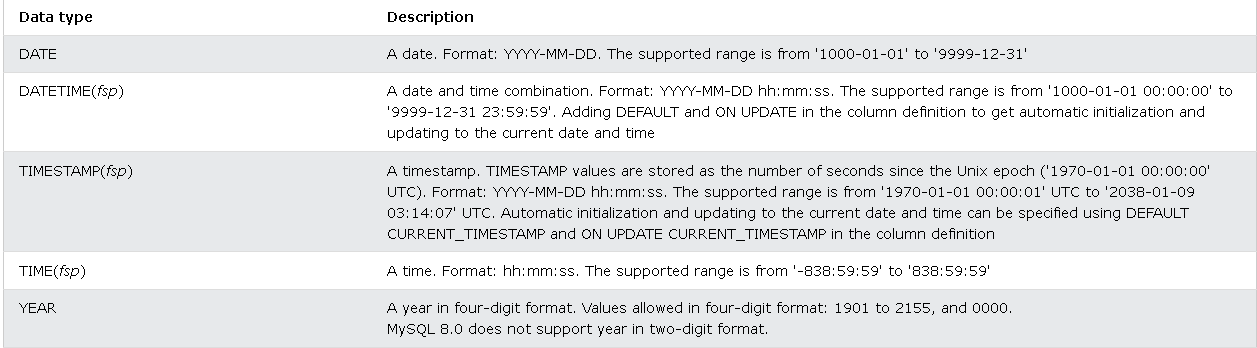
* **MySQL DATA Types** :

String Data Types :



Numeric Data Types : 

Date and Time Data Types :



* **MySQL Functions** :
  + Total : [here](https://www.w3schools.com/MySQL/mysql_ref_functions.asp).