| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SECO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 01**

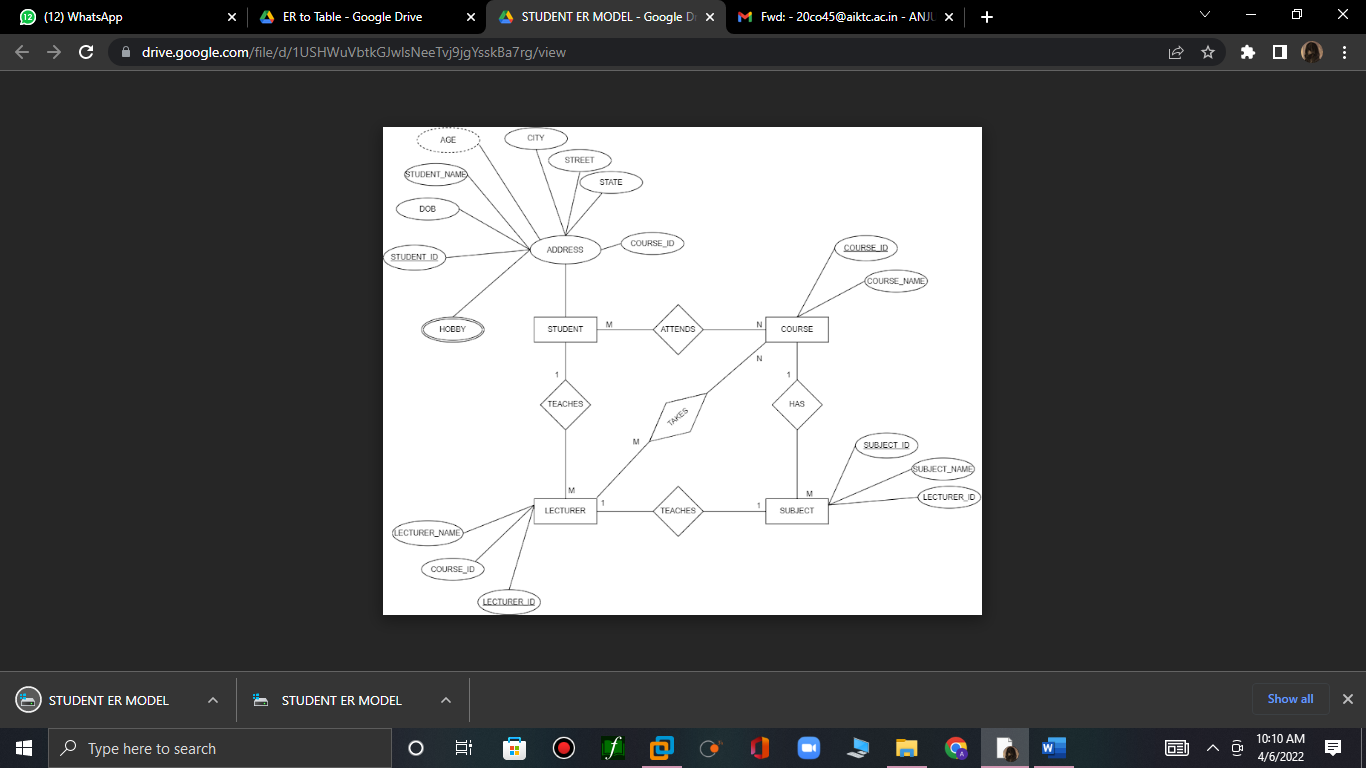
**Aim: Identify the case study / Mini Project and detail statement of problem. Design an E-R Entity Relationship / EER Extended Entity Relationship Model.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Detail Statement of Problem -**

| It is a system developed with an aim to solve the problem faced by organization while calculating salary of each Lecturer. |
| --- |
| There is a one-to-many relationship available in the table between the tables. |
| Each entity of the database such as (Student, Lecturer, Course) contain primary key. |
| The basic details of the Student and Lecturer are stored in respectively in tables in the database. |

**Output –**

****

**References –**

Dia

Draw.io

**Conclusion:**

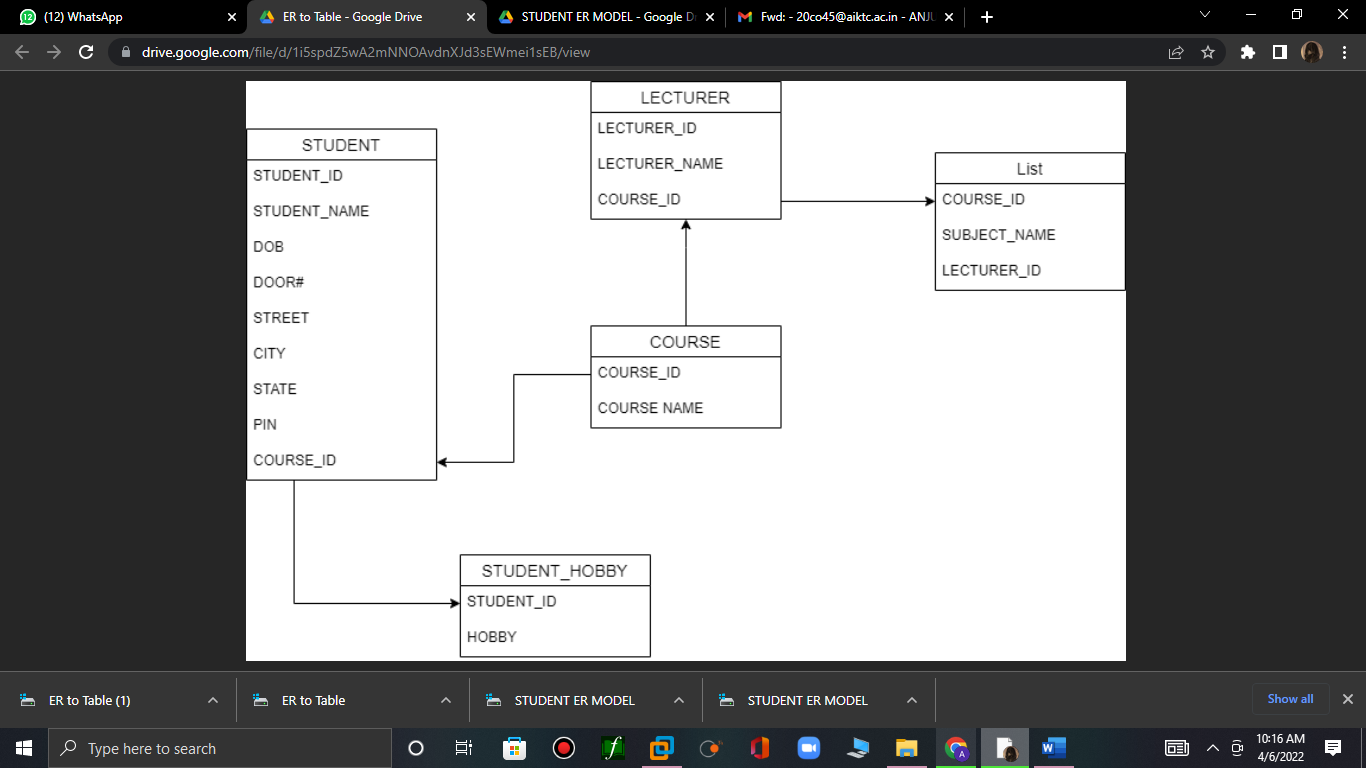
| From the above experiment I have understood how to create a ER/EER model and also understood all the related concepts. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SECO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 02**

**Aim: Mapping ER-EER to Relational Model.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Output – **

**References –**

Dia

Draw.io

https://dbdiagram.io

**Conclusion:**

| From the above experiment I have understood how to create a mapping ER-EER to Relation Model and also understood all the related concepts. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch:2020-21** |
| **Roll no:20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 03**

**Aim: Create a database using Data Definition Language (DDL) and apply integrity constraints for the specified system.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Execute the following commands on MySQL.**

**Data Definition Language (DDL)**

**==============================**

1. Create Database

create database databasename;

1. Create table Syntax

create table table\_name(

column1 datatype(size),

………

columnN datatype(size),

constraint(s) ) ;

1. Check the table structure

desc tablename

1. Alter table add / modify / drop

alter table table\_name add newcolumn\_name datatype(size)

alter table table\_name modify column\_name newdatatype(newsize)

1. Drop Table

Drop table tablename;

1. rename table oldtable to newtable
2. Using alter table add primary key and foreign key

alter table tablename modify colname datatype(size) primary key

alter table tablename add foreign key(colname) references tablename(colname);

**Integrity Constraints**

**Apply the following integrity constraints while using the DDL Commands.**

* Primary Key
  + - Column\_name datatype(size) primary key
* Foreign Key
  + - Foreign key (Fkey) references table\_name(PKey)
* NULL
  + - column\_name datatype(size) not null
* Default
  + - column\_name datatype(size) default “Mumbai**”**
* Unique
  + - Column\_name datatype(size) unique

