# Structured Query Language (SQL)

It is used to Creat, Retrieve, Update, Delete Database

- There are two types of database
  (i) Relational (SQL) ✓ (for the Study)
  (ii) Non Relational (NoSQL)

To write SQL we need a special peice of Software known as Database Management System (DBMS)

- Database
  - to Creat a database

Syntax

```
CREAT DATABASE Databasename;
```

Ex

```
CREAT DATABASE myDB;
```

- To use the database

Syntax

```
USE databasename;
```

Ex

```
USE myDB;
```

- To delete a database

Syntax

```
DROP DATABASE databasename;
```

Ex

```
DROP DATABASE myDB;
```

- To set the database as a Read only mode

Syntax

```
ALTER DATABASE databasename READ ONLY=1;
```

if we want to disable Read only mode then the value will be "0".

we can't be able to do any operation on that database if it's in Read Only mode.

- Tables
  - To Creat a table

Syntax

```
CREAT TABLE tablename (
    Column_name1    datatype,
    Column_name2    datatype,
    :
    Column_namen    datatype
);
```

Ex

```
CREAT TABLE employees (
    employee_id     INT,
    first_name      VARCHAR(50),    -> no of character
    Last_name       VARCHAR(50),
    hourly_pay      DECIMAL(5,2)    -> no of digit, -> no of digit after point
    hire_date       DATE
);
```

- To select a table

Syntax

```
SELECT * FROM tablename;
```

Ex

```
SELECT * FROM employees;
```

- To rename table

Syntax

```
RENAME TABLE oldname TO newname;
```

Ex

```
RENAME TABLE employees TO workers;
```

- To Drop a table

Syntax

```
DROP TABLE tablename;
```

o To add a new Column

Syntax

□ ALTER TABLE tablename ADD
ADD Columnname datatype ;

Ex

ALTER TABLE employees
ADD Phone_number VARCHAR(15);

o To rename a Column

Syntax

□ ALTER TABLE tablename
RENAME COLUMN oldColumnname TO new name ;

Ex

ALTER TABLE employees
RENAME COLUMN Phone_number TO email;

☆ It doesn't change the datatype.

o To Change the datatype of a Column

Syntax

□ ALTER TABLE tablename
MODIFY COLUMN Column_name newdatatype;

Ex

ALTER TABLE employees
MODIFY COLUMN email VARCHAR(100);

⭐ in MariaDB for rename and modify we do (for me)

Syntax

□ ALTER TABLE tablename
CHANGE oldColumn_name newname Datatype;

o To Change the position of the Columns

Syntax

□ ALTER TABLE tablenme
MODIFY COLUMN Columnname datatype
(I want to reposition)
AFTER column name ;
(After which i wanna place)

Ex

ALTER TABLE employees
MODIFY COLUMN email VARCHAR(100)
AFTER Last_name ;

if I want to reposition a Column at the first we have to do just write "FIRST" at the place of "AFTER Columnname"

Ex

ALTER TABLE employee
MODIFY COLUMN Columnname
FIRST ;

o TO DROP A Column

Syntax

□ ALTER TABLE tablename
DROP COLUMN Columnname ;

Ex

ALTER TABLE employee
DROP COLUMN email ;

• Insert ROWS

o To insert a single Row

Syntax

□ INSERT INTO tablenme
VALUES ("values according to column) ;

Ex

INSERT INTO employees
VALUES (1, "Soumo", "Karmakar", 100.00, "2023-01-03");

DATE Syntax
Year-month-Date
bcz Varchar
" value "

o To insert multiple Rows

Syntax

o INSERT INTO tablename
VALUES ( ), ( ), ( ), ( );
Value for each Row

o If we want to insert value in some specific columns but not all

Syntax

INSERT INTO tablenme (Column1, Column2,)
VALUES ( );

Ex

INSERT INTO employees (employee_id, firstname, last_name)
VALUES (1, "Soumo", "Karmakar") ;

- Select
  - To select all rows and Column

Syntax

- SELECT * FROM tablename;

Ex

SELECT * FROM employees;

- To select specific column from all or specific rows.

