|  |  |
| --- | --- |
| DDL | DML |
| Creating DB | Insert (Copy) |
| Creating Table | Read |
| Altering Table | Update |
| Adding Columns to existing Table | Delete |
| Drop db |  |
| Drop table |  |

SQL [DB] [Sequential Query Language]

**Data Types:**

|  |  |
| --- | --- |
| Data | Data Type in SQL |
| Text (If we know the size of the text) | CHAR (size) |
| Text (If it is not fixed size) | VARCHAR(maximum size=65535) |
| Text (in the form of bytes for fixed size text) | BINARY(size) |
| Text (in the form of bytes for variable size text) | VARBINARY(size) |
|  | TINYBLOB (255 Bytes) |
|  | TINYTEXT (255 characters) |
|  | TEXT (65535 bytes) |
|  | BLOB (65535 Bytes) |
|  | BIT (1-64) |
|  | TINYINT (-128 to 127 or 0-255) |
|  | INTEGER (INT (size)) |
|  | BIGINT (Large Integers) |
|  | FLOAT (size, d) |
|  | DOUBLe(size, d) |
|  | DECIMAL(size, d) |
|  | BOOL (0-false, anyother number-true) |
|  | BOOLEAN (TRUE/FALSE) |

**Query:**

What is query?

* Anything that we ask from DB Table

Types of Queries:

* DDL [Data Definition Language]
* Defining the structure of DB or Table or Columns
* DML [Data Manipulation Language]
* Any query that deals with data present in DB or Table or Column

**Structure of DB:**

* SQL Instance
* DB
* Table
* Column
* Column
* Table
* Column
* Column
* DB
* Table
* Column
* Column
* Table
* Column
* Column

**DDL:**

* Database:
  + create database MYDB;
  + show databases;
  + use mydb;
  + drop database mydb;
* Table:
  + create table student(id int, name varchar(50), marks int, report varchar(1));
  + show tables;
  + describe student;
  + alter table student add column grade varchar(1);
  + drop table student;

**DML:**

* Insertion/Copy -------C
* Read/FETCH -------R
* Updation/Modification----U
* Delete ------------------D

**C:**

* insert into student(id,name,marks,report) values (1,A,10,P);

or

* insert into student values(1,'A',10,'P');

**R:**

* select \* from student;
* select name,marks from student;

**U:**

* update student set marks=11;
* update student set marks=12, name='ANAND';

**D:**

* delete from student;

**Aggregation Operations:**

* select count(\*) from student; -------- count
* select min(marks) from student; ------ min
* select max(marks) from student; ------ max
* select avg(marks) from student; ----- avg
* select sum(marks) from student; ----- sum
* select \* from student limit 1; ------ row limitation for viewing

**Filtering Operations: [where clause --- condition clause]**

*Filtering on rows of the table as per conditioning column*

* select \* from student where report='P';
* select \* from student where report in ('p','f');
* select \* from student where report='p' and name='A' or marks=10;
* select \* from student where report='p' or name='A';

**Ordering Operations:**

* select \* from student order by marks asc; --ascending order
* select \* from student order by marks desc; --descending order

**Group By Operations:**

*Group By clause would be applied when we are grouping the rows based on columns and it will return always an aggregated output*

* select report, count(\*) from student group by report;
* select report , count(\*) as cnt from student group by report, marks;
* select report , count(\*) as cnt from student where report in ('p','f') group by report, marks;
* select report , count(\*) as cnt from student where report in ('p','f') group by report, marks order by cnt desc;

**Having Operations:**

*Having clause is applied only when group by clause is used in the query and also when filtering needs to be applied on aggregated output*

* select report , count(\*) as cnt from student where report in ('p','f') group by report, marks having cnt> 5 order by cnt desc;

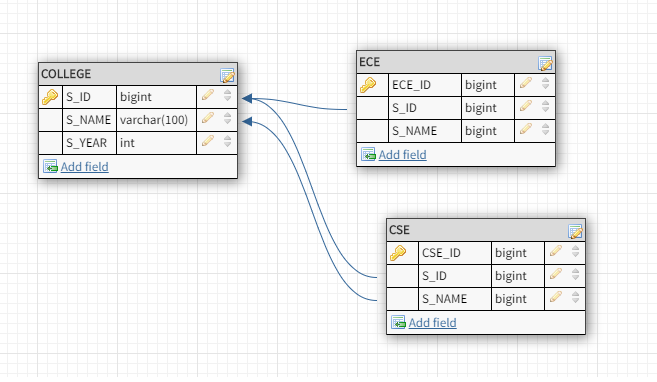
**Distinct Function:** *This function is used when you want to check the distinct values for a given column. It would not result aggregated function as output*

Select distinct(report) from student;

**Indexing:**

* Primary Key
* The PK is used to store and identify the every record in a table as unique
* One can assign only one column as PK
* One can give any number of columns as FK [Foriegn Key]
* Secondary Key / Foreign Key

PK in master table becomes FK in child table to ensure **uniqueness of the record in child** table and also **reference to master** table



**Inner Queries:**

SELECT \* FROM (SELECT report AS rp,COUNT(\*) AS cnt FROM student WHERE STATUS IN ('g','b') GROUP BY report HAVING cnt>1 ORDER BY cnt) A WHERE A.rp IN ('r','n') ORDER BY A.cnt DESC;

**Stored Procedures:**