**Attach the commands in Text format.**

**CREATE DATABASE CLASS;**

**CREATE TABLE STUDENT(**

**STU\_ID VARCHAR(10) NOT NULL PRIMARY KEY,**

**STU\_NAME CHAR(30),**

**STU\_DOB DATE,**

**STU\_ADD VARCHAR(40),**

**STU\_DEP\_ID INT);**

**ALTER TABLE STUDENT**

**ADD EMAIL VARCHAR(25);**

**ALTER TABLE STUDENT**

**DROP COLUMN EMAIL;**

**CREATE TABLE DEPARTMENT(**

**DEP\_ID INT NOT NULL PRIMARY KEY,**

**DEP\_NAME CHAR(20),**

**DEP\_FLOOR INT)**

**INSERT INTO DEPARTMENT**

**VALUES**

**(1,'COMPUTER',3),**

**(2,'ELECTRICAL',2),**

**(3,'MECHANICAL',1)**

**DROP TABLE DEPARTMENT;**

**Conclusion:**

| Thus, we have successfully completed the given practical. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch:2020-21** |
| **Roll no:20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 04**

**Aim: Apply Data Manipulation Language (DDL) Commands for the specified system.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Execute the following commands on MySQL.**

**DML Commands**

**==============**

**1. SELECT**

**select \* from table\_name;**

**select col1,col2,....colN from table\_name;**

**Note - select \* from table where condition;**

**2. INSERT**

**SYNTAX -**

**insert into table\_name (col1, col2....colN) values (val1,val2,....valN);**

**Multiple inserts are possible by using comma separated values.**

**String Values should be always in quotes.**

**Note - Column Names is mandatory if some column values are inserted**

**otherwise for all column values it is optional**

**3. UPDATE**

**Syntax - update tablename set column\_name = newvalue where condition**

**For multiple columns, after set keyword by using comma**

**4. DELETE**

**Syntax - delete from table\_name where condition;**

**Attach the commands in Text format.**

**INSERT INTO STUDENT(STU\_ID,STU\_NAME,STU\_DOB,STU\_ADD,STU\_DEP\_ID)**

**VALUES ('20CO22','SAQLAIN','01-01-2001','SEAWOOD',1);**

**INSERT INTO STUDENT**

**VALUES ('20CO38','AWAIZ','03-01-2001','PANVEL',1),**

**('20CO40','ZAKKI','04-01-2001','BHIWANDI',1),**

**('20CO41','DANISH','05-01-2001','CST',1),**

**('20CO43','ASAD','06-01-2001','TALOJA',1),**

**('20CO47','ALI','07-01-2001','BELAPUR',1),**

**('20CO38','JUNED','08-01-2001','BHIWANDI',1);**

**INSERT INTO STUDENT**

**VALUES('19CO50','AFAQUE','01-05-2002','BHIWANDI',2,'TE'),**

**('17CO14','ZAKIR','06-11-1998','TITWALA',3,'BE');**

**INSERT INTO STUDENT VALUES ('18CO14','ZAKIR','06-11-1998','TITWALA',3,'BE');**

**DELETE FROM STUDENT WHERE STU\_ID = '17CO14';**

**UPDATE STUDENT**

**SET STU\_ADD = 'GOVANDI'**

**WHERE STU\_ID = '19CO50';**

**Conclusion:**

| **DML** is short name of Data Manipulation Language which deals with data manipulation and includes most common **SQL statements** such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 05**

**Aim: Perform simple queries, string manipulation operations and aggregate functions.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Execute the following commands**

================================

1. **WHERE C1 AND C2**

Each selected row must satisfy both the conditions C1 and C2.

Syntax: select \* from table where column1 = ‘value’ AND column2 = ‘value’;

1. **WHERE C2 OR C2**

Each selected row must satisfy either the conditions C1 or the condition C2 or both the conditions C1 and C2.

Syntax: select \* from table where column1 = ‘value’ OR column2 = ‘value’;

AND and OR operators may be combined. For example, in a condition like A AND B OR C. In such compound conditions, the AND operation is done first.

1. **WHERE NOT C1 AND C2**

Each selected row must satisfy both the conditions C2 but not satisfy the condition C1.

Syntax: select \* from table where NOT column1 = ‘value’ and column2 = ‘value’;

1. **WHERE A IN**

Each selected row must have the value of A in the list that follows IN. NOT IN may be used to select all that are not in the list that follows IN.

Syntax: select \* from table where column1 IN (‘value1’,’value2’,’value3’);

1. **WHERE A OPERATOR ANY**

Each selected row must satisfy both the condition for each of the list that follows ANY. Any of the following operators may be used in such a clause, greater than (>), less than (<), less than equal to (<=) and greater than equal to (>=).

Syntax: select \* from table where column1 (>,<,>=,<=) = ‘value’;

1. **WHERE A LIKE X**

Each selected row must have a value of A that satisfies the string-matching condition specified in X. The expression that follow LIKE must be a character string and enclosed in apostrophes (‘).

Syntax: select \* from table where column1 LIKE ‘value1’ ;

LIKE is a powerful string-matching operator that uses two special characters (called as wildcards). These characters are underscore (\_) and percent (%). Underscore represents any single character while percent represents any sequence of n characters.

For eg. ‘Del%’, ‘Gup\_\_’, ‘%ing%’, and ‘%ame\_’

1. **WHERE A BETWEEN X AND Y**

Each selected row must have a values of A between X and Y including values X and Y. NOT BETWEEN may also be used to find rows that are outside the range (X,Y).

Syntax: select \* from table where column1 BETWEEN ‘value1’ AND ‘value2’;

1. **WHERE A IS NULL**

Each selected row must satisfy the condition that A is NULL.

Syntax: select \* from table where column1 IS NULL;

**9. AGGREGATE FUNCTIONS**

**COUNT –** Counts the occurrences of the rows.

**MIN –** Find the smallest value

**MAX** – Find the largest value

**SUM –** Computes the sum

**AVG –** Compute the average of unique values.

**Attach the output of the above commands in txt format.**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_ADD='BHIWANDI' AND STU\_DEP\_ID=1;**

**+--------+----------+----------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+----------+------------+**

**| 20CO40 | ZAKKI | BHIWANDI | 1 |**

**| 20CO38 | JUNED | BHIWANDI | 1 |**

**+--------+----------+----------+------------+**

**2 rows in set (0.002 sec)**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_DEP\_ID=2 OR STU\_YEAR='SE';**

**ERROR 1054 (42S22): Unknown column 'STU\_YEAR' in 'where clause'**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_DEP\_ID=2 OR STU\_YEAR='SE';**

**ERROR 1054 (42S22): Unknown column 'STU\_YEAR' in 'where clause'**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_ADD IN ('BHIWANDI','CST','TALOJA')**

**-> ;**

**+--------+----------+----------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+----------+------------+**

**| 20CO40 | ZAKKI | BHIWANDI | 1 |**

**| 20CO41 | DANISH | CST | 1 |**

**| 20CO43 | ASAD | TALOJA | 1 |**

**| 20CO38 | JUNED | BHIWANDI | 1 |**

**| 20EE40 | ZAKKI | BHIWANDI | 2 |**

**| 20EE45 | PRASAD | TALOJA | 2 |**

**| 20ME41 | HAMZA | CST | 3 |**

**| 20ME53 | RIZWAN | BHIWANDI | 3 |**

**+--------+----------+----------+------------+**

**8 rows in set (0.001 sec)**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_DEP\_ID > 2;**

**+--------+----------+----------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+----------+------------+**

**| 20ME05 | AWAIZ | NERUL | 3 |**

**| 20ME41 | HAMZA | CST | 3 |**

**| 20ME53 | RIZWAN | BHIWANDI | 3 |**

**+--------+----------+----------+------------+**

**3 rows in set (0.000 sec)**

**MariaDB [CLASS]> SELECT \* FROM STUDENT WHERE STU\_NAME LIKE 'A%';**

**+--------+----------+---------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+---------+------------+**

**| 20CO38 | AWAIZ | PANVEL | 1 |**

**| 20CO43 | ASAD | TALOJA | 1 |**

**| 20CO47 | ALI | BELAPUR | 1 |**

**| 20EE47 | AALAM | BELAPUR | 2 |**

**| 20ME05 | AWAIZ | NERUL | 3 |**

**+--------+----------+---------+------------+**

**5 rows in set (0.001 sec)**

**MariaDB [CLASS]>**

**Conclusion:**

| From the above experiment I have understood how to Perform simple queries, string manipulation operations and aggregate functions. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 06**

**Aim: Perform Join Operations.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Theory:**

**Joins**

Natural Join

The natural join of two tables R and S is obtained by applying a selection and a projection to the Cartesian Product R x S as follows;

1. For each column ‘a’ that is common to both tables R and S, we select rows that satisfy the condition R.a = S.a
2. For each column ‘a’ that is common to both tables R and S, we project out the column S.a. There if there are ‘m’ columns common to both tables, ‘m’ duplicate columns are removed from the Cartesian product.

Natural join command finds matching rows from the two tables that are being joined and reject rows that do not match.

A JOIN clause is used to combine rows from two or more tables, based on a related column between them.

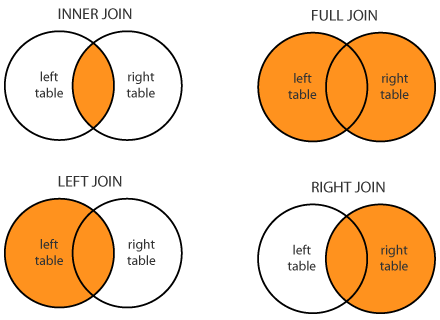
The INNER JOIN keyword selects records that have matching values in both tables.

