Name: Atharva C Mahamuni.
Class: M.Sc.C.S.-1
Practical Assignment 1: RDBMS Concepts using MySQL

(1) Student- Competition database
Consider the following database maintained by a school to record the details of all the
competitions organized by the school every year. The school maintains information about
students as well as
competitions. Competition type can be like ‘academics’ or ‘sports’
etc. Following are the tables
1. Student (sreg_no int , name char(30), class char(10))
2. Competition(c_no int , name char(20), type char(15))
The relationship is as follows .
Student-competition : M-M with described attributes rank and year.

```
mysql> desc Student;
+---------+----------+------+-----+---------+-------+
| Field   | Type     | Null | Key | Default | Extra |
+---------+----------+------+-----+---------+-------+
| sreg_no | int      | NO   | PRI | NULL    |       |
| name    | char(30) | YES  |     | NULL    |       |
| class   | char(10) | YES  |     | NULL    |       |
+---------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from Student;
+---------+-----------+-------+
| sreg_no | name      | class |
+---------+-----------+-------+
|       1 | Ak        | 5th   |
|       2 | Akya      | 7th   |
|       3 | AtharvaCM | 10th  |
|       4 | RJ        | 9th   |
|       5 | SSK       | 8th   |
+---------+-----------+-------+
5 rows in set (0.00 sec)
```

```
mysql> desc Competition;
+-------+----------+------+-----+---------+-------+
| Field | Type     | Null | Key | Default | Extra |
+-------+----------+------+-----+---------+-------+
| c_no  | int      | NO   | PRI | NULL    |       |
| name  | char(20) | YES  |     | NULL    |       |
| type  | char(15) | YES  |     | NULL    |       |
+-------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from Competition;
+------+-------------+-----------+
| c_no | name        | type      |
+------+-------------+-----------+
|  101 | Webmaster   | academics |
|  102 | C           | academics |
|  103 | Football    | sports    |
|  104 | Cricket     | sports    |
|  105 | CS          | esports   |
|  106 | RunningRace | sports    |
```

```
+------+-------------+-----------+
6 rows in set (0.00 sec)
```

```
mysql> desc Stud_Comp;
+---------+----------+------+-----+---------+-------+
| Field   | Type     | Null | Key | Default | Extra |
+---------+----------+------+-----+---------+-------+
| sreg_no | int      | NO   | MUL | NULL    |       |
| c_no    | int      | NO   | MUL | NULL    |       |
| rank1   | char(10) | YES  |     | NULL    |       |
| year    | int      | YES  |     | NULL    |       |
+---------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> select * from Stud_Comp limit 5;
+---------+------+-------+------+
| sreg_no | c_no | rank1 | year |
+---------+------+-------+------+
|       1 |  101 | 1st   | 1995 |
|       1 |  101 | 1st   | 1996 |
|       2 |  102 | 1st   | 1996 |
|       1 |  102 | 2nd   | 1996 |
|       1 |  102 | 2nd   | 1995 |
+---------+------+-------+------+
5 rows in set (0.00 sec)
```

```
***********(a) Queries:************

1] List out all the competitions held in the school.

mysql> select name from Competition;
+-----------+
| name      |
+-----------+
| Webmaster |
| C         |
| Football  |
| Cricket   |
| CS        |
+-----------+
5 rows in set (0.01 sec)
```

```
2] List the names of all the students who have secured 1st rank in running race
from 1995 to
1996.

mysql> select Student.name, Competition.name, Stud_Comp.year from Student,
Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Stud_Comp.rank1 = '1st' and Stud_Comp.year
between 1995 and 1996;
+------+-------------+------+
| name | name        | year |
+------+-------------+------+
| Ak   | Webmaster   | 1995 |
| Ak   | Webmaster   | 1996 |
| Akya | C           | 1996 |
| Akya | CS          | 1996 |
| Akya | RunningRace | 1996 |
```

```
+------+-------------+------+
5 rows in set (0.00 sec)
```

3] Give the name of a student who has won maximum number of competitions.

```
mysql> select Student.name, count(if (Student.sreg_no = Stud_Comp.sreg_no and
Stud_Comp.rank1 = '1st', 1, NULL)) as ez from Student, Stud_Comp group by
Student.sreg_no order by ez desc limit 1;
+------+----+
| name | ez |
+------+----+
| Akya |  7 |
+------+----+
1 row in set (0.00 sec)
```

4] Find out the total number of competitions organized in the school for competition type 'sports'.

```
mysql> select count(if (Competition.c_no = Stud_Comp.c_no and Competition.type =
'sports', 1, NULL)) as total_sports_competitions from Competition, Stud_Comp;
+---------------------------+
| total_sports_competitions |
+---------------------------+
|                        23 |
+---------------------------+
1 row in set (0.00 sec)

mysql> select Competition.name from Competition where type = 'sports';
+-------------+
| name        |
+-------------+
| Football    |
| Cricket     |
| RunningRace |
+-------------+
3 rows in set (0.00 sec)
```

