RDBMS Concepts and MySQL Commands

Database Languages

Database provides following facilities or languages for working.

1. **DDL (Data Definition Language)**
2. **DML (Data Manipulation Language)**
3. **TCL (Transition Control Language)**
4. **DCL (Data Control Language)**
5. **SCL (Session Control Language)**

DDL (Data Definition Language)

DDL language provides the various commands those works on the structure of database. It includes the following commands.

1. **CREATE TABLE**
2. **ALTER TABLE**
3. **MODIFY**
4. **DROP TABLE**

DML (Data Manipulation Language)

DML language provides the various commands those works on the data stored in the database. It includes the following commands.

1. **INSERT INTO**
2. **SELECT FROM**
3. **UPDATE**
4. **DELETE**

DDL Commands

1. **Create Database:**
   ```
   CREATE DATABASE school;
   ```
2. **Show Databases or Tables :**
   ```
   SHOW DATABASES;
   ```
3. **Show Tables :**
   ```
   SHOW TABLES
   ```
3. **Change Database:**
   ```
   USE student;
   ```

Create Table:

```
CREATE TABLE student
(
    Rollno INT primary key,
    Name CHAR(20) NOT NULL,
    Class VARCHAR(20),
    City VARCHAR(20),
    Mobile INT,
    Marks FLOAT
);
```

ALTER Table:

* Add New Attribute “Age” in student table.

```
ALTER TABLE student
ADD Age INT;
```

* Delete existing Attribute “Mobile” from student table.

```
ALTER TABLE student
DROP COLUMN Mobile;
```

*** Change size of Attribute City(20) to City(30).**

```
ALTER TABLE student
MODIFY City VARCHAR(30);
```

Exercise:

- **Change Datatype of any Attribute**
- **Add NOT NULL constraint to any Attribute**
- **Delete constraint from any Attribute.**
- **Change Datatype of Attribute**

```
ALTER TABLE student
MODIFY Name VARCHAR(30);
```

- **Add NOT NULL constraint to Attribute**

```
ALTER TABLE student
MODIFY Rollno INT NOT NULL;
```

- **Delete constraint from Attribute.**

```
ALTER TABLE student
DROP COLUMN Age;
```

Special:

Add Primary Key constraint to Rollno Attribute.

```
ALTER TABLE student
ADD PRIMARY KEY (Rollno);
```

Remove Primary Key constraint from Rollno Attribute.

```
ALTER TABLE student
DROP PRIMARY KEY
```

- **Remove the whole Table (structure) from database.**

```
DROP TABLE student;
```

(The table itself and all the records stored in it will be deleted permanently from the database and it will not be recovered.

DML Commands

INSERT INTO: Insert a New Record into Table

INSERT INTO student
VALUES(01, 'Aman', 'XI A', 'Jhunjhunu', 16);

INSERT INTO student(Name, Age, Rollno, City)
VALUES('Kamal', 15,02, 'Jhunjhunu');

Rule:

*** The Sequence of data values must be as the sequence of attribute in table.**

*** The Numeric Values should be without any quotation mark where the character, string and date values must be enclosed between single quotation marks.**

SELECT FROM: Select data from Table

1. SELECT * FROM student;

Rollno	Name	Class	City	Age
01	Aman	XI A	Jhunjhunu	16
02	Kamal		Jhunjhunu	15

SELECT FROM: Select data from Table

```
SELECT * FROM student
WHERE age=15;
```

Rollno	Name	Class	City	Age
02	Kamal		Jhunjhunu	15

Here * denote to all attributes. If we want to display only name and city of student then write the query as…

UPDATE: Change data of Table

Query: Change the class of all students as 'XI B'

```
UPDATE student
SET Class='XI B' ;
```

To display the changes write….

```
SELECT * FROM student;
```

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16
02	Kamal	XI B	Jhunjhunu	15

UPDATE: Change data of Table

Query: Change the class of student Kamal as 'XI C'

```
UPDATE student
SET Class='XI C' ;
WHERE Name='Kamal' ;
```

To display the changes….

```
SELECT * FROM student;
```

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16
02	Kamal	XI C	Jhunjhunu	15

Various SQL Commands / Operators

WHERE clause: used to add any criteria in query.