Syntax:

SELECT column\_name(s)

FROM table1

INNER JOIN table2 ON table1.column\_name = table2.column\_name;



Left Join

The LEFT JOIN keyword returns all records from the left table (table1), and the matched records from the right table (table2). The result is NULL from the right side, if there is no match.

Syntax : SELECT column\_name(s)

FROM table1

LEFT JOIN table2 ON table1.column\_name = table2.column\_name;

Right Join

The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.

Syntax:

SELECT column\_name(s)

FROM table1

RIGHT JOIN table2 ON table1.column\_name = table2.column\_name;

Full Join

The FULL OUTER JOIN keyword return all records when there is a match in either left (table1) or right (table2) table records.

Syntax:

SELECT column\_name(s)

FROM table1

LEFT JOIN table2 ON table1.column\_name = table2.column\_name

UNION

SELECT column\_name(s)

FROM table1

RIGHT JOIN table2 ON table1.column\_name = table2.column\_name;

Q4. **Inner join / Join**

select Player.PlayerID, Player.Fname, Player.Lname, Batting.NRuns, Batting.Fours, Batting.Six from Player join Batting ON Player.PlayerID = Batting.PID;

Q5. **Left Join**

select Player.PlayerID, Player.Fname, Player.Lname, Batting.NRuns, Batting.Fours, Batting.Six from Player left join Batting ON Player.PlayerID = Batting.PID;

Q6. **Right Join**

select Player.PlayerID, Player.Fname, Player.Lname, Batting.NRuns,g.Fours, Batting.Six from Player right join Batting ON Player.PlayerID = Batting.PID;

Q7. **Outer Join**

select Player.PlayerID, Player.Fname, Player.Lname, Batting.NRuns, Batting.Fours, Batting.Six from Player left join Batting ON Player.PlayerID = Batting.PID

-> UNION

-> select Player.PlayerID, Player.Fname, Player.Lname, Batting.NRuns, Batting.Fours, Batting.Six from Player right join Batting ON Player.PlayerID = Batting.PID;

**Attach the output of the join operations on your project in txt format.**

**MariaDB [CLASS]> show tables;**

**+-----------------+**

**| Tables\_in\_CLASS |**

**+-----------------+**

**| DEPARTMENT |**

**| STUDENT |**

**+-----------------+**

**2 rows in set (0.000 sec)**

**MariaDB [CLASS]> select \* from DEPARTMENT**

**-> ;**

**+--------+------------+-----------+**

**| DEP\_ID | DEP\_NAME | DEP\_FLOOR |**

**+--------+------------+-----------+**

**| 1 | COMPUTER | 3 |**

**| 2 | ELECTRICAL | 2 |**

**| 3 | MECHANICAL | 1 |**

**+--------+------------+-----------+**

**3 rows in set (0.001 sec)**

**MariaDB [CLASS]> select \* from STUDENT;**

**+--------+----------+----------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+----------+------------+**

**| 20CO23 | SAIM | NERUL | 1 |**

**| 20CO38 | AWAIZ | PANVEL | 1 |**

**| 20CO40 | ZAKKI | BHIWANDI | 1 |**

**| 20CO41 | DANISH | CST | 1 |**

**| 20CO43 | ASAD | TALOJA | 1 |**

**| 20CO47 | ALI | BELAPUR | 1 |**

**| 20CO38 | JUNED | BHIWANDI | 1 |**

**| 20EE40 | ZAKKI | BHIWANDI | 2 |**

**| 20EE45 | PRASAD | TALOJA | 2 |**

**| 20EE47 | AALAM | BELAPUR | 2 |**

**| 20ME05 | AWAIZ | NERUL | 3 |**

**| 20ME41 | HAMZA | CST | 3 |**

**| 20ME53 | RIZWAN | BHIWANDI | 3 |**

**+--------+----------+----------+------------+**

**13 rows in set (0.000 sec)**

**MariaDB [CLASS]> select \* from DEPARTMENT**

**-> ;**

**+--------+------------+-----------+**

**| DEP\_ID | DEP\_NAME | DEP\_FLOOR |**

**+--------+------------+-----------+**

**| 1 | COMPUTER | 3 |**

**| 2 | ELECTRICAL | 2 |**

**| 3 | MECHANICAL | 1 |**

**+--------+------------+-----------+**

**3 rows in set (0.000 sec)**

**MariaDB [CLASS]> select STU\_ID,STU\_NAME,STU\_DEP\_ID,DEP\_ID,DEP\_NAME from STUDENT as s,DEPARTMENT as d where s.STU\_DEP\_ID = d.DEP\_ID;**

**+--------+----------+------------+--------+------------+**

**| STU\_ID | STU\_NAME | STU\_DEP\_ID | DEP\_ID | DEP\_NAME |**

**+--------+----------+------------+--------+------------+**

**| 20CO23 | SAIM | 1 | 1 | COMPUTER |**

**| 20CO38 | AWAIZ | 1 | 1 | COMPUTER |**

**| 20CO40 | ZAKKI | 1 | 1 | COMPUTER |**

**| 20CO41 | DANISH | 1 | 1 | COMPUTER |**

**| 20CO43 | ASAD | 1 | 1 | COMPUTER |**

**| 20CO47 | ALI | 1 | 1 | COMPUTER |**

**| 20CO38 | JUNED | 1 | 1 | COMPUTER |**

**| 20EE40 | ZAKKI | 2 | 2 | ELECTRICAL |**

**| 20EE45 | PRASAD | 2 | 2 | ELECTRICAL |**

**| 20EE47 | AALAM | 2 | 2 | ELECTRICAL |**

**| 20ME05 | AWAIZ | 3 | 3 | MECHANICAL |**

**| 20ME41 | HAMZA | 3 | 3 | MECHANICAL |**

**| 20ME53 | RIZWAN | 3 | 3 | MECHANICAL |**

**+--------+----------+------------+--------+------------+**

**13 rows in set (0.001 sec)**

**MariaDB [CLASS]> select STU\_ID,STU\_NAME,STU\_DEP\_ID,DEP\_ID,DEP\_NAME from STUDENT as s,DEPARTMENT as d where s.STU\_DEP\_ID = d.DEP\_ID and d.DEP\_FLOOR = 3;**

**+--------+----------+------------+--------+----------+**

**| STU\_ID | STU\_NAME | STU\_DEP\_ID | DEP\_ID | DEP\_NAME |**

**+--------+----------+------------+--------+----------+**

**| 20CO23 | SAIM | 1 | 1 | COMPUTER |**

**| 20CO38 | AWAIZ | 1 | 1 | COMPUTER |**

**| 20CO40 | ZAKKI | 1 | 1 | COMPUTER |**

**| 20CO41 | DANISH | 1 | 1 | COMPUTER |**

**| 20CO43 | ASAD | 1 | 1 | COMPUTER |**

**| 20CO47 | ALI | 1 | 1 | COMPUTER |**

**| 20CO38 | JUNED | 1 | 1 | COMPUTER |**

**+--------+----------+------------+--------+----------+**

**7 rows in set (0.001 sec)**

**MariaDB [CLASS]> select STU\_ID,STU\_NAME,STU\_DEP\_ID,DEP\_ID,DEP\_NAME from STUDENT as s,DEPARTMENT as d where s.STU\_DEP\_ID = d.DEP\_ID and d.DEP\_FLOOR = 3 and STU\_NAME like '%I%';**

**+--------+----------+------------+--------+----------+**

**| STU\_ID | STU\_NAME | STU\_DEP\_ID | DEP\_ID | DEP\_NAME |**

**+--------+----------+------------+--------+----------+**

**| 20CO23 | SAIM | 1 | 1 | COMPUTER |**

**| 20CO38 | AWAIZ | 1 | 1 | COMPUTER |**

**| 20CO40 | ZAKKI | 1 | 1 | COMPUTER |**

**| 20CO41 | DANISH | 1 | 1 | COMPUTER |**

**| 20CO47 | ALI | 1 | 1 | COMPUTER |**

**+--------+----------+------------+--------+----------+**

**5 rows in set (0.001 sec)**

**MariaDB [CLASS]> select STU\_ID,STU\_NAME,STU\_DEP\_ID,DEP\_ID,DEP\_NAME from STUDENT inner join DEPARTMENT on STUDENT.STU\_DEP\_ID = DEPARTMENT.DEP\_ID;**

**+--------+----------+------------+--------+------------+**

**| STU\_ID | STU\_NAME | STU\_DEP\_ID | DEP\_ID | DEP\_NAME |**

**+--------+----------+------------+--------+------------+**

**| 20CO23 | SAIM | 1 | 1 | COMPUTER |**

**| 20CO38 | AWAIZ | 1 | 1 | COMPUTER |**

**| 20CO40 | ZAKKI | 1 | 1 | COMPUTER |**

**| 20CO41 | DANISH | 1 | 1 | COMPUTER |**

**| 20CO43| ASAD | 1 | 1 | COMPUTER |**

**| 20CO47 | ALI | 1 | 1 | COMPUTER |**

**| 20CO38 | JUNED | 1 | 1 | COMPUTER |**

**| 20EE40 | ZAKKI | 2 | 2 | ELECTRICAL |**

**| 20EE45 | PRASAD | 2 | 2 | ELECTRICAL |**

**| 20EE47 | AALAM | 2 | 2 | ELECTRICAL |**

**| 20ME05 | AWAIZ | 3 | 3 | MECHANICAL |**

**| 20ME41 | HAMZA | 3 | 3 | MECHANICAL |**

**| 20ME53 | RIZWAN | 3 | 3 | MECHANICAL |**

**+--------+----------+------------+--------+------------+**

**13 rows in set (0.001 sec)**

**MariaDB [CLASS]> select STU\_ID,STU\_NAME,STU\_DEP\_ID,DEP\_ID,DEP\_NAME from STUDENT left outer join DEPARTMENT on STUDENT.STU\_DEP\_ID = DEPARTMENT.DEP\_ID;**