5] Find out the details of students participating in different competitions.

```
mysql> select distinct Student.*, Competition.name from Student, Competition,
Stud_Comp where Competition.c_no = Stud_Comp.c_no and Student.sreg_no =
Stud_Comp.sreg_no;
+---------+-----------+-------+-------------+
| sreg_no | name      | class | name        |
+---------+-----------+-------+-------------+
|       1 | Ak        | 5th   | Webmaster   |
|       1 | Ak        | 5th   | C           |
|       1 | Ak        | 5th   | Cricket     |
|       1 | Ak        | 5th   | CS          |
|       1 | Ak        | 5th   | RunningRace |
|       2 | Akya      | 7th   | Webmaster   |
|       2 | Akya      | 7th   | C           |
|       2 | Akya      | 7th   | Football    |
|       2 | Akya      | 7th   | CS          |
|       2 | Akya      | 7th   | RunningRace |
|       3 | AtharvaCM | 10th  | C           |
|       3 | AtharvaCM | 10th  | Football    |
|       3 | AtharvaCM | 10th  | CS          |
```

```
|       3 | AtharvaCM | 10th  | RunningRace |
|       4 | RJ        | 9th   | Webmaster   |
|       4 | RJ        | 9th   | C           |
|       4 | RJ        | 9th   | Cricket     |
|       4 | RJ        | 9th   | CS          |
|       4 | RJ        | 9th   | RunningRace |
|       5 | SSK       | 8th   | Webmaster   |
|       5 | SSK       | 8th   | C           |
|       5 | SSK       | 8th   | Football    |
|       5 | SSK       | 8th   | Cricket     |
|       5 | SSK       | 8th   | CS          |
|       5 | SSK       | 8th   | RunningRace |
+---------+-----------+-------+-------------+
25 rows in set (0.00 sec)
```

**********(b) Stored Procedures:**********

a) Write a procedure to count the no of competitions which come under the type 'sports' and no of
competitions which come under the type 'academics'.

```
mysql> create procedure count_compi(out sports_cnt int, out academics_cnt int)
begin select count(if (Competition.c_no = Stud_Comp.c_no and Competition.type =
'sports', 1, NULL)) as total_sports_competitions, count(if (Competition.c_no =
Stud_Comp.c_no and Competition.type = 'academics', 1, NULL)) as
total_academics_competitions from Competition, Stud_Comp; end//
Query OK, 0 rows affected (0.14 sec)

mysql> call count_compi (@sports_cnt, @academics_cnt) //
+---------------------------+-----------------------------+
| total_sports_competitions | total_academics_competitions |
+---------------------------+-----------------------------+
|                        23 |                          15 |
+---------------------------+-----------------------------+


mysql> create procedure count_compi(out sports_cnt int, out academics_cnt int)
begin select count(if (Competition.type = 'sports', 1, NULL)) as
total_sports_competitions, count(if (Competition.type = 'academics', 1, NULL))
as total_academics_competitions from Competition; end//
Query OK, 0 rows affected (0.10 sec)

mysql> call count_compi (@sports_cnt, @academics_cnt) //
+---------------------------+-----------------------------+
| total_sports_competitions | total_academics_competitions |
+---------------------------+-----------------------------+
|                         3 |                           2 |
+---------------------------+-----------------------------+
1 row in set (0.00 sec)

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

---

b) Write a stored procedure which accepts year as input and gives a list of all competitions held in
that year.

```
mysql> create procedure list_compi (in yyyy int, out all_compis_held char(10))
begin select distinct Competition.name as all_compis_held from Competition,
Stud_Comp where Competition.c_no = Stud_Comp.c_no and Stud_Comp.year = yyyy;
end//
```

```
Query OK, 0 rows affected (0.14 sec)

mysql> call list_compi (1998, @all_compis_held) //
+-----------------+
| all_compis_held |
+-----------------+
| Webmaster       |
| C               |
| Football        |
| Cricket         |
| CS              |
| RunningRace     |
+-----------------+
6 rows in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec)
```

**********(c) Stored Functions:**********

a) Write a function which accepts a competition type and returns the total no of competitions held
under that type.

```
mysql> create function total_compis(ctype char(10)) returns int
    -> deterministic
    -> begin
    -> declare cnt int;
    -> select count(if (Competition.c_no = Stud_Comp.c_no and Competition.type =
ctype, 1, NULL)) into cnt from Competition, Stud_Comp;
    -> return cnt;
    -> end //
Query OK, 0 rows affected (0.12 sec)

mysql> select total_compis('sports')
    -> //
+------------------------+
| total_compis('sports') |
+------------------------+
|                     23 |
+------------------------+
1 row in set (0.00 sec)

mysql> create function total_compis2(ctype char(10)) returns int
    -> deterministic
    -> begin
    -> declare cnt int;
    -> select count(if (type = ctype, 1, NULL)) into cnt from Competition;
    -> return cnt;
    -> end //
Query OK, 0 rows affected (0.22 sec)

mysql> select total_compis2('academics') //
+---------------------------+
| total_compis2('academics') |
+---------------------------+
|                         2 |
+---------------------------+
1 row in set (0.00 sec)
```

---

b) Write a function which accepts a name of students and returns the total no of prizes won by that
student in the year 2001.

```
mysql> create function get_total_prizes(sname char(15), yyyy int) returns int
    -> deterministic
    -> begin
    -> declare cnt int;
    -> select count(if (Student.sreg_no = Stud_Comp.sreg_no and Student.name =
sname and Stud_Comp.year = yyyy and Stud_Comp.rank1 = '1st', 1, NULL)) into cnt
from Student, Stud_Comp;
    -> return cnt;
    -> end //
Query OK, 0 rows affected (0.13 sec)

mysql> select get_total_prizes('AK', 1996) //
+-----------------------------+
| get_total_prizes('AK', 1996) |
+-----------------------------+
|                           1 |
+-----------------------------+
1 row in set (0.00 sec)
```

**********(d) Cursors:**********

a) Write a procedure using cursor which will list all the competitions in which students studying in the
5th std have won 1st prize in 1995.

