Experiment No - 01

Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym, different constraints etc. Write at least 10 SQL queries on the suitable database application using SQL DML statements. Note: Instructor will design the queries which demonstrate the use of concepts like Insert, Select, Update, Delete with operators, functions, and set operator etc.

1.mysql> CREATE DATABASE school;

2.mysql> USE school;
```
Database changed
```
3.mysql> CREATE TABLE students (id INT PRIMARY KEY, name VARCHAR(100), age INT, grade VARCHAR(10));

4.mysql> CREATE TABLE students_copy AS SELECT * FROM students;

5.mysql> SHOW TABLES;
```
+------------------+
| Tables_in_school |
+------------------+
| students         |
| students_copy    |
+------------------+
2 rows in set (0.01 sec)
```

6.mysql> CREATE TABLE students_specific_fields AS SELECT id, name FROM students;

7.mysql> DESCRIBE students_copy;
```
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | NO   |     | NULL    |       |
| name  | varchar(100) | YES  |     | NULL    |       |
| age   | int          | YES  |     | NULL    |       |
| grade | varchar(10)  | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

8.mysql> DESCRIBE students_specific_fields;
```
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | NO   |     | NULL    |       |
| name  | varchar(100) | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

9.mysql> CREATE TABLE students_excluding_record AS SELECT * FROM students WHERE id != 1;

10.mysql> DESCRIBE students_excluding_record;

```
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | NO   |     | NULL    |       |
| name  | varchar(100) | YES  |     | NULL    |       |
| age   | int          | YES  |     | NULL    |       |
| grade | varchar(10)  | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

11.mysql> CREATE TABLE students_no_records AS SELECT * FROM students WHERE 1 = 0;

12.mysql> ALTER TABLE students ADD email VARCHAR(100);

13.mysql> DESCRIBE students;

```
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | NO   | PRI | NULL    |       |
| name  | varchar(100) | YES  |     | NULL    |       |
| age   | int          | YES  |     | NULL    |       |
| grade | varchar(10)  | YES  |     | NULL    |       |
| email | varchar(100) | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

14.mysql> ALTER TABLE students MODIFY name VARCHAR(200);

15.mysql> DESCRIBE students;

```
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | NO   | PRI | NULL    |       |
| name  | varchar(200) | YES  |     | NULL    |       |
| age   | int          | YES  |     | NULL    |       |
| grade | varchar(10)  | YES  |     | NULL    |       |
| email | varchar(100) | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

16.mysql> ALTER TABLE students DROP COLUMN email;

17.mysql> ALTER TABLE students RENAME TO pupils;

18.mysql> SHOW TABLES;

```
+---------------------------+
| Tables_in_school          |
+---------------------------+
| pupils                    |
| students_copy             |
| students_excluding_record |
| students_no_records       |
| students_specific_fields  |
+---------------------------+
5 rows in set (0.00 sec)
```

19.mysql> ALTER TABLE PUPILS RENAME COLUMN name TO full_name;

20.mysql> DESCRIBE pupils;

```
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| id        | int          | NO   | PRI | NULL    |       |
| full_name | varchar(200) | YES  |     | NULL    |       |
| age       | int          | YES  |     | NULL    |       |
| grade     | varchar(10)  | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

21.mysql> CREATE VIEW all_pupils_view AS SELECT * FROM pupils; SELECT * FROM all_pupils_view;

```
ERROR 1050 (42S01): Table 'all_pupils_view' already exists
+----+-----------+------+-------+
| id | full_name | age  | grade |
+----+-----------+------+-------+
|  1 | Ali       |   20 | TY    |
|  2 | Rohan     |   20 | TY    |
|  3 | Tanmay    |   20 | TY    |
|  4 | Atharava  |   20 | TY    |
+----+-----------+------+-------+
4 rows in set (0.00 sec)
```

22.mysql> CREATE VIEW specific_fields_pupils_view AS SELECT id, full_name FROM pupils; SELECT * FROM specific_fields_pupils_view;

```
+----+-----------+
| id | full_name |
+----+-----------+
|  1 | Ali       |
|  2 | Rohan     |
|  3 | Tanmay    |
|  4 | Atharava  |
+----+-----------+
4 rows in set (0.00 sec)
```

