Practical 2(a)

Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym, different constraints etc.

```
mysql> create Database Practical2a;
Query OK, 1 row affected (0.12 sec)

mysql> use Practical2a;
Database changed

mysql> create table Student(SID int primary key, SName varchar(20), SAge
varchar(20), SPhoneNo varchar(20), SPercentage varchar(20));
Query OK, 0 rows affected (0.18 sec)

mysql> show tables;
+----------------------+
| Tables_in_Practical1 |
+----------------------+
| Student              |
+----------------------+
1 row in set (0.01 sec)

mysql> desc Student;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| SID         | int         | NO   | PRI | NULL    |       |
| SName       | varchar(20) | YES  |     | NULL    |       |
| SAge        | varchar(20) | YES  |     | NULL    |       |
| SPhoneNo    | varchar(20) | YES  |     | NULL    |       |
| SPercentage | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.04 sec)

mysql> insert into Student values(1, "Mayur", 19, 8668832059,60),(2, "Akshay",
18, 7894567851, 59),(3, "Sanjay", 20, 9011779086, 80);
Query OK, 3 rows affected (0.20 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from Student;
+-----+--------+------+------------+-------------+
| SID | SName  | SAge | SPhoneNo   | SPercentage |
+-----+--------+------+------------+-------------+
|   1 | Mayur  | 19   | 8668832059 | 60          |
|   2 | Akshay | 18   | 7894567851 | 59          |
|   3 | Sanjay | 20   | 9011779086 | 80          |
+-----+--------+------+------------+-------------+
3 rows in set (0.00 sec)

mysql> select SName, SAge from Student;
+--------+------+
| SName  | SAge |
+--------+------+
| Mayur  | 19   |
| Akshay | 18   |
| Sanjay | 20   |
+--------+------+
3 rows in set (0.00 sec)

mysql> alter table Student add SClass varchar(20);
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

```
mysql> desc Student;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| SID         | int         | NO   | PRI | NULL    |       |
| SName       | varchar(20) | YES  |     | NULL    |       |
| SAge        | varchar(20) | YES  |     | NULL    |       |
| SPhoneNo    | varchar(20) | YES  |     | NULL    |       |
| SPercentage | varchar(20) | YES  |     | NULL    |       |
| SClass      | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> select * from Student;
+-----+--------+------+------------+-------------+--------+
| SID | SName  | SAge | SPhoneNo   | SPercentage | SClass |
+-----+--------+------+------------+-------------+--------+
|   1 | Mayur  | 19   | 8668832059 | 60          | NULL   |
|   2 | Akshay | 18   | 7894567851 | 59          | NULL   |
|   3 | Sanjay | 20   | 9011779086 | 80          | NULL   |
+-----+--------+------+------------+-------------+--------+
3 rows in set (0.00 sec)

mysql> update Student set SClass="TECO" where SID=1;
Query OK, 1 row affected (0.12 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Student set SClass="TECO" where SID=2;
Query OK, 1 row affected (0.11 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Student set SClass="TECO" where SID=3;
Query OK, 1 row affected (0.07 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Student;
+-----+--------+------+------------+-------------+--------+
| SID | SName  | SAge | SPhoneNo   | SPercentage | SClass |
+-----+--------+------+------------+-------------+--------+
|   1 | Mayur  | 19   | 8668832059 | 60          | TECO   |
|   2 | Akshay | 18   | 7894567851 | 59          | TECO   |
|   3 | Sanjay | 20   | 9011779086 | 80          | TECO   |
+-----+--------+------+------------+-------------+--------+
3 rows in set (0.00 sec)

mysql> create Index Student_Serach on Student(SID);
Query OK, 0 rows affected (0.22 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table Student rename to Stud_Info;
Query OK, 0 rows affected (0.20 sec)

mysql> alter table Stud_Info modify SPercentage int;
Query OK, 3 rows affected (0.44 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> desc Stud_Info;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| SID         | int         | NO   | PRI | NULL    |       |
| SName       | varchar(20) | YES  |     | NULL    |       |
| SAge        | varchar(20) | YES  |     | NULL    |       |
| SPhoneNo    | varchar(20) | YES  |     | NULL    |       |
| SPercentage | int         | YES  |     | NULL    |       |
| SClass      | varchar(20) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

```
mysql> create view Student as select SID, SName, SClass from Stud_Info;
Query OK, 0 rows affected (0.13 sec)

mysql> select * from Student;
+-----+--------+--------+
| SID | SName  | SClass |
+-----+--------+--------+
|   1 | Mayur  | TECO   |
|   2 | Akshay | TECO   |
|   3 | Sanjay | TECO   |
+-----+--------+--------+
3 rows in set (0.00 sec)
```

Practical 2(b)