```
mysql> create procedure d1()
    -> begin
    -> declare done int default 0;
    -> declare cname varchar(20) default "";
    -> declare cur1 cursor for
    -> select Competition.name from Competition, Student, Stud_Comp where
Student.sreg_no = Stud_Comp.sreg_no and Student.class = '5th' and
Competition.c_no = Stud_Comp.c_no and Stud_Comp.year = 1995 and Stud_Comp.rank1
= '1st';
    -> declare continue handler for SQLSTATE'02000' set done=1;
    -> open cur1;
    -> repeat
    -> fetch cur1 into cname;
    -> if not done then
    -> select cname;
    -> end if;
    -> until done
    -> end repeat;
    -> close cur1;
    -> end //
Query OK, 0 rows affected (0.14 sec)

mysql> call d1() //
+-----------+
| cname     |
+-----------+
| Webmaster |
+-----------+
1 row in set (0.00 sec)
```

---

b) Write a procedure using cursor to give competition wise 1st or 2nd rank holders for all the
competitions held in the year 2001

```
create procedure d2()
begin
declare done int default 0;
declare sname1 varchar(20) default "";
declare sname2 varchar(20) default "";
declare sname3 varchar(20) default "";
declare sname4 varchar(20) default "";
declare sname5 varchar(20) default "";
declare sname6 varchar(20) default "";
declare r1 varchar(10);
declare r2 varchar(10);
declare r3 varchar(10);
declare r4 varchar(10);
declare r5 varchar(10);
declare r6 varchar(10);
declare cur1 cursor for
      select Student.name as Webmaster, Stud_Comp.rank1 from Student,
Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Competition.c_no = 101 and Stud_Comp.year
= 1997 and Stud_Comp.rank1 = '1st' union select Student.name as Webmaster,
Stud_Comp.rank1 from Student, Competition, Stud_Comp where Student.sreg_no =
Stud_Comp.sreg_no and Competition.c_no = Stud_Comp.c_no and Competition.c_no =
101 and Stud_Comp.year = 1997 and Stud_Comp.rank1 = '2nd';
declare cur2 cursor for
      select Student.name as C, Stud_Comp.rank1 from Student, Competition,
Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and Competition.c_no =
Stud_Comp.c_no and Competition.c_no = 102 and Stud_Comp.year = 1997 and
Stud_Comp.rank1 = '1st' union select Student.name as C, Stud_Comp.rank1 from
Student, Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Competition.c_no = 102 and Stud_Comp.year
= 1997 and Stud_Comp.rank1 = '2nd';
declare cur3 cursor for
      select Student.name as Football, Stud_Comp.rank1 from Student,
Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Competition.c_no = 103 and Stud_Comp.year
= 1997 and Stud_Comp.rank1 = '1st' union select Student.name as Football,
Stud_Comp.rank1 from Student, Competition, Stud_Comp where Student.sreg_no =
Stud_Comp.sreg_no and Competition.c_no = Stud_Comp.c_no and Competition.c_no =
103 and Stud_Comp.year = 1997 and Stud_Comp.rank1 = '2nd';
declare cur4 cursor for
      select Student.name as Cricket, Stud_Comp.rank1 from Student, Competition,
Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and Competition.c_no =
Stud_Comp.c_no and Competition.c_no = 104 and Stud_Comp.year = 1997 and
Stud_Comp.rank1 = '1st' union select Student.name as Cricket, Stud_Comp.rank1
from Student, Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no
and Competition.c_no = Stud_Comp.c_no and Competition.c_no = 104 and
Stud_Comp.year = 1997 and Stud_Comp.rank1 = '2nd';
declare cur5 cursor for
      select Student.name as CS, Stud_Comp.rank1 from Student, Competition,
Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and Competition.c_no =
Stud_Comp.c_no and Competition.c_no = 105 and Stud_Comp.year = 1997 and
Stud_Comp.rank1 = '1st' union select Student.name as CS, Stud_Comp.rank1 from
Student, Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Competition.c_no = 105 and Stud_Comp.year
= 1997 and Stud_Comp.rank1 = '2nd';
declare cur6 cursor for
      select Student.name as RunningRace, Stud_Comp.rank1 from Student,
Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no and Competition.c_no = 106 and Stud_Comp.year
= 1997 and Stud_Comp.rank1 = '1st' union select Student.name as RunningRace,
```

```
Stud_Comp.rank1 from Student, Competition, Stud_Comp where Student.sreg_no =
Stud_Comp.sreg_no and Competition.c_no = Stud_Comp.c_no and Competition.c_no =
106 and Stud_Comp.year = 1997 and Stud_Comp.rank1 = '2nd';
declare continue handler for SQLSTATE'02000' set done=1;
open cur1;
open cur2;
open cur3;
open cur4;
open cur5;
open cur6;
repeat
fetch cur1 into sname1, r1;
fetch cur2 into sname2, r2;
fetch cur3 into sname3, r3;
fetch cur4 into sname4, r4;
fetch cur5 into sname5, r5;
fetch cur6 into sname6, r6;
if not done then
select sname1, r1;
select sname2, r2;
select sname3, r3;
select sname4, r4;
select sname5, r5;
select sname6, r6;
end if;
until done
end repeat;
close cur1;
close cur2;
close cur3;
close cur4;
close cur5;
close cur6;
end //
```

**********(e) Triggers:**********

a) Write a trigger before insertion on the relationship table. if the year entered is greater than current
year, it should be changed to current year.

```
mysql> create trigger e1 before insert on Stud_Comp for each row
    -> begin
    -> declare current_year int;
    -> select year(curdate()) into current_year;
    -> if new.year > current_year then
    -> set new.year = current_year;
    -> end if;
    -> end //
Query OK, 0 rows affected (0.21 sec)
```

---

b) Create a new table ‘tot_prize’ containing the fields stud_reg_no and no_of_prizes.
Write a trigger after insert into the relationship table between student and Competition. It should
increment the no_of_prizes in the table ‘tot_prize’ for the NEW stud_reg_no by 1.

