1. DDL

1. Objective To understand the syntax of basic DDL commands such as

- Create table
- Alter table
    - Add/Drop attribute
    - Modify type and size of an attribute
    - Rename attribute
- Truncate
- Drop

Database used: Table name : student

Student(rollno,name)
Add branch
Modify the width of Attribute branch
Modify the datatype of Attribute branch
Rename the attribute
Remove the attribute

```
mysql> create table students(rollno int,name char(20));
Query OK, 0 rows affected (0.37 sec)

mysql> desc students;
+--------+----------+------+-----+---------+-------+
| Field  | Type     | Null | Key | Default | Extra |
+--------+----------+------+-----+---------+-------+
| rollno | int(11)  | YES  |     | NULL    |       |
| name   | char(20) | YES  |     | NULL    |       |
+--------+----------+------+-----+---------+-------+
2 rows in set (0.07 sec)

mysql> alter table students add branch char(15);
Query OK, 0 rows affected (0.58 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc students;
+--------+----------+------+-----+---------+-------+
| Field  | Type     | Null | Key | Default | Extra |
+--------+----------+------+-----+---------+-------+
| rollno | int(11)  | YES  |     | NULL    |       |
| name   | char(20) | YES  |     | NULL    |       |
| branch | char(15) | YES  |     | NULL    |       |
+--------+----------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql> alter table students modify branch char(10);
Query OK, 0 rows affected (0.87 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> desc students;
+--------+----------+------+-----+---------+-------+
| Field  | Type     | Null | Key | Default | Extra |
+--------+----------+------+-----+---------+-------+
| rollno | int(11)  | YES  |     | NULL    |       |
| name   | char(20) | YES  |     | NULL    |       |
| branch | char(10) | YES  |     | NULL    |       |
+--------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> alter table students modify branch varchar(10);
Query OK, 0 rows affected (1.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
mysql> desc students;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| rollno | int(11)     | YES  |     | NULL    |       |
| name   | char(20)    | YES  |     | NULL    |       |
| branch | varchar(10) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> alter table students change  name  sname char(20);
Query OK, 0 rows affected (0.16 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc students;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| rollno | int(11)     | YES  |     | NULL    |       |
| sname  | char(20)    | YES  |     | NULL    |       |
| branch | varchar(10) | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> alter table students drop branch;
mysql> desc students;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| rollno | int(11)     | YES  |     | NULL    |       |
| sname  | char(20)    | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> truncate table students;
Query OK, 0 rows affected (0.47 sec)

mysql> drop table students;
Query OK, 0 rows affected (0.23 sec)
```

2. DML

2. Objective To understand the syntax of basic DML commands such as

- Insert
- Update
- Delete

Database used: students(rollno : int, name : char(20), branch : varchar(20))

```
mysql> create table students(rollno int,name char(20),branch varchar(20));
Query OK, 0 rows affected (0.60 sec)

mysql> insert into students values(1,'saketh','cse');
Query OK, 1 row affected (0.08 sec)

mysql> insert into students values(2,'arun','it'),(3,'varun','eee');
Query OK, 2 rows affected (0.09 sec)

mysql> insert into students values(4,'sathish','it'),(4,'sridhar','eee');
Query OK, 2 rows affected (0.07 sec)

mysql> select * from students;
+--------+---------+--------+
| rollno | name    | branch |
+--------+---------+--------+
|      1 | saketh  | cse    |
|      2 | arun    | it     |
|      3 | varun   | eee    |
|      4 | sathish | it     |
|      4 | sridhar | eee    |
+--------+---------+--------+
5 rows in set (0.00 sec)

mysql> update students set name='venkat' where rollno=2;
Query OK, 1 row affected (0.10 sec)

mysql> select * from students;
+--------+---------+--------+
| rollno | name    | branch |
+--------+---------+--------+
|      1 | saketh  | cse    |
|      2 | venkat  | it     |
|      3 | varun   | eee    |
|      4 | sathish | it     |
|      4 | sridhar | eee    |
+--------+---------+--------+
5 rows in set (0.00 sec)
mysql> update students set name='vinod',branch='ece' where rollno=3;
Query OK, 1 row affected (0.07 sec)
```

```
mysql> select * from students;
+--------+---------+--------+
| rollno | name    | branch |
+--------+---------+--------+
|      1 | saketh  | cse    |
|      2 | venkat  | it     |
|      3 | vinod   | ece    |
|      4 | sathish | it     |
|      4 | sridhar | eee    |
+--------+---------+--------+
5 rows in set (0.00 sec)
mysql> update students set name='vamshi' where rollno=4;
Query OK, 2 rows affected (0.48 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from students;

+--------+--------+--------+
| rollno | name   | branch |
+--------+--------+--------+
|      1 | saketh | cse    |
|      2 | venkat | it     |
|      3 | vinod  | ece    |
|      4 | vamshi | it     |
|      4 | vamshi | eee    |
+--------+--------+--------+
5 rows in set (0.00 sec)
mysql> delete from students where rollno=1;
Query OK, 1 row affected (0.34 sec)

mysql> select * from students;
+--------+--------+--------+
| rollno | name   | branch |
+--------+--------+--------+
|      2 | venkat | it     |
|      3 | vinod  | ece    |
|      4 | vamshi | it     |
|      4 | vamshi | eee    |
+--------+--------+--------+
4 rows in set (0.00 sec)

mysql> update students set branch='cse';
Query OK, 4 rows affected (0.07 sec)
Rows matched: 4  Changed: 4  Warnings: 0

mysql> select * from  students;
+--------+--------+--------+
| rollno | name   | branch |
+--------+--------+--------+
|      2 | venkat | cse    |
|      3 | vinod  | cse    |
|      4 | vamshi | cse    |
|      4 | vamshi | cse    |
+--------+--------+--------+
4 rows in set (0.01 sec)
mysql> delete from students;
Query OK, 4 rows affected (0.08 sec)
```

3.Constraints

3. Objective to understand the commands used to impose constraints on a table such as

- Not null
- unique
- primary key
- foreign key

Database used:
Branch()
Account()
Loan()
Customer()
Depositor()
Borrower()

```
create table Branch
   (branch_name            varchar(15)       ,
    branch_city   varchar(15)       not null,
    assets integer not null,
    primary key(branch_name));

create table Account
   (account_number  varchar(15) ,
    branch_name     varchar(15) not null,
    balance       integer     not null,
    primary key(account_number),
    foreign key(branch_name) references Branch(branch_name));

create table Customer
   (customer_name        varchar(15),
    customer_street      varchar(12)       not null,
    customer_city        varchar(15)       not null,
    customer_ph     varchar(10) unique,
    primary key(customer_name));

create table Depositor
   (customer_name        varchar(15)       ,
    account_number       varchar(15)       ,
    primary key(customer_name, account_number),
    foreign key(account_number) references Account(account_number),
    foreign key(customer_name) references Customer(customer_name));
```

```
create table Borrower

   (customer_name        varchar(15)      ,
    loan_number           varchar(15)      ,
    primary key(customer_name, loan_number),
    foreign key(customer_name) references Customer(customer_name),
    foreign key(loan_number) references Loan(loan_number));
```

<u>Adding primary key constraint.</u>

```
Create table cust(cust_no integer(10),cust_name varchar(20));

Query ok,0 rows effected

mysql> desc cust;
+---------+----------+------+-----+---------+-------+
| Field  | Type     | Null | Key | Default | Extra |
+---------+----------+------+-----+---------+-------+
| cust_no | int(11)  | NO |     | 0    |       |
| cust_name varchar(20) | YES  |     | NULL    |       |
+---------+----------+------+-----+---------+-------+
2 rows in set (0.07 sec)


Alter table cust add primary key(cust_no);

Query ok,0 rows effected

mysql> desc cust;
+---------+----------+------+-----+---------+-------+
| Field  | Type     | Null | Key | Default | Extra |
+---------+----------+------+-----+---------+-------+
| cust_no | int(11)  | NO |  PRI   | 0    |       |
| cust_name varchar(20) | YES  |     | NULL    |       |
+---------+----------+------+-----+---------+-------+
2 rows in set (0.07 sec)
```

<u>Adding foreign key constraint.</u>

```
mysql> create table cust1(cust_no int(10),cust_name varchar(20));
Query OK, 0 rows affected (0.26 sec)

mysql> desc cust1;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| cust_no   | int(10)     | YES  |     | NULL    |       |
| cust_name | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> alter table cust1 add foreign key (cust_no) references cust(cust_no);
Query OK, 0 rows affected (0.66 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> desc cust1;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| cust_no   | int(10)     | YES  | MUL | NULL    |       |
| cust_name | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

REMOVING A PRIMARY KEY CONSTRAINT

```
mysql> alter table cust drop primary key;
Query OK, 0 rows affected (0.72 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc cust;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| cust_no   | int(10)     | NO   |     | 0       |       |
| cust_name | varchar(20) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

4.Simple to Complex Queries

4. Objective: To understand the syntax of MySQL queries which use

- Select
  - distinct
  - order by
  - group by
- Comparison operators
  - Like, Not like
  - > ,<.>=,<=,==,<>
  - Between and, not between and
  - In, not in
  - Any, all
- Aggregate functions
  - Count, Sum, Average, Max, Min
- Set Operations
  - Union, Except, Intersect
- Nested Queries
  - Correlated

Database used:
emp()
dept()
Depositor()
Borrower()

Q) Display unique jobs of employees

```
mysql> select distinct job from emp;

+-----------+
| job       |
+-----------+
| CLERK     |
| SALESMAN  |
| MANAGER   |
| ANALYST   |
| PRESIDENT |
+-----------+
5 rows in set (0.05 sec)
```

Q) Display the employee names whose names starts with ‘J’.

```
mysql> select ename from emp where ename like 'J%';

+-------+
| ename |
+-------+
| JAMES |
| JONES |
+-------+
2 rows in set (0.03 sec)
```

Q) Display the employee names whose names DOES NOT starts with ‘J’.

```
mysql> select ename from emp where ename not like 'J%';
+--------+
| ename  |
+--------+
| ADAMS  |
| ALLEN  |
| BLAKE  |
| CLARK  |
| FORD   |
| KING   |
| MARTIN |
| MILLER |
| SCOTT  |
| SMITH  |
| TURNER |
| WARD   |
+--------+
12 rows in set (0.00 sec)
```

Q) Display the employee names whose names ENDS with ‘S’.

```
mysql> select ename from emp where ename  like '%S';
+-------+
| ename |
+-------+
| ADAMS |
| JAMES |
| JONES |
+-------+
3 rows in set (0.00 sec)
```

Q) Display the employee names whose names does not ENDS with ‘S’.

```
mysql> select ename from emp where ename not like '%S';
+--------+
| ename  |
+--------+
| ALLEN  |
| BLAKE  |
| CLARK  |
| FORD   |
| KING   |
| MARTIN |
| MILLER |
| SCOTT  |
| SMITH  |
| TURNER |
| WARD   |
+--------+
11 rows in set (0.00 sec)
```

Q) Display the employee names whose names contains exactly 4 letters.

```
mysql> select ename from emp where ename like '____';
+-------+
| ename |
+-------+
| FORD  |
| KING  |
| WARD  |
+-------+
3 rows in set (0.00 sec)
```

Q) Display the employee names whose names starts with ‘S’ and ends with ‘H’.

```
mysql> select ename from emp where ename like 'S%H';
+-------+
| ename |
+-------+
| SMITH |
+-------+
1 row in set (0.00 sec)
```

Q) Display the employee names whose names starts with ‘J’ 3RD LETTER ‘M’ AND 5TH LETTER ‘S’ .

```
mysql> select ename from emp where ename like 'J_M_S';
+-------+
| ename |
+-------+
| JAMES |
+-------+
1 row in set (0.00 sec)
```

Q) Display the employee names whose salary is in the range of 1500 and 2000.

```
mysql> select empno,ename from emp where sal between 1500 and 2000;
+-------+--------+
| empno | ename  |
+-------+--------+
|  7499 | ALLEN  |
|  7844 | TURNER |
+-------+--------+
2 rows in set (0.00 sec)
```