Write at least 10 SQL queries on the suitable database application using SQL DML statements.

```
mysql> create Database Practical2b;
Query OK, 1 row affected (0.07 sec)

mysql> use Practical2b;
Database changed

mysql> create table Student(SID int primary key, SName varchar(20), SAge
varchar(20), SPhoneNo varchar(20), SPercentage varchar(20));
Query OK, 0 rows affected (0.21 sec)

mysql> alter table Student add SClass varchar(20);
Query OK, 0 rows affected (0.21 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> insert into Student values(1, "Mayur", 19, 8668832059,60,"FECO"),(2,
"Akshay", 18, 7894567851, 59,"SECO"),(3, "Sanjay", 20, 9011779086, 80, "TECO");
Query OK, 3 rows affected (0.11 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> create table Staff (ID int primary key, Name varchar(20), Age
varchar(20), PhoneNo varchar(20), Class varchar(20));
Query OK, 0 rows affected (0.26 sec)

mysql> insert into Staff  values(1, "Suyash", 44, 7894651237,
"FECO"),(2,"Shubham",100, 4567891344, "SECO"),(3,"Mayur",30,789456853,"TECO");
Query OK, 3 rows affected (0.13 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from Staff;
+----+---------+------+------------+-------+
| ID | Name    | Age  | PhoneNo    | Class |
+----+---------+------+------------+-------+
|  1 | Suyash  | 44   | 7894651237 | FECO  |
|  2 | Shubham | 100  | 4567891344 | SECO  |
|  3 | Mayur   | 30   | 789456853  | TECO  |
+----+---------+------+------------+-------+
3 rows in set (0.00 sec)

mysql> alter table Staff add Experience varchar(20);
Query OK, 0 rows affected (0.24 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> update Staff set Experience=4 where ID=1;
Query OK, 1 row affected (0.11 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Staff set Experience=3 where ID=2;
Query OK, 1 row affected (0.07 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> update Staff set Experience=2 where ID=3;
Query OK, 1 row affected (0.04 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Staff;
+----+---------+------+------------+-------+------------+
| ID | Name    | Age  | PhoneNo    | Class | Experience |
+----+---------+------+------------+-------+------------+
|  1 | Suyash  | 44   | 7894651237 | FECO  | 4          |
|  2 | Shubham | 100  | 4567891344 | SECO  | 3          |
|  3 | Mayur   | 30   | 789456853  | TECO  | 2          |
+----+---------+------+------------+-------+------------+
3 rows in set (0.00 sec)
```

```
mysql> select * from Student Union select * from Staff;
+-----+---------+------+------------+-------------+--------+
| SID | SName   | SAge | SPhoneNo   | SPercentage | SClass |
+-----+---------+------+------------+-------------+--------+
|   1 | Mayur   | 19   | 8668832059 | 60          | FECO   |
|   2 | Akshay  | 18   | 7894567851 | 59          | SECO   |
|   3 | Sanjay  | 20   | 9011779086 | 80          | TECO   |
|   1 | Suyash  | 44   | 7894651237 | FECO        | 4      |
|   2 | Shubham | 100  | 4567891344 | SECO        | 3      |
|   3 | Mayur   | 30   | 789456853  | TECO        | 2      |
+-----+---------+------+------------+-------------+--------+
6 rows in set (0.07 sec)
```

```
mysql> select distinct SClass from Student where SClass in (select Class from
Staff);
+--------+
| SClass |
+--------+
| FECO   |
| SECO   |
| TECO   |
+--------+
3 rows in set (0.05 sec)
```

```
mysql> select distinct SName from Student where SName in (select Name from
Staff);
+-------+
| SName |
+-------+
| Mayur |
+-------+
1 row in set (0.00 sec)
```

```
mysql> select min(SPercentage) from Student;
+------------------+
| min(SPercentage) |
+------------------+
| 59               |
+------------------+
1 row in set (0.09 sec)
```

```
mysql> select Max(SPercentage) from Student;
+------------------+
| Max(SPercentage) |
+------------------+
| 80               |
+------------------+
1 row in set (0.06 sec)
```

```
mysql> select Avg(SPercentage) from Student;
+-------------------+
| Avg(SPercentage)  |
+-------------------+
| 66.33333333333333 |
+-------------------+
1 row in set (0.07 sec)
```

```
mysql> select sum(SPercentage) from Student;
+------------------+
| sum(SPercentage) |
+------------------+
|              199 |
+------------------+
1 row in set (0.06 sec)
```

mysql> select ucase(SName) from Student;

```
+--------------+
| ucase(SName) |
+--------------+
| MAYUR        |
| AKSHAY       |
| SANJAY       |
+--------------+
3 rows in set (0.00 sec)
```

mysql> select lcase(SName) from Student;

```
+--------------+
| lcase(SName) |
+--------------+
| mayur        |
| akshay       |
| sanjay       |
+--------------+
3 rows in set (0.01 sec)
```

mysql> select mid(Experience,1,3) from Staff;

```
+---------------------+
| mid(Experience,1,3) |
+---------------------+
| 4                   |
| 3                   |
| 2                   |
+---------------------+
3 rows in set (0.00 sec)
```

mysql> select mid(PhoneNo,1,3) from Staff;

```
+------------------+
| mid(PhoneNo,1,3) |
+------------------+
| 789              |
| 456              |
| 789              |
+------------------+
3 rows in set (0.01 sec)
```

mysql> select mid(PhoneNo,1,10) from Staff;

```
+-------------------+
| mid(PhoneNo,1,10) |
+-------------------+
| 7894651237        |
| 4567891344        |
| 789456853         |
+-------------------+
3 rows in set (0.00 sec)
```