```
create table tot_prize (stud_reg_no int not null, no_of_prizes int, constraint
stud_fk foreign key (stud_reg_no) references Student(sreg_no) on delete cascade
```

```
on update cascade);

mysql> create trigger e2 after insert on Stud_Comp for each row
    -> begin
    -> update tot_prize set no_of_prizes = no_of_prizes + 1 where stud_reg_no =
new.sreg_no;
    -> end //
Query OK, 0 rows affected (0.16 sec)
```

**********(f) Views:**********

a) Create a view over the competition table which contains only competition name and its type and it
should be sorted on type.

```
mysql> create view sorted_compi
    -> as
    -> select Competition.name, Competition.type from Competition order by type;
    -> //
Query OK, 0 rows affected (0.14 sec)

mysql> select * from sorted_compi;
    -> //
+-------------+-----------+
| name        | type      |
+-------------+-----------+
| Webmaster   | academics |
| C           | academics |
| CS          | esports   |
| Football    | sports    |
| Cricket     | sports    |
| RunningRace | sports    |
+-------------+-----------+
6 rows in set (0.00 sec)
```

b) Create a view containing student name, class, competition name, rank and year. the list should be
sorted by student name.

```
mysql> create view v2 as  select Student.name as Student_Name, Student.class,
Competition.name as Competition_Name, Stud_Comp.rank1, Stud_Comp.year from
Student, Competition, Stud_Comp where Student.sreg_no = Stud_Comp.sreg_no and
Competition.c_no = Stud_Comp.c_no order by Student.name;//
Query OK, 0 rows affected (0.17 sec)
```

==========

(2) Bank Database

==========

Consider the following database of Bank. A bank maintains the customer details, account details
and loan details . It has the Branch information also. Following are the tables :

1 Account(acc_no int, acc_type char(10), balance float(8,2))
2. Loan(loan_no int, loan_amt double(9,2) , no_of_years int)
3. Branch(branch_no int, branch_name char(20), branch_city varchar(20))
4 .Customer(cust_no int , cust_name char(20), cust_street char(15), cust_city varchar(20))

```
create table Customer(cust_no int primary key, cust_name char(20), cust_street
char(15), cust_city varchar(20));

create table Branch(branch_no int primary key, branch_name char(20) not null,
branch_city varchar(20));

create table Account(acc_no int primary key, acc_type char(10), balance
float(8,2), cust_no int, branch_no int, constraint fk_customer foreign key
(cust_no) references Customer(cust_no) on delete cascade on update cascade,
constraint fk_branch foreign key (branch_no) references Branch(branch_no) on
delete cascade on update cascade);

create table Loan(loan_no int primary key, loan_amt double(9, 2), no_of_years
int, cust_no int, branch_no int, constraint fk_customer2 foreign key (cust_no)
references Customer(cust_no) on delete cascade on update cascade, constraint
fk_branch2 foreign key (branch_no) references Branch(branch_no) on delete
cascade on update cascade);

mysql> select * from Customer;
+---------+-----------+-------------+-----------+
| cust_no | cust_name | cust_street | cust_city |
+---------+-----------+-------------+-----------+
|       1 | AtharvaCM | A           | Pune      |
|       2 | Aakash    | A           | Pune      |
|       3 | RJ        | B           | Pune      |
|       4 | Nachi     | B           | Mumbai    |
|       5 | SSk       | S           | Mumbai    |
|       6 | Jammy     | D           | Pune      |
|       7 | Nits      | S           | Pune      |
|       8 | Sanju     | F           | Kolkata   |
|       9 | Swup      | F           | Kolkata   |
+---------+-----------+-------------+-----------+
9 rows in set (0.00 sec)

mysql> select * from Account;
+--------+----------+-----------+---------+-----------+
| acc_no | acc_type | balance   | cust_no | branch_no |
+--------+----------+-----------+---------+-----------+
|  22101 | savings  |  29000.00 |       1 |       101 |
|  22102 | current  | 300000.00 |       2 |       101 |
|  22103 | savings  |  24000.00 |       2 |       102 |
|  22104 | savings  |   9000.00 |       1 |       102 |
|  22105 | savings  |  95000.00 |       3 |       103 |
|  22106 | current  |  70000.00 |       4 |       104 |
|  22107 | savings  |  71000.00 |       5 |       105 |
|  22108 | savings  |  67000.00 |       6 |       106 |
|  22109 | savings  |  66000.00 |       7 |       107 |
|  22110 | savings  |  54000.00 |       8 |       108 |
|  22111 | savings  |  51000.00 |       9 |       109 |
|  22112 | savings  |  28000.00 |       9 |       110 |
|  22113 | current  |  87000.00 |       7 |       111 |
+--------+----------+-----------+---------+-----------+
13 rows in set (0.00 sec)

mysql> select * from Branch;
+-----------+---------------+-------------+
| branch_no | branch_name   | branch_city |
+-----------+---------------+-------------+
|       101 | Sadashiv Peth | Pune        |
|       102 | Ravivar Peth  | Pune        |
|       103 | MG Road       | Pune        |
|       104 | Kurla         | Mumbai      |
|       105 | Bandra        | Mumbai      |
|       106 | CST           | Mumbai      |
```