23.mysql> CREATE VIEW specific_pupil_record_view AS SELECT * FROM pupils WHERE id = 1; SELECT * FROM specific_pupil_record_view;

```
+----+-----------+------+-------+
| id | full_name | age  | grade |
+----+-----------+------+-------+
|  1 | Ali       |   20 | TY    |
+----+-----------+------+-------+
1 row in set (0.00 sec)
```

24.mysql> CREATE OR REPLACE VIEW pupils_age_above_10_view AS SELECT * FROM pupils WHERE age > 10; SELECT * FROM pupils_age_above_10_view;

```
+----+-----------+------+-------+
| id | full_name | age  | grade |
+----+-----------+------+-------+
|  1 | Ali       |   20 | TY    |
|  2 | Rohan     |   20 | TY    |
|  3 | Tanmay    |   20 | TY    |
|  4 | Atharava  |   20 | TY    |
+----+-----------+------+-------+
4 rows in set (0.00 sec)
```

25.mysql> SELECT * FROM pupils;

```
+----+-----------+------+-------+
| id | full_name | age  | grade |
+----+-----------+------+-------+
|  1 | Ali       |   20 | TY    |
|  2 | Rohan     |   20 | TY    |
|  3 | Tanmay    |   20 | TY    |
|  4 | Atharava  |   20 | TY    |
+----+-----------+------+-------+
4 rows in set (0.00 sec)
```

26.mysql> SELECT * FROM pupils WHERE id = 1;

```
+----+-----------+------+-------+
| id | full_name | age  | grade |
+----+-----------+------+-------+
|  1 | Ali       |   20 | TY    |
+----+-----------+------+-------+
1 row in set (0.00 sec)
```

27.mysql> UPDATE pupils SET grade = 'Final Year' WHERE id = 1;

28.mysql> SELECT * FROM pupils WHERE id = 1;

```
+----+-----------+------+------------+
| id | full_name | age  | grade      |
+----+-----------+------+------------+
|  1 | Ali       |   20 | Final Year |
+----+-----------+------+------------+
1 row in set (0.00 sec)
```

29.mysql> DELETE FROM pupils WHERE id = 2;

30.mysql> SELECT * FROM pupils;

```
+----+-----------+------+------------+
| id | full_name | age  | grade      |
+----+-----------+------+------------+
|  1 | Ali       |   20 | Final Year |
|  3 | Tanmay    |   20 | TY         |
|  4 | Atharava  |   20 | TY         |
+----+-----------+------+------------+
3 rows in set (0.00 sec)
```

31.mysql> CREATE INDEX idx_full_name ON pupils (full_name);
show indexes from pupils;

```
+--------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------
+------+------------+---------+---------------+---------+------------+
| Table  | Non_unique | Key_name      | Seq_in_index | Column_name | Collation | Cardinality |
Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
+--------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------
+------+------------+---------+---------------+---------+------------+
| pupils |          0 | PRIMARY       |            1 | id          | A         |           3 |     NULL |   NULL |      |
BTREE      |         |               | YES     | NULL       |
| pupils |          1 | idx_full_name |            1 | full_name   | A         |           3 |     NULL |   NULL |
YES  | BTREE      |         |
  | YES     | NULL       |
+--------+------------+---------------+--------------+-------------+-----------+-------------+----------+--------
+------+------------+---------+---------------+---------+------------+
2 rows in set (0.01 sec)
```

