DML Operation

1 Change salary of employee 115 to 8000 if the existing salary is less than 6000.

update employees set salary=8000 where employee_id=115 and salary<6000;

```
1 row in set (0.00 sec)

mysql> update employees set salary=8000 where employee_id=115 and salary<6000;
Query OK, 1 row affected (0.27 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select salary from employees where employee_id=115;
+--------+
| salary |
+--------+
|   8000 |
+--------+
1 row in set (0.00 sec)
```

2 3Insert a new employee into employees with all the required details.

insert into employees
values(93259,'shreeraksha','hebbale','shreeraksha',9019801889,'2020_12_29','ST_MAN',27000,null,100,10);

```
ysql> insert into employees values(93259,'shreeraksha','hebbale','shreeraksha',9019801889,'2020_12_29','ST_MAN',27000,null,100,10);
uery OK, 1 row affected (0.22 sec)
```

3 Delete department 20.

Delete from employees where department_id=20;

4 Change job ID of employee 110 to IT_PROG if the employee belongs to department 10 and the existing job ID does not start with IT.

Update employees set job_id='IT_PROG' where department_id=10 and job_id not like 'IT%';

```
mysql> Update employees set job_id='IT_PROG' where department_id=10 and job_id not like 'IT%';
Query OK, 2 rows affected (0.17 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

5 Insert a row into departments table with manager ID 120 and location ID in any location ID for city Tokyo.

insert into departments values(1000,'finance',120,1200);

```
sql> insert into departments values(1000,'finance',120,1200);
ery OK, 1 row affected (0.28 sec)
```

6 Display job title, employee ID, number of days between ending date and starting date for all jobs in department 30 from job history

select employee_id,job_title,end_date-start_date Days from job_history a,jobs b where a.job_id=b.job_id and department_id=90;

```
mysql> select employee_id,job_title,end_date-start_date Days from job_history a,jobs b where a.job_id=b.job_id and depar
tment_id=90;
+-------------+-------------------------+-------+
| employee_id | job_title               | Days  |
+-------------+-------------------------+-------+
|         200 | Administration Assistant | 59700 |
|         200 | Public Accountant        | 40530 |
+-------------+-------------------------+-------+
2 rows in set (0.11 sec)
```

DDL Assignments with Constraints

Table ---> Customer

custId, firstName,lastName,age,city, mobileNumber, dob

Add the Constraints

custId is Primary Key

firstName not null

age must be greater than 21

mobile must be unique

create table Customer(custid int,firstName varchar(20) not null,lastName varchar(20),age int,city varchar(20),mobileNumber int,dob date,constraint c_pk primary key(custid),constraint a_pk check(age>21),constraint p_pk unique(mobileNumber));

```
mysql> create table Customer(custid int,firstName varchar(20) not null,lastName varchar(20),age int,city varchar(20),mob
ileNumber int,dob date,constraint c_pk primary key(custid),constraint a_pk check(age>21),constraint p_pk unique(mobileNu
mber));
Query OK, 0 rows affected (2.08 sec)
```

Table ----> Branch

branchId, branchName, city

Add the Constraints

branchId is Primary Key

branchName not null

city not null

create table Branch(branchId int,branchName varchar(20) not null,city varchar(20) not null,constraint b_pk primary key(branchId));

```
mysql> create table Branch(branchId int,branchName varchar(20) not null,city varchar(20) not null,constraint b_pk primar
y key(branchId));
Query OK, 0 rows affected (0.91 sec)
```

Table -----> Account

accountNumber, openingBalance, typeOfAccount, status,BankId,CustId

Add the Constraints

accountNumber is primary key

openingBalance must be greater than 5000

typeOfAccount must be saving/current

BankId is foreign key refer to BranchId(Primary key) Branch table

CustId is foreign key refer to Customer(Primary key) Customer table

Create table Account(accountNumber varchar(20),openingBalance int,typeOfAccount enum('savings','current'),status varchar(20),bankid int,constraint a_pk primary key(accountNumber),constraint ob_pk check(openingBalance>5000),constraint bi_pk foreign key(bankid) references Branch(branchId));