UPDATE student
SET Class='XI C'
WHERE Name='Kamal' ;

OR

SELECT * FROM student
WHERE Age>15 ;

DISTINCT: used to display data without repetition.

SELECT DISTINCT(City) FROM student

OUTPUT: **DISTINCT(City)**

Jhunjhunu

Look at the output of following query:

SELECT City FROM student

City

Jhunjhunu
Jhunjhunu

LIKE clause: Pattern Matching in criteria. Two wild cards are used for pattern matching.

1. **%** : It represent to any number of characters.
2. **_** : It represent to only one character.

SELECT * FROM student
WHERE Name LIKE '%n;

It will show the records whose name ends with 'n' alphabet.

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16

ORDER BY clause: Display the records either in Ascending order or Descending Order.

1. ASC : It used for Ascending Order
2. DESC: It used for Descending Order

(Note: By default order is ascending.)

```
SELECT * FROM student
ORDER BY Name Desc;
```

Rollno	Name	Class	City	Age
02	Kamal	XI C	Jhunjhunu	15
01	Aman	XI B	Jhunjhunu	16

IS: Used to Search NULL Values only.

```
SELECT * FROM student
WHERE City IS NULL ;
```

It will show only those records whose city values are NULL.

IS NOT: Used to Search other than NULL Values only.

```
SELECT * FROM student
WHERE City IS NOT NULL ;
```

It will show records which values of city are other than NULL.

IN: Search in multiple strings..

```
SELECT * FROM student
WHERE City IN ('Jhunjhunu', 'Jaipur', 'Ajmer');
```

IN operator works similar to Logical OR operator.

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16
02	Kamal	XI C	Jhunjhunu	15

GROUP BY Clause:

It will display the records in groups with similar type of values.

```
SELECT * FROM student
GROUP BY City;
```

It will show records in groups based on city.

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16
02	Kamal	XI C	Jhunjhunu	15

HAVING Clause: It used to include a condition with GROUP BY Clause. (Where command is not allowed with group)

```
SELECT * FROM student
GROUP BY City
HAVING Age>14;
```

It will show records in groups based on city and age should greater than 14.

Rollno	Name	Class	City	Age
01	Aman	XI B	Jhunjhunu	16
02	Kamal	XI C	Jhunjhunu	15

MySQL Functions

(A) Mathematical Functions

AVG() Function

- **The AVG() Function**
  The AVG() function returns the average value of a numeric column.
- **SQL AVG() Syntax**
  SELECT AVG(column_name) FROM table_name;
  - **SELECT AVG(OrderPrice) AS OrderAverage FROM Orders;**

ABS() FUNCTION

- **The ABS() function returns the Absolute value (Positive value) in the selected column or given numeric value.**
- **SQL ABS() Syntax**

SELECT ABS(column_name) FROM table_name;

- **SELECT MAX(OrderPrice) AS LargestOrderPrice FROM Orders;**

Example: ABS(-25.32) Output: 25.32

ABS(-25) output: 25 ABS(-25/3.0) output: 8.333

Count() function

- **SELECT COUNT(*) FROM table_name;**

(COUNT(*): NULL values in some of columns will be counted as row number)

- **The COUNT(column_name) function returns the number of values other than NULL.**

(COUNT(column_name) : NULL values columns will not count)

MAX() Function

- The MAX() function returns the largest value in the selected column.
- SQL MAX() Syntax

SELECT MAX(column_name) FROM table_name;

- SELECT MAX(OrderPrice) AS LargestOrderPrice FROM Orders;

MIN() Function

- The MIN() function returns the smallest value of the selected column.
- SQL MIN() Syntax

SELECT MIN(column_name) FROM table_name;

- SELECT MIN(OrderPrice) AS SmallestOrderPrice FROM Orders;

SUM() Function

- The SUM() function returns the total sum of a numeric column.
- SQL SUM() Syntax

SELECT SUM(column_name) FROM table_name;

- SELECT SUM(OrderPrice) AS OrderTotal FROM Orders;