Practical 3

SQL Queries - all types of Join, Sub-Query and View: Write at least 10 SQL queries for suitable database application using SQL DML statements

```
mysql> Create Database Practical3;

Query OK, 1 row affected (0.08 sec)


mysql> use Practical3;
Database changed

mysql> create Table Emp(EID int Primary Key, EFName varchar(20), ELName
varchar(20), ESalary varchar(20), EPhone varchar(20));
Query OK, 0 rows affected (0.25 sec)

mysql> create Table Manager(MID int Primary Key, MFName varchar(20), MLName
varchar(20), MSalary varchar(20), MPhone varchar(20));
Query OK, 0 rows affected (0.11 sec)

mysql> Desc Emp;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| EID     | int         | NO   | PRI | NULL    |       |
| EFName  | varchar(20) | YES  |     | NULL    |       |
| ELName  | varchar(20) | YES  |     | NULL    |       |
| ESalary | varchar(20) | YES  |     | NULL    |       |
| EPhone  | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)

mysql> Desc Manager;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| MID     | int         | NO   | PRI | NULL    |       |
| MFName  | varchar(20) | YES  |     | NULL    |       |
| MLName  | varchar(20) | YES  |     | NULL    |       |
| MSalary | varchar(20) | YES  |     | NULL    |       |
| MPhone  | varchar(20) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql> insert into Emp values(1, "Mayur","Jagtap",40000, 8668832059),(2,
"Sanjay","Jagtap",50000,9011779086),(3,"Suyash","Jagtap",60000,8830322552);
Query OK, 3 rows affected (0.03 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> insert into Manager values(1, "Sunil","Jagtap",400000, 8668845659),(2,
"Shivram","Jagtap",500000,4561779086),(3,"Nilesh","Jagtap",600000,883789552);
Query OK, 3 rows affected (0.07 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> select * from Emp;
+-----+--------+--------+---------+------------+
| EID | EFName | ELName | ESalary | EPhone     |
+-----+--------+--------+---------+------------+
|   1 | Mayur  | Jagtap | 40000   | 8668832059 |
|   2 | Sanjay | Jagtap | 50000   | 9011779086 |
|   3 | Suyash | Jagtap | 60000   | 8830322552 |
+-----+--------+--------+---------+------------+
3 rows in set (0.00 sec)

mysql> select * from Manager;
+-----+---------+--------+---------+------------+
| MID | MFName  | MLName | MSalary | MPhone     |
+-----+---------+--------+---------+------------+
|   1 | Sunil   | Jagtap | 400000  | 8668845659 |
|   2 | Shivram | Jagtap | 500000  | 4561779086 |
|   3 | Nilesh  | Jagtap | 600000  | 883789552  |
+-----+---------+--------+---------+------------+
3 rows in set (0.00 sec)

mysql> select Emp.EID, Emp.EFName, Manager.MID, Manager.MFName from Emp inner
join Manager on Emp.EID=Manager.MID;
+-----+--------+-----+---------+
| EID | EFName | MID | MFName  |
+-----+--------+-----+---------+
|   1 | Mayur  |   1 | Sunil   |
|   2 | Sanjay |   2 | Shivram |
|   3 | Suyash |   3 | Nilesh  |
+-----+--------+-----+---------+
3 rows in set (0.00 sec)

mysql> select Emp.EID, Emp.EFName, Manager.MID, Manager.MFName from Emp left
join Manager on Emp.EID=Manager.MID;
+-----+--------+------+---------+
| EID | EFName | MID  | MFName  |
+-----+--------+------+---------+
|   1 | Mayur  |    1 | Sunil   |
|   2 | Sanjay |    2 | Shivram |
|   3 | Suyash |    3 | Nilesh  |
+-----+--------+------+---------+
3 rows in set (0.00 sec)

mysql> select Emp.EID, Emp.EFName, Manager.MID, Manager.MFName from Emp right
join Manager on Emp.EID=Manager.MID;
+------+--------+-----+---------+
| EID  | EFName | MID | MFName  |
+------+--------+-----+---------+
|    1 | Mayur  |   1 | Sunil   |
|    2 | Sanjay |   2 | Shivram |
|    3 | Suyash |   3 | Nilesh  |
+------+--------+-----+---------+
3 rows in set (0.00 sec)

mysql> select Emp.EID,Emp.EFName,Emp.ELName,Emp.ESalary,Emp.EPhone, Manager.MID
from Emp left join Manager on Emp.EID=Manager.MID union select
Emp.EID,Emp.EFName,Emp.ELName,Emp.ESalary,Emp.EPhone, Manager.MID from Emp right
join Manager on Emp.EID=Manager.MID;
+------+--------+--------+---------+------------+------+
| EID  | EFName | ELName | ESalary | EPhone     | MID  |
+------+--------+--------+---------+------------+------+
|    1 | Mayur  | Jagtap | 40000   | 8668832059 |    1 |
|    2 | Sanjay | Jagtap | 50000   | 9011779086 |    2 |
|    3 | Suyash | Jagtap | 60000   | 8830322552 |    3 |
+------+--------+--------+---------+------------+------+
3 rows in set (0.00 sec)

mysql> select * from Emp E, Manager M where E.EID=M.MID;
+-----+--------+--------+---------+------------+-----+---------+--------+-------
--+------------+
| EID | EFName | ELName | ESalary | EPhone     | MID | MFName  | MLName |
MSalary | MPhone     |
+-----+--------+--------+---------+------------+-----+---------+--------+-------
--+------------+
|   1 | Mayur  | Jagtap | 40000   | 8668832059 |   1 | Sunil   | Jagtap | 400000
| 8668845659 |
|   2 | Sanjay | Jagtap | 50000   | 9011779086 |   2 | Shivram | Jagtap | 500000
| 4561779086 |
|   3 | Suyash | Jagtap | 60000   | 8830322552 |   3 | Nilesh  | Jagtap | 600000
| 883789552  |
+-----+--------+--------+---------+------------+-----+---------+--------+-------
--+------------+
3 rows in set (0.00 sec)
```

```
mysql> select * from Emp where EID=(select EID from Emp where EFName="Mayur");
+-----+--------+--------+---------+------------+
| EID | EFName | ELName | ESalary | EPhone     |
+-----+--------+--------+---------+------------+
|   1 | Mayur  | Jagtap | 40000   | 8668832059 |
+-----+--------+--------+---------+------------+
1 row in set (0.04 sec)
s
mysql> select * from Emp where EId=(select Manager.MID from Manager where
MFName="Nilesh");
+-----+--------+--------+---------+------------+
| EID | EFName | ELName | ESalary | EPhone     |
+-----+--------+--------+---------+------------+
|   3 | Suyash | Jagtap | 60000   | 8830322552 |
+-----+--------+--------+---------+------------+
1 row in set (0.00 sec)
```