Q) Display the employee names whose salary is not in the range of 1500 and 2000.

```
mysql> select empno,ename from emp where sal not between 1500 and 2000;
+-------+--------+
| empno | ename  |
+-------+--------+
|  7876 | ADAMS  |
|  7698 | BLAKE  |
|  7782 | CLARK  |
|  7902 | FORD   |
|  7900 | JAMES  |
|  7566 | JONES  |
|  7839 | KING   |
|  7654 | MARTIN |
|  7934 | MILLER |
|  7788 | SCOTT  |
|  7369 | SMITH  |
|  7521 | WARD   |
+-------+--------+
12 rows in set (0.00 sec)
```

Q) Display the employee number and names in the ascending order of employee name

```
mysql> select empno,ename from emp order by ename;
+-------+--------+
| empno | ename  |
+-------+--------+
|  7876 | ADAMS  |
|  7499 | ALLEN  |
|  7698 | BLAKE  |
|  7782 | CLARK  |
|  7902 | FORD   |
|  7900 | JAMES  |
|  7566 | JONES  |
|  7839 | KING   |
|  7654 | MARTIN |
|  7934 | MILLER |
|  7788 | SCOTT  |
|  7369 | SMITH  |
|  7844 | TURNER |
|  7521 | WARD   |
+-------+--------+
14 rows in set (0.00 sec)
```

Q) Display the employee number and names in the descinding order of employee name

```
mysql> select empno,ename from emp order by ename desc;
+-------+--------+
| empno | ename  |
+-------+--------+
|  7521 | WARD   |
|  7844 | TURNER |
|  7369 | SMITH  |
|  7788 | SCOTT  |
|  7934 | MILLER |
|  7654 | MARTIN |
|  7839 | KING   |
|  7566 | JONES  |
|  7900 | JAMES  |
|  7902 | FORD   |
|  7782 | CLARK  |
|  7698 | BLAKE  |
|  7499 | ALLEN  |
|  7876 | ADAMS  |
+-------+--------+
14 rows in set (0.00 sec)
```

Q) Display the number and names of the employees who are working in department 10 and 20

```
mysql> select empno,ename from emp where deptno in(10,20);
+-------+--------+
| empno | ename  |
+-------+--------+
|  7369 | SMITH  |
|  7566 | JONES  |
|  7782 | CLARK  |
|  7788 | SCOTT  |
|  7839 | KING   |
|  7876 | ADAMS  |
|  7902 | FORD   |
|  7934 | MILLER |
+-------+--------+
8 rows in set (0.02 sec)
```

Q) Display the number and names of the employees who are working either in department 10 or 20

```
mysql> select empno,ename from emp where deptno=10 or deptno=20;
+-------+--------+
| empno | ename  |
+-------+--------+
|  7369 | SMITH  |
|  7566 | JONES  |
|  7782 | CLARK  |
|  7788 | SCOTT  |
|  7839 | KING   |
|  7876 | ADAMS  |
|  7902 | FORD   |
|  7934 | MILLER |
+-------+--------+
8 rows in set (0.00 sec)
```

Q) Display the number and names of the employees who are not having manager.

```
mysql> select empno,ename from emp where mgr is null;
+-------+-------+
| empno | ename |
+-------+-------+
|  7839 | KING  |
+-------+-------+
1 row in set (0.00 sec)
```

Q) Display the number of tuples in employee reletaion.

```
mysql> select count(*),count(empno),count(sal),count(comm) from emp;
+----------+--------------+------------+-------------+
| count(*) | count(empno) | count(sal) | count(comm) |
+----------+--------------+------------+-------------+
|       14 |           14 |         14 |           4 |
+----------+--------------+------------+-------------+
1 row in set (0.01 sec)
```

Q) Display the number of tuples,sum,avg,max,min of salary from employee reletaion.

```
mysql> select count(*),sum(sal),avg(sal),max(sal),min(sal) from emp;
+----------+----------+-----------+----------+----------+
| count(*) | sum(sal) | avg(sal)  | max(sal) | min(sal) |
+----------+----------+-----------+----------+----------+
|       14 |    29025 | 2073.2143 |     5000 |      800 |
+----------+----------+-----------+----------+----------+
1 row in set (0.00 sec)
```

Q) Display the department number and average salary of each department.

```
mysql> select deptno,avg(sal) from emp group by deptno;
+--------+-----------+
| deptno | avg(sal)  |
+--------+-----------+
|     10 | 2916.6667 |
|     20 | 2175.0000 |
|     30 | 1566.6667 |
+--------+-----------+
3 rows in set (0.02 sec)
```

Q) Display the job and total salary of each job.

```
mysql> select job,sum(sal) from emp group by job;
+-----------+----------+
| job       | sum(sal) |
+-----------+----------+
| ANALYST   |     6000 |
| CLERK     |     4150 |
| MANAGER   |     8275 |
| PRESIDENT |     5000 |
| SALESMAN  |     5600 |
+-----------+----------+
5 rows in set (0.00 sec)
```

Q) Display the department number and average salary of each department.

```
mysql> select deptno,count(empno) from emp group by deptno;
+--------+--------------+
| deptno | count(empno) |
+--------+--------------+
|     10 |            3 |
|     20 |            5 |
|     30 |            6 |
+--------+--------------+

3 rows in set (0.00 sec)
```

Q) Display the department with more than 3 employees.

```
mysql> select deptno,count(empno) from emp group by deptno having count(empno)>3;
+--------+--------------+
| deptno | count(empno) |
+--------+--------------+
|     20 |            5 |
|     30 |            6 |
+--------+--------------+
2 rows in set (0.00 sec)
```

Q) Display the Employee name who is having highest salary

```
mysql> select ename from emp where sal=(select max(sal) from emp);
+-------+
| ename |
+-------+
| KING  |
+-------+
1 row in set (0.02 sec)
```

Q) Display the Employee number and salary is equal to the employees who are working in deptno 10.

```
mysql> SELECT e1.empno, e1.sal FROM   emp e1 WHERE  e1.sal in (SELECT e2.sal FROM emp e2
            WHERE e2.deptno = 10);
+-------+------+
| empno | sal  |
+-------+------+
|  7782 | 2450 |
|  7839 | 5000 |
|  7934 | 1300 |
+-------+------+
3 rows in set (0.02 sec)
```

Q) Display the Employee number and salary is greater than all employees working in dept.no 10.

```
mysql> SELECT e1.empno, e1.sal FROM   emp e1 WHERE  e1.sal > ALL (SELECT e2.sal FROM emp e2
            WHERE e2.deptno = 10);

Empty set (0.00 sec)
```

Q) Display the Employee number and salary is greater than ANY employee working in dept.no 10.

```
mysql> SELECT e1.empno, e1.sal FROM   emp e1 WHERE  e1.sal > ANY (SELECT e2.sal FROM emp e2
            WHERE e2.deptno=10);
+-------+------+
| empno | sal  |
+-------+------+
|  7499 | 1600 |
|  7698 | 2850 |
|  7782 | 2450 |
|  7902 | 3000 |
|  7566 | 2975 |
|  7839 | 5000 |
|  7788 | 3000 |
|  7844 | 1500 |
+-------+------+
8 rows in set (0.00 sec)
```

Q) Display the Employee NAMES that are present in emp relation.

```
mysql> select ename from emp where exists (select * from emp);
+--------+
| ename  |
+--------+
| ADAMS  |
| ALLEN  |
| BLAKE  |
| CLARK  |
| FORD   |
| JAMES  |
| JONES  |
| KING   |
| MARTIN |
| MILLER |
| SCOTT  |
| SMITH  |
| TURNER |
| WARD   |
+--------+
14 rows in set (0.02 sec)
```

Q) Display the Employee number and salary whose salary is greater than the employee salary who are working in deptno 10.

```
mysql> SELECT e1.empno, e1.sal FROM   emp e1 WHERE  EXISTS (SELECT e2.sal FROM emp e2
          WHERE e2.deptno = 10 AND   e1.sal > e2.sal);
+-------+------+
| empno | sal  |
+-------+------+
|  7499 | 1600 |
|  7698 | 2850 |
|  7782 | 2450 |
|  7902 | 3000 |
|  7566 | 2975 |
|  7839 | 5000 |
|  7788 | 3000 |
|  7844 | 1500 |
+-------+------+
8 rows in set (0.02 sec)
```

Queries on BANK Database

```
mysql> desc customer;
+-----------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-----------------+-------------+------+-----+---------+-------+
| customer_name | varchar(15) | NO | PRI | NULL | |
| customer_street | varchar(12) | NO | | NULL | |
| customer_city | varchar(15) | NO | | NULL | |
+-----------------+-------------+------+-----+---------+-------+
3 rows in set (0.07 sec)
mysql> desc branch;
+-------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| branch_name | varchar(15) | NO | PRI | NULL | |
| branch_city | varchar(15) | NO | | NULL | |
| assets | int(11) | NO | | NULL | |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.07 sec)
mysql> desc account;
+----------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| account_number | varchar(15) | NO | PRI | NULL | |
| branch_name | varchar(15) | NO | MUL | NULL | |
| balance | int(11) | NO | | NULL | |
+----------------+-------------+------+-----+---------+-------+
3 rows in set (0.03 sec)
mysql> desc loan;
+-------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| loan_number | varchar(15) | NO | PRI | NULL | |
| branch_name | varchar(15) | NO | MUL | NULL | |
| amount | int(11) | NO | | NULL | |
+-------------+-------------+------+-----+---------+-------+
3 rows in set (0.07 sec)
mysql> desc borrowers;
ERROR 1146 (42S02): Table 'bankingdb.borrowers' doesn't exist
mysql> desc borrower;
+---------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| customer_name | varchar(15) | NO | PRI | NULL | |
| loan_number | varchar(15) | NO | PRI | NULL | |
+---------------+-------------+------+-----+---------+-------+
2 rows in set (0.06 sec)
mysql> desc depositor;
+----------------+-------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------------+-------------+------+-----+---------+-------+
| customer_name | varchar(15) | NO | PRI | NULL | |
| account_number | varchar(15) | NO | PRI | NULL | |
+----------------+-------------+------+-----+---------+-------+
2 rows in set (0.06 sec)
mysql> select * from branch;
+-------------+-------------+---------+
| branch_name | branch_city | assets |
+-------------+-------------+---------+
| Brighton | Brooklyn | 7000000 |
| Central | Rye | 400280 |
| Downtown | Brooklyn | 900000 |
| Mianus | Horseneck | 400200 |
| North Town | Rye | 3700000 |
| Perryridge | Horseneck | 1700000 |
| Pownal | Bennington | 400000 |
| Redwood | Palo Alto | 2100000 |
| Round Hill | Horseneck | 8000000 |
+-------------+-------------+---------+
9 rows in set (0.04 sec)
mysql> select * from account;
+----------------+-------------+---------+
| account_number | branch_name | balance |
+----------------+-------------+---------+
| A-101 | Downtown | 500 |
| A-102 | Perryridge | 400 |
| A-201 | Perryridge | 900 |
| A-215 | Mianus | 700 |
| A-217 | Brighton | 750 |
| A-222 | Redwood | 700 |
| A-305 | Round Hill | 350 |
| A-333 | Central | 850 |
| A-444 | North Town | 625 |
+----------------+-------------+---------+
9 rows in set (0.00 sec)

mysql> select * from loan;

+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| L-11 | Round Hill | 900 |
| L-14 | Downtown | 1500 |
| L-15 | Perryridge | 1500 |
| L-16 | Perryridge | 1300 |
| L-17 | Downtown | 1000 |
| L-20 | North Town | 7500 |
| L-21 | Central | 570 |
| L-23 | Redwood | 2000 |
| L-93 | Mianus | 500 |
+-------------+-------------+--------+
9 rows in set (0.00 sec)
mysql> select *from customer;
+---------------+-----------------+---------------+
| customer_name | customer_street | customer_city |
+---------------+-----------------+---------------+
| Adams | Spring | Pittsfield |
| Brooks | Senator | Brooklyn |
| Curry | North | Rye |
| Glenn | Sand Hill | Woodside |
| Green | Walnut | Stamford |
| Hayes | Main | Harrison |
| Jackson | University | Salt Lake |
| Johnson | Alma | Palo Alto |
| Jones | Main | Harrison |
| Lindsay | Park | Pittsfield |
| Majeris | First | Rye |
| McBride | Safety | Rye |
| Smith | Main | Rye |
| Turner | Putnam | Stamford |
| Williams | Nassau | Princeton |
+---------------+-----------------+---------------+
15 rows in set (0.00 sec)
mysql> select * from borrower;
+---------------+-------------+
| customer_name | loan_number |
+---------------+-------------+
| Smith | L-11 |
| Jackson | L-14 |
| Adams | L-16 |
| Jones | L-17 |
| Williams | L-17 |
| McBride | L-20 |
| Smith | L-21 |
| Curry | L-93 |
8 rows in set (0.06 sec)
mysql> select * from depositor;
+---------------+----------------+
| customer_name | account_number |
+---------------+----------------+
| Hayes | A-101 |
| Johnson | A-101 |
| Hayes | A-102 |
| Johnson | A-201 |
| Smith | A-215 |
| Jones | A-217 |
| Lindsay | A-222 |
| Turner | A-305 |
| Majeris | A-333 |
| Smith | A-444 |
+---------------+----------------+
10 rows in set (0.03 sec)
```