Experiment No - 02

SQL Queries – all types of Join, Sub-Query and View: Write at least10 SQL queries for suitable database application using SQL DML statements. Note: Instructor will design the queries which demonstrate the use of concepts like all types of Join ,Sub-Query and View

```
mysql> use school;
Database changed
mysql> CREATE TABLE pupils (id INT, full_name VARCHAR(50), age INT, grade VARCHAR(20)); CREATE TABLE classes (class_id INT, subject VARCHAR(50), teacher VARCHAR(50)); INSERT INTO pupils (id, full_name, age, grade) VALUES (1, 'Ali', 20, 'TY'), (2, 'Rohan', 20, 'TY'), (3, 'Tanmay', 20, 'TY'), (4, 'Atharava', 20, 'TY'); INSERT INTO classes (class_id, subject, teacher) VALUES (1, 'Math', 'Mr. Sharma'), (2, 'Science', 'Ms. Gupta'), (3, 'History', 'Mr. Khan');
```

```
mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils INNER JOIN classes ON pupils.id = classes.class_id;
+-----------+---------+------------+
| full_name | subject | teacher    |
+-----------+---------+------------+
| Ali       | Math    | Mr. Sharma |
| Rohan     | Science | Ms. Gupta  |
| Tanmay    | History | Mr. Khan   |
+-----------+---------+------------+
3 rows in set (0.00 sec)
```

```
mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils LEFT JOIN classes ON pupils.id = classes.class_id;
+-----------+---------+------------+
| full_name | subject | teacher    |
+-----------+---------+------------+
| Ali       | Math    | Mr. Sharma |
| Rohan     | Science | Ms. Gupta  |
| Tanmay    | History | Mr. Khan   |
| Atharava  | NULL    | NULL       |
+-----------+---------+------------+
4 rows in set (0.00 sec)
```

```
mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils RIGHT JOIN classes ON pupils.id = classes.class_id;
+-----------+---------+------------+
| full_name | subject | teacher    |
+-----------+---------+------------+
| Ali       | Math    | Mr. Sharma |
| Rohan     | Science | Ms. Gupta  |
| Tanmay    | History | Mr. Khan   |
+-----------+---------+------------+
3 rows in set (0.00 sec)
```

mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils FULL OUTER JOIN classes ON pupils.id = classes.class_id;

mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils LEFT JOIN classes ON pupils.id = classes.class_id UNION SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils RIGHT JOIN classes ON pupils.id = classes.class_id;

```
+-----------+---------+------------+
| full_name | subject | teacher    |
+-----------+---------+------------+
| Ali       | Math    | Mr. Sharma |
| Rohan     | Science | Ms. Gupta  |
| Tanmay    | History | Mr. Khan   |
| Atharava  | NULL    | NULL       |
+-----------+---------+------------+
4 rows in set (0.00 sec)
```

mysql> SELECT pupils.full_name, classes.subject, classes.teacher FROM pupils CROSS JOIN classes;

```
+-----------+---------+------------+
| full_name | subject | teacher    |
+-----------+---------+------------+
| Ali       | History | Mr. Khan   |
| Ali       | Science | Ms. Gupta  |
| Ali       | Math    | Mr. Sharma |
| Rohan     | History | Mr. Khan   |
| Rohan     | Science | Ms. Gupta  |
| Rohan     | Math    | Mr. Sharma |
| Tanmay    | History | Mr. Khan   |
| Tanmay    | Science | Ms. Gupta  |
| Tanmay    | Math    | Mr. Sharma |
| Atharava  | History | Mr. Khan   |
| Atharava  | Science | Ms. Gupta  |
| Atharava  | Math    | Mr. Sharma |
+-----------+---------+------------+
12 rows in set (0.00 sec)
```

mysql> SELECT full_name FROM pupils WHERE id = (SELECT MAX(id) FROM pupils);

```
+-----------+
| full_name |
+-----------+
| Atharava  |
+-----------+
1 row in set (0.00 sec)
```

mysql> SELECT full_name FROM pupils WHERE grade = (SELECT grade FROM pupils WHERE full_name = 'Ali');

```
+-----------+
| full_name |
+-----------+
| Ali       |
| Rohan     |
| Tanmay    |
| Atharava  |
+-----------+
4 rows in set (0.00 sec)
```

mysql> CREATE VIEW view_all_pupils AS SELECT * FROM pupils; SELECT * FROM view_all_pupils;

```
+------+-----------+------+-------+
| id   | full_name | age  | grade |
+------+-----------+------+-------+
|    1 | Ali       |   20 | TY    |
|    2 | Rohan     |   20 | TY    |
|    3 | Tanmay    |   20 | TY    |
|    4 | Atharava  |   20 | TY    |
+------+-----------+------+-------+
4 rows in set (0.00 sec)
```

mysql> CREATE VIEW view_pupil_details AS SELECT full_name, grade FROM pupils; SELECT * FROM view_pupil_details;

```
+-----------+-------+
| full_name | grade |
+-----------+-------+
| Ali       | TY    |
| Rohan     | TY    |
| Tanmay    | TY    |
| Atharava  | TY    |
+-----------+-------+
4 rows in set (0.00 sec)
```

mysql> SELECT full_name, age FROM pupils WHERE age > (SELECT AVG(age) FROM pupils);

```
Empty set (0.00 sec)
```

