1. Write about SQL and its features along with classification of SQL Command SET.

SQL or Structured Query Language is the language that we use to work with the relational database management systems. SQL is a language that communicates with databases.

Features of SQL

1. Data Definition Language (DDL)

→ changes the structure of table like creating, deleting, altering a table.

→ Commands under DDL

a. CREATE

Syntax: CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES(..));

Eg: CREATE TABLE EMPLOYEE (Name VARCHAR2(20), Email VARCHAR2(100), DOB DATE);

b. DROP

Syntax: DROP TABLE table_name;

Eg: DROP TABLE EMPLOYEE;

c. ALTER

Syntax: ALTER TABLE table_name ADD Column_name COLUMN-def;

Eg: ALTER TABLE STUDETAILS ADD(ADDRESS VARCHAR2(20));

d. TRUNCATE

Syntax: TRUNCATE TABLE table_name;

Eg: TRUNCATE TABLE EMPLOYEE;

2. Data Manipulation Language (DML)

→ used to modify the database. It is responsible for all form of changes in database.

→ Commands under DML

a. INSERT

Syntax: INSERT INTO TABLE_NAME
VALUES (Value1, Value2, Value3,.... ValueN);

Eg: INSERT INTO javatpoint (Author, subject) VALUES ("sonoo", "DBMS");

b. UPDATE

Syntax: UPDATE table_name SET[Column_name1 = Value1, ... Column_nameN
= ValueN] [WHERE CONDITION]

Eg: UPDATE Students
SET User_Name = 'Sonoo'
WHERE Student_Id = '3'

C. DELETE

Syntax: DELETE FROM table_name [WHERE Condition];

Eg: DELETE FROM javatpoint
WHERE Author = 'sonoo';

3. Data Control Language (DCL)

→ used to grant & take back authority from any database user

→ Commands under DCL

a. GRANT

Eg: GRANT SELET, UPDATE ON MY_TABLE TO SOME_USER,
ANOTHER_USER;

b. REVOKE

Eg: REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;

4. Transaction Contrl Language (TCL)

→ can only use with DML Commands like INSERT, DELETE & UPDATE only.

→ Commands under TCL

a. COMMIT

Syntax: COMMIT;

Eg, DELETE FROM CUSTOMERS
WHERE AGE=25;
COMMIT;

b. ROLLBACK

Syntax: ROLLBACK;

Eg, DELETE FROM CUSTOMERS
WHERE AGE =25;
ROLLBACK;

C. SAVEPOINT

Syntax: SAVEPOINT SAVEPOINT_NAME;

5. Data Query Language (DQL)

→ used to fetch the data from database

→ Command under DQL

a. SELECT

Syntax: SELECT expressions
FROM TABLES
WHERE Conditions;

Eg, SELECT emp_name
FROM employee
WHERE age > 20;

2. Create tables for the following:

a) Train (train Number, name, source, destination, start_time, reach_time, traveltime, distance, class, days, type)

Syntax:

```
create database dbmslab;
use dbmslab;
create table Train(trainno INT(6) PRIMARY KEY, name VARCHAR(20),
source VARCHAR(20), destination VARCHAR(20), start_time
DATETIME, reach_time DATETIME, traveltime TIME, distance
FLOAT (6,2), class VARCHAR(10), days INT(2), type VARCHAR(5));
```

Output:

```
Mysql> desc train;
```

Field	Type	Null	Key	Default	Extra
trainno	int	NO	PRI	NULL	
name	varchar(20)	YES		NULL	
Source	varchar(20)	YES		NULL	
destination	varchar(20)	YES		NULL	
Start_time	datetime	YES		NULL	
reach_time	datetime	YES		NULL	
traveltime	time	YES		NULL	
distance	float(6,2)	YES		NULL	
class	varchar(10)	yes		NULL	
days	int	YES		NULL	
type	varchar	yes		NULL	

6) Ticket (PNRNO, Transactionid, from_station, To_station, date-of_journey, class date-of_booking, total_ticket_fare, train number)