Syntax

- SELECT Column_name1, Column_name2 FROM tablename;

Ex

SELECT first_name, last_name from employees;

- To select specific rows
  - SELECT * FROM tablename WHERE condition ;

Ex

SELECT * FROM employees WHERE employee_id = 1 ;

- if any column has null value then

Syntax

SELECT * FROM tablename where hire_date IS / IS NOT null;

(depends on query)

- UPDATE & DELETE
  - To Update any Column value

Syntax

- UPDATE tablename SET Columnname = value WHERE Condition;

Ex

UPDATE employees SET hire_date = '2023-01-07' where employee_id = 6

- To set Null value

Syntax

- UPDATE tablename SET Colname = null · where condition.

- To delete a specific row

Syntax

- DELETE FROM tablename where condition (for delete) ;

Ex

DELETE FROM employees
where employee_id = 6 ;

- AUTOCOMMIT, COMMIT, ROLLBACK
  - AUTOCOMMIT = ON (in default) for that it save everything all the time

    AUTOCOMMIT = OFF to off auto save
  - COMMIT ; to save manually
  - Rollback; to Undo any operation.
- CURRENT_DATE() & CURRENT_TIME()

: first Creat a table

```
CREAT TABLE test(
    mydate DATE,
    mytime TIME,
    mydatetime DATETIME
);
```

to

- CURRENT_DATE() //for current date
- CURRENT_TIME() //for current time
- NOW() // for current date time
- insert the value
  - Insert into test Values (CURRENT_DATE, CURRENT_TIME, NOW());

- Unique

it conform all values in a column are different.

- if we want to make any column Unique when we creat table.

Syntax

- CREAT TABLE tablename(

  Columnname1 datatype UNIQUE,

  Columnname2 datatype

  :

  );

- And if we do this after Creating a table

Syntax

- ALTER TABLE tablename

  ADD CONSTRAINT

  UNIQUE (Column_name);

By that we can't do duplicate at that Column

- Not NULL

- If we want to make a Column which value can't be NULL (At the time of table creation)

Syntax

- CREAT TABLE tablename(

  Columnname1 datatype,

  Columnname2 datatype NOT NULL,

  :

  );

- if we do that After table Creation

Syntax

- ALTER TABLE tablename

  MODIFY Columnname datatype NOT NULL;

  (Columnname: which column i wanna change)

after that we can't do that

Columnname = NULL;

- CHECK

It Check a specified Condition.

- if we want to set Check at the time of table Creation.

Syntax

- CREAT TABLE tablename(

  Col-name1 datatype,

  Col-name2 datatype,

  :

  CONSTRAINT Constraintname CHECK (condition on the Column)

  );

- if we do after creating a table

SYNTAX

- ALTER TABLE tablename

  ADD CONSTRAINT Constraintname CHECK

  (condition);

- If we want to delet check

Syntax

- ALTER TABLE tablename

  DROP CHECK Constraint name;

Syntax for me

- ALTER TABLE tablename

  DROP CONSTRAINT Constraintname;

- DEFAULT

It is used when someone is not giving value for a column so we set a default value at the place of NULL

- if we do it at the time of table Creation

Syntax

- CREAT TABLE tablename (
  Col-1 datatype,
  Col-2 datatype DEFAULT value
  :
  Col-n datatype
  );

- if we do it after table creation.

Syntax

- ALTER TABLE tablename
  ALTER Columnname SET DEFAULT value;
  (Columnname: on which I wanna set)

- Primary Keys

  if we use PRIMARY KEY Constraint on a column that's value will be both Unique and not null.

- if we do it at the time of table Creation

Syntax

- CREAT TABLE tablename (
  Col-1 datatype PRIMARY KEY,
  Col-2 datatype,
  :
  );

- if we do it after table creation

Syntax

- ALTER TABLE tablename
  ADD CONSTRAINT
  PRIMARY KEY (Column name);

Or me

- ALTER TABLE tablename
  ADD PRIMARY KEY (Column name);

- AUTO_INCREMENT

  Normally Primary Keys are auto incremented.

- if we set AUTO increment at the time of table creation

Syntax

- CREAT TABLE tablename (
  Col-1 datatype PRIMARY KEY AUTO_INCREMENT,
  Col-2 datatype,
  :
  );

- By default auto increment starts from 1

- To change auto increment starting value

Syntax

- ALTER TABLE tablename
  AUTO_INCREMENT = value;

- Foreign KEY

  when one table's primary key is present in another table's column. It Creat link between two tables.