Experiment No - 03

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

```
$ mongo

> use book
switched to db book

> show collections
library

> db.createCollection("library")
{ "ok" : 1 }

> db.library.insert({"bid":1,"name":"C++"})
WriteResult({ "nInserted" : 1 })

> db.library.insert({"bid":2,"name":"SEPM","author":"Pressman"})
WriteResult({ "nInserted" : 1 })

> db.library.insert({"bid":3,"name":"CN","author":"Forouzan","cost":700})
WriteResult({ "nInserted" : 1 })

> db.library.find().pretty()
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"),
        "bid" : 1,
        "name" : "C++"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}

> db.library.remove({"bid":1})
WriteResult({ "nRemoved" : 1 })
```

```
> db.library.count()
2

> db.library.find().pretty()
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}

> db.library.insert({"bid":1,"name":"C++"})
WriteResult({ "nInserted" : 1 })

> db.library.find().pretty()
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"),
        "bid" : 1,
        "name" : "C++"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"),
        "bid" : 1,
        "name" : "C++"
}

> db.library.find().sort({"bid":1})
```

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" :
"Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"), "bid" : 3, "name" : "CN", "author" : "Forouzan",
"cost" : 700 }
```

> db.library.insert({"bid":4,"name":"SPOS","author":"Pearson","cost":500})

```
WriteResult({ "nInserted" : 1 })
```

> db.library.find().pretty()

```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"),
        "bid" : 1,
        "name" : "C++"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"),
        "bid" : 4,
        "name" : "SPOS",
        "author" : "Pearson",
        "cost" : 500
}
```

> db.library.find().sort({"bid":1})

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" :
"Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"), "bid" : 3, "name" : "CN", "author" : "Forouzan",
"cost" : 700 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"), "bid" : 4, "name" : "SPOS", "author" : "Pearson",
"cost" : 500 }
```

> db.library.find({$and:[{"name":"CN"},{"cost":700}]}).pretty()

```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}
```

> db.library.insert({"bid":5,"name":"TOC","author":"Addison-Wesley","cost":600})
```
WriteResult({ "nInserted" : 1 })
```

> db.library.insert({"bid":6,"name":"AI","author":"McGraw Hill Education","cost":800})
```
WriteResult({ "nInserted" : 1 })
```

> db.library.find().pretty()
```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"),
        "bid" : 1,
        "name" : "C++"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"),
        "bid" : 4,
        "name" : "SPOS",
        "author" : "Pearson",
        "cost" : 500
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"),
        "bid" : 5,
        "name" : "TOC",
        "author" : "Addison-Wesley",
        "cost" : 600
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"),
        "bid" : 6,
        "name" : "AI",
        "author" : "McGraw Hill Education",
        "cost" : 800
}
```

> db.library.find({$or:[{"cost":500},{"cost":800}]}).pretty()

```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"),
        "bid" : 4,
        "name" : "SPOS",
        "author" : "Pearson",
        "cost" : 500
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"),
        "bid" : 6,
        "name" : "AI",
        "author" : "McGraw Hill Education",
        "cost" : 800
}
```

> db.library.find({"cost":{$ne:500}})

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" : "Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"), "bid" : 3, "name" : "CN", "author" : "Forouzan", "cost" : 700 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"), "bid" : 5, "name" : "TOC", "author" : "Addison-Wesley", "cost" : 600 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"), "bid" : 6, "name" : "AI", "author" : "McGraw Hill Education", "cost" : 800 }
```

> db.library.find({$nor:[{"cost":500},{"author":"Forouzan"}]})

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" : "Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"), "bid" : 5, "name" : "TOC", "author" : "Addison-Wesley", "cost" : 600 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"), "bid" : 6, "name" : "AI", "author" : "McGraw Hill Education", "cost" : 800 }
```