```
|       107 | SS            | Kolkata     |
|       108 | IT Park       | Kolkata     |
|       109 | Ganesh Peth   | Pune        |
|       110 | FC Road       | Pune        |
|       111 | MC            | Pune        |
|       112 | CIDCO         | Kolkata     |
+-----------+---------------+-------------+
12 rows in set (0.00 sec)

mysql> select * from Loan;
+---------+----------+-------------+---------+-----------+
| loan_no | loan_amt | no_of_years | cust_no | branch_no |
+---------+----------+-------------+---------+-----------+
|   33101 | 15000.00 |           1 |       1 |       101 |
|   33102 | 41000.00 |           4 |       2 |       112 |
|   33103 | 45000.00 |           4 |       3 |       112 |
|   33104 | 20000.00 |           2 |       4 |       102 |
|   33105 | 21000.00 |           2 |       5 |       105 |
|   33106 | 21200.00 |           2 |       6 |       106 |
|   33107 | 21200.00 |           2 |       7 |       107 |
|   33108 | 21200.00 |           2 |       8 |       108 |
|   33109 | 43000.00 |           5 |       9 |       110 |
|   33110 | 60000.00 |           6 |       9 |       109 |
|   33111 | 98000.00 |           7 |       7 |       111 |
|   33112 | 40000.00 |           4 |       4 |       112 |
|   33113 | 21000.00 |           3 |       5 |       112 |
+---------+----------+-------------+---------+-----------+
13 rows in set (0.00 sec)

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------

************(a) Queries:*************

1. Find out customer name having loan amt >10000

mysql> select Customer.cust_name, Loan.loan_amt from Customer, Loan where
Customer.cust_no=Loan.cust_no and Loan.loan_amt > 10000;
+-----------+----------+
| cust_name | loan_amt |
+-----------+----------+
| AtharvaCM | 15000.00 |
| Aakash    | 41000.00 |
| RJ        | 45000.00 |
| Nachi     | 20000.00 |
| SSk       | 21000.00 |
| Jammy     | 21200.00 |
| Nits      | 21200.00 |
| Sanju     | 21200.00 |
| Swup      | 43000.00 |
| Swup      | 60000.00 |
| Nits      | 98000.00 |
| Nachi     | 40000.00 |
| SSk       | 21000.00 |
+-----------+----------+
13 rows in set (0.00 sec)

--------------------------------------------------------------------------------

2. Select customers having account but not loan

mysql> select cust_name from Customer where cust_no in (select cust_no from
Account where cust_no not in (select cust_no from Loan));
+-----------+
```

```
| cust_name |
+-----------+
| Ravi      |
+-----------+
1 row in set (0.00 sec)
```

---

3. Select customers having account as well as loan.

```
mysql> select cust_name from Customer where cust_no in (select cust_no from
Account where cust_no in (select cust_no from Loan));
+-----------+
| cust_name |
+-----------+
| AtharvaCM |
| Aakash    |
| RJ        |
| Nachi     |
| SSk       |
| Jammy     |
| Nits      |
| Sanju     |
| Swup      |
+-----------+
9 rows in set (0.00 sec)
```

---

4. Find out customer names having loan at ‘Pimpri’ branch

```
mysql> select Customer.cust_name, Branch.branch_name from Customer, Branch, Loan
where Customer.cust_no = Loan.cust_no and Branch.branch_no = Loan.branch_no and
Branch.branch_name = 'MC';
+-----------+-------------+
| cust_name | branch_name |
+-----------+-------------+
| Nits      | MC          |
+-----------+-------------+
1 row in set (0.00 sec)
```

---

5. Find out customer names having Saving account.

```
mysql> select cust_name from Customer, Account where Customer.cust_no =
Account.cust_no and acc_type = 'savings';
+-----------+
| cust_name |
+-----------+
| AtharvaCM |
| Aakash    |
| AtharvaCM |
| RJ        |
| SSk       |
| Jammy     |
| Nits      |
| Sanju     |
| Swup      |
| Swup      |
| Ravi      |
+-----------+
11 rows in set (0.00 sec)
```

---

6. Find outthe total of all the loan amount at Nagar Branch.

```
mysql> select sum(loan_amt) as total_loan_amt from Loan, Branch where
Loan.branch_no = Branch.branch_no and Branch.branch_name = 'FC Road';
+----------------+
| total_loan_amt |
+----------------+
|       43000.00 |
+----------------+
1 row in set (0.01 sec)
```

---

7. List the names of customers who have taken loan from the branch in the same city they live

```
mysql> select distinct Customer.cust_name, Customer.cust_city,
Branch.branch_city from Customer, Branch, Loan where Customer.cust_no =
Loan.cust_no and Branch.branch_no = Loan.branch_no and Customer.cust_city =
Branch.branch_city;
+-----------+-----------+-------------+
| cust_name | cust_city | branch_city |
+-----------+-----------+-------------+
| AtharvaCM | Pune      | Pune        |
| SSk       | Mumbai    | Mumbai      |
| Nits      | Pune      | Pune        |
| Sanju     | Kolkata   | Kolkata     |
+-----------+-----------+-------------+
4 rows in set (0.00 sec)
```


**********(b) Stored Procedures:**********

a) Write a procedure to transfer amount of 1000 Rs. from account_no 10 to account_no 20.