Q) Find the names and loan numbers of all customers who have a loan at the Perryridge branch.

```
mysql> select customer_name,borrower.loan_number
-> from borrower,loan
-> where borrower.loan_number=loan.loan_number and branch_name='Perryridge';
+---------------+-------------+
| customer_name | loan_number |
+---------------+-------------+
| Adams | L-16 |
+---------------+-------------+
1 row in set (0.00 sec)
```

Q) Find the names of all branches that have assets greater than that of each branch in Brooklyn.

```
mysql> select distinct t.branch_name
-> from branch t,branch s
-> where t.assets>s.assets and s.branch_city='Brooklyn';
+-------------+
| branch_name |
+-------------+
| Brighton |
| North Town |
| Perryridge |
| Redwood |
| Round Hill |
+-------------+
5 rows in set (0.00 sec)
```

Q) Find the names of all customers whose street address includes the substring ‘Main’.

```
mysql> select customer_name from customer where customer_street like '%Main%';
+---------------+
| customer_name |
+---------------+
| Hayes |
| Jones |
| Smith |
+---------------+
3 rows in set (0.00 sec)
```

Q) Find the names and loan numbers of all customers who have a loan at the Perryridge branch.

```
mysql> select distinct customer_name
-> from borrower,loan
-> where borrower.loan_number = loan.loan_number and branch_name='Perryridge';
+---------------+
| customer_name |
+---------------+
| Adams |
+---------------+
1 row in set (0.00 sec)
```

Q) Find the tuples of the loan relation in descending order of amount and ascending order of loan number.

```
mysql> select *
-> from loan
-> order by amount desc, loan_number asc;
+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| L-20 | North Town | 7500 |
| L-23 | Redwood | 2000 |
| L-14 | Downtown | 1500 |
| L-15 | Perryridge | 1500 |
| L-16 | Perryridge | 1300 |
| L-17 | Downtown | 1000 |
| L-11 | Round Hill | 900 |
| L-21 | Central | 570 |
| L-93 | Mianus | 500 |
+-------------+-------------+--------+
9 rows in set (0.00 sec)
```

Q) Find the average account balance at the Perryridge branch.

```
mysql> select avg (balance) from account where branch_name='Perryridge';
+---------------+
| avg (balance) |
+---------------+
| 650.0000 |
+---------------+
1 row in set (0.00 sec)
```

Q) Find the average balance and branch name of each branch.

```
mysql> select branch_name,avg(balance)from account group by branch_name;
+-------------+--------------+
| branch_name | avg(balance) |
+-------------+--------------+
| Brighton | 750.0000 |
| Central | 850.0000 |
| Downtown | 500.0000 |
| Mianus | 700.0000 |
| North Town | 625.0000 |
| Perryridge | 650.0000 |
| Redwood | 700.0000 |
| Round Hill | 350.0000 |
+-------------+--------------+
8 rows in set (0.00 sec)
```

Q) Find the number of depositors each branch along with branch name.

```
mysql> select branch_name, count(distinct customer_name) from depositor, account
-> where depositor.account_number = account.account_number
-> group by branch_name;
+-------------+-------------------------------+
| branch_name | count(distinct customer_name) |
+-------------+-------------------------------+
| Brighton | 1 |
| Central | 1 |
| Downtown | 2 |
| Mianus | 1 |
| North Town | 1 |
| Perryridge | 2 |
| Redwood | 1 |
| Round Hill | 1 |
+-------------+-------------------------------+
8 rows in set (0.04 sec)
mysql>
```

Q) Find all the branch names whose average balance greater than 1,200.

```
mysql> select branch_name, avg (balance)
-> from account
-> group by branch_name
-> having avg (balance) > 1200;
Empty set (0.00 sec)
mysql>
mysql> select avg (balance)from account;
+---------------+
| avg (balance) |
+---------------+
| 641.6667 |
+---------------+
1 row in set (0.00 sec)
```

Q) Find the number of tuples in customer relation.

```
mysql> select count(*) from customer;
+----------+
| count(*) |
+----------+
| 15 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> select d.customer_name, avg(a.balance)
-> from depositor d , account a, customer c
-> where d.account_number = a.account_number and d.customer_name = c.customer_name ;
+---------------+----------------+
| customer_name | avg(a.balance) |
+---------------+----------------+
| Hayes | 627.5000 |
+---------------+----------------+
1 row in set (0.00 sec)
```

Q) Find the average balance of each customer who lives in Harrison

```
mysql> select d.customer_name, avg(a.balance)
-> from depositor d , account a, customer c
-> where d.account_number = a.account_number and d.customer_name = c.customer_name and
c.customer_city='Harrison';
+---------------+----------------+
| customer_name | avg(a.balance) |
+---------------+----------------+
| Hayes | 550.0000 |
+---------------+----------------+
1 row in set (0.00 sec)

mysql> select d.customer_name, avg(a.balance)
-> from depositor d , account a, customer c
-> where d.account_number = a.account_number and d.customer_name = c.customer_name and
c.customer_city='Harrison' group by d.customer_name;
+---------------+----------------+
| customer_name | avg(a.balance) |
+---------------+----------------+
| Hayes | 450.0000 |
| Jones | 750.0000 |
+---------------+----------------+
2 rows in set (0.00 sec)
```

Q) Find the average balance of each customer who lives in Harrison and has at least 3 accounts.

```
mysql> select d.customer_name, avg(a.balance)
-> from depositor d , account a, customer c
-> where d.account_number = a.account_number and d.customer_name = c.customer_name and
c.customer_city='Harrison'
group by d.customer_name having count(distinct d.account_number)>3;
Empty set (0.00 sec)
mysql> (select customer_name
-> from Depositor)
-> union
-> (select customer_name
-> from Borrower);
+---------------+
| customer_name |
+---------------+
| Hayes |
| Johnson |
| Smith |
| Jones |
| Lindsay |
| Turner |
| Majeris |
| Jackson |
| Adams |
| Williams |
```
Vnktrmnb/DBMS LAB/Queries on Banking Schema
```
| McBride |
| Curry |
+---------------+
12 rows in set (0.00 sec)
```

Q) Find all the customers who is having an account and a loan.

```
mysql> select customer_name from depositor where customer_name in(select customer_name from
borrower);
+---------------+
| customer_name |
+---------------+
| Smith |
| Jones |
| Smith |
+---------------+
3 rows in set (0.00 sec)
```

Q) Find all the customers who is having an account but not a loan.

```
mysql> select customer_name from depositor where customer_name not in(select customer_name from
borrower);
+---------------+
| customer_name |
+---------------+
| Hayes |
| Johnson |
| Hayes |
| Johnson |
| Lindsay |
| Turner |
| Majeris |
+---------------+
7 rows in set (0.00 sec)
```

5. JOINS

5. Objective To understand the working of various kinds of join operations in MySQL such as

- Cartesian Product
- Inner Join
  - Self Join
  - Equi join / Natural join
  - Non equi join
- Outer Join
  - Left
  - Right

Database used: emp(empno :int, ename : varchar(20), job : varchar(20), mgr : int, hiredate : date,
Sal : decimal(10,2), comm : decimal (10,2), deptno int)
Dept(deptno : int, dname : varcahar(20), location : varchar(20))
Salgrade(grade : int, losal : int, hisal : int)

Cartesian Product

```
mysql> select ename,loc from emp,dept;
+--------+----------+
| ename  | loc      |
+--------+----------+
| SMITH  | NEW YORK |
| SMITH  | DALLAS   |
| SMITH  | CHICAGO  |
| SMITH  | BOSTON   |
| ALLEN  | NEW YORK |
| ALLEN  | DALLAS   |
| ALLEN  | CHICAGO  |
| ALLEN  | BOSTON   |
| WARD   | NEW YORK |
| WARD   | DALLAS   |
| WARD   | CHICAGO  |
| WARD   | BOSTON   |
| JONES  | NEW YORK |
| JONES  | DALLAS   |
| JONES  | CHICAGO  |
| JONES  | BOSTON   |
| MARTIN | NEW YORK |
| MARTIN | DALLAS   |
| MARTIN | CHICAGO  |
| MARTIN | BOSTON   |
| BLAKE  | NEW YORK |
| BLAKE  | DALLAS   |
| BLAKE  | CHICAGO  |
| BLAKE  | BOSTON   |
| CLARK  | NEW YORK |
| CLARK  | DALLAS   |
| CLARK  | CHICAGO  |
| CLARK  | BOSTON   |
| SCOTT  | NEW YORK |
| SCOTT  | DALLAS   |
| SCOTT  | CHICAGO  |
| SCOTT  | BOSTON   |
| KING   | NEW YORK |
| KING   | DALLAS   |
| KING   | CHICAGO  |
| KING   | BOSTON   |
| TURNER | NEW YORK |
| TURNER | DALLAS   |
| TURNER | CHICAGO  |
| TURNER | BOSTON   |
| ADAMS  | NEW YORK |
| ADAMS  | DALLAS   |
| ADAMS  | CHICAGO  |
| ADAMS  | BOSTON   |
| JAMES  | NEW YORK |
| JAMES  | DALLAS   |
| JAMES  | CHICAGO  |
| JAMES  | BOSTON   |
| FORD   | NEW YORK |
| FORD   | DALLAS   |
| FORD   | CHICAGO  |
| FORD   | BOSTON   |
| MILLER | NEW YORK |
| MILLER | DALLAS   |
| MILLER | CHICAGO  |
| MILLER | BOSTON   |
+--------+----------+
56 rows in set (0.13 sec)
```

Equi -Join

```
mysql> select emp.ename,dept.loc from emp,dept where emp.deptno=dept.deptno;
```

Another way

```
mysql> select e.ename "employee name" ,d.loc "location" from emp e,dept d  where e.deptno=d.deptno;
```

Another Method Equi join

```
mysql> select e.ename,d.loc from emp e join dept d on e.deptno=d.deptno;
```

Another Method using natural join

```
mysql> select e.ename,d.loc from emp e natural join dept d;
```

```
+--------+----------+
| ename  | loc      |
+--------+----------+
| CLARK  | NEW YORK |
| KING   | NEW YORK |
| MILLER | NEW YORK |
| SMITH  | DALLAS   |
| JONES  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| FORD   | DALLAS   |
| ALLEN  | CHICAGO  |
| WARD   | CHICAGO  |
| MARTIN | CHICAGO  |
| BLAKE  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
+--------+----------+
14 rows in set (0.04 sec)
```