Syntax:

```
Create table Ticket (PNRNO INT(10) PRIMARY KEY, transactionid INT(10),
from_station VARCHAR(20), to_station VARCHAR(20), date-of_
journey DATETIME, class VARCHAR(10), date_of_booking DATETIME,
total_ticket_fare INT(5), trainno INT(6));
```

Output:

```
mysql> desc ticket;
```

Field	Type	NULL	Key	Default	Extra
PNRNO	int	NO	PRI	NULL	
transactionid	int	YES		NULL	
from_station	varchar(20)	YES		NULL	
to_station	varchar(20)	YES		NULL	
date_of_journey	datetime	YES		NULL	
class	varchar(10)	YES		NULL	
date_of_booking	datetime	YES		NULL	
total_ticket_fare	int	YES		NULL	
trainno	int	YES		NULL	

c) Passenger (PNR No, Serial no, Name, Age, Reservation_status)

Syntax:

```
Create table Passenger (PNRNo INT(10) Primary key, Serialno INT(10),
Name VARCHAR(20), Age INT(3), Reservation_status VARCHAR(10));
```

Output:

```
mysql> desc Passenger;
```

Field	Type	NULL	KEY	Default	Extra
PNRNO	int	NO	PRI	NULL	
SerialNO	int	YES		NULL	
Name	varchar(20)	YES		NULL	
Age	int	YES		NULL	
Reservation_Status	varchar(10)	YES		NULL	

d) Train_Route (Train_NO, route_no, station_code, name, arrival_time, depart_time, distance, day)

Syntax:

create table Train_Route (trainno INT(6) Primary key, route_no INT(6), station_code VARCHAR(5), name VARCHAR(20), arrival_time TIME, depart_time TIME, distance FLOAT(6,2), day INT(2));

Output:

mysql> desc Train_Route;

Field	Type	NULL	key	default	Extra
trainno	int	NO	PRI	NULL	
route_no	int	YES		NULL	
station_code	varchar(5)	YES		NULL	
name	varchar(20)	YES		NULL	
arrival_time	time	YES		NULL	
depart_time	time	YES		NULL	
distance	float(6,2)	YES		NULL	
day	int	YES		NULL	

c) Train_Ticket_fare (Train_No, class, base_fare, reservation_charge, Superfast_charge, other_charge, tatkal_charge, service_tax)

Syntax:

Create table Train_Ticket_fare(trainno INT(6) primary key, class VARCHAR(10), base_fare INT(4), reservation_charge INT(4), superfast_charge INT(4), other_charge INT(4), tatkal_charge INT(4), Service_tax INT(4);

Output:

mysql>desc Train_Ticket

field	Type	Null	key	Default	Extra
trainno	int	NO	PRI	NULL	
class	Varchar(10)	YES		NULL	
base_fare	int	YES		NULL	
reservation_charge	int	YES		NULL	
superfast_charge	int	YES		NULL	
other_charge	int	YES		NULL	
tatkal_charge	int	YES		NULL	
Service_tax	int	YES		NULL	

3. Write Simple DDL / DML Queries to

a) List the trains details within a range of numbers

Syntax:

Select * from train where trainno between 123400 & 123450;

Output:

mysql > Select * from train where trainno between 123400 and 123450;

trainno	name	Source	destination	Start_time	reach_time	travel time	distance	Class	days	type
123422	rajdhani	Simla	Chennai	2022-05-18 10:00:00	2022-05-28 10:00:00	240:00:00	999.00	chair car	10	3tier
123422	ayodhya	up	AP	2022-05-20 12:00:00	2022-06-20 12:00:00	720:00:00	1499.00	general	1	2tier

b) change the Superjast charge value in train fare as zero, if it is null.

Syntax:

Update train_ticket_fare set Superfast_charge = 0 where superfast_charge is NULL;

Output:

mysql> Update train_ticket_fare set superfast_charge = 0 where superfast_charge = 200;

Query ok, 1 row affected

Rows matched: 1 Changed: 1 Warnings: 0

) List the passenger names whose tickets are not confirmed.