Practical 4

Unnamed PL/SQL code block: Use of Control structure and Exception handling is mandatory. Suggested Problem statement:
Consider Tables:

1. **Borrower(Roll_no, Name, DateofIssue, NameofBook, Status)**
2. **Fine(Roll_no,Date,Amt)**

- **Accept Roll_no and NameofBook from user.**
- **Check the number of days (from date of issue).**
- **If days are between 15 to 30 then fine amount will be Rs 5per day.**
- **If no. of days>30, per day fine will be Rs 50 per day and for days less than 30, Rs. 5 per day.**
- **After submitting the book, status will change from I to R.**
- **If condition of fine is true, then details will be stored into fine table.**
- **Also handles the exception by named exception handler or user define exception handler.**

```
mysql> create database Practical4;

Query OK, 1 row affected (0.02 sec)

mysql> use Practical4;
Database changed

mysql> create table Borrower(Rollin int primary key, Name varchar(20),
DateofIssue date, NameofBook varchar(20), Status varchar(20));
Query OK, 0 rows affected (0.12 sec)

mysql> create table Fine(Roll_no int primary key, Date date, Amt varchar(20));
Query OK, 0 rows affected (0.15 sec)

mysql> insert into Borrower values(1, "Mayur","2022-07-01","HIJ","I"),(2,
"Suyash","2022-08-01","EFG","I"),(3, "Sanjay","2018-07-01","XYZ","I"),(4,
"Sunil","2023-07-01","ABC","I");
Query OK, 4 rows affected (0.06 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> select * from Borrower;
+--------+--------+-------------+------------+--------+
| Rollin | Name   | DateofIssue | NameofBook | Status |
+--------+--------+-------------+------------+--------+
|      1 | Mayur  | 2022-07-01  | HIJ        | I      |
|      2 | Suyash | 2022-08-01  | EFG        | I      |
|      3 | Sanjay | 2018-07-01  | XYZ        | I      |
|      4 | Sunil  | 2023-07-01  | ABC        | I      |
+--------+--------+-------------+------------+--------+
4 rows in set (0.00 sec)

mysql> delimiter $$
mysql> create procedure B(RollNo int, BookName varchar(20))
    ->     begin
    ->     declare X integer;
    ->     declare Continue handler for not found
    ->     begin
    ->     select &apos;NOT FOUND&apos;;
    ->     end;
    ->     select datediff(curdate(),DateofIssue) into X from Borrower where
Rollin=RollNo;
    ->     if(X>15&&X<30)
    ->     then
    ->     insert into Fine values(RollNo, curdate(),(X*5));
    ->     end if;
    ->     if(X>30)
    ->     then
    ->     insert into Fine values(RollNo,curdate(),(X*50));
    ->     end if;
    ->     update Borrower set status = &apos;R&apos; where Rollin=RollNo;
    ->     end;
    ->     $$
Query OK, 0 rows affected, 1 warning (0.05 sec)
```

```
mysql> call B(1,&apos;HIJ&apos;);$$
Query OK, 1 row affected (0.04 sec)
```

```
mysql> select * from Fine $$
+---------+------------+-------+
| Roll_no | Date       | Amt   |
+---------+------------+-------+
|       1 | 2023-11-04 | 24550 |
+---------+------------+-------+
1 row in set (0.00 sec)
```

```
mysql> select * from Borrower;$$
+--------+--------+-------------+------------+--------+
| Rollin | Name   | DateofIssue | NameofBook | Status |
+--------+--------+-------------+------------+--------+
|      1 | Mayur  | 2022-07-01  | HIJ        | R      |
|      2 | Suyash | 2022-08-01  | EFG        | I      |
|      3 | Sanjay | 2018-07-01  | XYZ        | I      |
|      4 | Sunil  | 2023-07-01  | ABC        | I      |
+--------+--------+-------------+------------+--------+
4 rows in set (0.00 sec)
```

Practical 5

Named PL/SQL Block: PL/SQL Stored Procedure and Stored Function.