Non-Equijoin

```
mysql> select e.empno,e.ename,s.grade from emp e,salgrade s where e.sal between s.losal and s.hisal;
+-------+--------+-------+
| empno | ename  | grade |
+-------+--------+-------+
|  7369 | SMITH  |  1.00 |
|  7499 | ALLEN  |  3.00 |
|  7521 | WARD   |  2.00 |
|  7566 | JONES  |  4.00 |
|  7654 | MARTIN |  2.00 |
|  7698 | BLAKE  |  4.00 |
|  7782 | CLARK  |  4.00 |
|  7788 | SCOTT  |  4.00 |
|  7839 | KING   |  5.00 |
|  7844 | TURNER |  3.00 |
|  7876 | ADAMS  |  1.00 |
|  7900 | JAMES  |  1.00 |
|  7902 | FORD   |  4.00 |
|  7934 | MILLER |  2.00 |
+-------+--------+-------+
14 rows in set (0.00 sec)
```

Note :

```
mysql> select e.ename,d.loc from emp e natural join dept d on e.deptno=d.deptno;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL
server version for the right syntax to use near 'on e.deptno=d.deptno' at line 5
```

Self join

```
mysql> select e1.ename "employee name",e2.ename "Manager name"
    -> from emp e1,emp e2
    -> where e1.mgr=e2.empno;
+---------------+--------------+
| employee name | Manager name |
+---------------+--------------+
| SMITH         | FORD         |
| ALLEN         | BLAKE        |
| WARD          | BLAKE        |
| JONES         | KING         |
| MARTIN        | BLAKE        |
| BLAKE         | KING         |
| CLARK         | KING         |
| SCOTT         | JONES        |
| TURNER        | BLAKE        |
| ADAMS         | SCOTT        |
| JAMES         | BLAKE        |
| FORD          | JONES        |
| MILLER        | CLARK        |
| MILLER        | CLARK        |
| JAMES         | CLARK        |
+---------------+--------------+
```

Outer Joins
Left Join

```
mysql> select e.ename,d.loc from emp e left join dept d on e.deptno=d.deptno;
+--------+----------+
| ename  | loc      |
+--------+----------+
| CLARK  | NEW YORK |
| KING   | NEW YORK |
| MILLER | NEW YORK |
| SMITH  | DALLAS   |
| JONES  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| FORD   | DALLAS   |
| ALLEN  | CHICAGO  |
| WARD   | CHICAGO  |
| MARTIN | CHICAGO  |
| BLAKE  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
+--------+----------+
14 rows in set (0.00 sec)
```

Right join

```
mysql> select e.ename,d.loc from emp e right join dept d on e.deptno=d.deptno;
+--------+----------+
| ename  | loc      |
+--------+----------+
| CLARK  | NEW YORK |
| KING   | NEW YORK |
| MILLER | NEW YORK |
| SMITH  | DALLAS   |
| JONES  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| FORD   | DALLAS   |
| ALLEN  | CHICAGO  |
| WARD   | CHICAGO  |
| MARTIN | CHICAGO  |
| BLAKE  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
| NULL   | BOSTON   |
+--------+----------+
15 rows in set (0.00 sec)

mysql> select e.ename,d.loc from emp e full join dept d on e.deptno=d.deptno;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL
server version for the right syntax to use near 'full join dept d on e.deptno=d.deptno' at line 3
```

Full join

```
mysql> (select e.ename,d.loc from emp e left join dept d on e.deptno=d.deptno)
        union
        (select e.ename,d.loc from emp e right join dept d on e.deptno=d.deptno);
+--------+----------+
| ename  | loc      |
+--------+----------+
| CLARK  | NEW YORK |
| KING   | NEW YORK |
| MILLER | NEW YORK |
| SMITH  | DALLAS   |
| JONES  | DALLAS   |
| SCOTT  | DALLAS   |
| ADAMS  | DALLAS   |
| FORD   | DALLAS   |
| ALLEN  | CHICAGO  |
| WARD   | CHICAGO  |
| MARTIN | CHICAGO  |
| BLAKE  | CHICAGO  |
| TURNER | CHICAGO  |
| JAMES  | CHICAGO  |
| NULL   | BOSTON   |
+--------+----------+
15 rows in set (0.01 sec)
```

6. DCL

6. Objective To understand the syntax of DCL commands such as

- Grant
- Revoke

Database used: Table name : student(rollno : int , name : char(30))

```
mysql --host=127.0.0.1 --user=root

mysql> select user();
+----------------+
| user()         |
+----------------+
| root@localhost |
+----------------+
1 row in set (0.00 sec)

mysql> create user 'varun'@'localhost'  identified by 'abcde';
Query OK, 0 rows affected (0.02 sec)

mysql> create database mydb;
Query OK, 1 row affected (0.03 sec)

mysql> use mydb;
Database changed

mysql> create table student(rollno int,name char(30) not  null,primary key(rollno));
Query OK, 0 rows affected (0.40 sec

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| maheshdb           |
| mydb               |
| mysql              |
| new                |
| performance_schema |
| phpmyadmin         |
| rahuldb            |
| store              |
| test                    |
+--------------------+
10 rows in set (0.00 sec)

mysql> grant select on mydb.student to 'varun'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql --host=127.0.0.1 --user=varun --password=abcde
```

```
mysql> show databases;

+--------------------+
| Database           |
+--------------------+
| information_schema |
| mydb               |
| test               |
+--------------------+
3 rows in set (0.00 sec)

mysql> select * from mydb.student;
Empty set (0.00 sec)

mysql> delete from mydb.student;
ERROR 1142 (42000): DELETE command denied to user 'varun'@'localhost' for table 'student'

mysql --host=127.0.0.1 --user=root

mysql> revoke select on mydb.student from 'varun'@'localhost';
Query OK, 0 rows affected (0.00 sec)

mysql --host=127.0.0.1 --user=varun --password=abcde

mysql> select * from mydb.student;
ERROR 1142 (42000): SELECT command denied to user 'varun'@'localhost' for table 'student'
```

7. Views

7. Objective To understand the syntax of view
- Create view
- Modification of views

Database used:Books,Customer,Loan,Branch

```
mysql> create table books(isbn int,title char(20),author char(20),cost float,primary key(isbn));
Query OK, 0 rows affected (0.69 sec)

mysql> desc books;
+--------+----------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+--------+----------+------+-----+---------+-------+
| isbn | int(11) | NO | PRI | NULL | |
| title | char(20) | YES | | NULL | |
| author | char(20) | YES | | NULL | |
| cost | float | YES | | NULL | |
+--------+----------+------+-----+---------+-------+
4 rows in set (0.01 sec)

mysql> insert into books values(101,'C','Preetham',200);
Query OK, 1 row affected (0.13 sec)
mysql> insert into books values(102,'CPP','Pranav',300);
Query OK, 1 row affected (0.07 sec)
mysql> insert into books values(103,'CPP','Ganesh',400);
Query OK, 1 row affected (0.07 sec)
mysql> insert into books values(104,'JAVA','Ganesh',400);
Query OK, 1 row affected (0.09 sec)

mysql> select * from books;
+------+-------+----------+------+
| isbn | title | author | cost |
+------+-------+----------+------+
| 101 | C | Preetham | 200 |
| 102 | CPP | Pranav | 300 |
| 103 | CPP | Ganesh | 400 |
| 104 | JAVA | Ganesh | 400 |
+------+-------+----------+------+
4 rows in set (0.00 sec)

mysql> create view mybooks as select isbn,title,cost from books;
Query OK, 0 rows affected (0.10 sec)

mysql> select *from mybooks;
+------+-------+------+
| isbn | title | cost |
+------+-------+------+
| 101 | C | 200 |
| 102 | CPP | 300 |
| 103 | CPP | 400 |
| 104 | JAVA | 400 |
+------+-------+------+
4 rows in set (0.01 sec)
mysql> insert into mybooks values(107,'VB',350);
Query OK, 1 row affected (0.08 sec)

mysql> select * from mybooks;
```

```
+------+-------+------+
| isbn | title | cost |
+------+-------+------+
| 101 | C | 200 |
| 102 | CPP | 300 |
| 103 | CPP | 400 |
| 104 | JAVA | 400 |
| 107 | VB | 350 |
+------+-------+------+
5 rows in set (0.00 sec)

mysql> select * from books;
+------+-------+----------+------+
| isbn | title | author | cost |
+------+-------+----------+------+
| 101 | C | Preetham | 200 |
| 102 | CPP | Pranav | 300 |
| 103 | CPP | Ganesh | 400 |
| 104 | JAVA | Ganesh | 400 |
| 107 | VB | NULL | 350 |
+------+-------+----------+------+
5 rows in set (0.00 sec)

mysql> delete from books where isbn=101;
Query OK, 1 row affected (0.11 sec)

mysql> select * from books;
+------+-------+--------+------+
| isbn | title | author | cost |
+------+-------+--------+------+
| 102 | CPP | Pranav | 300 |
| 103 | CPP | Ganesh | 400 |
| 104 | JAVA | Ganesh | 400 |
| 107 | VB | NULL | 350 |
+------+-------+--------+------+
4 rows in set (0.00 sec)

mysql> select *from mybooks;
+------+-------+------+
| isbn | title | cost |
+------+-------+------+
| 102 | CPP | 300 |
| 103 | CPP | 400 |
| 104 | JAVA | 400 |
| 107 | VB | 350 |
+------+-------+------+
4 rows in set (0.00 sec)
```

```
mysql> update mybooks set title='RDBMS' where isbn=107;
Query OK, 1 row affected (0.09 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

```
mysql> select * from mybooks;
+------+-------+------+
| isbn | title | cost |
+------+-------+------+
| 102 | CPP | 300 |
| 103 | CPP | 400 |
| 104 | JAVA | 400 |
| 107 | RDBMS | 350 |
+------+-------+------+
4 rows in set (0.00 sec)

mysql> select * from books;
+------+-------+--------+------+
| isbn | title | author | cost |
+------+-------+--------+------+
| 102 | CPP | Pranav | 300 |
| 103 | CPP | Ganesh | 400 |
| 104 | JAVA | Ganesh | 400 |
| 107 | RDBMS | NULL | 350 |
+------+-------+--------+------+
4 rows in set (0.00 sec)

mysql> drop view mybooks;
Query OK, 0 rows affected (0.00 sec)
mysql> select * from mybooks;
ERROR 1146 (42S02): Table 'empdeptdb.mybooks' doesn't exist

mysql> select * from books;
+------+-------+--------+------+
| isbn | title | author | cost |
+------+-------+--------+------+
| 102 | CPP | Pranav | 300 |
| 103 | CPP | Ganesh | 400 |
| 104 | JAVA | Ganesh | 400 |
| 107 | RDBMS | NULL | 350 |
+------+-------+--------+------+
4 rows in set (0.00 sec)


mysql> select * from customer;
+---------------+-----------------+---------------+
| customer_name | customer_street | customer_city |
+---------------+-----------------+---------------+
| Adams | Spring | Pittsfield |
| Brooks | Senator | Brooklyn |
| Curry | North | Rye |
| Glenn | Sand Hill | Woodside |
| Green | Walnut | Stamford |
| Hayes | Main | Harrison |
| Jackson | University | Salt Lake |
| Johnson | Alma | Palo Alto |
| Jones | Main | Harrison |
| Lindsay | Park | Pittsfield |
| Majeris | First | Rye |
| McBride | Safety | Rye |
| Smith | Main | Rye |
| Turner | Putnam | Stamford |
| Williams | Nassau | Princeton |
+---------------+-----------------+---------------+
15 rows in set (0.00 sec)
```