> db.library.find({"cost":{$not:{$gt:800}}})

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" : "Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"), "bid" : 3, "name" : "CN", "author" : "Forouzan", "cost" : 700 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"), "bid" : 5, "name" : "TOC", "author" : "Addison-
Wesley", "cost" : 600 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"), "bid" : 6, "name" : "AI", "author" : "McGraw
Hill Education", "cost" : 800 }
```

> db.library.insert({"bid":7,"name":"CC","author":"Wiley Publications","cost":400})

```
WriteResult({ "nInserted" : 1 })
```

> db.library.find()

```
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"), "bid" : 1, "name" : "C++" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"), "bid" : 2, "name" : "SEPM", "author" :
"Pressman" }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"), "bid" : 3, "name" : "CN", "author" : "Forouzan",
"cost" : 700 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"), "bid" : 4, "name" : "SPOS", "author" : "Pearson",
"cost" : 500 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"), "bid" : 5, "name" : "TOC", "author" : "Addison-
Wesley", "cost" : 600 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"), "bid" : 6, "name" : "AI", "author" : "McGraw
Hill Education", "cost" : 800 }
{ "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"), "bid" : 7, "name" : "CC", "author" : "Wiley
Publications", "cost" : 400 }
```

> db.library.update({'cost':400},{$set:{'cost':600}})

```
Modified document count: 1
```

> db.library.update({'cost':800},{$set:{'cost':1200}})

```
Modified document count: 1
```

> db.library.find().pretty()

```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9c"),
        "bid" : 1,
        "name" : "C++"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9a"),
        "bid" : 2,
        "name" : "SEPM",
        "author" : "Pressman"
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9b"),
        "bid" : 3,
        "name" : "CN",
        "author" : "Forouzan",
        "cost" : 700
}
```

```
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9d"),
        "bid" : 4,
        "name" : "SPOS",
        "author" : "Pearson",
        "cost" : 500
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"),
        "bid" : 5,
        "name" : "TOC",
        "author" : "Addison-Wesley",
        "cost" : 600
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9f"),
        "bid" : 6,
        "name" : "AI",
        "author" : "McGraw Hill Education",
        "cost" : 1200
}
{
        "_id" : ObjectId("60c72b2f5f1b2c001c8fbb9e"),
        "bid" : 7,
        "name" : "CC",
        "author" : "Wiley Publications",
        "cost" : 600
}
```

Experiment No - 04

Use of Control structure and Exception handling is mandatory. Library & Fine Management

```
mysql> CREATE DATABASE library_db;
Query OK, 1 row affected (0.01 sec)

mysql> USE library_db;
Database changed

mysql> CREATE TABLE Borrower (
   -> Roll_no INT PRIMARY KEY,
   -> Name VARCHAR(100),
   -> Date_of_Issue DATE,
   -> Name_of_Book VARCHAR(100),
   -> Status CHAR(1) CHECK (Status IN ('I', 'R'))
   -> );
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE Fine (
   -> Roll_no INT,
   -> Date DATE,
   -> Amt DECIMAL(10, 2),
   -> FOREIGN KEY (Roll_no) REFERENCES Borrower(Roll_no)
   -> );
Query OK, 0 rows affected (0.02 sec)

mysql> INSERT INTO Borrower (Roll_no, Name, Date_of_Issue, Name_of_Book, Status)
VALUES (1, 'John Doe', '2024-09-01', 'C++', 'I');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO Borrower (Roll_no, Name, Date_of_Issue, Name_of_Book, Status)
VALUES (2, 'Jane Smith', '2024-09-15', 'SEPM', 'I');
Query OK, 1 row affected (0.01 sec)