Syntax:

Select name from passenger where reservation_status = "Pending";

Output:

mysql> Select name from passenger where reservation_status = "Pending";

name
mohan

1 row in set.

d) List the base_fare of all AC coaches available in each train

Syntax:

Select BASE_FARE from Train_Ticket_fare where CLASS="chair car";

Output:

mysql> Select BASE_FARE from Train_Ticket_fare where CLASS = "sleeper";

BASE_FARE
2200

1 row in set

mysql> select BASE_FARE from Train_Ticket_fare where CLASS = "chair car";

BASE_FARE
2000

1 row in set

c) find the train names that are from Secunderabad to Mumbai, but do not have the source or destination in its name.

Syntax:

```
Select name from Train where SOURCE = 'SECUNDERABAD'
AND DESTINATION = 'MUMBAI' AND NAME != '%SECUNDERABAD%
MUMBAI%' AND NAME != '%MUMBAI%SECUNDERABAD%';
```

Output:

```
mysql> Select name from Train where SOURCE = 'new delhi' AND
DESTINATION = 'dawoai' AND NAME != '%SECUNDERABAD%MUMBAI%'
AND NAME != '%SECUNDERABAD%MUMBAI%SECUNDERABAD%';
```

name
Konark express

1 row in set

4. Use SQL PLUS functions to the following

a) print the passenger names with left padding character.

Syntax:

```
SELECT LPAD(Name, 10, '***') AS LeftPadName from passenger;
```

Output:

LeftPadName
***ram
***sam
***mohan

3 rows in set

b) Print the station codes replacing K with H

Syntax:

```
Select replace(station_code, 'K', 'H') from Train_Route;
```

Output:

replace(station_code,'K','H')
TIR42
TIO42
HID42

3 rows in set.

c) find the passengers whose date of journey is one month from today

Syntax:

```
Select date_of_journey from ticket where date_of_journey >
    date_add(now(), interval 30 day);
```

Output:

date_of_journey
2022-05-20 12:00:00

1 row in set

d) Find the average age of passengers in one ticket

Syntax:

```
Select avg(age) from Passenger;
```

Output:

```
mysql> Select avg(age) from Passenger;
```

avg(age)
43.0000

e) Find the train names that are from Secunderabad to Mumbai, but do not have the source or destination in its name.

Task 7

Q. Write a simple PL/SQL block to

1. print the factorial of a given number

Queries:

```
DELIMITER //
CREATE PROCEDURE fact (IN x INT)
BEGIN
DECLARE result INT;
DECLARE i INT;
SET result = 1;
SET i = 1;
WHILE i<= x DO
SET result = result * i;
SET i = i+1;
END WHILE;
SELECT x AS Number, result as factorial;
END //
```

2. Print the fibonacci series

Queries:

```
DELIMITER//
CREATE PROCEDURE nonsec_fib(n INT, OUT out_fib INT)
BEGIN
DECLARE m INT default 0;
DECLARE k INT default 1;
DECLARE i INT;
DECLARE tmp INT;
Set m=0;
Set k=1;
Set i=1;
WHILE (i<=n) DO
   SET tmp=m+K;
   SET m=k;
   SET k=tmp;
   SET i=i+1;
END WHILE;
SET out_fib=m;
END //
```

Output:

Call fact(5)

//

Number	Factorial
5	120

Ex:9

Q. Write a cursor for the following: Declare a cursor that defines a result set. Fetch the data into local variables as needed from the cursor, one row at a time. Close the cursor when done.

Example Queries

```
CREATE TABLE Sailors (sid INT, sname VARCHAR(20), rating INT,
age FLOAT, PRIMARY KEY (sid));
INSERT INTO sailors VALUES (22,'Dustin',7,45);
INSERT INTO sailors VALUES (29,'Brutus',1,33);
INSERT INTO sailors VALUES (31,'Lubber',8,56);
INSERT INTO sailors VALUES(32,'Andy',8,26);
INSERT INTO sailors VALUES(58,'Rusty',10,35);
INSERT INTO sailors VALUES(74,'Horatio',9,35);
INSERT INTO sailors VALUES(64,'Horatio',7,35);
INSERT INTO sailors VALUES(95,'Bobo',3,64);
INSERT INTO sailors VALUES(85,'Art',3,26);
INSERT INTO sailors VALUES (71,'Zorba',10,16);

DELIMITER //
create procedure mycur1(sa_id int)
begin
declare v_sid int;
declare v_sname varchar(30);
declare v_rating int;
declare v_age int;
declare c1 cursor for select sname, rating, age from sailors where
    sid = sa_id;
open c1;
fetch c1 into v_sid, v_sname, v_rating, v_age;
Select v_sid, v_sname, v_rating, v_age;
close c1;
end//
```