```
mysql> select * from loan;
+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| L-11 | Round Hill | 900 |
| L-14 | Downtown | 1500 |
| L-15 | Perryridge | 1500 |
| L-16 | Perryridge | 1300 |
| L-17 | Downtown | 1000 |
| L-20 | North Town | 7500 |
| L-21 | Central | 570 |
| L-23 | Redwood | 2000 |
| L-93 | Mianus | 500 |
+-------------+-------------+--------+
9 rows in set (0.00 sec)
mysql> select *from borrower;
+---------------+-------------+
| customer_name | loan_number |
+---------------+-------------+
| Smith | L-11 |
| Jackson | L-14 |
| Adams | L-16 |
| Jones | L-17 |
| Williams | L-17 |
| McBride | L-20 |
| Smith | L-21 |
| Curry | L-93 |
+---------------+-------------+
8 rows in set (0.00 sec)

mysql> create view loan_branch as select loan_number,branch_name from loan;
Query OK, 0 rows affected (0.06 sec)
mysql> select * from loan_branch;
+-------------+-------------+
| loan_number | branch_name |
+-------------+-------------+
| L-21 | Central |
| L-14 | Downtown |
| L-17 | Downtown |
| L-93 | Mianus |
| L-20 | North Town |
| L-15 | Perryridge |
| L-16 | Perryridge |
| L-23 | Redwood |
| L-11 | Round Hill |
+-------------+-------------+
9 rows in set (0.02 sec)
mysql> select * from loan;
+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| L-11 | Round Hill | 900 |
| L-14 | Downtown | 1500 |
| L-15 | Perryridge | 1500 |
| L-16 | Perryridge | 1300 |
| L-17 | Downtown | 1000 |
| L-20 | North Town | 7500 |
| L-21 | Central | 570 |
| L-23 | Redwood | 2000 |
| L-93 | Mianus | 500 |
+-------------+-------------+--------+
9 rows in set (0.00 sec)
mysql> insert into loan_branch values('L-37','Perryridge');
ERROR 1423 (HY000): Field of view 'bankingdb.loan_branch' underlying table doesn't have a default
value
mysql> insert into loan values('L-37','Perryridge',null);
ERROR 1048 (23000): Column 'amount' cannot be null

mysql> create view loan_info as select b.customer_name,l.loan_number from borrower b,loan l where
b.loan_number=l.loan_number;
Query OK, 0 rows affected (0.06 sec)

mysql> select * from loan_info;
+---------------+-------------+
| customer_name | loan_number |
+---------------+-------------+
| Smith | L-11 |
| Jackson | L-14 |
| Adams | L-16 |
| Jones | L-17 |
| Williams | L-17 |
| McBride | L-20 |
| Smith | L-21 |
| Curry | L-93 |
+---------------+-------------+
8 rows in set (0.05 sec)
mysql> insert into loan_info values('Johnson',1900);
ERROR 1394 (HY000): Can not insert into join view 'bankingdb.loan_info' without fields list

mysql> create view branch_total
-> as
-> select branch_name ,sum(amount)
-> from loan
-> group by branch_name;
Query OK, 0 rows affected (0.06 sec)
mysql> select * from branch_total;
+-------------+-------------+
| branch_name | sum(amount) |
+-------------+-------------+
| Central | 570 |
| Downtown | 2500 |
| Mianus | 500 |
| North Town | 7500 |
| Perryridge | 2800 |
| Redwood | 2000 |
| Round Hill | 900 |
+-------------+-------------+
7 rows in set (0.08 sec)
mysql> insert into branch_total values('Nadergul',3000);
ERROR 1471 (HY000): The target table branch_total of the INSERT is not insertable-into
```

Numeric Functions	String Functions	Date and Time Functions
ABS	ASCII, BIN	ADDDATE, ADDTIME
ACOS	BINARY OPERATOR	CONVERT_TZ
ASIN	BIT_LENGTH	CURDATE, CURRENT_DATE
ATAN	CAST, CHAR FUNCTION	CURRENT_TIME
ATAN2	HARACTER_LENGTH	URRENT_TIMESTAMP
CEIL	CHAR_LENGTH	CURTIME, DATE FUNCTION
CEILING	CONCAT, CONCAT_WS	DATEDIFF, DATE_ADD
CONV	CONVERT, ELT	DATE_FORMAT, DATE_SUB
COS	EXPORT_SET	DAY, DAYNAME
COT	EXTRACTVALUE	DAYOFMONTH, DAYOFWEEK
CRC32	FIELD, FIND_IN_SET	DAYOFYEAR, EXTRACT

8. Built-in Functions

8. Objective To understand the use of functions in MySQL queries

- Numeric Functions
- String Functions
- Date and Time Functions

DEGREES DIV EXP FLOOR LN LOG LOG10 LOG2 MOD PI POW POWER RADIANS RAND ROUND SIGN SIN SQRT TAN TRUNCATE	FORMAT, FROM_BASE64 HEX, INSERT FUNCTION INSTR, LCASE LEFT, LENGTH LIKE, LOAD_FILE LOCATE, LOWER LPAD, LTRIM MAKE_SET MATCH AGAINST MID, NOT LIKE NOT REGEXP OCT, OCTET_LENGTH ORD, POSITION QUOTE, REGEXP REPEAT FUNCTION REPLACE FUNCTION REVERSE, RIGHT RPAD, RTRIM SOUNDEX, SOUNDS LIKE SPACE, STRCMP SUBSTR, SUBSTRING SUBSTRING_INDEX TO_BASE64 TRIM, UCASE UNHEX, UPDATEXML UPPER, WEIGHT_STRING	FROM_DAYS, FROM_UNIXTIME GET_FORMAT HOUR, LAST_DAY LOCALTIME LOCALTIMESTAMP MAKEDATE, MAKETIME MICROSECOND MINUTE, MONTH MONTHNAME NOW, PERIOD_ADD PERIOD_DIFF QUARTER, SECOND SEC_TO_TIME STR_TO_DATE SUBDATE, SUBTIME SYSDATE, TIME FUNCTION TIMEDIFF TIMESTAMP FUNCTION TIMESTAMPADD TIMESTAMPDIFF TIME_FORMAT TIME_TO_SEC TO_DAYS, TO_SECONDS UNIX_TIMESTAMP UTC_DATE, UTC_TIME UTC_TIMESTAMP WEEK, WEEKDAY WEEKOFYEAR YEAR, YEARWEEK

```
mysql> select abs(-5),abs(0),abs(5);

+---------+--------+--------+
| abs(-5) | abs(0) | abs(5) |
+---------+--------+--------+
|       5 |      0 |      5 |
+---------+--------+--------+
1 row in set (0.03 sec)

mysql> select ceil(-5.6),ceil(0),ceil(-5.6);
+------------+---------+------------+
| ceil(-5.6) | ceil(0) | ceil(-5.6) |
+------------+---------+------------+
|         -5 |       0 |         -5 |
+------------+---------+------------+
1 row in set (0.00 sec)

mysql> select floor(-5.6),floor(0),floor(-5.6);
+-------------+----------+-------------+
| floor(-5.6) | floor(0) | floor(-5.6) |
+-------------+----------+-------------+
|          -6 |        0 |          -6 |
+-------------+----------+-------------+
1 row in set (0.00 sec)

mysql> select ln(10),log(10),exp(3),power(2,3);
+-------------------+-------------------+--------------------+------------+
| ln(10)            | log(10)           | exp(3)             | power(2,3) |
+-------------------+-------------------+--------------------+------------+
| 2.302585092994046 | 2.302585092994046 | 20.085536923187668 |          8 |
+-------------------+-------------------+--------------------+------------+
1 row in set (0.03 sec)

mysql> select pow(2,3),sqrt(35) ;
+----------+-------------------+
| pow(2,3) | sqrt(35)          |
+----------+-------------------+
|        8 | 5.916079783099616 |
+----------+-------------------+
1 row in set (0.00 sec)

mysql> select round(5.6789,2),truncate(5.6789,2);
+-----------------+--------------------+
| round(5.6789,2) | truncate(5.6789,2) |
+-----------------+--------------------+
|            5.68 |               5.67 |
+-----------------+--------------------+
1 row in set (0.00 sec)
mysql> select lower('HELLO'),upper('hello'),length('hello'),reverse('hello');
+----------------+----------------+-----------------+------------------+
| lower('HELLO') | upper('hello') | length('hello') | reverse('hello') |
+----------------+----------------+-----------------+------------------+
| hello          | HELLO          |               5 | olleh            |
+----------------+----------------+-----------------+------------------+
1 row in set (0.03 sec)


mysql> select curdate();
+------------+
| curdate()  |
+------------+
| 2016-10-03 |
+------------+
1 row in set (0.02 sec)

mysql> select last_day('2016-10-03'),datediff('2016-10-03','2016-08-03');
+------------------------+-------------------------------------+
| last_day('2016-10-03') | datediff('2016-10-03','2016-08-03') |
+------------------------+-------------------------------------+
| 2016-10-31             |                                  61 |
+------------------------+-------------------------------------+
1 row in set (0.02 sec)
```

9. MySQL Table Locking

In this experiment, you will learn how to use MySQL locking for cooperating table access between sessions.

The simple form of acquiring a lock for a table is as follows:

LOCK TABLES table_name [READ | WRITE]

We will go into detail of each lock type in this section.

To release a lock for a table, you use the following statement:

UNLOCK TABLES;

```
mysql> SELECT CONNECTION_ID();
+-----------------+
| connection_id() |
+-----------------+
|               2 |
+-----------------+
1 row in set (0.00 sec)
mysql> INSERT INTO books VALUES(112233,'DBMS','korth');
Query OK, 1 row affected (0.08 sec)
```

After that, to acquire a lock, you use the LOCK TABLE statement.

```
mysql> LOCK TABLE books READ;
Query OK, 0 rows affected (0.00 sec)
```

Finally, in the same session, if you try to insert a new row into the books table, you will get an error message.

```
mysql> INSERT INTO books VALUES(112244,'Data base System','Sam');
ERROR 1099 (HY000): Table 'books' was locked with a READ lock and can't be updated
```

So the once READ lock is acquired, you cannot write data into the table within the same session. Let’s check the READ lock from a different session.

First, connect to the MYDB and check the connection id:

```
mysql> USE mydb;
Database changed

mysql> SELECT CONNECTION_ID();
+-----------------+
| connection_id() |
+-----------------+
|               3 |
+-----------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM books;
+--------+-------+--------+
| isbn   | title | author |
+--------+-------+--------+
| 112233 | DBMS  | korth  |
+--------+-------+--------+
1 row in set (0.00 sec)
```

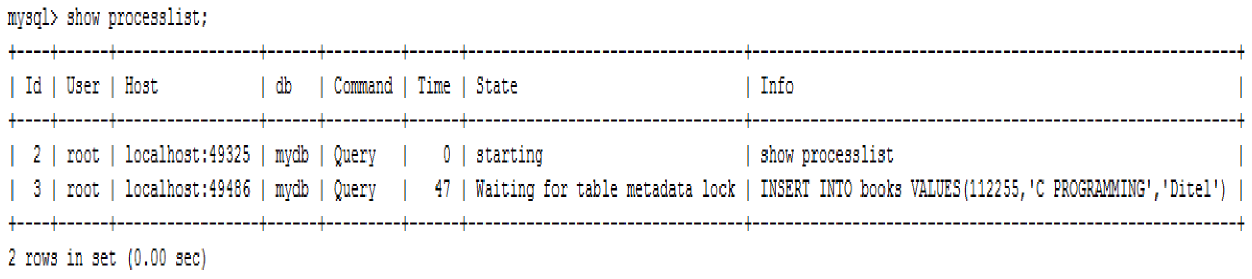
Next, insert a new row into the books table from the second session.

```
mysql> INSERT INTO books VALUES(112255,'C PROGRAMMING','Ditel');--waiting
Query OK, 1 row affected (1 min 15.78 sec)
```

The insert operation from the second session is in the waiting state because a READ lock already acquired on the books table by the first session and it has not released yet.

You can see the detailed information from the SHOW PROCESSLIST statement.