- **Write a Stored Procedure namely proc_Grade for the categorization of student. If marks scoredby students in examination is <=1500 and marks>=990 then student will be placed in distinction category if marks scored are between 989 and 900 category is first class, if marks 899 and 825 category is Higher Second Class.**
- **Write a PL/SQL block to use procedure created with above requirement.**
  1. Stud_Marks(name, total_marks)
  2. Result(Roll,Name, Class)

```
mysql> create database Practical5;

Query OK, 1 row affected (0.02 sec)

mysql> use Practical5;
Database changed

mysql> create table Stud_Marks(Roll int primary key, name varchar(20),
total_marks varchar(20));
Query OK, 0 rows affected (0.17 sec)

mysql> create table Result(Roll int primary key, name varchar(20), class
varchar(20));
Query OK, 0 rows affected (0.18 sec)

mysql> insert into Stud_Marks values(1, "Mayur",1400),(2,"Sanjay",1400),(3,
"Sunil",1000),(4,"Suyash",980);
Query OK, 4 rows affected (0.07 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> delimiter $$
mysql> create procedure proc_Grade(in marks int, out class varchar(20))
    ->     begin
    ->     if(marks<1500&&marks>990)
    ->     then
    ->     set class='Distinction';
    ->     end if;
    ->     if(marks<989&&marks>890)
    ->     then
    ->     set class = 'First class';
    ->     end if;
    ->     if(marks<889 && marks>825)
    ->     then
    ->     set class='Higher Second Class';
    ->     end if;
    ->     if(marks<824&&marks>750)
    ->     then
    ->     set class='Second Class';
    ->     end if;
    ->     if (marks<749&&marks>650)
    ->     then
    ->     set class='Passed';
    ->     end if;
    ->     if(marks<649)
    ->     then
    ->     set class = 'Fail';
    ->     end if;
    ->     end;
    ->     $$
Query OK, 0 rows affected, 5 warnings (0.13 sec)
```

```
mysql> create function final_result(R1 int)
    ->     returns int
    ->     DETERMINISTIC
    ->     begin
    ->     declare fmarks integer;
    ->     declare grade varchar(20);
    ->     declare stud_name varchar(20);
    ->     select Stud_Marks.total_marks, Stud_Marks.name into fmarks,stud_name
from Stud_Marks where Stud_Marks.Roll=R1;
    ->     call proc_Grade(fmarks,@grade);
    ->     insert into Result values(R1,stud_name,@grade);
    ->     return R1;
    ->     end;
    ->     $$
Query OK, 0 rows affected (0.06 sec)

mysql> select final_result(1);
    -> $$
+-----------------+
| final_result(1) |
+-----------------+
|               1 |
+-----------------+
1 row in set (0.13 sec)

mysql> select final_result(2); $$
+-----------------+
| final_result(2) |
+-----------------+
|               2 |
+-----------------+
1 row in set (0.12 sec)
mysql> select final_result(3)$$
+-----------------+
| final_result(3) |
+-----------------+
|               3 |
+-----------------+
1 row in set (0.08 sec)

mysql> select final_result(4)$$
+-----------------+
| final_result(4) |
+-----------------+
|               4 |
+-----------------+
1 row in set (0.07 sec)

mysql> select * from Result$$
+------+--------+-------------+
| Roll | name   | class       |
+------+--------+-------------+
|    1 | Mayur  | Distinction |
|    2 | Sanjay | Distinction |
|    3 | Sunil  | Distinction |
|    4 | Suyash | First class |
+------+--------+-------------+
4 rows in set (0.00 sec)
```

Practical 6

Cursors: (All types: Implicit, Explicit, Cursor FOR Loop, Parameterized Cursor)

- **Write a PL/SQL block of code using parameterized Cursor that will merge the data availablein the newly created table N_RollCall with the data available in the table O_RollCall. If the data in the first table already exist in the second table then that data should be skipped.**

```
mysql> create database Practical6;

Query OK, 1 row affected (0.11 sec)

mysql> use Practical6;
Database changed

mysql> create table N_RollCall(roll_no int primary key, name varchar(20),
address varchar(20));
Query OK, 0 rows affected (0.11 sec)

mysql> create table O_RollCall(roll_no int primary key, name varchar(20),
address varchar(20));
Query OK, 0 rows affected (0.11 sec)

mysql> insert into O_RollCall values(1, "Mayur","Ozar"),(2,
"Sanjay","Nashik"),(3, "Sunil","Pune"),(4, "Suyash","Mumbai");
Query OK, 4 rows affected (0.07 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql> delimiter $$

mysql> create procedure p(in r1 int)
    ->     begin
    ->     declare r2 int;
    ->     declare exit_loop boolean;
    ->     declare c1 cursor for select roll_no from O_RollCall where
roll_no>r1;
    ->     declare continue handler for not found set exit_loop=true;
    ->     open c1;
    ->     e_loop:loop
    ->     fetch c1 into r2;
    ->     if not exists(select * from N_RollCall where roll_no=r2)
    ->     then
    ->     insert into N_RollCall select * from O_RollCall where roll_no=r2;
    ->     end if;
    ->     if exit_loop
    ->     then
    ->     close c1;
    ->     leave e_loop;
    ->     end if;
    ->     end loop e_loop;
    ->     end
    ->     $$
Query OK, 0 rows affected (0.07 sec)

mysql> call p(1);$$
Query OK, 0 rows affected (0.13 sec)

mysql> select * from N_RollCall$$
+---------+--------+---------+
| roll_no | name   | address |
+---------+--------+---------+
|       2 | Sanjay | Nashik  |
|       3 | Sunil  | Pune    |
|       4 | Suyash | Mumbai  |
+---------+--------+---------+
3 rows in set (0.00 sec)
```

mysql> call p (0);$$

```
Query OK, 0 rows affected (0.10 sec)
```

mysql> select * from N_RollCall$$

```
+---------+--------+---------+
| roll_no | name   | address |
+---------+--------+---------+
|       1 | Mayur  | Ozar    |
|       2 | Sanjay | Nashik  |
|       3 | Sunil  | Pune    |
|       4 | Suyash | Mumbai  |
+---------+--------+---------+
4 rows in set (0.00 sec)
```

Practical 7

Database Trigger (All Types: Row level and Statement level triggers, Before and After Triggers).