- At the time of table Creation

Syntax

- CREAT TABLE tablename (
  Col-1 datatype,
  Col-2 datatype,
  :
  FOREIGN KEY (Col-2) REFERENCES othertable name where Col-2 is primary key
  (Col2)
  );

Ex

FOREIGN KEY (customer_id) References Customer(customer

- To drop a Foreign key

Syntax

- ALTER TABLE tablename
  DROP FOREIGN KEY (foreign key name);

- To give a nickname to foreign key

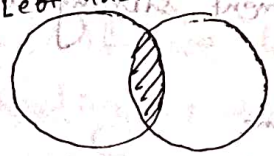
Syntax

- ALTER TABLE tablename
  ADD CONSTRAINT newname
  FOREIGN KEY (column-name) References
  otertablename (column-name);

- JOIN

It is used to combine rows from two or more table based on a related column

Inner Join




Syntax

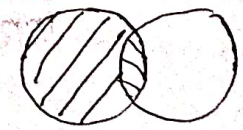
- SELECT * [redacted]
  FROM lefttable INNER JOIN right table
  ON lefttable.foreignkey = righttable.Primarykey

Ex

SELECT *
FROM transactions INNER JOIN Customers
ON transactions.Customerid = Customer.Customer-id;

It return Common between two Column.

LEFT JOIN

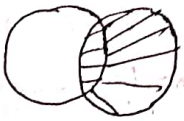


Syntax

- SELECT *
  FROM left table LEFT JOIN right table
  ON lefttable. foreign key = righttable. primary key;

It return Common between two tables and also all elements of Left table.

Right Join



- Syntax
  - SELECT *
    FROM Lefttable RIGHT JOIN right table
    ON lefttable. Foreignkey = Righttable. Primary key;

It will return the common elements and all elements of right table.

- Functions
  - Count()

    It Count no of rows [redacted] in a Column

Syntax

- SELECT COUNT ([redacted]col-name)
  FROM tablename;
  - MAX()

    It extract the max of a Column

Syntax

- SELECT MAX(Col-name)
  FROM tablename;
  - Min()

    it extract the min of a column

Syntax

- SELECT MIN(Col-name)
  FROM tablename;
  - AVG()

    it return the average of the Column data

Syntax

- SELECT AVG(Colname)
  FROM tablename;
  - SUM()

    it return the sum of the column data

Syntax

- SELECT SUM(column_name)
  FROM table name;

- CONCAT()

  It Concat to Column

  Syntax (to give a good name (alias))

  - SELECT CONCAT(col-1, col-2) AS newname From: tablename ;

- AND, OR, NOT

  - AND

    Syntax

    - SELECT * FROM table name Where Condition 1 AND condition 2;

      Here both condition have to be true.

  - OR

    Syntax

    - SELECT * FROM table name Where Condition1 OR Condition2;

      Here only one Condition need to be true.

  - NOT

    Syntax

    - SELECT * FROM tablename where NOT Condition

      it implies that the reverse condition is true.

  - Between

    Syntax

    - SELECT * FROM tablenmme where col-name Between value1 AND value2

      it return values Between value1 and value 2

  - IN

    Syntax

    - SELECT * FROM tablenmme where Columname IN (value1, value2, ...);

      Specified valued row will be displayed.

- Wild Card

  Wild card Charactens %, _

  Used to substitute one or more character in a string.

  - %

    'Char%. → Which start by that specified Character

    Ex

    - S%. → which start by S

    %'char' → which start by that specified characte

    Ex

    %'S → which end with S

    Syntax

    - SELECT * FROM tablename where Colname LIKE '%'char'/char'%'

    Ex

    SELECT * FROM tablenme WHERE Job LIKE "S%"

  - _

    _'Char' → which's second char is specified

    'char'_ → which's after has only one char.

    no of '_' indicate no of character present before or after specified charecte

    Syntax

    - SELECT * FROM tablenmme WHERE Col-name LIKE "__'Char'__"

    Ex

    - SELECT * FROM tablenmme WHERE Job LIKE "__a"

- Order By

  - ASCENDING

    Syntax

    - SELECT * FROM tablename ORDER BY Columnname ;

  - Descending

    Syntax

    - SELECT * FROM tablename ORDER BY Col-name DESC ;

for multiple Column

Syntax

- SELECT * FROM tablenme ORDER BY col-1 ASC, col-2 DESC, ... ;

- LIMIT
  - LIMIT Clause is used to limit the number of records.
  - Useful if you're working with a lot of data.
  - Can be used to display a large data on Pages (Pagination).

