SQL:-

- SQL stands for Structured Query Language.
- It is a language which is used to create, remove, alter the database.
- We can store, retrieve, update the data in a database using SQL.
- SQL works for all modern relational database management systems, like SQL Server, Oracle, MySQL, etc.

Different types of SQL commands are:

1. DDL – Data Definition Language
2. DQL – Data Query Language
3. DML – Data Manipulation Language
4. DCL – Data Control Language
5. TCL- Transaction Control Language

MySQL:-

is a relational database management system that is a RDBMS developed by Oracle based on structured query language (SQL).

Difference:-

SQL	**MySQL**
SQL is a Structured Query Language. It is useful to manage relational databases.	MySQL is an RDBMS to store, retrieve, modify and administrate a database using SQL.
SQL is a query **language**.	MYSQL is used as an **RDBMS** database.
To query and operate database systems.	Allows data handling, storing, modifying, deleting in a tabular format.

We have 3 main types of data types-

String Data Types:

Datatype	Description
CHAR(size)	A FIXED length string (can contain letters, numbers, and special characters). The *size* parameter specifies the column length in characters - can be from 0 to 255. Default is 1
VARCHAR(size)	A VARIABLE length string (can contain letters, numbers, and special characters). The *size* parameter specifies the maximum column length in characters - can be from 0 to 65535
BINARY(size)	Equal to CHAR(), but stores binary byte strings. The *size* parameter specifies the column length in bytes. Default is 1
VARBINARY(size)	Equal to VARCHAR(), but stores binary byte strings. The *size* parameter specifies the maximum column length in bytes.

Numeric Data Types:

Datatype	Description
BIT(*size*)	A bit-value type. The number of bits per value is specified in *size*. The *size* parameter can hold a value from 1 to 64. The default value for *size* is 1.
TINYINT(*size*)	A very small integer. Signed range is from -128 to 127. Unsigned range is from 0 to 255. The *size* parameter specifies the maximum display width (which is 255)
BOOLEAN (Not in MySQL)	Zero is considered as false, nonzero values are considered as true.

INT(*size*)/ INTEGER(*size*)	Signed range is from -2147483648 to 2147483647. Unsigned range is from 0 to 4294967295. The *size* parameter specifies the maximum display width (which is 255)
FLOAT(*p*)	A floating point number. MySQL uses the *p* value to determine whether to use FLOAT or DOUBLE for the resulting data type. If *p* is from 0 to 24, the data type becomes FLOAT(). If *p* is from 25 to 53, the data type becomes DOUBLE()
DECIMAL(*size*, *d*)	An exact fixed-point number. The total number of digits is specified in *size*. The number of digits after the decimal point is specified in the *d* parameter. The maximum number for *size* is 65. The maximum number for *d* is 30. The default value for *size* is 10. The default value for *d* is 0.

Date and Time Data Types:

Datatype	**Description**
DATE	Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'
DATETIME	A date and time combination. Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'.
TIME	Format: hh:mm:ss. The supported range is from '-838:59:59' to '838:59:59'
TIMESTAMP	TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD hh:mm:ss. The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC.

Commands and their functionalities:

- **DDL (Data Definition Language):**

CREATE	Create TABLE, DATABASE, INDEX or VIEW
DROP	Delete TABLE, DATABASE, or INDEX
ALTER TABLE	Add/Remove columns from table
TRUNCATE	Removes all records from a table.
RENAME	Rename an object existing in the database.

- **DML (Data Manipulation Language):**

INSERT	Insert data into a table.
UPDATE	Update table data.
DELETE	Delete rows from a table.

- **DQL (Data Query Language):**

SELECT	Select data from database.

- **DCL (Data Control Language):**

GRANT	Access privileges to the database.
REVOKE	Withdraws the user's access privileges.

- **TCL (Transaction Control Language):**

BEGIN TRANSACTION	used to begin a transaction.
COMMIT	used to apply changes and end transactions.
ROLLBACK	used to discard changes and end transactions.
SAVEPOINT	points within the groups of transactions in which to ROLLBACK.

- **Few more important clauses:**

AS	Rename an attribute or table with an alias.
FROM	Specifying the table we are accessing the data from.
WHERE	Conditional statement to filter the data.