```
mysql> SHOW PROCESSLIST;
```



```
mysql> show processlist;
+----+------+-----------------+------+---------+------+---------------------------------+----------------------------------------------------------+
| Id | User | Host            | db   | Command | Time | State                           | Info                                                     |
+----+------+-----------------+------+---------+------+---------------------------------+----------------------------------------------------------+
|  2 | root | localhost:49325 | mydb | Query   |    0 | starting                        | show processlist                                         |
|  3 | root | localhost:49486 | mydb | Query   |   47 | Waiting for table metadata lock | INSERT INTO books VALUES(112255,'C PROGRAMMING','Ditel') |
+----+------+-----------------+------+---------+------+---------------------------------+----------------------------------------------------------+
2 rows in set (0.00 sec)
```

After that, go back to the first session and release the lock by using the UNLOCK TABLES statement.
After you release the READ lock from the first session, the INSERT operation in the second session executed.
Finally, check it the data of the books table to see if the INSERT operation from the second session really executed.

mysql> **UNLOCK TABLES;**
Query OK, 0 rows affected (0.00 sec)

```
mysql> INSERT INTO books VALUES(112255,'C PROGRAMMING','Ditel');
Query OK, 1 row affected (1 min 15.78 sec)
```

MySQL table locking for WRITE

The table lock for WRITE has the following features:

1. Only session that holds the lock of a table can read and write data from the table.
2. Other sessions cannot read and write from the table until the WRITE lock is released.

Let’s go into detail to see how the WRITE lock works.
First, acquire a WRITE lock from the first session.

mysql> **LOCK TABLE books WRITE;**

```
mysql> SELECT * FROM books;
+--------+---------------+--------+
| isbn   | title         | author |
+--------+---------------+--------+
| 112233 | DBMS          | korth  |
| 112255 | C PROGRAMMING | Ditel  |
+--------+---------------+--------+
2 rows in set (0.00 sec)
```

Then, insert a new row into the books table.

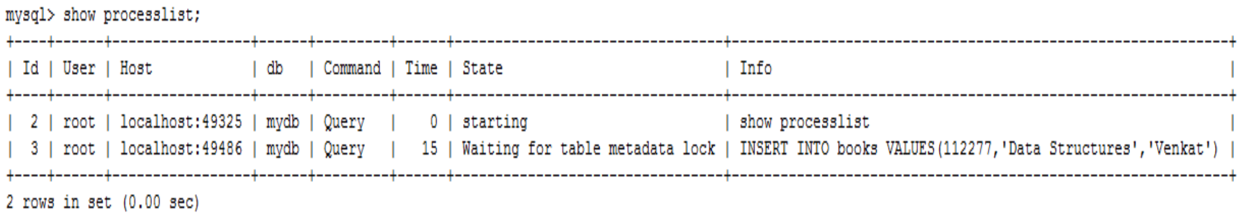
```
mysql> INSERT INTO books VALUES(112266,'C/C++','Ditel');
Query OK, 1 row affected (0.07 sec)
```

It works.
Next, read data from the books table.
It is fine.

After that, from the second session, try to write and read data:

mysql> INSERT INTO books VALUES(112255,'C PROGRAMMING','Ditel');
mysql> SELECT * FROM books;

MySQL puts those operations into a waiting state. You can check it using the SHOW PROCESSLIST statement.
mysql> SHOW PROCESSLIST;



```
mysql> show processlist;
+----+------+-----------------+------+---------+------+---------------------------------+--------------------------------------------------------------+
| Id | User | Host            | db   | Command | Time | State                           | Info                                                         |
+----+------+-----------------+------+---------+------+---------------------------------+--------------------------------------------------------------+
|  2 | root | localhost:49325 | mydb | Query   |    0 | starting                        | show processlist                                             |
|  3 | root | localhost:49486 | mydb | Query   |   15 | Waiting for table metadata lock | INSERT INTO books VALUES(112277,'Data Structures','Venkat') |
+----+------+-----------------+------+---------+------+---------------------------------+--------------------------------------------------------------+
2 rows in set (0.00 sec)
```

Finally, release the lock from the first session.

mysql> UNLOCK TABLES;
Query OK, 0 rows affected (0.00 sec)
You will see all pending operations from the second session executed.

PLSQL

10. SUB PROGRAMS

9. Objective To understand PL/SQL syntax for implementing

- Procedures

- Functions

Database : EMP(empno :int, ename : varchar(20), job : varchar(20), mgr : int, hiredate : date,
Sal : decimal(10,2), comm : decimal (10,2), deptno int)

9.1. Procedure to print digit in words

```
DROP PROCEDURE IF EXISTS digitName //
CREATE PROCEDURE digitName(IN x INT)
BEGIN

  DECLARE result VARCHAR(20);

 CASE x
WHEN 0 THEN SET result = 'Zero';
WHEN 1 THEN SET result = 'One';
 WHEN 2 THEN SET result = 'Two';
WHEN 3 THEN SET result = 'Three';
WHEN 4 THEN SET result = 'Four';
WHEN 5 THEN SET result = 'Five';
 WHEN 6 THEN SET result = 'Six';
WHEN 7 THEN SET result = 'Seven';
 WHEN 8 THEN SET result = 'Eight';
WHEN 9 THEN SET result = 'Nine';
 ELSE SET result = 'Not a digit';
END CASE;

 SELECT x AS Digit, result AS Name;

END;
//

CALL digitName(0) //
CALL digitName(4) //
CALL digitName(100) //

+-------+------+
| Digit | Name |
+-------+------+
|     0 | Zero |
+-------+------+
1 row in set (0.00 sec)

+-------+------+
| Digit | Name |
+-------+------+
|     4 | Four |
+-------+------+
1 row in set (0.00 sec)
```

```
+-------+-------------+
| Digit | Name        |
+-------+-------------+
|   100 | Not a digit |
+-------+-------------+
1 row in set (0.00 sec)
```

9.2. Procedure to find sign of a number

```
DROP PROCEDURE IF EXISTS mySign //
CREATE PROCEDURE mySign(IN x INT)
BEGIN
  IF x > 0 THEN
    SELECT x AS Number, '+' AS Sign;
  ELSEIF x < 0 THEN
    SELECT x AS Number, '-' AS Sign;
  ELSE
    SELECT x AS Number, 'Zero' AS Sign;
  END IF;
END;
//

CALL mySign(2) //
CALL mySign(-5) //
CALL mySign(0) //

+--------+------+
| Number | Sign |
+--------+------+
|      2 | +    |
+--------+------+
1 row in set (0.00 sec)

+--------+------+
| Number | Sign |
+--------+------+
|     -5 | -    |
+--------+------+
1 row in set (0.00 sec)

+--------+------+
| Number | Sign |
+--------+------+
|      0 | Zero |
+--------+------+
1 row in set (0.00 sec)
```

9.3. Procedure to compute factorial of given number

```
DROP PROCEDURE IF EXISTS fact //
CREATE PROCEDURE fact(IN x INT)
BEGIN

  DECLARE result INT DEFAULT 1;  /* notice you can declare a variable*/
  DECLARE i INT DEFAULT 1;       /* and give it a value in one line */

  REPEAT
    SET result = result * i;
    SET i = i + 1;
  UNTIL i > x
  END REPEAT;

  SELECT x AS Number, result as Factorial;

END;
//

CALL fact(1) //
CALL fact(2) //
CALL fact(4) //
CALL fact(0) //

+--------+-----------+
| Number | Factorial |
+--------+-----------+
|      1 |         1 |
+--------+-----------+
1 row in set (0.00 sec)

+--------+-----------+
| Number | Factorial |
+--------+-----------+
|      2 |         2 |
+--------+-----------+
1 row in set (0.00 sec)

+--------+-----------+
| Number | Factorial |
+--------+-----------+
|      4 |        24 |
+--------+-----------+
1 row in set (0.00 sec)

+--------+-----------+
| Number | Factorial |
+--------+-----------+
|      0 |         1 |
+--------+-----------+
1 row in set (0.00 sec)
```

9.4. Function to compute simple interest

```
mysql> DELIMITER //
mysql> CREATE FUNCTION ptr( p float,t int,r float )
    -> RETURNS float
    -> BEGIN
    ->    DECLARE interest float;
    ->    SET interest =(p*t*r)/100;
    ->    RETURN interest ;
    -> END; //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> select ptr(4000,5,0.05);
+------------------+
| ptr(4000,5,0.05) |
+------------------+
|               10 |
+------------------+
1 row in set (0.00 sec)
```

9.5. Function to print the number in reverse

```
mysql> DELIMITER //
mysql> CREATE FUNCTION reverseNumber( n int )
    -> RETURNS int
    -> BEGIN
    ->    DECLARE r int;
    ->    DECLARE s int;
    ->    SET s = 0;
    ->   WHILE n>0 DO
    ->      SET r =mod(n,10);
    ->      SET s =s*10+r;
    ->      SET n =truncate(n/10,0);
    ->    END WHILE ;
    ->    RETURN s;
    -> END; //
Query OK, 0 rows affected (0.00 sec)

mysql> select reverseNumber(123);

+--------------------+
| reverseNumber(123) |
+--------------------+
|                321 |
+--------------------+
1 row in set (0.00 sec)
```

9.6. Function to getname of employee of the given empno;

```
mysql> DELIMITER //
mysql> CREATE FUNCTION getName( eid  int )
    -> RETURNS varchar(20)
    -> BEGIN
    -> DECLARE name varchar(20);
    -> select ename into name from emp where empno=eid;
    -> RETURN name;
    -> END; //
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> select getName(1);
+------------+
| getName(1) |
+------------+
| saketh     |
+------------+
1 row in set (0.00 sec)
```

11. CURSORS

Objective: To understand PL/SQL syntax for implementing cursor
Find the names and their jobs who are working in the particular department.

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE get_empdetails_using_cursor(
-> IN deptid INT
-> )
-> BEGIN
-> DECLARE record_not_found INTEGER DEFAULT 0;
-> DECLARE namee VARCHAR(20) DEFAULT "";
-> DECLARE jobe VARCHAR(10) DEFAULT "";
-> DECLARE c CURSOR FOR
-> SELECT ename,job FROM emp
-> WHERE deptno = deptid;
-> DECLARE CONTINUE HANDLER FOR NOT FOUND SET record_not_found = 1;
-> OPEN c;
-> edetails_loop: LOOP
-> FETCH c INTO namee,jobe;
-> IF record_not_found THEN
-> LEAVE edetails_loop;
-> END IF;
-> SELECT namee,jobe;
-> END LOOP edetails_loop;
-> CLOSE c;
-> END //
Query OK, 0 rows affected (0.00 sec)
mysql> DELIMITER ;
mysql> call get_empdetails_using_cursor(10);
+-------+---------+
| namee | jobe |
+-------+---------+
| CLARK | MANAGER |
+-------+---------+
1 row in set (0.01 sec)
+-------+-----------+
| namee | jobe |
+-------+-----------+
| KING | PRESIDENT |
+-------+-----------+
1 row in set (0.01 sec)
+--------+-------+
| namee | jobe |
+--------+-------+
| MILLER | CLERK |
+--------+-------+
1 row in set (0.01 sec)
+--------+-------+
| namee | jobe |
+--------+-------+
| MILLER | CLERK |
+--------+-------+
1 row in set (0.02 sec)
Query OK, 0 rows affected (0.02 sec)
```

12. TRIGGER

Objective: To understand PL/SQL syntax for implementing trigger

Create a trigger that may insert a tuple into T5 when a tuple is inserted into T4. Specifically, the trigger checks whether the new tuple has a first component 10 or less, and if so inserts the reverse tuple into T5

```
mysql> create  table t4(a int,b char);
Query OK, 0 rows affected (0.28 sec)
mysql> create table t5(c char,d int);
Query OK, 0 rows affected (0.25 sec)

mysql> delimiter //
mysql> create trigger trig45
 after insert on t4
 for each row
begin
 if new.a>25 then
insert into t5 values(new.b,new.a);
end if;
end;
  //
Query OK, 0 rows affected (0.08 sec)

mysql> insert into t4 values(27,'x');

Query OK, 1 row affected (0.06 sec)

mysql> select * from t4;

+------+------+
| a    | b    |
+------+------+
|   27 | x    |
+------+------+
1 row in set (0.00 sec)

mysql> select * from t5;
+------+------+
| c    | d    |
+------+------+
| x    |   27 |
+------+------+
1 row in set (0.00 sec)

mysql> insert into t4 values(10,'y');
 Query OK, 1 row affected (0.04 sec)

mysql> select * from t4;
+------+------+
| a    | b    |
+------+------+
|   27 | x    |
|   10 | y    |
+------+------+
2 rows in set (0.00 sec)
mysql> select * from t5;
+------+------+
| c    | d    |
+------+------+
| x    |   27 |
+------+------+
1 row in set (0.00 sec)

mysql> drop trigger trig45;

Query OK, 0 rows affected (0.00 sec)
```