Syntax

- SELECT * FROM tablenme LIMIT n;

  It return 0 to n valued rows.

Syntax

- SELECT * FROM tablenme LIMIT m,n;

  It returns n no of element after m no of elements are discard from the start. m is offset no. It returns m+1 to m+n elements

- LIMIT mainly used with order by

Syntax

- SELECT * FROM tablename Order_By Col_nme LIMIT n;

- Union.

  Union Combine the result of two or more SELECT statements.

Syntax

- SELECT Col_1, col_2... FROM table1
  UNION
  SELECT Col_1, col_2... FROM table2;

* You have to select same no of column for UNION from table1 and table2. If we select all from each then you both table should have same no of Column.

- for Print duplicates

Syntax

SELECT * FROM table1
UNION ALL
SELECT * FROM table2;

- SELF JOIN
  - another copy of a table to itself
  - Used to compare rows of the same table
  - helps to display a heirarchy of data.

SYNTAX

- SELECT *
  FROM tablenme as a
  INNER JOIN (can be any join as per requirement) table name as b
  ON a.Col_1 = b.Col_2 ;

Ex

SELECT a.first_name, a.last_name
CONCAT(b.firstname, " ", b.lastname) as "referred"
FROM Customers as a
INNER JOIN
Customer as Bb
ON a.referralId = b.Customer_id;

- Views
  - a virtual table based on the result-set of an SQL Statement.
  - The fields in a view are fields from one or more real tables in the database
  - They're not real tables, but can be interacted with if they were

Syntax

- CREAT VIEW viewname AS
  SELECT * FROM tablename ;

ex

CREAT VIEW employee_attendence AS
(SELECT Firstname, last-name
FROM employees.

to drop a view

Syntax

- DROP VIEW viewname ;

- INDEXES
  - it is BTree data structure.
  - used to find value within a specific column more quickly
  - MySQL normally searches sequentially through a column.
  - The longer the column, the more expensive the operation is.
  - UPDATE takes more time, SELECT takes less time.

- To know How many Indexes are Present

Syntax

- SHOW INDEX FROM tablename

- To Creat an Index

Syntax

- CREAT INDEX indexname
  ON tablename (Columns of the table)

- To DROP an Index

Syntax

- ALTER TABLE tablename
  DROP INDEX indexname

- Subqueries.
  - a query within another query
  - query (subquery)

Ex

SELECT first_name, last_name FROM
Customers where customer_id IN
(SELECT customer_id From transaction where
customer_id IS NOT null);

- Group By

aggregate all rows by a specific column often used with aggregate functions
Ex: SUM(), MAX(), MIN(), AVG(), COUNT()

Syntax

- SELECT SUM(col_name) FROM table name (it embe max, min, avg).
  GROUP BY Columnname
  → According to which the operation will be perform

Ex

SELECT sum(amount), customerid FROM transactions
GROUP BY customer_id;

we can't use where clause with groupby so if any condition we have to add

Syntax

- SELECT SUM(col_name) FROM tablename
  GROUP BY Columnname
  HAVING condition;

- ROLLUP
  - extension of GROUP BY Clause
  - Produces another row and shows the GRAND TOTAL (super-aggregate value)

Syntax

- SELECT COUNT(col_name) FROM tablename (it embe any thing)
  GROUP BY Columnname WITH ROLLUP ;

it returns total.

- ON DELETE
  - ON DELETE SET NULL = When a PFK is deleted, replace FK with NULL
  - ON DELETE CASCADE = When a FK is deleted, delete row.

- if i want to do it at the time of table Creation

```
CREAT TABLE tablenmme (
    Col-1 datatype,
    Col-2 datatype,
    ;
    Foreign key (col-2) References other table
                                    (col_nme),
    ON DELETE SET NULL / ON DELETE
                        CASCADE
);
```

- alter table Creation

Syntax

```
ALTER TABLE table nme
ADD CONSTRAINT foreignkey nme
Foreign KEY (col-1) Referene other table(col-2)
ON DELETE SET NULL/ ON DELETE CASCADE;
```