**+--------+----------+------------+--------+------------+**

**| STU\_ID | STU\_NAME | STU\_DEP\_ID | DEP\_ID | DEP\_NAME |**

**+--------+----------+------------+--------+------------+**

**| 20CO23 | SAIM | 1 | 1 | COMPUTER |**

**| 20CO38 | AWAIZ | 1 | 1 | COMPUTER |**

**| 20CO40 | ZAKKI | 1 | 1 | COMPUTER |**

**| 20CO41 | DANISH | 1 | 1 | COMPUTER |**

**| 20CO43 | ASAD | 1 | 1 | COMPUTER |**

**| 20CO47 | ALI | 1 | 1 | COMPUTER |**

**| 20CO38 | JUNED | 1 | 1 | COMPUTER |**

**| 20EE40 | ZAKKI | 2 | 2 | ELECTRICAL |**

**| 20EE45 | PRASAD | 2 | 2 | ELECTRICAL |**

**| 20EE47 | AALAM | 2 | 2 | ELECTRICAL |**

**| 20ME05 | AWAIZ | 3 | 3 | MECHANICAL |**

**| 20ME41 | HAMZA | 3 | 3 | MECHANICAL |**

**| 20ME53 | RIZWAN | 3 | 3 | MECHANICAL |**

**+--------+----------+------------+--------+------------+**

**13 rows in set (0.001 sec)**

MariaDB [CLASS]>

**Conclusion:**

| From the above experiment I have understood how to Perform Join Operations. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 07**

**Aim: Perform nested and complex queries.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Theory:**

A Subquery or Inner query or a Nested query is a query within another SQL query and embedded within the WHERE clause.

A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Subqueries can be used with the SELECT, INSERT, UPDATE, and DELETE statements along with the operators like =, <, >, >=, <=, IN, BETWEEN, etc.

There are a few rules that subqueries must follow −

* Subqueries must be enclosed within parentheses.
* A subquery can have only one column in the SELECT clause, unless multiple columns are in the main query for the subquery to compare its selected columns.
* An ORDER BY command cannot be used in a subquery, although the main query can use an ORDER BY. The GROUP BY command can be used to perform the same function as the ORDER BY in a subquery.
* Subqueries that return more than one row can only be used with multiple value operators such as the IN operator.
* The BETWEEN operator cannot be used with a subquery. However, the BETWEEN operator can be used within the subquery.

**Subqueries with the SELECT Statement**

Subqueries are most frequently used with the SELECT statement. The basic syntax is as follows −

SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

WHERE column\_name OPERATOR

(SELECT column\_name [, column\_name ]

FROM table1 [, table2 ]

[WHERE])

**Attach the output of the nested queries on your project in txt format.**

**MariaDB [CLASS]> select STU\_NAME,STU\_ID from STUDENT where STU\_ID in (select STU\_ID from STUDENT where STU\_ADD='TALOJA');**

**+----------+--------+**

**| STU\_NAME | STU\_ID |**

**+----------+--------+**

**| ASAD | MUJAWAR JUNED |**

**| PRASAD | 20EE45 |**

**+----------+--------+**

**2 rows in set (0.098 sec)**

**MariaDB [CLASS]> select STU\_NAME,STU\_ADD,STU\_DEP\_ID from STUDENT where STU\_ID in (select STU\_ID from DEPARTMENT where DEP\_FLOOR > 2);**

**+----------+----------+------------+**

**| STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+----------+----------+------------+**

**| SAIM | NERUL | 1 |**

**| AWAIZ | PANVEL | 1 |**

**| ZAKKI | BHIWANDI | 1 |**

**| DANISH | CST | 1 |**

**| ASAD | TALOJA | 1 |**

**| ALI | BELAPUR | 1 |**

**| JUNED | BHIWANDI | 1 |**

**| ZAKKI | BHIWANDI | 2 |**

**| PRASAD | TALOJA | 2 |**

**| AALAM | BELAPUR | 2 |**

**| AWAIZ | NERUL | 3 |**

**| HAMZA | CST | 3 |**

**| RIZWAN | BHIWANDI | 3 |**

**+----------+----------+------------+**

**13 rows in set (0.001 sec)**

**MariaDB [CLASS]> select STU\_NAME,STU\_ID from STUDENT where STU\_ID in (select STU\_ID from STUDENT where STU\_ADD = 'BHIWANDI');**

**+----------+--------+**

**| STU\_NAME | STU\_ID |**

**+----------+--------+**

**| ZAKKI | 20CO40 |**

**| JUNED | 20CO38 |**

**| ZAKKI | 20EE40 |**

**| RIZWAN | 20ME53 |**

**+----------+--------+**

**4 rows in set (0.000 sec)**

**MariaDB [CLASS]> select \* from STUDENT where STU\_ID in (select STU\_ID from DEPARTMENT where STU\_DEP\_ID>=1);**

**+--------+----------+----------+------------+**

**| STU\_ID | STU\_NAME | STU\_ADD | STU\_DEP\_ID |**

**+--------+----------+----------+------------+**

**| 20CO23 | SAIM | NERUL | 1 |**

**| 20CO38 | AWAIZ | PANVEL | 1 |**

**| 20CO40 | ZAKKI | BHIWANDI | 1 |**

**| 20CO41 | DANISH | CST | 1 |**

**| 20CO43 | ASAD | TALOJA | 1 |**

**| 20CO47 | ALI | BELAPUR | 1 |**

**| 20CO38 | JUNED | BHIWANDI | 1 |**

**| 20EE40 | ZAKKI | BHIWANDI | 2 |**

**| 20EE45 | PRASAD | TALOJA | 2 |**

**| 20EE47 | AALAM | BELAPUR | 2 |**

**| 20ME05 | AWAIZ | NERUL | 3 |**

**| 20ME41 | HAMZA | CST | 3 |**

**| 20ME53 | RIZWAN | BHIWANDI | 3 |**

**+--------+----------+----------+------------+**

**13 rows in set (0.001 sec)**

**Conclusion:**

| From the above experiment I have understood how to Perform nested and complex queries. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 08**

**Aim: Implement DCL and TCL Commands**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Theory:**

**DCL(Data Control Language):** DCL includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions and other controls of the database system.

DCL Commands

Create a user -

Create user 'username'@'servername' identifed by 'password';

Mysql -u newuser -p

Password -

Grant - Used to give permission to perform DDL, DML commands to the users.

Syntax

Grant roles/permission on database.tables to 'user'@'servername';

Roles - ddl or dml commands (select , insert, update, delete, create , alter….)

Revoke - Used to remove permission to perform DDL, DML commands to the users.

Syntax -

Revoke roles/permission on database.tables from 'user'@'servername';

**TCL(Transaction Control Language):** TCL commands deal with the transaction within the database.

COMMIT– commits a Transaction.

ROLLBACK– rollbacks a transaction in case of any error occurs.

SAVEPOINT–sets a save point within a transaction.

SET TRANSACTION–specify characteristics for the transaction.

**Attach the output of the nested queries on your project in txt format.**

C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u root -p

Enter password: \*\*\*\*

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 14

Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create user Awaiz identified by '12345';

Query OK, 0 rows affected (0.19 sec)

mysql> ^Z

Bye

C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u root -p

Enter password: \*\*\*\*

Welcome to the MySQL monitor. Commands end with ; or \g.

Your MySQL connection id is 16

Server version: 8.0.28 MySQL Community Server - GPL

Copyright (c) 2000, 2022, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its

affiliates. Other names may be trademarks of their respective

owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

**mysql> create database School;**

**Query OK, 1 row affected (0.01 sec)**

**mysql> use School;**

**Database changed**

**mysql> create table students1(Roll\_No int(10),Name varchar(30),Date\_Of\_Birth varchar(30),Addr varchar(30));**

**Query OK, 0 rows affected, 1 warning (0.05 sec)**

**mysql> insert into students1 values(25,'Awaiz','29-10-2002','Mumbai');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> insert into students1 values(53,'JUNED','29-10-2002','Mumbai');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> insert into students1 values(45,'Asad','29-10-2002','Mumbai');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> insert into students1 values(40,'Zakki','15-10-2002','Mumbai');**

**Query OK, 1 row affected (0.01 sec)**

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**4 rows in set (0.00 sec)**

**mysql> set autocommit=0;**

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

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**4 rows in set (0.00 sec)**

**mysql> commit;**

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

**mysql> insert into students1 values(14,'Ebad','10-12-2002','Mumbai');**

**Query OK, 1 row affected (0.00 sec)**

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**| 14 | Ebad | 10-12-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**5 rows in set (0.00 sec)**

**mysql> rollback;**

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

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**4 rows in set (0.00 sec)**