Create a Trigger to ensure the salary of the employee is not decreased.

```
mysql>create table t6(eno int,sal int);
Query OK, 0 rows affected (0.00 sec)

mysql>insert into t6 values(101,15000);
mysql>insert into t6 values(102,10000);

Query OK, 0 rows affected (0.00 sec)

mysql> create trigger trg_emp_sal_check before update on t6 for each row
begin
   if new.sal<=old.sal then
           set new.sal=old.sal;
   end if;
 end
 //
Query OK, 0 rows affected (0.08 sec)

mysql> Update t6 set sal=14000;

mysql>select * from t6;
```

show results

13. Transaction control language (TCL)

Objective: To understand the syntax of TCL commands such as

- Commit
- Savepoint
- Rollback

Database used: books(isbn : int, title : varchar(20),author : varchar(20))

```
mysql> create table books(isbn int,
    -> title varchar(20) not null,
    -> author varchar(20) not null,
    -> primary key(isbn));
Query OK, 0 rows affected (0.35 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set autocommit=0;
Query OK, 0 rows affected (0.00 sec)

mysql> insert into books values(112233,'Database Design','Sam');
Query OK, 1 row affected (0.01 sec)

mysql> insert into books values(112244,'Database System','Korth');
Query OK, 1 row affected (0.00 sec)

mysql> select * from books;
+--------+-----------------+--------+
| isbn   | title           | author |
+--------+-----------------+--------+
| 112233 | Database Design | Sam    |
| 112244 | Database System | Korth  |
+--------+-----------------+--------+
2 rows in set (0.00 sec)

mysql> savepoint p;
Query OK, 0 rows affected (0.00 sec)

mysql> delete from books;
Query OK, 2 rows affected (0.02 sec)

mysql> select * from books;
Empty set (0.00 sec)

mysql> rollback to p;
Query OK, 0 rows affected (0.02 sec)

mysql> select * from books;
+--------+-----------------+--------+
| isbn   | title           | author |
+--------+-----------------+--------+
| 112233 | Database Design | Sam    |
| 112244 | Database System | Korth  |
+--------+-----------------+--------+
2 rows in set (0.00 sec)
```

Show commit

JDBC PROGRAM

Objective: To understand JDBC API uses jdbc drivers to connect and execute query with the database.

Steps involved in building a JDBC application

1. Import the packages
2. Register the JDBC driver
3. Open a connection:

4. Execute a query:
5. Extract data from result set
6. Clean up the environment

Java program to connect to mysql database and printing the details of all employees present in EMP table

MysqlCon.java

```
import java.sql.*;

class MysqlCon{
public static void main(String args[]){
try{

Class.forName("com.mysql.jdbc.Driver");

Connection con=DriverManager.getConnection("jdbc:mysql://localhost:3306/mydb ","root","pwd");
//here mydb is database name, root is username and pwd is password

Statement stmt=con.createStatement();

ResultSet rs=stmt.executeQuery("select * from emp");
while(rs.next())
System.out.println(rs.getInt(1)+"  "+rs.getString(2)+"  "+rs.getString(3));

con.close();

}catch(Exception e){ System.out.println(e);}

}
}
```

Appendix-A

Database Application

Tables used in this note: Sailors(<u>sid: integer</u>, sname: string, rating: integer, age: real);
Boats(<u>bid: integer</u>, bname: string, color: string);
Reserves(<u>sid: integer</u>, <u>bid: integer</u>, <u>day: date</u>).

```
mysql> create table sailors(
   ->        sid integer,
   ->        sname varchar(30),
   ->        rating integer,
   ->        age real,
   ->        primary key(sid)
   ->        );
Query OK, 0 rows affected (0.39 sec)

mysql> create table boats(
   ->        bid integer,
   ->        bname varchar(30),
   ->        color varchar(10),
   ->        primary key(bid)
   ->        );
Query OK, 0 rows affected (0.31 sec)
mysql> create table reserves(
   ->        sid integer,
   ->        bid integer,
   ->        day date,
   ->        primary key(sid,bid,day),
   ->        foreign key(sid) references sailors(sid),
   ->        foreign key(bid) references boats(bid) );
 Query OK, 0 rows affected (0.32 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (22, 'Dustin', 7, 45.0);
Query OK, 1 row affected (0.08 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (29, 'Brutus', 1, 33.0);
Query OK, 1 row affected (0.07 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (31, 'Lubber', 8, 55.5);
Query OK, 1 row affected (0.15 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (32, 'Andy', 8, 25.5);
Query OK, 1 row affected (0.09 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (58, 'Rusty', 10, 35.0);
Query OK, 1 row affected (0.05 sec)

mysql> insert into sailors (sid, sname, rating, age)
   ->        values (64, 'Horatio', 7, 35.0);
Query OK, 1 row affected (0.03 sec)
mysql> insert into sailors (sid, sname, rating, age)
   ->        values (71, 'Zorba', 10, 16.0);
Query OK, 1 row affected (0.05 sec)
```

```
mysql> insert into sailors (sid, sname, rating, age)
    ->        values (74, 'Horatio', 9, 35.0);
Query OK, 1 row affected (0.03 sec)

mysql> insert into sailors (sid, sname, rating, age)
    ->        values (85, 'Art', 3, 25.5);
Query OK, 1 row affected (0.04 sec)

mysql> insert into sailors (sid, sname, rating, age)
    ->        values (95, 'Bob', 3, 63.5);
Query OK, 1 row affected (0.03 sec)

mysql> insert into boats (bid, bname, color)
    ->        values (101, 'Interlake', 'blue');
Query OK, 1 row affected (0.06 sec)

mysql> insert into boats (bid, bname, color)
    ->        values (102, 'Interlake', 'red');
Query OK, 1 row affected (0.03 sec)

mysql> insert into boats (bid, bname, color)
    ->        values (103, 'Clipper', 'green');
Query OK, 1 row affected (0.06 sec)

mysql> insert into boats (bid, bname, color)
    ->        values (104, 'Marine', 'red');
Query OK, 1 row affected (0.06 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (22, 101, '1998-10-10');
Query OK, 1 row affected (0.08 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (22, 102, '1998-10-10');
Query OK, 1 row affected (0.05 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (22, 103, '1998-10-8');
Query OK, 1 row affected (0.13 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (22, 104, '1998-10-7');
Query OK, 1 row affected (0.03 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (31, 102, '1998-11-10');
Query OK, 1 row affected (0.03 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (31, 103, '1998-11-6');
Query OK, 1 row affected (0.05 sec)

mysql> insert into reserves (sid, bid, day)
    ->        values (31, 104, '1998-11-12');
Query OK, 1 row affected (0.03 sec)
```

```
mysql> insert into reserves (sid, bid, day)

    ->         values (64, 101, '1998-9-5');
Query OK, 1 row affected (0.03 sec)

mysql> insert into reserves (sid, bid, day)
    ->         values (64, 102, '1998-8-9');
Query OK, 1 row affected (0.07 sec)

mysql> insert into reserves (sid, bid, day)
    ->         values (74, 103, '1998-8-9');
Query OK, 1 row affected (0.03 sec)

mysql> select * from sailors;
+-----+---------+--------+------+
| sid | sname   | rating | age  |
+-----+---------+--------+------+
|  22 | Dustin  |      7 |   45 |
|  29 | Brutus  |      1 |   33 |
|  31 | Lubber  |      8 | 55.5 |
|  32 | Andy    |      8 | 25.5 |
|  58 | Rusty   |     10 |   35 |
|  64 | Horatio |      7 |   35 |
|  71 | Zorba   |     10 |   16 |
|  74 | Horatio |      9 |   35 |
|  85 | Art     |      3 | 25.5 |
|  95 | Bob     |      3 | 63.5 |
+-----+---------+--------+------+
10 rows in set (0.02 sec)

mysql> select * from boats;

+-----+-----------+-------+
| bid | bname     | color |
+-----+-----------+-------+
| 101 | Interlake | blue  |
| 102 | Interlake | red   |
| 103 | Clipper   | green |
| 104 | Marine    | red   |
+-----+-----------+-------+
4 rows in set (0.00 sec)

mysql> select * from reserves;
+-----+-----+------------+
| sid | bid | day        |
+-----+-----+------------+
|  22 | 101 | 1998-10-10 |
|  64 | 101 | 1998-09-05 |
|  22 | 102 | 1998-10-10 |
|  31 | 102 | 1998-11-10 |
|  64 | 102 | 1998-08-09 |
|  22 | 103 | 1998-10-08 |
|  31 | 103 | 1998-11-06 |
|  74 | 103 | 1998-08-09 |
|  22 | 104 | 1998-10-07 |
|  31 | 104 | 1998-11-12 |
+-----+-----+------------+
10 rows in set (0.01 sec)
```
Questions

1. **Display sailors name and age**

```
mysql> SELECT sname, age FROM sailors;
+---------+------+
| sname   | age  |
+---------+------+
| Dustin  |   45 |
| Brutus  |   33 |
| Lubber  | 55.5 |
| Andy    | 25.5 |
| Rusty   |   35 |
| Horatio |   35 |
| Zorba   |   16 |
| Horatio |   35 |
| Art     | 25.5 |
| Bob     | 63.5 |
+---------+------+
10 rows in set (0.00 sec)
```

2. Find all information of sailors who have reserved boat number 101.

```
mysql> SELECT S.* FROM Sailors S, Reserves R WHERE S.sid = R.sid AND R.bid = 103;
+-----+---------+--------+------+
| sid | sname   | rating | age  |
+-----+---------+--------+------+
|  22 | Dustin  |      7 |   45 |
|  31 | Lubber  |      8 | 55.5 |
|  74 | Horatio |      9 |   35 |
+-----+---------+--------+------+
3 rows in set (0.03 sec)
```

3. Find the names of sailors who have reserved a red boat, and list in the order of age.

```
mysql> SELECT S.sname, S.age FROM Sailors S, Reserves R, Boats B
          WHERE S.sid = R.sid AND R.bid = B.bid AND B.color ='red' ORDER BY S.age;
+---------+------+
| sname   | age  |
+---------+------+
| Horatio |   35 |
| Dustin  |   45 |
| Dustin  |   45 |
| Lubber  | 55.5 |
| Lubber  | 55.5 |
+---------+------+
5 rows in set (0.00 sec)
```

4. Find the names of sailors who have reserved at least one boat

```
mysql> SELECT sname FROM Sailors S, Reserves R  WHERE S.sid = R.sid;
+---------+
| sname   |
+---------+
| Dustin  |
| Dustin  |
| Dustin  |
| Dustin  |
| Lubber  |
| Lubber  |
| Lubber  |
| Horatio |
| Horatio |
| Horatio |
+---------+
10 rows in set (0.00 sec)
```

5. Find the ids and names of sailors who have reserved two different boats on the same day.

```
mysql> SELECT DISTINCT S.sid, S.sname FROM Sailors S, Reserves R1, Reserves R2
          WHERE S.sid = R1.sid AND S.sid = R2.sid AND R1.day = R2.day AND R1.bid <> R2.bid;
+-----+--------+
| sid | sname  |
+-----+--------+
|  22 | Dustin |
+-----+--------+
1 row in set (0.01 sec)
```

6. Using Expressions and Strings in the SELECT Command

```
mysql> select sname,age,rating+1 from sailors  where (((2*rating)-1)<10) and sname like 'B_%b';
+-------+------+----------+
| sname | age  | rating+1 |
+-------+------+----------+
| Bob   | 63.5 |        4 |
+-------+------+----------+
1 row in set (0.05 sec)
```

<u>**Set operations**</u>

7. Find the ids of sailors who have reserved a red boat or a green boat.

```
(SELECT R.sid FROM Boats B, Reserves R WHERE R.bid = B.bid AND B.color = 'RED');
UNION
(SELECT R2.sid FROM Boats B2, Reserves R2 WHERE R2.bid = B2.bid AND B2.color = 'GREEN');
+-----+
| sid |
+-----+
|  22 |
|  31 |
|  64 |
|  74 |
+-----+
4 rows in set (0.01 sec)
```