Output

Call mycur1(69) //

V_sname	V_rating	V_age
Brutus	1	33

Example 2:

Queries:

```
DELIMITER //
create procedure mycur2(sa_rating int)
begin
declare V_Sname varchar(30);
declare v_sid int;
declare V_age int;
declare V_rating int;
declare c1 cursor for select sid, sname, age from sailors where
                rating = sa_rating;
Openc1;
fetch c1 into V_sid, V_sname, V_age;
Select V_sid, V_sname, V_age;
Close c1;
end//
```

Output

Call mycur2(3) //

V_sid	V_Sname	V_age
85	Art	26

Example 3

Queries

```
DELIMITER //
create procedure mycur3 (sa_rating int)
begin
declare finished int default 0;
declare Count int default 0;
declare V_Sname varchar(30);
declare V_Sid int;
declare V_age int;
declare C1 cursor for select sid, sname, age from sailors where
                      rating = sa_rating;
declare Continue handler for not found to set finished=1;
openc1;
getcur : loop
fech C1 into V_Sid, V_Sname, V_age;
if finished = 1 then
leave getcur;
end if;
Set Count = Count + 1;
Select V_Sid, V_Sname, V_age;
end loop;
Close C1;
Select Count;
end //
```

Output

Call myaw13(1) //

v_sid	v_sname	v_age
29	Boutus	33

Count
1

Task 10

Q. Write a PL/SQL procedure to: Creation of stored procedure, Execution of procedure and modification of procedure.

Queries:

```
CREATE TABLE Sailors (sid INT, sname VARCHAR(10), rating INT,
 age FLOAT, PRIMARY KEY (sid));
INSERT INTO Sailors VALUES (22,'Dustin',7,45);
INSERT INTO Sailors VALUES (29,'Brutus',1,33);
INSERT INTO Sailors VALUES (31,'Lubber',8,56);
INSERT INTO Sailors VALUES (32,'Andy',8,26);
INSERT INTO Sailors VALUES (58,'Rusty',10,35);
INSERT INTO Sailors VALUES (74,'Horatio',9,35);
INSERT INTO Sailors VALUES (64,'Horatio',7,35);
INSERT INTO Sailors VALUES (95,'Bob',3,64);
INSERT INTO Sailors VALUES (85,'Art',3,26);
INSERT INTO Sailors VALUES (71,'Zorba',10,16);

DELIMITER //
create procedure p1 (P_age int)
begin
SELECT S.rating, S.age
FROM Sailors S
WHERE S.age>= P_age;
end
//
```

Output

Call P1(30) //

rating	age
7	45
1	33
8	56
7	35
9	35
3	64

Task 11

11. write a trigger for the following
Creation of insert trigger, delete trigger, Update trigger

Update Trigger:

Queries:

```
CREATE TABLE Boats(bid INT, bname VARCHAR(10), Color VARCHAR(10),
PRIMARY KEY(bid));

DESC Boats;
INSERT INTO Boats VALUES(101,'interlake','blue');
INSERT INTO Boats VALUES(102,'interlake','red');
INSERT INTO Boats VALUES(103,'clipper','green');
INSERT INTO Boats VALUES(104,'Marine','red');

DELIMITER //
create trigger t1 before update on boats
for each row
begin
if new.color='red' then
set new.color=old.color;
else
set new.color=new.color;
end if;
end//
```