● **JOIN**	Combine rows from 2 or more tables.
● **AND**	Combine conditions in the query. All must be met.
● **OR**	Combine conditions in a query. One must be met.
● **LIKE**	Search for patterns in a column. (Regex operations)
● **IN**	Specify multiple values when using WHERE.
● **IS NULL**	Return only rows with a NULL value.
● **LIMIT**	Limit the number of rows returned.
● **CASE**	Return value on a specified condition.

Filtering and Sorting Data

We use the **WHERE** clause in our query as a Conditional statement to filter the data.
Like- SELECT column_name(s) FROM T_name WHERE conditions;

When filtering the strings:

Wildcards:

Symbol	**Description**
%	Represents zero or more characters
_	Represents a single character

Few examples:
- 'a%' - Find any value that starts with "a".
- '%or%' - Finds any values that have "or" in any position.
- '_r%'- Finds any values that have "r" in the second position.
- 'a%o'- Finds any values that starts with "a" and ends with "o"

We use wildcards with LIKE operators.
Query:- SELECT column_name(s) FROM T_name WHERE column_name LIKE '%o_r%';

Sorting:
For sorting we use ORDER BY.

ORDER BY Set order of result. Use DESC to reverse order, ASC is default.

Query:- SELECT column_name(s) FROM table_name ORDER BY column_name(s) ASC|DESC;

Grouping Data

GROUP BY Group rows that have the same values into summary rows.

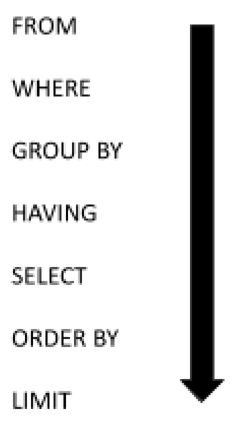
Query:- SELECT column_name(s) FROM T_name WHERE condition GROUP BY column_name(s);

HAVING Same as WHERE but used for aggregate functions.

Query:- SELECT column_name(s) FROM T_name WHERE condition GROUP BY column_name(s) HAVING condition;

- **Aggregate Functions:**
    - **SUM** Returns sum of column
      Query:- SELECT SUM(items) AS TotalItems FROM Order;
    - **AVG** Returns average of column
      Query:- SELECT AVG(Price) AS AveragePrice FROM Products;
    - **MIN** Returns min value of column
      Query:- SELECT MIN(Price) AS CheapestItemcost FROM Products;
    - **MAX** Returns max value of column
      Query:- SELECT MAX(Price) AS Costliest FROM Products;
    - **COUNT** Count number of rows
      Query:- SELECT COUNT(ProductID) AS NumberOfProducts FROM Products;

Order of Execution:


FROM

WHERE

GROUP BY

HAVING

SELECT

ORDER BY

LIMIT

Managing Tables:

- Create a new table with three columns:
  CREATE TABLE T_name(
  id INT PRIMARY KEY,
  name VARCHAR NOT NULL
  price INT DEFAULT 0
  course_id INT
  FOREIGN KEY(course_id) REFERENCES parent_T_name(course_id)
  );

- Delete the table from the database
  DROP TABLE T_name;
- Add a new column to the table
  ALTER TABLE T_name ADD column;

- Drop column c from the table
  ALTER TABLE T_name DROP COLUMN c;

- Add a constraint
  ALTER TABLE T_name ADD constraint;
  Note: possible constraints could be like Foregin key, unique,or checks.

- Drop a constraint
  ALTER TABLE T_name DROP constraint;

<u>Constraints:</u>

Constraint	Description
CHECK	determines whether the value is valid or not from a logical expression.
FOREIGN KEY	Link between two tables by one specific column of both tables. The specified column in one table

	must be a PRIMARY KEY and referred by the column of another table known as FOREIGN KEY.
UNIQUE	Maintains the uniqueness of a column in a table. More than one UNIQUE column can be used in a table.
NOT NULL	column can not contain any NULL value
PRIMARY KEY	Enforces the table to accept unique data for a specific column and is a unique index for accessing the table faster.