**mysql> commit;**

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

**mysql> insert into students1 values(14,'Ebad','10-12-2002','Mumbai');**

**Query OK, 1 row affected (0.00 sec)**

**mysql> savepoint s1;**

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

**mysql> insert into students1 values(15,'Adnan','20-12-2002','Mumbai');**

**Query OK, 1 row affected (0.00 sec)**

**mysql> savepoint s2;**

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

**mysql> insert into students1 values(18,'Ammar','20-12-2002','Mumbai');**

**Query OK, 1 row affected (0.00 sec)**

**mysql> rollback to s2;**

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

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**| 14 | Ebad | 10-12-2002 | Mumbai |**

**| 15 | Adnan | 20-12-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**6 rows in set (0.00 sec)**

**mysql> rollback to s1;**

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

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**| 14 | Ebad | 10-12-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**5 rows in set (0.00 sec)**

**mysql> rollback;**

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

**mysql> select \* from Students1;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | 29-10-2002 | Mumbai |**

**| 53 | JUNED | 29-10-2002 | Mumbai |**

**| 45 | Asad | 29-10-2002 | Mumbai |**

**| 40 | Zakki | 15-10-2002 | Mumbai |**

**+---------+-------+---------------+--------+**

**4 rows in set (0.00 sec)**

**mysql> set autocommit=1;**

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

**mysql> select \* from Students;**

**+---------+-------+---------------+--------+**

**| Roll\_No | Name | Date\_Of\_Birth | Addr |**

**+---------+-------+---------------+--------+**

**| 25 | Awaiz | -1983 | Mumbai |**

**| 53 | JUNED | -1996 | Mumbai |**

**| 40 | Zakki | -1996 | Mumbai |**

**| 45 | Asad | -1996 | Mumbai |**

**+---------+-------+---------------+--------+**

**4 rows in set (0.01 sec)**

**Conclusion:**

| From the above experiment I have understood how to Implement DCL and TCL Commands. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 09**

**Aim: Implement stored procedures and functions.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Theory:**

**Functions:** Scalar functions in SQL Server will return only a single value where as table-valued functions will return a table. Functions cannot make any permanent alterations to SQL Server (i.e., no INSERT or UPDATE operations are allowed). If function returns a scalar value, it can be used inline in SQL statements.

**Stored Procedure:** Stored Procedures can be defined as the set of SQL statements that are stored in the server. The users can refer from the stored procedure and does not have to write individual statements. Stored Procedures is a tool that is used to perform any specific operations like Insert, Update or Delete in our database recursively and it can be used to alter or update any records in database. Unless function that returns only single value, stored procedures can return zero and many values at a time.

**Attach the output of the nested queries on your project in txt format.**

**# MYSQL -U ROOT -P**

**MYSQL: option &#39;-P&#39; requires an argument**

**shkab@DESKTOP-PJJ8MRD c:\xampp**

**# mysql -u root -p**

**Enter password:**

**MariaDB [(none)]&gt; USE CRICKET;**

**Database changed**

**MariaDB [CRICKET]&gt; SHOW TABLES;**

**+-------------------+**

**| Tables\_in\_cricket |**

**+-------------------+**

**| batsmen |**

**| batting |**

**| bowling |**

**| bowling2689 |**

**| matches |**

**| player |**

**+-------------------+**

**6 rows in set (0.001 sec)**

**MariaDB [CRICKET]&gt; DELIMITER #**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE BEST\_PLAYERS()**

**-&gt; BEGIN**

**-&gt; SELECT PLAYERID, FNAME, LNAME, NRUNS,FOURS, SIX**

**-&gt; FROM PLAYER, BATTING**

**-&gt; WHERE PLAYER.PLAYERID = BATTING.PID;**

**-&gt; END**

**-&gt; #**

**Query OK, 0 rows affected (0.006 sec)**

**MariaDB [CRICKET]&gt; SHOW CREATE PROCEDURE STATUS WHERE DB=&#39;cricket&#39;;**

**-&gt; #**

**ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to**

**your MariaDB server version for the right syntax to use near &#39;WHERE DB=&#39;cricket&#39;&#39; at line 1**

**MariaDB [CRICKET]&gt;**

**MariaDB [CRICKET]&gt; CALL BEST\_PLAYERS() #**

**+----------+-----------+------------+-------+-------+------+**

**| PLAYERID | FNAME | LNAME | NRUNS | FOURS | SIX |**

**+----------+-----------+------------+-------+-------+------+**

**| 23001 | Yuvraj | Singh | 38 | 2 | 2 |**

**| 25001 | MS | Dhoni | 36 | 2 | 1 |**

**| 27001 | Praveen | Kumar | 7 | 1 | 0 |**

**| 89001 | Sachin | Tendulkar | 91 | 7 | 0 |**

**| 98002 | Harbhajan | Singh | 3 | 0 | 0 |**

**| 23001 | Yuvraj | Singh | 0 | 0 | 0 |**

**| 25001 | MS | Dhoni | 71 | 4 | 0 |**

**| 27001 | Praveen | Kumar | 2 | 0 | 0 |**

**| 91001 | Sanath | Jayasuriya | 60 | 8 | 2 |**

**| 94002 | Chaminda | Vaas | 17 | 1 | 0 |**

**| 98002 | Harbhajan | Singh | 2 | 0 | 0 |**

**| 99999 | DANIYAL | DANISH | 100 | 4 | 5 |**

**+----------+-----------+------------+-------+-------+------+**

**12 rows in set (0.001 sec)**

**Query OK, 0 rows affected (0.026 sec)**

**MariaDB [CRICKET]&gt; DELIMITER @**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE GET\_PLAYERS( IN VAR1 INT)**

**-&gt; BEGIN**

**-&gt; SELECT \* FROM PLAYER LIMIT VAR1**

**-&gt; ;**

**-&gt; SELECT COUNT(\*) AS TOTAL\_PLAYERS FROM PLAYER;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.007 sec)**

**MariaDB [CRICKET]&gt; CALL GET\_PLAYERS(5) @**

**+----------+----------+---------------+-----------+-------+------------+-------+**

**| PlayerID | Lname | Fname | Country | YBorn | BPlace | FTest |**

**+----------+----------+---------------+-----------+-------+------------+-------+**

**| 23001 | Singh | Yuvraj | India | 1981 | Chandigarh | 2003 |**

**| 24001 | Symonds | Andrew | Australia | 1976 | Wollongong | 1999 |**

**| 25001 | Dhoni | MS | India | 1981 | Ranchi | 2005 |**

**| 25002 | Siddiqui | MahefuzRehman | India | 1996 | Bihar | 2015 |**

**| 27001 | Kumar | Praveen | India | 1986 | Meerut | NULL |**

**+----------+----------+---------------+-----------+-------+------------+-------+**

**+---------------+**

**| TOTAL\_PLAYERS |**

**+---------------+**

**| 25 |**

**+---------------+**

**Query OK, 0 rows affected (0.022 sec)**

**MariaDB [CRICKET]&gt; DELIMITER @**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE TOTALRUNS()**

**-&gt; BEGIN**

**-&gt; DECLARE TOTALRUNS INT DEFAULT 0;**

**-&gt; SELECT SUM(NRUNS) INTO TOTALRUNS FROM BATTING;**

**-&gt; SELECT TOTALRUNS;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.214 sec)**

**MariaDB [CRICKET]&gt; CALL TOTALRUNS()**

**-&gt; @**

**+-----------+**

**| TOTALRUNS |**

**+-----------+**

**| 527 |**

**+-----------+**

**Query OK, 1 row affected (0.007 sec)**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE GETRUNS (OUT TOTAL INT)**

**-&gt; BEGIN**

**-&gt; SELECT SUM(NRUNS) INTO TOTAL FROM BATTING;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.006 sec)**

**MariaDB [CRICKET]&gt; delimiter ;**

**MariaDB [CRICKET]&gt; call getruns(@total);**

**Query OK, 1 row affected (0.000 sec)**

**MariaDB [CRICKET]&gt;**

**MariaDB [CRICKET]&gt;**

**MariaDB [CRICKET]&gt;**

**MariaDB [CRICKET]&gt; call GETRUNS(@total);**

**Query OK, 1 row affected (0.001 sec)**

**MariaDB [CRICKET]&gt; CALL GETRUNS(@TOTAL);**

**Query OK, 1 row affected (0.001 sec)**

**MariaDB [CRICKET]&gt; SELECT @TOTAL;**

**+--------+**

**| @TOTAL |**

**+--------+**

**| 527 |**

**+--------+**

**MariaDB [CRICKET]&gt; SELECT \* FROM BATTING;**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**| MID | PID | NOrder | HOut | FOW | NRuns | Nballs | Fours | Six |**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**| 2689 | 23001 | 4 | C | 175 | 38 | 38 | 2 | 2 |**

**| 2689 | 25001 | 5 | C | 240 | 36 | 37 | 2 | 1 |**

**| 2689 | 27001 | 9 | C | 255 | 7 | 7 | 1 | 0 |**

**| 2689 | 89001 | 2 | C | 205 | 91 | 121 | 7 | 0 |**

**| 2689 | 98002 | 8 | LBW | 249 | 3 | 3 | 0 | 0 |**

**| 2755 | 23001 | 3 | C | 51 | 0 | 6 | 0 | 0 |**

**| 2755 | 25001 | 5 | C | 232 | 71 | 80 | 4 | 0 |**

**| 2755 | 27001 | 9 | B | 257 | 2 | 6 | 0 | 0 |**

**| 2755 | 91001 | 1 | C | 74 | 60 | 52 | 8 | 2 |**

**| 2755 | 94002 | 7 | LBW | 157 | 17 | 29 | 1 | 0 |**

**| 2755 | 98002 | 8 | RO | 253 | 2 | 4 | 0 | 0 |**

**| 9999 | 88888 | 1 | C | 1 | 100 | 40 | 4 | 5 |**

**| 9999 | 99999 | 1 | C | 1 | 100 | 40 | 4 | 5 |**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**MariaDB [CRICKET]&gt;**