```
ysql> Create table Account(accountNumber varchar(20),openingBalance int,typeOfAccount  enum('savings','current'),status
 varchar(20),bankid int,constraint a_pk primary key(accountNumber),constraint ob_pk check(openingBalance>5000),constrain
t bi_pk  foreign key(bankid) references Branch(branchId));
uery OK, 0 rows affected (1.44 sec)
```

Table ----> Transaction

transactionId, transactionDate, MediumOfTransaction, TransactionAmount

Add the Constraints

transactionId is primary key

create table Transaction (transactionId int,transactionDate date,MediumOfTransaction varchar(20),TransactionAmount int,constraint ti_pk primary key(transactionId));

create table Transaction (transactionId int,transactionDate date,MediumOfTransaction varchar(20),TransactionAmount int,constraint ti_pk primary key(transactionId));

```
mysql> create table Transaction (transactionId int,transactionDate date,MediumOfTransaction varchar(20),TransactionAmoun
t int,constraint ti_pk  primary key(transactionId));
Query OK, 0 rows affected (0.91 sec)
```

Table ----> Loan

LoanId, loanAmount, customerId and bankdId

Add the Constraints

loadId is primary key

loanAmount must be +ve

BankId is foreign key refer to BranchId(Primary key) Branch table

Create table Loan(LoanId int,loanAmount int,customerId int,bankid int,constraint li_pk primary key(LoanId),constraint la_pk check(loanAmount>=0),constraint bi_pk foreign key(bankid) references Branch(branchId));

```
mysql> Create table Loan(LoanId int,loanAmount int,customerId int,bankid int,constraint  li_pk primary key(LoanId),const
raint la_pk check(loanAmount>=0),constraint bid_pk foreign key(bankid) references Branch(branchId));
Query OK, 0 rows affected (1.28 sec)
```

Subquery

Display details of departments managed by ‘John’.

select * from departments where manager_id in(select employee_id from employees where first_name='John');

```
mysql> select * from departments where manager_id in(select employee_id from employees where first_name='John');
+---------------+-----------------+------------+-------------+
| department_id | department_name | manager_id | location_id |
+---------------+-----------------+------------+-------------+
|            80 | Sales           |        145 |        2500 |
+---------------+-----------------+------------+-------------+
1 row in set (0.08 sec)
```

Display employees who did not do any job in the past.

SELECT EMPLOYEE_ID FROM JOB_HISTORY GROUP BY EMPLOYEE_ID HAVING COUNT(*) > 1;

```
mysql> SELECT EMPLOYEE_ID FROM JOB_HISTORY GROUP BY EMPLOYEE_ID HAVING COUNT(*) > 1;
+-------------+
| EMPLOYEE_ID |
+-------------+
|         101 |
|         176 |
|         200 |
+-------------+
3 rows in set (0.11 sec)
```

Display job title and average salary for employees who did a job in the past.

select job_title,avg(salary) from employees e,jobs j where e.job_id=j.job_id group by job_title;

```
mysql> select job_title,avg(salary) from employees e,jobs j where e.job_id=j.job_id group by job_title;
+----------------------------------+-------------+
| job_title                        | avg(salary) |
+----------------------------------+-------------+
| Public Accountant                |        8300 |
| Accounting Manager               |       12000 |
| President                        |       24000 |
| Administration Vice President    |       17000 |
| Accountant                       |        7920 |
| Finance Manager                  |       12000 |
| Human Resources Representative   |        6500 |
| Programmer                       |        8600 |
| Marketing Manager                |       13000 |
| Marketing Representative         |        6000 |
| Public Relations Representative  |       10000 |
| Purchasing Clerk                 |        3760 |
| Purchasing Manager               |       11000 |
| Sales Manager                    |       12200 |
| Sales Representative             |        8350 |
| Shipping Clerk                   |        3215 |
| Stock Clerk                      |        2785 |
| Stock Manager                    |        7280 |
+----------------------------------+-------------+
```

Display country name, city, and number of departments where department has more than 5 employees.

SELECT COUNTRY_NAME, CITY, COUNT(DEPARTMENT_ID) FROM COUNTRIES JOIN LOCATIONS USING (COUNTRY_ID) JOIN DEPARTMENTS USING (LOCATION_ID) WHERE DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES GROUP BY DEPARTMENT_ID HAVING COUNT(DEPARTMENT_ID)>5) GROUP BY COUNTRY_NAME, CITY;