- **Write a database trigger on Library table. The System should keep track of the records that are being updated or deleted. The old value of updated or deleted records should be added in Library_Audit table.**

```
mysql> create database Practical7;
Query OK, 1 row affected (0.06 sec)

mysql> use Practical7;
Database changed

mysql> create table Borrower(Rollin int primary key, Name varchar(20),
DateofIssue date, NameofBook varchar(20));
Query OK, 0 rows affected (0.12 sec)

mysql> create table Borrower(Rollin int primary key, Name varchar(20),
DateofIssue date, NameofBook varchar(20), ts TIMESTAMP(0));
Query OK, 0 rows affected (0.12 sec)

mysql> insert into Borrower values(1, "Mayur","2018-06-10","Wings of
Fire","Available", "APJ"),(2, "Suyash","2019-06-10","XYZ","Available",
"MSJ"),(3,"Sunil","2022-10-4","ABC","Available","SSJ");
Query OK, 3 rows affected (0.05 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql> insert into audit1 values(1, "Suyash","2019-06-10","XYZ","Available",
"MSJ","15:46:13"),(2,"Sunil","2022-10-4","ABC","Available","SSJ","15:46:13");
Query OK, 2 rows affected (0.03 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from Borrower;
+--------+--------+-------------+---------------+-----------+--------+
| Rollin | Name   | DateofIssue | NameofBook    | Status    | Author |
+--------+--------+-------------+---------------+-----------+--------+
|      1 | Mayur  | 2018-06-10  | Wings of Fire | Available | APJ    |
|      2 | Suyash | 2019-06-10  | XYZ           | Available | MSJ    |
|      3 | Sunil  | 2022-10-04  | ABC           | Available | SSJ    |
+--------+--------+-------------+---------------+-----------+--------+
3 rows in set (0.00 sec)

mysql> select * from Borrower;
+--------+--------+-------------+---------------+-----------+--------+
| Rollin | Name   | DateofIssue | NameofBook    | Status    | Author |
+--------+--------+-------------+---------------+-----------+--------+
|      1 | Mayur  | 2018-06-10  | Wings of Fire | Available | APJ    |
|      2 | Suyash | 2019-06-10  | XYZ           | Available | MSJ    |
|      3 | Sunil  | 2022-10-04  | ABC           | Available | SSJ    |
+--------+--------+-------------+---------------+-----------+--------+
3 rows in set (0.00 sec)
```

mysql> delimiter $$

mysql> create trigger library after insert on Borrower for each row

```
    ->     begin
    ->     insert into audit1 values(new.RollIn, new.Name, new.DateofIssue,
new.NameofBook,new.Status, new.Author, current_timestamp);
    ->     end;
    ->     $$
Query OK, 0 rows affected (0.12 sec)
```

mysql> select * from Borrower; $$

```
+--------+--------+-------------+---------------+-----------+--------+
| Rollin | Name   | DateofIssue | NameofBook    | Status    | Author |
+--------+--------+-------------+---------------+-----------+--------+
|      1 | Mayur  | 2018-06-10  | Wings of Fire | Available | APJ    |
|      2 | Suyash | 2019-06-10  | XYZ           | Available | MSJ    |
|      3 | Sunil  | 2022-10-04  | ABC           | Available | SSJ    |
+--------+--------+-------------+---------------+-----------+--------+
3 rows in set (0.00 sec)
```

mysql> select * from audit1$$

```
+--------+--------+-------------+------------+-----------+--------+----------+
| RollIn | Name   | DateofIssue | NameofBook | Status    | Author | ts       |
+--------+--------+-------------+------------+-----------+--------+----------+
|      1 | Suyash | 2019-06-10  | XYZ        | Available | MSJ    | 15:46:13 |
|      2 | Sunil  | 2022-10-4   | ABC        | Available | SSJ    | 15:46:13 |
+--------+--------+-------------+------------+-----------+--------+----------+
2 rows in set (0.00 sec)
```

mysql> delimiter $$

mysql> create trigger library1 after update on Borrower for each row

```
    ->     begin
    ->     insert into audit1 values(new.RollIn, new.Name, new.DateofIssue,
new.NameofBook, new.Status, new.Author, current_timestamp);
    ->     end;
    ->     $$
Query OK, 0 rows affected (0.03 sec)
```

mysql> update Borrower set RollIn=4, NameofBook="Nityaseva" where Name="Mayur"$$

```
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

mysql> select * from Borrower; -> $$

```
+--------+--------+-------------+------------+-----------+--------+
| Rollin | Name   | DateofIssue | NameofBook | Status    | Author |
+--------+--------+-------------+------------+-----------+--------+
|      2 | Suyash | 2019-06-10  | XYZ        | Available | MSJ    |
|      3 | Sunil  | 2022-10-04  | ABC        | Available | SSJ    |
|      4 | Mayur  | 2018-06-10  | Nityaseva  | Available | APJ    |
+--------+--------+-------------+------------+-----------+--------+
3 rows in set (0.00 sec)
```

Practical 8

Database Connectivity:

- **Write a program to implement MySQL/Oracle database connectivity with any front end language to implement Database navigation operations (add, delete, edit etc.)**