**MariaDB [CRICKET]&gt; DELIMITER @**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE HUNDRENRUNS( INOUT RUNS INT)**

**-&gt; BEGIN**

**-&gt; SELECT NRUNS INTO RUNS FROM BATTNG;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.007 sec)**

**MariaDB [CRICKET]&gt; DELIMITER ;**

**MariaDB [CRICKET]&gt; SET @RUNS=&#39;100&#39;;**

**MariaDB [CRICKET]&gt; DROP PROCEDURE HUNDRENRUNS ;**

**Query OK, 0 rows affected (0.005 sec)**

**MariaDB [CRICKET]&gt; DELIMITER @**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE HUNDREN (INOUT RUNS INT)**

**-&gt; BEGIN**

**-&gt; SELECT NRUNS INTO RUNS FROM BATTING;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.006 sec)**

**MariaDB [CRICKET]&gt; DELIMITER ;**

**MariaDB [CRICKET]&gt; SET @RUNS=&#39;100&#39;;**

**Query OK, 0 rows affected (0.000 sec)**

**MariaDB [CRICKET]&gt; DELIMITER @**

**MariaDB [CRICKET]&gt; CREATE PROCEDURE TOPBATSMEN(INOUT RUNS INT)**

**-&gt; BEGIN**

**-&gt; SELECT NRUNS INTO RUNS FROM BATTING WHERE NRUNS = RUNS;**

**-&gt; END**

**-&gt; @**

**Query OK, 0 rows affected (0.006 sec)**

**MariaDB [CRICKET]&gt; DELIMITER ;**

**MariaDB [CRICKET]&gt; set @RUNS=&#39;100&#39;;**

**Query OK, 0 rows affected (0.000 sec)**

**MariaDB [CRICKET]&gt; call topbatsmen(@RUNS);**

**Query OK, 1 row affected (0.001 sec)**

**MariaDB [CRICKET]&gt; select @RUNS;**

**+-------+**

**| @RUNS |**

**+-------+**

**| 100 |**

**+-------+**

**MariaDB [CRICKET]&gt;**

**Conclusion:**

| From the above experiment I have understood how to Perform stored procedures and functions. |
| --- |

| **Course Code: CSL402** | **Course Name: Database Management System Lab** |
| --- | --- |
| **Class: SE-CO** | **Batch: 2020-21** |
| **Roll no: 20CO38** | **Name: MUJAWAR JUNED** |

**Experiment: 10**

**Aim: Implementation of View and Triggers.**

**Case Study Title – STUDENT MANAGEMENT SYSTEM**

**Theory:**

**Views:**

When the database is defined using the CREATE TABLE command in SQL, the user is instructing the DBMS to create tables and store them on disk. These tables can be retrieved, updated or deleted as the user wishes. These are real tables or base tables. The result of any SQL query is itself a table.

In normal query sessions, a query is executed and table is materialized, displayed and once the query is completed, discarded. In some situations, it may be convenient to store the query definition as a definition of a table that could be used in other queries. Such a table is called a view of the database.

A view is a virtual table that does not really exist physically as the base tables do and it has no real existence. Since the view is a virtual table, it is automatically updated when the base tables are updated.

Views are useful in controlling access to a database. Users may be permitted to see and manipulate only that data which is visible through some views. It also provides logical independence as the user dealing with the database through a view does not need to be aware of the tables that exist, since a view may be based on one or more base tables.

**Triggers: -**

A trigger is a named database object that is associated with a table, and that activates when a particular event occurs for the table. Some uses for triggers are to perform checks of values to be inserted into a table or to perform calculations on values involved in an update.

A trigger is defined to activate when a statement inserts, updates, or deletes rows in the associated table. These row operations are trigger events. For example, rows can be inserted by INSERT or LOAD DATA statements, and an insert trigger activates for each inserted row. A trigger can be set to activate either before or after the trigger event. For example, you can have a trigger activate before each row that is inserted into a table or after each row that is updated.

**Syntax:**

**DELIMITER //**

**CREATE TRIGGER TRIGGERNAME**

**BEFORE / AFTER**

**INSERT / UPDATE / DELETE**

**ON TABLE**

**FOR EACH ROW**

**BEGIN**

**-------CODE-----**

**END**

**//**

**Attach the output of the nested queries on your project in txt format.**

aiktc@aiktc10:~$ sudo mysql -u root -p

Enter password:

Welcome to the MySQL monitor. Commands end with ; or \g.

**mysql> show databases;**

**+--------------------+**

**| Database |**

**+--------------------+**

**| information\_schema |**

**| mysql |**

**| odimatch |**

**| org |**

**| performance\_schema |**

**| phpmyadmin |**

**| programmer |**

**| pythonExp |**

**| sys |**

**+--------------------+**

**9 rows in set (0.00 sec)**

**mysql> use org;**

**Reading table information for completion of table and column names**

**You can turn off this feature to get a quicker startup with -A**

**Database changed**

**mysql> show tables;**

**+---------------+**

**| Tables\_in\_org |**

**+---------------+**

**| bestworker |**

**| bestworkerfeb |**

**| bonus |**

**| log |**

**| title |**

**| worker |**

**+---------------+**

**6 rows in set (0.01 sec)**

**mysql> select \* from worker;**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | JOINING\_DATE | DEPARTMENT |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| 1 | Monika | Arora | 0 | 2014-02-20 09:00:00 | HR |**

**| 2 | Niharika | Verma | 80000 | 2014-06-11 09:00:00 | Admin |**

**| 3 | Vishal | Singhal | 300000 | 2014-02-20 09:00:00 | HR |**

**| 4 | Amitabh | Singh | 500000 | 2014-02-20 09:00:00 | Admin |**

**| 5 | Vivek | Bhati | 500000 | 2014-06-11 09:00:00 | Admin |**

**| 6 | Vipul | Diwan | 200000 | 2014-06-11 09:00:00 | Account |**

**| 7 | Satish | Kumar | 75000 | 2014-01-20 09:00:00 | Account |**

**| 8 | Geetika | Chauhan | 90000 | 2014-04-11 09:00:00 | Admin |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**8 rows in set (0.00 sec)**

**mysql> select \* from bonus;**

**+---------------+--------------+---------------------+**

**| WORKER\_REF\_ID | BONUS\_AMOUNT | BONUS\_DATE |**

**+---------------+--------------+---------------------+**

**| 1 | 5000 | 2016-02-20 00:00:00 |**

**| 2 | 3000 | 2016-06-11 00:00:00 |**

**| 3 | 4000 | 2016-02-20 00:00:00 |**

**| 1 | 4500 | 2016-02-20 00:00:00 |**

**| 2 | 3500 | 2016-06-11 00:00:00 |**

**+---------------+--------------+---------------------+**

**5 rows in set (0.00 sec)**

**mysql> CREATE VIEW BESTWORKER ( WORKER\_ID , FIRST\_NAME, LAST\_NAME, SALARY, BONUS\_AMOUNT, BONUS\_DATE) AS SELECT WORKER\_ID, FIRST\_NAME,LAST\_NAME,SALARY,BONUS\_AMOUNT,BONUS\_DATE FROM worker,bonus WHERE**

**WORKER\_ID = WORKER\_REF\_ID;**

**Query OK, 0 rows affected (0.17 sec)**

**mysql> select \* from BESTWORKER;**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | BONUS\_AMOUNT | BONUS\_DATE |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| 1 | Monika | Arora | 0 | 5000 | 2016-02-20 00:00:00 |**

**| 2 | Niharika | Verma | 80000 | 3000 | 2016-06-11 00:00:00 |**

**| 3 | Vishal | Singhal | 300000 | 4000 | 2016-02-20 00:00:00 |**

**| 1 | Monika | Arora | 0 | 4500 | 2016-02-20 00:00:00 |**

**| 2 | Niharika | Verma | 80000 | 3500 | 2016-06-11 00:00:00 |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**5 rows in set (0.00 sec)**

**mysql> SELECT \* FROM BESTWORKER WHERE MONTH(BONUS\_DATE) = 2;**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | BONUS\_AMOUNT | BONUS\_DATE |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| 1 | Monika | Arora | 0 | 5000 | 2016-02-20 00:00:00 |**

**| 3 | Vishal | Singhal | 300000 | 4000 | 2016-02-20 00:00:00 |**

**| 1 | Monika | Arora | 0 | 4500 | 2016-02-20 00:00:00 |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**3 rows in set (0.00 sec)**

**mysql> CREATE VIEW BESTWORKERFEBRUARY ( WORKER\_ID , FIRST\_NAME, LAST\_NAME, SALARY, BONUS\_AMOUNT, BONUS\_DATE) AS SELECT WORKER\_ID, FIRST\_NAME,LAST\_NAME,SALARY,BONUS\_AMOUNT,BONUS\_DATE FROM worker,bonu**