mysql> DELIMITER //

mysql> CREATE PROCEDURE CalculateFine(IN roll_no INT, IN name_of_book
VARCHAR(255))
   -> BEGIN
   ->    DECLARE v_date_of_issue DATE;
   ->    DECLARE v_status CHAR(1);
   ->    DECLARE v_fine_amount DECIMAL(10, 2);
   ->    DECLARE v_days_borrowed INT;
   ->    DECLARE v_error_message VARCHAR(255);
   ->
   ->     -- Exception handler for book not found
   ->     DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
   ->     BEGIN
   ->        SET v_error_message = 'Error: The book is not currently borrowed by the user.';
   ->        SELECT v_error_message;
   ->     END;
   ->     SELECT Date_of_Issue, Status INTO v_date_of_issue, v_status
   ->     FROM Borrower WHERE Roll_no = roll_no AND Name_of_Book = name_of_book;
   ->     IF v_status != 'I' THEN
   ->        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Book not borrowed by this
user.';
   ->     END IF;
   ->     SET v_days_borrowed = DATEDIFF(CURDATE(), v_date_of_issue);
   ->     IF v_days_borrowed BETWEEN 15 AND 30 THEN
   ->        SET v_fine_amount = v_days_borrowed * 5;
   ->     ELSEIF v_days_borrowed > 30 THEN
   ->        SET v_fine_amount = (30 * 5) + ((v_days_borrowed - 30) * 50);
   ->     ELSE
   ->        SET v_fine_amount = 0;
   ->     END IF;
   ->     IF v_fine_amount > 0 THEN
   ->              UPDATE Borrower SET Status = 'R' WHERE Roll_no = roll_no AND
Name_of_Book = name_of_book;
   ->             INSERT INTO Fine (Roll_no, Date, Amt) VALUES (roll_no, CURDATE(),
v_fine_amount);
   ->     END IF;
   ->
   ->     -- Final message
   ->     SELECT 'Fine calculation completed successfully. Amount: ', v_fine_amount;
   ->
   -> END //
Query OK, 0 rows affected (0.04 sec)

mysql> DELIMITER ;

mysql> CALL CalculateFine(1, 'C++');
+-------------------------------------------------+-------------+
| Fine calculation completed successfully. Amount: | v_fine_amount |
+-------------------------------------------------+-------------+
|                          5.00                          |        5.00        |
+-------------------------------------------------+-------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM Fine;
+--------+------------+-------+
| Roll_no | Date       | Amt   |
+--------+------------+-------+
|      1 | 2024-10-12 |  5.00 |
+--------+------------+-------+
1 row in set (0.00 sec)
```

mysql> SELECT * FROM Borrower;

```
+--------+------------+--------------+-----------+--------+
| Roll_no | Name       | Date_of_Issue | Name_of_Book | Status |
+--------+------------+--------------+-----------+--------+
|      1 | John Doe   | 2024-09-01   | C++       | R      |
|      2 | Jane Smith | 2024-09-15   | SEPM      | I      |
+--------+------------+--------------+-----------+--------+
2 rows in set (0.00 sec)
```

mysql> CALL CalculateFine(2, 'SEPM');

```
+-------------------------------------+-------------+
| Fine calculation completed successfully. Amount: | v_fine_amount |
+-------------------------------------+-------------+
|                                0.00                              |         0.00        |
+-------------------------------------+-------------+
1 row in set (0.00 sec)
```

mysql> SELECT * FROM Fine;

```
+--------+------------+-------+
| Roll_no | Date       | Amt   |
+--------+------------+-------+
|      1 | 2024-10-12 |  5.00 |
+--------+------------+-------+
|      2 | 2024-10-12 |  0.00 |
+--------+------------+-------+
2 rows in set (0.00 sec)
```

mysql> SELECT * FROM Borrower;

```
+--------+------------+--------------+-----------+--------+
| Roll_no | Name       | Date_of_Issue | Name_of_Book | Status |
+--------+------------+--------------+-----------+--------+
|      1 | John Doe   | 2024-09-01   | C++       | R      |
|      2 | Jane Smith | 2024-09-15   | SEPM      | I      |
+--------+------------+--------------+-----------+--------+
2 rows in set (0.00 sec)
```

Experiment No - 05

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 available in the newly created table N_Roll_Call with the data available in the table O_Roll_Call. If the data in the first table already exists in the second table then that data should be skipped.