```
mysql> create procedure transfer()
    -> begin
    -> update Account set balance = balance -1000 where acc_no = 22101
    -> ;
    -> update Account set balance = balance + 1000 where acc_no = 22103;
    -> end //
Query OK, 0 rows affected (0.22 sec)

mysql> call transfer()//
Query OK, 1 row affected (0.19 sec)

mysql> select * from Account//
+--------+----------+-----------+---------+-----------+
| acc_no | acc_type | balance   | cust_no | branch_no |
+--------+----------+-----------+---------+-----------+
|  22101 | savings  |  28000.00 |       1 |       101 |
|  22102 | current  | 300000.00 |       2 |       101 |
|  22103 | savings  |  25000.00 |       2 |       102 |
|  22104 | savings  |   9000.00 |       1 |       102 |
|  22105 | savings  |  95000.00 |       3 |       103 |
|  22106 | current  |  70000.00 |       4 |       104 |
|  22107 | savings  |  71000.00 |       5 |       105 |
|  22108 | savings  |  67000.00 |       6 |       106 |
|  22109 | savings  |  66000.00 |       7 |       107 |
```

```
|  22110 | savings  |  54000.00 |       8 |       108 |
|  22111 | savings  |  51000.00 |       9 |       109 |
|  22112 | savings  |  28000.00 |       9 |       110 |
|  22113 | current  |  87000.00 |       7 |       111 |
|  22114 | savings  |  69000.00 |      10 |       111 |
+--------+----------+-----------+---------+-----------+
14 rows in set (0.00 sec)
```

---

b) Write a procedure withdrawal for the following
 1. Accept balance amount and acc_no for withdrawal as input parameters.
 2. Check if input amount is less than actual balance of accounts.
 3. If input amount is less ,givethe message “withdrawal allowed from account” otherwise give the message “withdrawal not allowed from account”. Update the balance field.

```
  mysql> create procedure withdraw (in amt float, in acc_num int)
    ->   begin
    ->   declare bal float;
    ->   select balance into bal from Account where  acc_no = acc_num;
    ->   if bal > amt then
    ->   update Account set balance = balance -amt where acc_no = acc_num;
    ->   select 'Withdrawalallowed from Account' as Message;
    ->   else
    ->   select 'Withdrawal not allowed from Account' as Message;
    ->   end if;
    ->   end //
Query OK, 0 rows affected (0.19 sec)

mysql> select * from Account
    -> //
+--------+----------+-----------+---------+-----------+
| acc_no | acc_type | balance   | cust_no | branch_no |
+--------+----------+-----------+---------+-----------+
|  22101 | savings  |  28000.00 |       1 |       101 |
|  22102 | current  | 300000.00 |       2 |       101 |
|  22103 | savings  |  25000.00 |       2 |       102 |
|  22104 | savings  |   9000.00 |       1 |       102 |
|  22105 | savings  |  95000.00 |       3 |       103 |
|  22106 | current  |  70000.00 |       4 |       104 |
|  22107 | savings  |  71000.00 |       5 |       105 |
|  22108 | savings  |  67000.00 |       6 |       106 |
|  22109 | savings  |  66000.00 |       7 |       107 |
|  22110 | savings  |  54000.00 |       8 |       108 |
|  22111 | savings  |  51000.00 |       9 |       109 |
|  22112 | savings  |  28000.00 |       9 |       110 |
|  22113 | current  |  87000.00 |       7 |       111 |
|  22114 | savings  |  69000.00 |      10 |       111 |
+--------+----------+-----------+---------+-----------+
14 rows in set (0.00 sec)

mysql> call withdraw(30000, 22103)
    -> //
+-----------------------------------+
| Message                           |
+-----------------------------------+
| Withdrawal not allowed from Account |
+-----------------------------------+
1 row in set (0.04 sec)

Query OK, 0 rows affected (0.04 sec)

mysql> call withdraw(1000, 22103)//
```

```
+--------------------------------+
| Message                        |
+--------------------------------+
| Withdrawalallowed from Account |
+--------------------------------+
1 row in set (0.08 sec)

Query OK, 0 rows affected (0.08 sec)

mysql> select * from Account//
+--------+----------+-----------+---------+-----------+
| acc_no | acc_type | balance   | cust_no | branch_no |
+--------+----------+-----------+---------+-----------+
|  22101 | savings  |  28000.00 |       1 |       101 |
|  22102 | current  | 300000.00 |       2 |       101 |
|  22103 | savings  |  24000.00 |       2 |       102 |
|  22104 | savings  |   9000.00 |       1 |       102 |
|  22105 | savings  |  95000.00 |       3 |       103 |
|  22106 | current  |  70000.00 |       4 |       104 |
|  22107 | savings  |  71000.00 |       5 |       105 |
|  22108 | savings  |  67000.00 |       6 |       106 |
|  22109 | savings  |  66000.00 |       7 |       107 |
|  22110 | savings  |  54000.00 |       8 |       108 |
|  22111 | savings  |  51000.00 |       9 |       109 |
|  22112 | savings  |  28000.00 |       9 |       110 |
|  22113 | current  |  87000.00 |       7 |       111 |
|  22114 | savings  |  69000.00 |      10 |       111 |
+--------+----------+-----------+---------+-----------+
14 rows in set (0.00 sec)

--------------------------------------------------------------------------------
--------------------------------------------------------------------------------

************(c) Stored Functions:*************

a) Write a function that returns the total loan amount of a particular branch. (
Accept branch name as input parameter.)

mysql> select * from Loan//
+---------+----------+-------------+---------+-----------+
| loan_no | loan_amt | no_of_years | cust_no | branch_no |
+---------+----------+-------------+---------+-----------+
|   33101 | 15000.00 |           1 |       1 |       101 |
|   33102 | 41000.00 |           4 |       2 |       112 |
|   33103 | 45000.00 |           4 |       3 |       112 |
|   33104 | 20000.00 |           2 |       4 |       102 |
|   33105 | 21000.00 |           2 |       5 |       105 |
|   33106 | 21200.00 |           2 |       6 |       106 |
|   33107 | 21200.00 |           2 |       7 |       107 |
|   33108 | 21200.00 |           2 |       8 |       108 |
|   33109 | 43000.00 |           5 |       9 |       110 |
|   33110 | 60000.00 |           6 |       9 |       109 |
|   33111 | 98000.00 |           7 |       7 |       111 |
|   33112 | 40000.00 |           4 |       4 |       112 |
|   33113 | 21000.00 |           3 |       5 |       112 |
+---------+----------+-------------+---------+-----------+
13 rows in set (0.00 sec)

mysql> select * from Branch//
+-----------+---------------+-------------+
| branch_no | branch_name   | branch_city |
+-----------+---------------+-------------+
|       101 | Sadashiv Peth | Pune        |
|       102 | Ravivar Peth  | Pune        |
```