- Rename a table from T_name to T_new_name

  ALTER TABLE T_name RENAME TO T_new_name;

- Rename column c1 to c2

  ALTER TABLE T_name RENAME c1 TO c2;

- Remove all data in a table

  TRUNCATE TABLE T_name;

★ **Difference between Delete, Drop and Truncate:**

Delete	**Drop**	**Truncate**
DML command	DDL command	DDL command
Removes one, some or all the records in the table.	Removes the entire table structure.	Removes all the records from the table.
Is a slow operation	Relatively faster	Fastest of all.

★ **Difference between Modify, Alter, Change:**

Alter	Change	Modify
Used to set or remove the default value for a column	Used to rename a column, change its datatype, or move it within the schema.	Can't rename a column, rest works the same as CHANGE.
Eg: ALTER TABLE T_name ALTER COLUMN floc SET DEFAULT 'bar';	Eg: ALTER TABLE T_name CHANGE COLUMN floc VARCHAR(32) NOT NULL FIRST;	Eg: ALTER TABLE T_name MODIFY COLUMN floc VARCHAR(32) NOT NULL AFTER contact_no;

Modifying Data:

- Insert one row into a table
  INSERT INTO T_name(column_name(s)) VALUES(value_list);

- Insert multiple rows into a table
  INSERT INTO T_name(column_name(s)) VALUES (value_list), (value_list), (value_list),....,(value_list);

- Insert rows from T_name into T_new_name
  INSERT INTO T_new_name(column_name(s)) SELECT column_name(s) FROM T_name;

- Update new value in the column c1 for all rows
  UPDATE T_name SET c1= new_value;

- Update values in the column c1, c2 that match the condition
  UPDATE T_name SET c1= new_value, c2= new value WHERE condition;

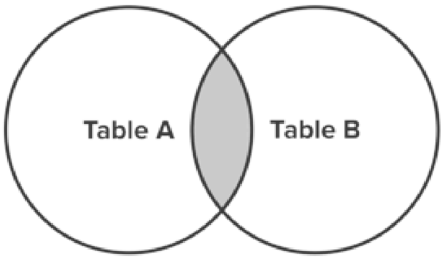
- Delete all data in a table
  DELETE FROM T_name;

- Drop the table.
  DROP TABLE T_name;

- Replace command to Insert a new row into the table, and if a duplicate key error occurs it internally first deletes the already present key and inserts the new one.
  REPLACE [INTO] T_name(column_name(s)) VALUES(value_list);

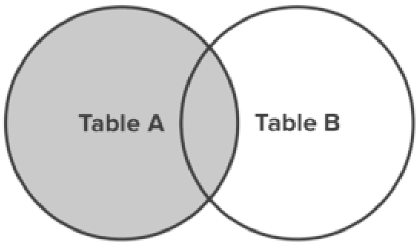
Joins:

- **Inner Join:**
  A Inner Join B,



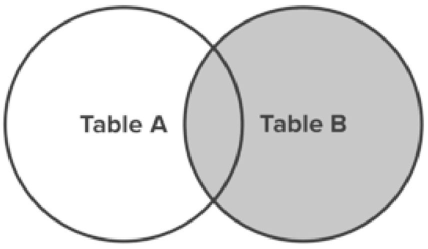
  Query:- SELECT column_name(s) FROM A INNER JOIN B ON A.column_name = B.column_name;

- **Left Join:**
  A Left Join B,



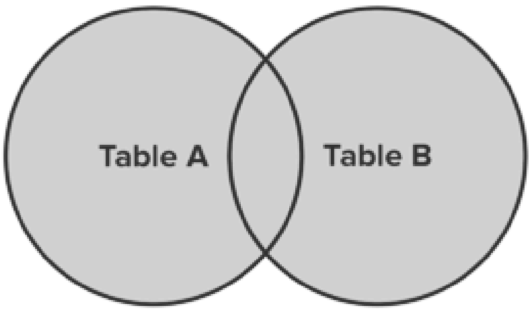
  Query:- SELECT column_name(s) FROM A LEFT JOIN B ON A.column_name = B.column_name;

- **Right Join:**
  A Right Join B,



  Query:- SELECT column_name(s) FROM A RIGHT JOIN B ON A.column_name = B.column_name;

- **Full Join:**
  A Full Join B,



Query:-

SQL:

SELECT column_name(s) FROM A FULL JOIN B ON A.column_name = B.column_name;

MySQL:

SELECT column_name(s) FROM A LEFT JOIN B ON A.column_name = B.column_name
UNION
SELECT column_name(s) FROM A RIGHT JOIN B ON A.column_name = B.column_name;

Set Operators:
Let the two tables be A and B.