```
mysql> SELECT COUNTRY_NAME, CITY, COUNT(DEPARTMENT_ID)
    -> FROM COUNTRIES JOIN LOCATIONS USING (COUNTRY_ID) JOIN DEPARTMENTS USING (LOCATION_ID)
    -> WHERE DEPARTMENT_ID IN
    ->     (SELECT DEPARTMENT_ID FROM EMPLOYEES
    ->  GROUP BY DEPARTMENT_ID
    ->  HAVING COUNT(DEPARTMENT_ID)>5)
    -> GROUP BY COUNTRY_NAME, CITY;
+-------------------------+--------------------+----------------------+
| COUNTRY_NAME            | CITY               | COUNT(DEPARTMENT_ID) |
+-------------------------+--------------------+----------------------+
| United States of America | South San Francisco |                    1 |
| United States of America | Seattle            |                    2 |
| United Kingdom          | Oxford             |                    1 |
+-------------------------+--------------------+----------------------+
3 rows in set (0.00 sec)
```

Display details of manager who manages more than 5 employees.

Select first_name from employees where employee_id in(select manager_id from employees group by manager_id having count(*)>5);

```
mysql> Select  first_name from employees where employee_id in(select manager_id  from employees group by manager_id havi
ng count(*)>5);
+------------+
| first_name |
+------------+
| Steven     |
| Matthew    |
| Adam       |
| Payam      |
| Shanta     |
| Kevin      |
| John       |
| Karen      |
| Alberto    |
| Gerald     |
| Eleni      |
+------------+
11 rows in set (0.00 sec)
```

Display details of current job for employees who worked as IT Programmers in the past.

Select * from jobs where job_id in(select job_id from employees where employee_id in (select employee_id from job_history where job_id=’IT_PROG’));

```
mysql> Select * from jobs where job_id in(select job_id from employees where employee_id in (select employee_id from job
_history where job_id='IT_PROG'));
+--------+----------------------------+------------+------------+
| job_id | job_title                  | min_salary | max_salary |
+--------+----------------------------+------------+------------+
| AD_VP  | Administration Vice President |      15000 |      30000 |
+--------+----------------------------+------------+------------+
1 row in set (0.00 sec)
```

7. Display the details of employees drawing the highest salary in the department

Select department_id,max(salary) from employees group by department_id;

```
mysql> Select department_id,max(salary) from employees group by department_id;
+---------------+-------------+
| department_id | max(salary) |
+---------------+-------------+
|          NULL |        7000 |
|            10 |       27000 |
|            20 |       13000 |
|            30 |       11000 |
|            40 |        6500 |
|            50 |        8200 |
|            60 |        9000 |
|            70 |       10000 |
|            80 |       14000 |
|            90 |       24000 |
|           100 |       12000 |
|           110 |       12000 |
+---------------+-------------+
12 rows in set (0.11 sec)
```

8. Display third highest salary of all employees

Select max(salary) from employees where salary<(select max(salary) from employees where salary<(select max(salary) from employees));

```
mysql> Select max(salary) from employees where salary<(select max(salary) from employees where salary<(select max(salary
) from employees));
+-------------+
| max(salary) |
+-------------+
|       17000 |
+-------------+
1 row in set (0.02 sec)
```