8. Find the names of sailors who have reserved boat 103.

```
mysql> SELECT S.sname FROM Sailors S WHERE S.sid IN ( SELECT R.sid FROM Reserves R WHERE R.bid =
103 );
+---------+
| sname   |
+---------+
| Dustin  |
| Lubber  |
| Horatio |
+---------+
3 rows in set (0.04 sec)
```

(Correlated Nested Queries)

```
mysql> SELECT S.sname FROM Sailors S WHERE EXISTS ( SELECT * FROM Reserves R
        WHERE R.bid = 103 AND R.sid = S.sid );
+---------+
| sname   |
+---------+
| Dustin  |
| Lubber  |
| Horatio |
+---------+
3 rows in set (0.00 sec)
```

9.Find the name and the age of the youngest sailor.

```
mysql> SELECT S.sname, S.age FROM Sailors S WHERE S.age <= ALL ( SELECT age FROM Sailors );
+-------+------+
| sname | age  |
+-------+------+
| Zorba |   16 |
+-------+------+
1 row in set (0.04 sec)
```

10. Find the names and ratings of sailor whose rating is better than some sailor called Horatio.

```
mysql> SELECT S.sname, S.rating FROM Sailors S
        WHERE S.rating > ANY ( SELECT S2.rating FROM Sailors S2 WHERE S2.sname = 'Horatio');
+---------+--------+
| sname   | rating |
+---------+--------+
| Lubber  |      8 |
| Andy    |      8 |
| Rusty   |     10 |
| Zorba   |     10 |
| Horatio |      9 |
+---------+--------+
5 rows in set (0.00 sec)
```

11. Find the names of sailors who have reserved all boats.

```
mysql> SELECT S.sname FROM Sailors S
        WHERE NOT EXISTS ( SELECT B.bid FROM Boats B
        WHERE NOT EXISTS ( SELECT R.bid FROM Reserves R WHERE R.bid = B.bid AND R.sid = S.sid ) );
+--------+
| sname  |
+--------+
| Dustin |
+--------+
1 row in set (0.00 sec)
```

Aggregate Functions

12.Count the number of different sailor names.

```
mysql> SELECT COUNT( DISTINCT S.sname ) FROM Sailors S;
+---------------------------+
| COUNT( DISTINCT S.sname ) |
+---------------------------+
|                         9 |
+---------------------------+
1 row in set (0.01 sec)
```

13: Calculate the average age of all sailors.

```
mysql> SELECT AVG(s.age) FROM Sailors S;
+------------+
| AVG(s.age) |
+------------+
|       36.9 |
+------------+
1 row in set (0.00 sec)
```

14: Find the name and the age of the youngest sailor

```
mysql> SELECT S.sname, S.age FROM Sailors S WHERE S.age = (SELECT MIN(S2.age) FROM Sailors S2 );
+-------+------+
| sname | age  |
+-------+------+
| Zorba |   16 |
+-------+------+
1 row in set (0.00 sec)
```

15. Find the average age of sailors for each rating level.

```
mysql> SELECT S.rating, AVG(S.age) AS avg_age  FROM Sailors S GROUP BY S.rating;
+--------+---------+
| rating | avg_age |
+--------+---------+
|      1 |      33 |
|      3 |    44.5 |
|      7 |      40 |
|      8 |    40.5 |
|      9 |      35 |
|     10 |    25.5 |
+--------+---------+
6 rows in set (0.04 sec)
```

16. Find the average age of sailors for each rating level that has at least two sailors.

```
mysql> SELECT S.rating, AVG(S.age) AS avg_age FROM Sailors S GROUP BY S.rating HAVING COUNT(*) >
1;
+--------+---------+
| rating | avg_age |
+--------+---------+
|      3 |    44.5 |
|      7 |      40 |
|      8 |    40.5 |
|     10 |    25.5 |
+--------+---------+
4 rows in set (0.00 sec)
```

17. An example shows difference between WHERE and HAVING

```
mysql> SELECT S.rating, AVG(S.age) as avg_age FROM Sailors S WHERE S.age >=40 GROUP BY S.rating;
+--------+---------+
| rating | avg_age |
+--------+---------+
|      3 |    63.5 |
|      7 |      45 |
|      8 |    55.5 |
+--------+---------+
3 rows in set (0.00 sec)
```

```
mysql> SELECT S.rating, AVG(S.age) as avg_age FROM Sailors S GROUP BY S.rating HAVING AVG(S.age)
>= 40;
+--------+---------+
| rating | avg_age |
+--------+---------+
|      3 |    44.5 |
|      7 |      40 |
|      8 |    40.5 |
+--------+---------+
3 rows in set (0.00 sec)
```

<u>Join</u>

```
mysql> SELECT sailors.sid, sailors.sname, reserves.bid FROM sailors
        LEFT JOIN reserves ON reserves.sid = sailors.sid ORDER BY sailors.sid;
+-----+---------+------+
| sid | sname   | bid  |
+-----+---------+------+
|  22 | Dustin  |  101 |
|  22 | Dustin  |  102 |
|  22 | Dustin  |  103 |
|  22 | Dustin  |  104 |
|  29 | Brutus  | NULL |
|  31 | Lubber  |  102 |
|  31 | Lubber  |  103 |
|  31 | Lubber  |  104 |
|  32 | Andy    | NULL |
|  58 | Rusty   | NULL |
|  64 | Horatio |  101 |
|  64 | Horatio |  102 |
|  71 | Zorba   | NULL |
|  74 | Horatio |  103 |
|  85 | Art     | NULL |
|  95 | Bob     | NULL |
+-----+---------+------+
16 rows in set (0.00 sec)
```

```
mysql> select * from customer;
+---------------+-----------------+---------------+
| customer_name | customer_street | customer_city |
+---------------+-----------------+---------------+
| Adams         | Spring          | Pittsfield    |
| Brooks        | Senator         | Brooklyn      |
| Curry         | North           | Rye           |
| Glenn         | Sand Hill       | Woodside      |
| Green         | Walnut          | Stamford      |
| Hayes         | Main            | Harrison      |
| Jackson       | University      | Salt Lake     |
| Johnson       | Alma            | Palo Alto     |
| Jones         | Main            | Harrison      |
| Lindsay       | Park            | Pittsfield    |
| Majeris       | First           | Rye           |
| McBride       | Safety          | Rye           |
| Smith         | Main            | Rye           |
| Turner        | Putnam          | Stamford      |
| Williams      | Nassau          | Princeton     |
+---------------+-----------------+---------------+
15 rows in set (0.26 sec)

mysql> select * from account;
+----------------+-------------+---------+
| account_number | branch_name | balance |
+----------------+-------------+---------+
| A-101          | Downtown    |     500 |
| A-102          | Perryridge  |     400 |
| A-201          | Perryridge  |     900 |
| A-215          | Mianus      |     700 |
| A-217          | Brighton    |     750 |
| A-222          | Redwood     |     700 |
| A-305          | Round Hill  |     350 |
| A-333          | Central     |     850 |
| A-444          | North Town  |     625 |
+----------------+-------------+---------+
9 rows in set (0.17 sec)

mysql> select * from loan;
+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| L-11        | Round Hill  |    900 |
| L-14        | Downtown    |   1500 |
| L-15        | Perryridge  |   1500 |
| L-16        | Perryridge  |   1300 |
| L-17        | Downtown    |   1000 |
| L-20        | North Town  |   7500 |
| L-21        | Central     |    570 |
| L-23        | Redwood     |   2000 |
| L-93        | Mianus      |    500 |
+-------------+-------------+--------+
9 rows in set (0.11 sec)

mysql> select * from depositor;
+---------------+----------------+
| customer_name | account_number |
+---------------+----------------+
| Hayes         | A-101          |
| Johnson       | A-101          |
| Hayes         | A-102          |
| Johnson       | A-201          |
| Smith         | A-215          |
| Jones         | A-217          |
| Lindsay       | A-222          |
| Turner        | A-305          |
| Majeris       | A-333          |
| Smith         | A-444          |
+---------------+----------------+
10 rows in set (0.13 sec)

mysql> select * from borrower;
+---------------+-------------+
| customer_name | loan_number |
+---------------+-------------+
| Smith         | L-11        |
| Jackson       | L-14        |
| Adams         | L-16        |
| Jones         | L-17        |
| Williams      | L-17        |
| McBride       | L-20        |
| Smith         | L-21        |
| Curry         | L-93        |
+---------------+-------------+
8 rows in set (0.13 sec)

mysql> select * from branch;
+-------------+-------------+---------+
| branch_name | branch_city | assets  |
+-------------+-------------+---------+
| Brighton    | Brooklyn    | 7000000 |
| Central     | Rye         |  400280 |
| Downtown    | Brooklyn    |  900000 |
| Mianus      | Horseneck   |  400200 |
| North Town  | Rye         | 3700000 |
| Perryridge  | Horseneck   | 1700000 |
| Pownal      | Bennington  |  400000 |
| Redwood     | Palo Alto   | 2100000 |
| Round Hill  | Horseneck   | 8000000 |
+-------------+-------------+---------+
9 rows in set (0.11 sec)
```

```
mysql> select * from emp;
+-------+--------+-----------+------+------------+---------+---------+--------+
| EMPNO | ENAME  | JOB       | MGR  | HIREDATE   | SAL     | COMM    | DEPTNO |
+-------+--------+-----------+------+------------+---------+---------+--------+
|  7369 | SMITH  | CLERK     | 7902 | 1980-12-17 |  800.00 |    NULL |     20 |
|  7499 | ALLEN  | SALESMAN  | 7698 | 1981-02-20 | 1600.00 |  300.00 |     30 |
|  7521 | WARD   | SALESMAN  | 7698 | 1981-02-22 | 1250.00 |  500.00 |     30 |
|  7566 | JONES  | MANAGER   | 7839 | 1981-04-02 | 2975.00 |    NULL |     20 |
|  7654 | MARTIN | SALESMAN  | 7698 | 1981-09-28 | 1250.00 | 1400.00 |     30 |
|  7698 | BLAKE  | MANAGER   | 7839 | 1981-05-01 | 2850.00 |    NULL |     30 |
|  7782 | CLARK  | MANAGER   | 7839 | 1981-06-09 | 2450.00 |    NULL |     10 |
|  7788 | SCOTT  | ANALYST   | 7566 | 1982-12-09 | 3000.00 |    NULL |     20 |
|  7839 | KING   | PRESIDENT | NULL | 1981-11-17 | 5000.00 |    NULL |     10 |
|  7844 | TURNER | SALESMAN  | 7698 | 1981-09-08 | 1500.00 |    0.00 |     30 |
|  7876 | ADAMS  | CLERK     | 7788 | 1983-01-12 | 1100.00 |    NULL |     20 |
|  7900 | JAMES  | CLERK     | 7698 | 1981-12-03 |  950.00 |    NULL |     30 |
|  7902 | FORD   | ANALYST   | 7566 | 1981-12-03 | 3000.00 |    NULL |     20 |
|  7934 | MILLER | CLERK     | 7782 | 1982-01-23 | 1300.00 |    NULL |     10 |
+-------+--------+-----------+------+------------+---------+---------+--------+
14 rows in set (0.16 sec)

mysql> select * from dept;
+--------+------------+----------+
| DEPTNO | DNAME      | LOC      |
+--------+------------+----------+
|     10 | ACCOUNTING | NEW YORK |
|     20 | RESEARCH   | DALLAS   |
|     30 | SALES      | CHICAGO  |
|     40 | OPERATIONS | BOSTON   |
+--------+------------+----------+
4 rows in set (0.09 sec)

mysql> select * from salgrade;
+-------+---------+---------+
| GRADE | LOSAL   | HISAL   |
+-------+---------+---------+
|  1.00 |  700.00 | 1200.00 |
|  2.00 | 1201.00 | 1400.00 |
|  3.00 | 1401.00 | 2000.00 |
|  4.00 | 2001.00 | 3000.00 |
|  5.00 | 3001.00 | 9999.00 |
+-------+---------+---------+
5 rows in set (0.14 sec)
```