SQRT() FUNCTION

- The SQRT() function returns the square root of given POSITIVE number.
- SQL SQRT() Syntax

SELECT SQRT (column_name) FROM table_name;

Example:

SQRT(25)	OUTPUT: 5
SQRT(-25)	OUTPUT: NULL
SQRT(25.16)	OUTPUT: 5.0159

ROUND() Function

- The ROUND() function returns the nearest round off number of the given number as per arguments.
- SQL round() Syntax

SELECT ROUND(column_name) FROM table_name;

Example:

ROUND(25.238,1)	OUTPUT: 25.2
ROUND(25.238,2)	OUTPUT: 25.24
ROUND(25.238,0)	OUTPUT: 25
ROUND(25.238,-1)	OUTPUT: 30
ROUND(151.238,-1)	OUTPUT: 150
ROUND(151.238,-2)	OUTPUT: 200

TRUNCATE() FUNCTION:

- The TRUNCATE() function returns the number after removing the part of number as per arguments.
- SQL TRUNCATE () Syntax

SELECT TRUNCATE (column_name) FROM table_name;

Example:

TRUNCATE(25.238,1)	OUTPUT: 25.2
TRUNCATE(25.238,2)	OUTPUT: 25.23
TRUNCATE(25.238,0)	OUTPUT: 25
TRUNCATE(25.238,-1)	OUTPUT: 25
TRUNCATE(151.238,-1)	OUTPUT: 150
TRUNCATE(151.238,-2)	OUTPUT: 100

(B) String Functions

UCASE() / UPPER() Function

- The UCASE() function converts the value of a field to uppercase.
- SQL UCASE() Syntax

SELECT UCASE(column_name) FROM table_name;

- SELECT UCASE(LastName) as LastName,FirstName FROM Persons;

LCASE() / LOWER() Function

- The LCASE() function converts the value of a field to lowercase.
- SQL LCASE() Syntax

SELECT LCASE(column_name) FROM table_name;

- SELECT LCASE(LastName) as LastName,FirstName FROM Persons;

LENGTH() Function

- The LENGTH() function returns the length of the value in a text field.
- Length of string includes blank spaces also.
- SQL LENGTH() Syntax

SELECT LENGTH(column_name) FROM table_name;

- SQL LENGTH() Example

SELECT LENGTH(Address) as LengthOfAddress FROM Persons;

Example:

LENGTH("KVS RO JPR") Output: 10

LENGTH("KVS R.O. JPR!") Output: 13

CONCAT( ) Function

It takes two string arguments and returns a string after concatenating both strings.

CONCAT(“KVS”,”JHUNJHUNU”)

Output: KVSJHUNJHUNU

CONCAT(“KVS”, “ “, ”JHUNJHUNU”)

Output: KVS JHUNJHUNU

SUBSTR() FUNCTION

It returns a sub string that get from main string as per given locations.

Syntax:

SUBSTR(main string, start, length)

Where main string is given string from which sub string to be find.

Start is initial number of character in main string from where sub string started.

Length is total number of characters to be in sub string.

SUBSTR(“rajasthan”,3,6)

Output: jastha

SUBSTR(”JHUNJHUNU”,5)

OUTPUT: JHUNU

Note: If length is not given then by default length will be upto end of string.

INSTR() FUNCTION

This function get two string arguments as main string and sub string. It finds the sub string in main string and returns its initial location as number.

If sub string not finds in main string then it returns 0 (zero).

SYNTAX: INSTR(main string, sub string)

Example:

INSTR("Rajasthan","THAN") Output: 6

INSTR("Rajasthan","khan") Output: 0

TRIM() Function:

This function will remove the blank space character from leading and trailing side of string.

LTRIM() Function:

This function will remove the blank space character from leading side of string.

RTRIM() Function:

This function will remove the blank space character from trailing side of string.

Examples:

SELECT (" RAJASTHAN ") OUTPUT: __RAJASTHAN__

RTRIM(" RAJASTHAN ") OUTPUT: __RAJASTHAN

LTRIM(" RAJASTHAN ") OUTPUT: RAJASTHAN__

TRIM(" RAJASTHAN ") OUTPUT: RAJASTHAN

:: Finished ::