```
|       103 | MG Road       | Pune        |
|       104 | Kurla         | Mumbai      |
|       105 | Bandra        | Mumbai      |
|       106 | CST           | Mumbai      |
|       107 | SS            | Kolkata     |
|       108 | IT Park       | Kolkata     |
|       109 | Ganesh Peth   | Pune        |
|       110 | FC Road       | Pune        |
|       111 | MC            | Pune        |
|       112 | CIDCO         | Kolkata     |
+-----------+---------------+-------------+
12 rows in set (0.00 sec)

mysql>  create function total_loan(b_name varchar(30)) returns float
    ->  deterministic
    ->  begin
    ->  declare op float;
    ->  select sum(loan_amt) as total_loan_amt into op from loan, branch where
branch.branch_no = loan.branch_no and branch_name = b_name;
    ->  return(op);
    ->  end //
Query OK, 0 rows affected (0.18 sec)

mysql>  select total_loan('CIDCO')//
+---------------------+
| total_loan('CIDCO') |
+---------------------+
|              147000 |
+---------------------+
1 row in set (0.00 sec)
```

---

b) Write a function to count the no. of customers of particular branch. (Accept branch name as input parameter).

```
mysql> create function count_cust(br_name varchar(30)) returns int
deterministic   begin   declare op int;   select count(cust_no) as no_of_cust
into op from Account, Branch where Branch.branch_no = Account.branch_no and
Branch_name = br_name;   return(op);   end //
Query OK, 0 rows affected (0.16 sec)

 mysql> select count_cust('MC');
    -> //
+------------------+
| count_cust('MC') |
+------------------+
|                2 |
+------------------+
1 row in set (0.01 sec)
```


**********(d) Cursors:**********

a) Write a procedure using cursor to display the customers having loan amounts between 40000 and 50000 from branch name 'CIDCO'.

```
mysql> create procedure C1()   begin   declare done int default 0;   declare
customer_name varchar(30);  declare loan_amount float;   declare branch_n
varchar(30);
declare cur1 cursor for select Customer.cust_name, Loan.loan_amt,
Branch.branch_name from Customer, Branch, Loan where Customer.cust_no =
```

```
Loan.cust_no and Branch.branch_no = Loan.branch_no and Loan_amt between 40000
and 50000 and Branch.branch_name = 'CIDCO';   declare continue handler for
SQLSTATE '02000' set done = 1;   open cur1;   repeat   fetch cur1 into
customer_name, loan_amount, branch_n;   if not done then   select customer_name,
loan_amount, branch_n;   end if;   until done   end repeat;   close cur1;
end//
Query OK, 0 rows affected (0.18 sec)

mysql> call C1()//
+---------------+-------------+----------+
| customer_name | loan_amount | branch_n |
+---------------+-------------+----------+
| Aakash        |       41000 | CIDCO    |
+---------------+-------------+----------+
1 row in set (0.00 sec)

+---------------+-------------+----------+
| customer_name | loan_amount | branch_n |
+---------------+-------------+----------+
| RJ            |       45000 | CIDCO    |
+---------------+-------------+----------+
1 row in set (0.00 sec)

+---------------+-------------+----------+
| customer_name | loan_amount | branch_n |
+---------------+-------------+----------+
| Nachi         |       40000 | CIDCO    |
+---------------+-------------+----------+
1 row in set (0.00 sec)

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

---

b) Write a procedure using cursor add an interest of 3% to the balance of all accounts having balance > 5000

```
mysql>  create procedure C2()
    ->  begin
    ->  declare done int default 0;
    ->  declare bal float;
    ->  declare account_no int;
    ->  declare cur1 cursor for select balance from Account where balance >
5000;
    ->  declare cur2 cursor for select acc_no from Account where balance > 5000;
    ->  declare continue handler for SQLSTATE '02000' set done = 1;
    ->  open cur1;
    ->  open cur2;
    ->  repeat
    ->  fetch cur1 into bal;
    ->  fetch cur2 into account_no;
    ->  if not done then
    ->  set bal = (bal * 3 * 0.01) + bal;
    ->  update Account set balance = bal where acc_no = account_no;
    ->  select Account_no, bal;
    ->  end if;
    ->  until done
    ->  end repeat;
    ->  close cur1;
    ->  close cur2;
    ->  end //
Query OK, 0 rows affected (0.16 sec)

mysql> call C2()//
```

```
+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22101 | 28840 |
+------------+-------+
1 row in set (0.37 sec)

+------------+--------+
| Account_no | bal    |
+------------+--------+
|      22102 | 309000 |
+------------+--------+
1 row in set (0.50 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22103 | 24720 |
+------------+-------+
1 row in set (0.57 sec)

+------------+------+
| Account_no | bal  |
+------------+------+
|      22104 | 9270 |
+------------+------+
1 row in set (0.70 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22105 | 97850 |
+------------+-------+
1 row in set (0.80 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22106 | 72100 |
+------------+-------+
1 row in set (0.90 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22107 | 73130 |
+------------+-------+
1 row in set (1.00 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22108 | 69010 |
+------------+-------+
1 row in set (1.10 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22109 | 67980 |
+------------+-------+
1 row in set (1.34 sec)

+------------+-------+
```

```
| Account_no | bal   |
+------------+-------+
|      22110 | 55620 |
+------------+-------+
1 row in set (1.59 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22111 | 52530 |
+------------+-------+
1 row in set (1.70 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22112 | 28840 |
+------------+-------+
1 row in set (1.77 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22113 | 89610 |
+------------+-------+
1 row in set (1.85 sec)

+------------+-------+
| Account_no | bal   |
+------------+-------+
|      22114 | 71070 |
+------------+-------+
1 row in set (1.95 sec)

Query OK, 0 rows affected (1.95 sec)
```