Output

Select * from boats //

bid	bname	Color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

Insert Trigger

Queries

```
CREATE TABLE sailors(sid INT, sname VARCHAR(20), rating INT,
age FLOAT, PRIMARY KEY(sid));
INSERT INTO Sailors VALUES (22,'Dustin',7,45);
INSERT INTO Sailors VALUES(29,'Brutus',1,33);
INSERT INTO Sailors VALUES(31,'Lubber',8,56);
INSERT INTO Sailors VALUES(32,'Andy',8,26);
INSERT INTO sailors VALUES(58,'Rusty',10,35);
INSERT INTO Sailors VALUES(74,'Horatio',9,35);
INSERT INTO Sailors VALUES(64,'Horatio',7,35);
INSERT INTO Sailors VALUES(95,'Bob',3,64);
INSERT INTO Sailors VALUES(85,'Art',3,26);
INSERT INTO Sailors VALUES(71,'Zorba',10,16);
DELIMITER //
Create trigger t2
before insert on Sailors
for each row
begin
if new.age>40 then
Set new.rating='10';
Else
Set new.rating=new.rating;
end if;
end//
```

Output

Select * from sailors //

sid	sname	rating	age
22	Dustin	7	45
29	Brutus	1	33
31	Lubber	8	56
32	Andy	8	26
64	Horatio	7	35
71	Zorba	10	16
74	Horatio	9	35
85	Art	3	26
95	Bob	3	64

DELETE TRIGGER

Queries:

```
CREATE TABLE Reserves (sid INT, bid INT, day DATE NOT NULL,
PRIMARY KEY (sid, bid), FOREIGN KEY (sid) REFERENCES Sailors(sid)
ON DELETE CASCADE, FOREIGN KEY(bid) REFERENCES Boats
(bid) ON DELETE CASCADE);
DESC Reserves;
INSERT INTO Reserves VALUES (22,101,'2012/10/10');
INSERT INTO Reserves VALUES (22,102,'2012/10/9');
INSERT INTO Reserves VALUES(22,103,'2012/08/10');
INSERT INTO Reserves VALUES(22,104,'2012/07/10');
INSERT INTO Reserves VALUES(31,102,'2012/11/10');
INSERT INTO Reserves VALUES(31,103,'2012/06/11');
INSERT INTO Reserves VALUES(31,104,'2012/12/11');
INSERT INTO Reserves VALUES (64,101,'2012/05/09');
INSERT INTO Reserves VALUES(64,102,'2012/08/09');
INSERT INTO Reserves VALUES(74,103,'2012/08/09');
DELIMITER//
Create trigger t3 before delete on reserves
for each row
begin
insert into cancel values(old.sid, old.bid, old.day);
end//
```

Output

Select * from reserves //

Sid	bid	day
22	101	2012-10-10
22	102	2012-10-09
22	103	2012-08-10
22	104	2012-07-10
31	102	2012-11-10
31	103	2012-06-11
31	104	2012-12-11
64	101	2012-05-09
64	102	2012-08-09
74	103	2012-08-09

Task 12

12. Use TCL commands for your transactions

1. Commit
2. Rollback
3. Save point

Queries:

```
CREATE TABLE Sailors(sid INT, sname VARCHAR(20), rating INT,
age FLOAT, PRIMARY KEY(sid));
INSERT INTO Sailors VALUES(22, 'Dustin', 7,45);
INSERT INTO sailors VALUES(29,'Brutus',1,33);
INSERT INTO Sailors VALUES(31, `Lubber', 8,56);
INSERT INTO Sailors VALUES(32,'Andy',8,26);
INSERT INTO Sailors VALUES(58,`Rusty', 10,35);
INSERT INTO Sailors VALUES(74,`Horatio',9,35);
INSERT INTO Sailors VALUES(64,`Horatio',7,35);
INSERT INTO Sailors VALUES(95,`Bob', 3,64);
INSERT INTO Sailors VALUES(85,`Art', 3,26);
INSERT INTO Sailors VALUES(71,`Zorba',10,16);
Select *from Sailors;
START TRANSACTION;
COMMIT;
SET autoCommit=0;
SAVEPOINT Insertion;
UPDATE Sailors SET rating=10 WHERE age=35;
SAVEPOINT Updation;
ROLLBACK To Insertion;
```

Output

Select * from sailors;

Sid	Sname	rating	age
22	Dustin	7	45
29	Brutus	1	33
31	Lubber	8	56
32	Andy	8	26
64	Horatio	7	35
71	Zorba	10	16
74	Horatio	9	35
85	Art	3	26
95	Bob	3	64