```
mysql> CREATE DATABASE class;
Query OK, 1 row affected (0.01 sec)

mysql> USE class;
Database changed

mysql> CREATE TABLE O_RollCall (
   -> roll_no INT(3),
   -> name VARCHAR(20)
   -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE N_RollCall (
   -> roll_no INT(3),
   -> name VARCHAR(20)
   -> );
Query OK, 0 rows affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (1, 'Himanshu');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (2, 'Ram');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (3, 'Soham');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (5, 'Mohan');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (6, 'Om');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (9, 'Yash');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO O_RollCall VALUES (11, 'Mayur');
Query OK, 1 row affected (0.01 sec)
```

```
mysql> SELECT * FROM O_RollCall;
+---------+----------+
| roll_no | name     |
+---------+----------+
|       1 | Himanshu |
|       2 | Ram      |
|       3 | Soham    |
|       5 | Mohan    |
|       6 | Om       |
|       9 | Yash     |
|      11 | Mayur    |
+---------+----------+
7 rows in set (0.00 sec)

mysql> SELECT * FROM N_RollCall;
Empty set (0.00 sec)

mysql> DELIMITER //

mysql> CREATE PROCEDURE cursor_proc_p1()
    -> BEGIN
    ->     DECLARE fin INTEGER DEFAULT 0;
    ->     DECLARE old_roll INT(3);
    ->     DECLARE old_name VARCHAR(20);
    ->     DECLARE new_roll INT(3);
    ->     DECLARE old_csr CURSOR FOR SELECT roll_no, name FROM O_RollCall;
    ->     DECLARE new_csr CURSOR FOR SELECT roll_no FROM N_RollCall;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin=1;
    ->
    ->     OPEN old_csr;
    ->     OPEN new_csr;
    ->
    ->     ss: LOOP
    ->         FETCH old_csr INTO old_roll, old_name;
    ->         FETCH new_csr INTO new_roll;
    ->
    ->         IF fin=1 THEN
    ->             LEAVE ss;
    ->         END IF;
    ->
    ->         IF old_roll <> new_roll THEN
    ->             INSERT INTO N_RollCall VALUES(old_roll, old_name);
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE old_csr;
    ->     CLOSE new_csr;
    -> END //
```

```
Query OK, 0 rows affected (0.04 sec)

mysql> DELIMITER ;

mysql> DELIMITER //

mysql> CREATE PROCEDURE cursor_proc_p2(IN r1 INT)
   -> BEGIN
   ->     DECLARE r2 INT;
   ->     DECLARE exit_loop BOOLEAN DEFAULT FALSE;
   ->     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 (0.03 sec)

mysql> DELIMITER ;

mysql> CALL cursor_proc_p2(5);
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM O_RollCall;
+---------+----------+
| roll_no | name     |
+---------+----------+
|       1 | Himanshu |
|       2 | Ram      |
|       3 | Soham    |
|       5 | Mohan    |
|       6 | Om       |
|       9 | Yash     |
|      11 | Mayur    |
+---------+----------+
7 rows in set (0.00 sec)

mysql> SELECT * FROM N_RollCall;
+---------+----------+
| roll_no | name     |
+---------+----------+
|       6 | Om       |
|       9 | Yash     |
|      11 | Mayur    |
+---------+----------+
3 rows in set (0.00 sec)

mysql> CALL cursor_proc_p2(3);
Query OK, 0 rows affected (0.02 sec)

mysql> CALL cursor_proc_p1();
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT * FROM O_RollCall;
+---------+----------+
| roll_no | name     |
+---------+----------+
|       1 | Himanshu |
|       2 | Ram      |
|       3 | Soham    |
|       5 | Mohan    |
|       6 | Om       |
|       9 | Yash     |
|      11 | Mayur    |
+---------+----------+
7 rows in set (0.00 sec)

mysql> SELECT * FROM N_RollCall;
+---------+----------+
| roll_no | name     |
+---------+----------+
|       6 | Om       |
|       9 | Yash     |
|      11 | Mayur    |
|       1 | Himanshu |
|       2 | Ram      |
|       3 | Soham    |
|       5 | Mohan    |
+---------+----------+
7 rows in set (0.00 sec)
```