**********(e) Triggers:**********

a) Write a trigger which will execute when account_no is less than 0 . Display the appropriate message.

```
create trigger T1 before update on Account for each row
begin
declare x int;
if new.acc_no < 0 then
select 'Account number cannot be less than zero' into x from Account;
end if;
end //

mysql> update Account set acc_no = -1 where acc_no = 22101//
ERROR 1366 (HY000): Incorrect integer value: 'Account number cannot be less than
zero' for column 'x' at row 1
```

---

b) Write a trigger which will execute when loan_amount is updated. Do not allow to update. Display the message that ' loan amount once given cannot be updated.”

```
mysql> select * from Loan//
+---------+----------+-------------+---------+-----------+
| loan_no | loan_amt | no_of_years | cust_no | branch_no |
```

```
+---------+----------+-------------+---------+-----------+
|   33101 | 15000.00 |           1 |       1 |       101 |
|   33102 | 41000.00 |           4 |       2 |       112 |
|   33103 | 45000.00 |           4 |       3 |       112 |
|   33104 | 20000.00 |           2 |       4 |       102 |
|   33105 | 21000.00 |           2 |       5 |       105 |
|   33106 | 21200.00 |           2 |       6 |       106 |
|   33107 | 21200.00 |           2 |       7 |       107 |
|   33108 | 21200.00 |           2 |       8 |       108 |
|   33109 | 43000.00 |           5 |       9 |       110 |
|   33110 | 60000.00 |           6 |       9 |       109 |
|   33111 | 98000.00 |           7 |       7 |       111 |
|   33112 | 40000.00 |           4 |       4 |       112 |
|   33113 | 21000.00 |           3 |       5 |       112 |
+---------+----------+-------------+---------+-----------+
13 rows in set (0.00 sec)

 mysql>  create trigger T2 before update on Loan for each row
    ->  begin
    ->  declare x int;
    ->  if new.loan_amt then
    ->  select 'loan amount once given cannot be updated.' into x from Loan;
    ->  end if;
    ->  end //
Query OK, 0 rows affected (0.14 sec)

mysql>  update Loan set loan_amt = 17000 where loan_no = 33101//
ERROR 1366 (HY000): Incorrect integer value: 'loan amount once given cannot be
updated.' for column 'x' at row 1
```

**********(f) Views:**********

a) Create a view which contains all the customer details along with the details of all accounts of that customer.

```
create view V1 as select Customer.*, Account.acc_no, Account.acc_type,
Account.balance, Account.branch_no from Customer, Account where Customer.cust_no
= Account.cust_no;

mysql> select * from V1//
+---------+----------+-------------+----------+--------+----------+----------
+-----------+
| cust_no | cust_name | cust_street | cust_city | acc_no | acc_type | balance
| branch_no |
+---------+----------+-------------+----------+--------+----------+----------
+-----------+
|       1 | AtharvaCM | A           | Pune      |  22101 | savings  |  28840.00
|       101 |
|       1 | AtharvaCM | A           | Pune      |  22104 | savings  |   9270.00
|       102 |
|       2 | Aakash    | A           | Pune      |  22102 | current  | 309000.00
|       101 |
|       2 | Aakash    | A           | Pune      |  22103 | savings  |  24720.00
|       102 |
|       3 | RJ        | B           | Pune      |  22105 | savings  |  97850.00
|       103 |
|       4 | Nachi     | B           | Mumbai    |  22106 | current  |  72100.00
|       104 |
|       5 | SSk       | S           | Mumbai    |  22107 | savings  |  73130.00
|       105 |
|       6 | Jammy     | D           | Pune      |  22108 | savings  |  69010.00
```

```
|       106 |
|       7 | Nits      | S           | Pune      |  22109 | savings  |  67980.00
|       107 |
|       7 | Nits      | S           | Pune      |  22113 | current  |  89610.00
|       111 |
|       8 | Sanju     | F           | Kolkata   |  22110 | savings  |  55620.00
|       108 |
|       9 | Swup      | F           | Kolkata   |  22111 | savings  |  52530.00
|       109 |
|       9 | Swup      | F           | Kolkata   |  22112 | savings  |  28840.00
|       110 |
|      10 | Ravi      | D           | Pune      |  22114 | savings  |  71070.00
|       111 |
+---------+-----------+-------------+-----------+--------+----------+-----------
+-----------+
14 rows in set (0.00 sec)
```

---

b) Create a view which contains details of all loans from the ‘sadashiv peth’ branch.

```
create view V2 as select Loan.* from Loan, Branch where Branch.branch_no =
Loan.branch_no and branch_name = 'Sadashiv Peth'

mysql> select * from V2//
+---------+----------+-------------+---------+-----------+
| loan_no | loan_amt | no_of_years | cust_no | branch_no |
+---------+----------+-------------+---------+-----------+
|   33101 | 15000.00 |           1 |       1 |       101 |
+---------+----------+-------------+---------+-----------+
1 row in set (0.00 sec)
```