```
import java.awt.*;
import java.awt.event.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.*;
public class student extends JFrame implements ActionListener{
      JFrame f;
      JLabel l1, l2,l3,l4;
      JTextField t1, t2,t3;
      JButton b1, b2, b3, b4, b5;
      Connection c;
      Statement s;
      ResultSet  r;
      student (){
            try{
                  f=new JFrame("Student Form");
                  f.setLayout(null);f.setVisible(true);
                  f.setSize(700, 500);
                  l4=new JLabel("Student Management System");

                  l4.setBounds(100, 30, 400, 30);
                  f.add(l4);
                  l4.setForeground(Color.blue);
                  l4.setFont(new Font("Serif", Font.BOLD,30));
                  l1=new JLabel("Stud_RollNo");
                  l1.setBounds(50, 70, 100, 50);
                  f.add(l1);
                  l2=new JLabel("Stud_Name");
                  l2.setBounds(50, 120, 100, 50);
                  f.add(l2);
                  l3=new
                  JLabel("Stud_Dept");
                  l3.setBounds(50, 170, 100, 50);
                  f.add(l3);
                  t1=new JTextField();
                  t1.setBounds(150, 90, 100, 30);
                  f.add(t1);
                  t2=new JTextField();
                  t2.setBounds(150, 140, 100, 30);
                  f.add(t2);t3=new JTextField();
                  t3.setBounds(150, 190, 100, 30);
                  f.add(t3);
                  b1= new JButton("ADD");
                  b1.setBounds(200, 300, 75, 50);
                  f.add(b1);
                  b1.addActionListener(this);
                  b2= new JButton("EDIT");
                  b2.setBounds(300, 300, 75, 50);
                  f.add(b2);
                  b2.addActionListener(this);
                  b3= new JButton("DELETE");b3.setBounds(400, 300, 75, 50);
                  f.add(b3);
                  b3.addActionListener(this);
```

```
                        b5= new JButton("EXIT");
                        b5.setBounds(500, 300, 75, 50);
                        f.add(b5);
                        b5.addActionListener(this);
                        Class.forName("com.mysql.jdbc.Driver");

        c=DriverManager.getConnection("jdbc:mysql://localhost:3306/info","root","r
oot");s=c.createStatement();
                }catch(Exception e){
                        System.out.println(e);
                }
        }
        public void actionPerformed(ActionEvent ae){
                try{
                        if(ae.getSource()==b1){
                                String s1="INSERT INTO
result(stud_RollNo,stud_Name,stud_Dept) VALUES("+t1.getText()+"','"+t2.getText()
+"','"+t3.getText() + "')";
                                System.out.println(s1);
                                s.executeUpdate(s1);
                                r=s.executeQuery("SELECT * FROM result");
                                t1.setText("");
                                t2.setText("");
                                t3.setText("");
                        }else if(ae.getSource()==b2){
                                String s2="UPDATE result SET
stud_Name='"+t2.getText()+"' WHERE stud_RollNo="+t1.getText();
                                System.out.println(s2);
                                s.executeUpdate(s2);
                                r=s.executeQuery("SELECT * FROM result");
                                t1.setText("");
                                t2.setText("");t3.setText("");
                        }else if(ae.getSource()==b3) {
                                String s3="DELETE FROM result WHERE
stud_RollNo="+t1.getText();
                                System.out.println(s3);
                                s.executeUpdate(s3);
                                r=s.executeQuery("SELECT * FROM result");
                                t1.setText("");
                                t2.setText("");
                                t3.setText("");}else
if(ae.getSource()==b5){System.exit(0);
                        }
                }catch(Exception e){
                        System.out.println(e);
                }
        }
        public static void main(String args[]){
                new student();
        }
}
```

Practical 9

MongoDB Queries:

- **Design and Develop MongoDB Queries using CRUD operations. (Use CRUD operations, SAVE method, logical operators etc.).**

```
test> use Mayur;
switched to db Mayur

Mayur> db.createCollection("Student");
{ ok: 1 }

Mayur> db.Student.insert( { 'RollNo':'1','Name':'Mayur','Class':'TECO'});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne,
insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545b8daf85a719d8aa02c78") }
}

Mayur> db.Student.insert( {
'RollNo':'2','Name':'Sunil','Class':'TECO'},{'RollNo':'3','Name':'Sanjay','Class
':'TECE'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545b935f85a719d8aa02c79") }
}

Mayur> db.Student.insert({'RollNo':'3','Name':'Sanjay','Class':'TECE'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545b959f85a719d8aa02c7a") }
}
Mayur> db.Student.find();
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]

Mayur> db.Student.find().pretty();
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]
```

```
Mayur> db.Student.update({'Name':'Mayur'},{$set:{'Name':'Mayur Jagtap'}});
DeprecationWarning: Collection.update() is deprecated. Use updateOne,
updateMany, or bulkWrite.
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

```
Mayur> db.Student.remove({'ADD':'MP'});
DeprecationWarning: Collection.remove() is deprecated. Use deleteOne,
deleteMany, findOneAndDelete, or bulkWrite.
{ acknowledged: true, deletedCount: 0 }
```

```
Mayur> db.Student.find({$and:[{'Name':'Mayur Jagtap'},{'RollNo':'1'}]});
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur Jagtap',
    Class: 'TECO'
  }
]
Mayur> db.Student.find({$or:[{'Name':'Mayur Jagtap'},{'RollNo':'2'}]});
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur Jagtap',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  }
]
```

```
Mayur> db.Student.find({$nor:[{'Name':'Mayur Jagtap'},{'Class':'TECO'}]});
[
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]

Mayur> db.Student.find({'RollNo':{$not:{$lt:'3'}}}).pretty();
[
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]

Mayur> db.Student.find({'RollNo':{$eq:'3'}}).pretty();
[
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]

Mayur> db.Student.find({'RollNo':{$ne:'3'}}).pretty();
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur Jagtap',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  }
]

Mayur> db.Student.find({'RollNo':{$gt:'2'}}).pretty();
[
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]

Mayur> db.Student.find({'RollNo':{$gte:'2'}}).pretty();
[
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b959f85a719d8aa02c7a"),
    RollNo: '3',
    Name: 'Sanjay',
    Class: 'TECE'
  }
]
```

```
Mayur> db.Student.find({'RollNo':{$lt:'2'}}).pretty();
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur Jagtap',
    Class: 'TECO'
  }
]

Mayur> db.Student.find({'RollNo':{$lte:'2'}}).pretty();
[
  {
    _id: ObjectId("6545b8daf85a719d8aa02c78"),
    RollNo: '1',
    Name: 'Mayur Jagtap',
    Class: 'TECO'
  },
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  }
]

Mayur> db.Student.find({'RollNo':{$lt:'3',$gt:'1'}}).pretty();
[
  {
    _id: ObjectId("6545b935f85a719d8aa02c79"),
    RollNo: '2',
    Name: 'Sunil',
    Class: 'TECO'
  }
]
```