- **Union:**
  ```
  SELECT column_name(s) FROM A
  UNION
  SELECT column_name(s) FROM B;
  ```

- **Union All:**
  ```
  SELECT column_name(s) FROM A
  UNION ALL
  SELECT column_name(s) FROM B;
  ```

- **Intersect:**
  Basic Syntax:
  ```
  SELECT column_name(s) FROM A
  INTERSECT
  SELECT column_name(s) FROM B;
  ```

  Above syntax doesn’t work in mysql workbench, so to emulate that we use:
  ```
  SELECT DISTINCT column_name(s) FROM A
  INNER JOIN B ON A.column_name = B.column_name;
  ```

- **Minus:**
  Basic Syntax:
  ```
  SELECT column_name(s) FROM A
  MINUS
  SELECT column_name(s) FROM B;
  ```

  Above syntax doesn’t work in mysql workbench, so to emulate that we use:

```
SELECT column_name(s) FROM A LEFT JOIN B ON
A.column_name = B.column_name WHERE B.column_name IS NULL;
```

★ **Difference between Joins and Union:**

Join	**Union**
It combines data from multiple tables based on a matched condition between them.	It combines the result of two or more SELECT statements.
New columns added to a table.	Rows are modified.
Can select different no. of columns from different tables.	Number of columns selected are the same.

Subqueries:

It exists in three clauses-

- **a WHERE clause:**

  Query:- SELECT column_list (s) FROM T_name WHERE column_name OPERATOR (SELECT column_list (s) FROM T_name [WHERE])

  Note: Operators could be equal to, IN, NOT IN, etc.

- **a FROM clause:**

  Query:- SELECT column_list (s) FROM T_name, (SELECT column_list(s) FROM T2_name GROUP BY column_list(s))
  WHERE condition;

- **a SELECT clause:**

  Query:- SELECT (SELECT column_list(s) FROM T_name WHERE condition), column_list(s) FROM T2_name WHERE condition;

TCL (Transaction Control Language):

- **BEGIN TRANSACTION** used to begin a transaction.
  Query:- BEGIN TRANSACTION transaction_name;
- **COMMIT** used to apply changes and end transactions.
  Query:- COMMIT;
- **ROLLBACK** used to discard changes and end transactions.
  Query:- ROLLBACK;
- **SAVEPOINT** points within the groups of transactions in which to ROLLBACK.
  Query:- SAVEPOINT SAVEPOINT_NAME;

Locks:

- **READ LOCK:** This lock allows a user to only read the data from a table.
- **WRITE LOCK:** This lock allows a user to do both reading and writing into a table.

Query:- LOCK TABLES T_name [READ | WRITE];

We can lock multiple tables together too.

```
Query:- LOCK TABLES T1_name [READ | WRITE],
                    T2_name [READ | WRITE],...... ,
                    Tn_name [READ | WRITE];
```

Importing :

When importing from a local computer , the client program reads the file on the client and sends it to the MySQL server.
The file will be uploaded into the database server operating system.

```
Query: - LOAD DATA LOCAL INFILE  'c:/tmp/xyz.csv'
            INTO TABLE T_name
            FIELDS TERMINATED BY ','
            ENCLOSED BY '"'
            LINES TERMINATED BY '\n'
            IGNORE 1 ROWS;
```

Exporting :
To export our data into a CSV file.

```
Query:- SELECT column_name(s) FROM T_name WHERE id = 1
            INTO OUTFILE 'C:/tmp/xyz_exported.csv'
            FIELDS ENCLOSED BY '"'
            TERMINATED BY ';'
            ESCAPED BY '"'
            LINES TERMINATED BY '\r\n';
```