**s WHERE WORKER\_ID = WORKER\_REF\_ID AND MONTH(BONUS\_DATE)=2;**

**Query OK, 0 rows affected (0.16 sec)**

**mysql> SELECT \* FROM BESTWORKERFEBRUARY;**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | BONUS\_AMOUNT | BONUS\_DATE |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**| 1 | Monika | Arora | 0 | 5000 | 2016-02-20 00:00:00 |**

**| 3 | Vishal | Singhal | 300000 | 4000 | 2016-02-20 00:00:00 |**

**| 1 | Monika | Arora | 0 | 4500 | 2016-02-20 00:00:00 |**

**+-----------+------------+-----------+--------+--------------+---------------------+**

**3 rows in set (0.00 sec)**

**mysql> use odimatch;**

**Reading table information for completion of table and column names**

**You can turn off this feature to get a quicker startup with -A**

**Database changed**

**mysql> select \* from batting;**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**| MID | PID | NOrder | HOut | FOW | NRuns | Nballs | Fours | Six |**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**| 2689 | 23001 | 4 | C | 175 | 38 | 38 | 2 | 2 |**

**| 2689 | 25001 | 5 | C | 240 | 36 | 37 | 2 | 1 |**

**| 2689 | 27001 | 9 | C | 255 | 7 | 7 | 1 | 0 |**

**| 2689 | 89001 | 2 | C | 205 | 91 | 121 | 7 | 0 |**

**| 2689 | 98002 | 8 | LBW | 249 | 3 | 3 | 0 | 0 |**

**| 2755 | 23001 | 3 | C | 51 | 0 | 6 | 0 | 0 |**

**| 2755 | 25001 | 5 | C | 232 | 71 | 80 | 4 | 0 |**

**| 2755 | 27001 | 9 | B | 257 | 2 | 6 | 0 | 0 |**

**| 2755 | 91001 | 1 | C | 74 | 60 | 52 | 8 | 2 |**

**| 2755 | 94002 | 7 | LBW | 157 | 17 | 29 | 1 | 0 |**

**| 2755 | 98002 | 8 | RO | 253 | 2 | 4 | 0 | 0 |**

**+------+-------+--------+------+------+-------+--------+-------+------+**

**11 rows in set (0.00 sec)**

**mysql> select \* from player;**

**+----------+------------+---------------+--------------+-------+----------------+-------+**

**| PlayerID | Lname | Fname | Country | YBorn | BPlace | FTest |**

**+----------+------------+---------------+--------------+-------+----------------+-------+**

**| 23001 | Singh | Yuvraj | India | 1981 | Chandigarh | 2003 |**

**| 24001 | Symonds | Andrew | Australia | 1976 | Wollongong | 1999 |**

**| 25001 | Dhoni | MS | India | 1981 | Ranchi | 2005 |**

**| 25002 | Siddiqui | MahefuzRehman | India | 1996 | Bihar | 2015 |**

**| 27001 | Kumar | Praveen | India | 1986 | Meerut | NULL |**

**| 27002 | Sharma | Ishant | India | 1988 | Delhi | 2007 |**

**| 89001 | Tendulkar | Sachin | India | 1973 | Mumbai | 1989 |**

**| 90001 | Lara | Brian | West Indies | 1969 | Barbados | 1990 |**

**| 91001 | Jayasuriya | Sanath | Sri Lanka | 1969 | Matara | 1991 |**

**| 92001 | Warne | Shane | Australia | 1969 | Melbourne | 1992 |**

**| 92003 | Ul-Haq | Inzamam | Pakistan | 1970 | Multan | 1992 |**

**| 94002 | Vaas | Chaminda | Sri Lanka | 1974 | Mattumagala | 1994 |**

**| 95001 | Ponting | Ricky | Australia | 1974 | Launceston | 1995 |**

**| 95002 | Pollock | Shaun | South Africa | 1973 | Port Elizabeth | 1995 |**

**| 96001 | Dravid | Rahul | India | 1973 | Indore | 1996 |**

**| 96002 | Gibbs | Herschelle | South Africa | 1974 | Lahore | 1996 |**

**| 98001 | Afridi | Shahid | Pakistan | 1980 | Khyber Agency | 1998 |**

**| 98002 | Singh | Harbhajan | India | 1980 | Jalandhar | 1998 |**

**| 99001 | Lee | Brett | Australia | 1976 | Wollongong | 1999 |**

**| 99002 | Gilchrist | Adam | Australia | 1971 | Bellingen | 1991 |**

**| 99003 | Vaughan | Michael | England | 1974 | Manchester | 1999 |**

**| 99004 | Shaikh | Abdul Salam | India | 1991 | Mumbai | 2006 |**

**| 99005 | Ansari | Mukhtar | India | 1987 | Mumbai | 2004 |**

**| 99006 | Syed | Amer | India | 1991 | Parbhani | 2003 |**

**+----------+------------+---------------+--------------+-------+----------------+-------+**

**24 rows in set (0.00 sec)**

**mysql> CREATE VIEW BATSMAN ( PLAYER\_ID, FIRST\_NAME, LAST\_NAME, N\_RUNS, COUNTRY) AS SELECT PlayerID,Fname,Lname,NRuns,Country from player,batting where PlayerID = PID;**

**Query OK, 0 rows affected (0.13 sec)**

**mysql> CREATE VIEW BOLWER ( PLAYER\_ID, FIRST\_NAME, LAST\_NAME, N\_OVER,MAIDENS,N\_RUNS, N\_WICKETS) AS SELECT PlayerID,Fname,Lname,NOvers,Maidens,NRuns,NWickets from player,bowling where PlayerID = PID;**

**Query OK, 0 rows affected (0.25 sec)**

**mysql> select \* from BOLWER;**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| PLAYER\_ID | FIRST\_NAME | LAST\_NAME | N\_OVER | MAIDENS | N\_RUNS | N\_WICKETS |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| 23001 | Yuvraj | Singh | 3 | 0 | 27 | 1 |**

**| 24001 | Andrew | Symonds | 10 | 0 | 58 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 44 | 1 |**

**| 99001 | Brett | Lee | 10 | 0 | 58 | 1 |**

**| 23001 | Yuvraj | Singh | 10 | 0 | 53 | 2 |**

**| 91001 | Sanath | Jayasuriya | 4 | 0 | 29 | 0 |**

**| 92001 | Shane | Warne | 10 | 0 | 56 | 1 |**

**| 94002 | Chaminda | Vaas | 9 | 1 | 40 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 40 | 3 |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**9 rows in set (0.00 sec)**

**mysql> DELIMITER /**

**mysql> SELECT \* FROM BOLWER/**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| PLAYER\_ID | FIRST\_NAME | LAST\_NAME | N\_OVER | MAIDENS | N\_RUNS | N\_WICKETS |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| 23001 | Yuvraj | Singh | 3 | 0 | 27 | 1 |**

**| 24001 | Andrew | Symonds | 10 | 0 | 58 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 44 | 1 |**

**| 99001 | Brett | Lee | 10 | 0 | 58 | 1 |**

**| 23001 | Yuvraj | Singh | 10 | 0 | 53 | 2 |**

**| 91001 | Sanath | Jayasuriya | 4 | 0 | 29 | 0 |**

**| 92001 | Shane | Warne | 10 | 0 | 56 | 1 |**

**| 94002 | Chaminda | Vaas | 9 | 1 | 40 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 40 | 3 |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**9 rows in set (0.00 sec)**

**mysql> SELECT \* FROM BOLWER;**

**-> /**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| PLAYER\_ID | FIRST\_NAME | LAST\_NAME | N\_OVER | MAIDENS | N\_RUNS | N\_WICKETS |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**| 23001 | Yuvraj | Singh | 3 | 0 | 27 | 1 |**

**| 24001 | Andrew | Symonds | 10 | 0 | 58 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 44 | 1 |**

**| 99001 | Brett | Lee | 10 | 0 | 58 | 1 |**

**| 23001 | Yuvraj | Singh | 10 | 0 | 53 | 2 |**

**| 91001 | Sanath | Jayasuriya | 4 | 0 | 29 | 0 |**

**| 92001 | Shane | Warne | 10 | 0 | 56 | 1 |**

**| 94002 | Chaminda | Vaas | 9 | 1 | 40 | 1 |**

**| 98002 | Harbhajan | Singh | 10 | 0 | 40 | 3 |**

**+-----------+------------+------------+--------+---------+--------+-----------+**

**9 rows in set (0.00 sec)**

**mysql> CREATE TABLE LOGS ( WORKER\_ID INT(4), SALARYCHANGE BIGINT(10),SDATE TIMESTAMP)/**

**Query OK, 0 rows affected, 2 warnings (1.27 sec)**

**mysql> DESC LOGS/**

**+--------------+-----------+------+-----+---------+-------+**

**| Field | Type | Null | Key | Default | Extra |**

**+--------------+-----------+------+-----+---------+-------+**

**| WORKER\_ID | int | YES | | NULL | |**

**| SALARYCHANGE | bigint | YES | | NULL | |**

**| SDATE | timestamp | YES | | NULL | |**

**+--------------+-----------+------+-----+---------+-------+**

**3 rows in set (0.00 sec)**

**mysql> use org/**

**Reading table information for completion of table and column names**

**You can turn off this feature to get a quicker startup with -A**

**Database changed**

**mysql> CREATE TABLE LOGS ( WORKER\_ID INT(4), SALARYCHANGE BIGINT(10),SDATE TIMESTAMP)/**

**Query OK, 0 rows affected, 2 warnings (0.52 sec)**

**mysql> CREATE TRIGGER SALARYUPDATES AFTER UPDATE ON worker FOR EACH ROW INSERT INTO LOG VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); END/**