Practical 10

MongoDB - Aggregation and Indexing:

- **Design and Develop MongoDB Queries using aggregation and indexing with suitable example using MongoDB.**

```
test> use Practical10
switched to db Practical10

Practical10> db.createCollection('website');
{ ok: 1 }

Practical10>
db.website.insert({'roll':'1','name':'harsh','amount':1000,'url':'www.yahoo.com'
});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne,
insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545be50582566230a408804") }
}
Practical10>
db.website.insert({'roll':'2','name':'jitesh','amount':2000,'url':'www.yahoo.com
'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545be50582566230a408805") }
}
Practical10>
db.website.insert({'roll':'3','name':'rina','amount':3000,'url':'www.google.com'
});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545be53582566230a408806") }
}

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$sum:"$amount"}}});
[
  { _id: 'harsh', total: 1000 },
  { _id: 'rina', total: 3000 },
  { _id: 'jitesh', total: 2000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$avg:"$amount"}}});
[
  { _id: 'jitesh', total: 2000 },
  { _id: 'harsh', total: 1000 },
  { _id: 'rina', total: 3000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$min:"$amount"}}});
[
  { _id: 'jitesh', total: 2000 },
  { _id: 'harsh', total: 1000 },
  { _id: 'rina', total: 3000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$max:"$amount"}}});
```

```
[
  { _id: 'harsh', total: 1000 },
  { _id: 'jitesh', total: 2000 },
  { _id: 'rina', total: 3000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$first:"$amount"}}});
[
  { _id: 'jitesh', total: 2000 },
  { _id: 'harsh', total: 1000 },
  { _id: 'rina', total: 3000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$last:"$amount"}}});
[
  { _id: 'jitesh', total: 2000 },
  { _id: 'rina', total: 3000 },
  { _id: 'harsh', total: 1000 }
]

Practical10>
db.website.aggregate({$group:{_id:"$name","total":{$push:"$amount"}}});
[
  { _id: 'jitesh', total: [ 2000 ] },
  { _id: 'rina', total: [ 3000 ] },
  { _id: 'harsh', total: [ 1000 ] }
]

Practical10> db.website.aggregate({$group:{_id:"$name","total": {$sum:1}}});
[
  { _id: 'jitesh', total: 1 },
  { _id: 'rina', total: 1 },
  { _id: 'harsh', total: 1 }
]

Practical10> db.createCollection('website1');
{ ok: 1 }

Practical10> db.website1.insert({'r':1,'name':'harsh'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545beef582566230a408807") }
}

Practical10> db.website1.find().pretty()
[ { _id: ObjectId("6545beef582566230a408807"), r: 1, name: 'harsh' } ]

Practical10> db.website1.createIndex({'name':1})
name_1

Practical10> db.website1.createIndex({'name':-1})
name_-1

Practical10> db.website1.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: 1 }, name: 'name_1' },
  { v: 2, key: { name: -1 }, name: 'name_-1' }
]

Practical10> db.website1.dropIndex({'name':1})
{ nIndexesWas: 3, ok: 1 }
```

```
Practical10> db.website1.getIndexes()
[
  { v: 2, key: { _id: 1 }, name: '_id_' },
  { v: 2, key: { name: -1 }, name: 'name_-1' }
]
```

Practical 11

MongoDB - Map reduces operations:

- **Implement Map reduces operation with suitable example using MongoDB.**

```
test> use Practical11
switched to db Practical11

Practical11> db.createCollection('Journal');
{ ok: 1 }

Practical11>
db.Journal.insert({'book_id':1,'book_name':'JavacdOOP','amt':500,'status':'Avail
able'});
DeprecationWarning: Collection.insert() is deprecated. Use insertOne,
insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545c08f3dd8efae092febc2") }
}

Practical11>
db.Journal.insert({'book_id':1,'book_name':'JavaOOP','amt':400,'status':'NotAvai
lable'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545c09c3dd8efae092febc3") }
}

Practical11>
db.Journal.insert({'book_id':1,'book_name':'Java','amt':300,'status':'Not
Available'});
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("6545c0a93dd8efae092febc4") }
}

Practical11> var mapfunction=function(){ emit(this.book_id,this.amt)};

Practical11> var reducefunction=function(key,value){return Array.sum(value);};

Practical11> db.Journal.mapReduce(mapfunction,reducefunction,{'out':'new'});
DeprecationWarning: Collection.mapReduce() is deprecated. Use an aggregation
instead.
See https://docs.mongodb.com/manual/core/map-reduce for details.
{ result: 'new', ok: 1 }
```