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

**ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'END' at line 1**

**mysql> SELECT \* FROM LOGS/**

**Empty set (0.00 sec)**

**mysql> SELECT \* FROM worker/**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | JOINING\_DATE | DEPARTMENT |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| 1 | Monika | Arora | 0 | 2014-02-20 09:00:00 | HR |**

**| 2 | Niharika | Verma | 80000 | 2014-06-11 09:00:00 | Admin |**

**| 3 | Vishal | Singhal | 300000 | 2014-02-20 09:00:00 | HR |**

**| 4 | Amitabh | Singh | 500000 | 2014-02-20 09:00:00 | Admin |**

**| 5 | Vivek | Bhati | 500000 | 2014-06-11 09:00:00 | Admin |**

**| 6 | Vipul | Diwan | 200000 | 2014-06-11 09:00:00 | Account |**

**| 7 | Satish | Kumar | 75000 | 2014-01-20 09:00:00 | Account |**

**| 8 | Geetika | Chauhan | 90000 | 2014-04-11 09:00:00 | Admin |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**8 rows in set (0.01 sec)**

**mysql> SELECT \* FROM LOGS/**

**Empty set (0.00 sec)**

**mysql> SHOW TABLES;**

**-> /**

**+--------------------+**

**| Tables\_in\_org |**

**+--------------------+**

**| BESTWORKER |**

**| BESTWORKERFEBRUARY |**

**| LOGS |**

**| bestworker |**

**| bestworkerfeb |**

**| bonus |**

**| log |**

**| title |**

**| worker |**

**+--------------------+**

**9 rows in set (0.01 sec)**

**mysql> SELECT \* FROM LOGS/**

**Empty set (0.00 sec)**

**mysql> DESC LOGS/**

**+--------------+-----------+------+-----+---------+-------+**

**| Field | Type | Null | Key | Default | Extra |**

**+--------------+-----------+------+-----+---------+-------+**

**| WORKER\_ID | int | YES | | NULL | |**

**| SALARYCHANGE | bigint | YES | | NULL | |**

**| SDATE | timestamp | YES | | NULL | |**

**+--------------+-----------+------+-----+---------+-------+**

**3 rows in set (0.00 sec)**

**mysql> DESC worker/**

**+--------------+----------+------+-----+---------+----------------+**

**| Field | Type | Null | Key | Default | Extra |**

**+--------------+----------+------+-----+---------+----------------+**

**| WORKER\_ID | int | NO | PRI | NULL | auto\_increment |**

**| FIRST\_NAME | char(25) | YES | | NULL | |**

**| LAST\_NAME | char(25) | YES | | NULL | |**

**| SALARY | int | YES | | NULL | |**

**| JOINING\_DATE | datetime | YES | | NULL | |**

**| DEPARTMENT | char(25) | YES | | NULL | |**

**+--------------+----------+------+-----+---------+----------------+**

**6 rows in set (0.00 sec)**

**mysql> CREATE TRIGGER SALARYUPDATE1 AFTER UPDATE ON worker FOR EACH ROW INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); END/**

**Query OK, 0 rows affected (0.20 sec)**

**ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'END' at line 1**

**mysql> drop trigger salaryupdates;**

**-> /**

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

**mysql> show tables;**

**-> /**

**+--------------------+**

**| Tables\_in\_org |**

**+--------------------+**

**| BESTWORKER |**

**| BESTWORKERFEBRUARY |**

**| LOGS |**

**| bestworker |**

**| bestworkerfeb |**

**| bonus |**

**| log |**

**| title |**

**| worker |**

**+--------------------+**

**9 rows in set (0.00 sec)**

**mysql> drop table log;**

**-> /**

**Query OK, 0 rows affected (0.35 sec)**

**mysql> show triggers;**

**-> /**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**| Trigger | Event | Table | Statement | Timing | Created | sql\_mode | Definer | character\_set\_client | collation\_connection | Database Collation |**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**| BEFOREINSERT | INSERT | worker | BEGIN**

**INSERT INTO LOG VALUES (NEW.WORKER\_ID, NEW.SALARY, NOW(), 'INSERT');**

**END | BEFORE | 2022-04-05 10:31:51.62 | NO\_AUTO\_VALUE\_ON\_ZERO | root@localhost | utf8mb4 | utf8mb4\_general\_ci | utf8mb4\_0900\_ai\_ci |**

**| SALARYUPDATE | UPDATE | worker | INSERT INTO LOG VALUES (OLD.WORKER\_ID, NEW.SALARY, NOW() ) | AFTER | 2022-04-05 10:31:51.79 | NO\_AUTO\_VALUE\_ON\_ZERO | root@localhost | utf8mb4 | utf8mb4\_general\_ci | utf8mb4\_0900\_ai\_ci |**

**| SALARYUPDATE1 | UPDATE | worker | INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); | AFTER | 2022-04-05 11:32:08.51 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**| DELETEEMP | DELETE | worker | BEGIN**

**INSERT INTO LOG VALUES (OLD.WORKER\_ID, OLD.SALARY, NOW(), 'DELETE');**

**END | BEFORE | 2022-04-05 10:31:51.96 | NO\_AUTO\_VALUE\_ON\_ZERO | root@localhost | utf8mb4 | utf8mb4\_general\_ci | utf8mb4\_0900\_ai\_ci |**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**4 rows in set (0.01 sec)**

**mysql> drop TRIGGER SALARYUPDATE/**

**Query OK, 0 rows affected (0.13 sec)**

**mysql> drop TRIGGER DELETEEMP/**

**Query OK, 0 rows affected (0.12 sec)**

**mysql> SELECT \* FROM LOGS/**

**Empty set (0.00 sec)**

**mysql> UPDATE worker SET SALARY = 0 WHERE WORKER\_ID = 2/**

**Query OK, 1 row affected (0.12 sec)**

**Rows matched: 1 Changed: 1 Warnings: 0**

**mysql> SELECT \* FROM LOGS/**

**+-----------+--------------+---------------------+**

**| WORKER\_ID | SALARYCHANGE | SDATE |**

**+-----------+--------------+---------------------+**

**| 2 | 0 | 2022-04-05 11:36:06 |**

**+-----------+--------------+---------------------+**

**1 row in set (0.00 sec)**

**mysql> ALTER TABLE LOGS ADD ACTION VARCHAR(10)/**

**Query OK, 0 rows affected (0.44 sec)**

**Records: 0 Duplicates: 0 Warnings: 0**

**mysql> SHOW TRIGGERS/**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**| Trigger | Event | Table | Statement | Timing | Created | sql\_mode | Definer | character\_set\_client | collation\_connection | Database Collation |**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**| BEFOREINSERT | INSERT | worker | BEGIN**

**INSERT INTO LOG VALUES (NEW.WORKER\_ID, NEW.SALARY, NOW(), 'INSERT');**

**END | BEFORE | 2022-04-05 10:31:51.62 | NO\_AUTO\_VALUE\_ON\_ZERO | root@localhost | utf8mb4 | utf8mb4\_general\_ci | utf8mb4\_0900\_ai\_ci |**

**| SALARYUPDATE1 | UPDATE | worker | INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); | AFTER | 2022-04-05 11:32:08.51 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**+---------------+--------+--------+----------------------------------------------------------------------------------+--------+------------------------+-----**

**2 rows in set (0.00 sec)**

**mysql> DROP TRIGGER BEFOREINSERT/**

**Query OK, 0 rows affected (0.17 sec)**

**mysql> SHOW TRIGGERS/**

**+---------------+--------+--------+-----------------------------------------------------------+--------+------------------------+----------------------------**

**| Trigger | Event | Table | Statement | Timing | Created | sql\_mode | Definer | character\_set\_client | collation\_connection | Database Collation |**

**---------------+----------------------+----------------------+---------------**

**| SALARYUPDATE1 | UPDATE | worker | INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); | AFTER | 2022-04-05 11:32:08.51 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**+---------------+--------+--------+------------------------------------------**

**1 row in set (0.00 sec)**

**mysql> CREATE TRIGGER BEFOREINSERT BEFORE INSERT ON worker FOR EACH ROW BEGIN INSERT INTO LOGS VALUES (NEW.WORKER\_ID,NEW.SALARY,NOW(),'INSERT');END/**

**Query OK, 0 rows affected (0.20 sec)**

**mysql> INSERT INTO worker VALUES (9,'JUNED','SHAIKH',50000,'2020-02-20 09:00:00' ,'HR')/**

**Query OK, 1 row affected (0.20 sec)**

**mysql> SELECT \* FROM worker/**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| WORKER\_ID | FIRST\_NAME | LAST\_NAME | SALARY | JOINING\_DATE | DEPARTMENT |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**| 1 | Monika | Arora | 0 | 2014-02-20 09:00:00 | HR |**

**| 2 | Niharika | Verma | 0 | 2014-06-11 09:00:00 | Admin |**

**| 3 | Vishal | Singhal | 300000 | 2014-02-20 09:00:00 | HR |**

**| 4 | Amitabh | Singh | 500000 | 2014-02-20 09:00:00 | Admin |**

**| 5 | Vivek | Bhati | 500000 | 2014-06-11 09:00:00 | Admin |**

**| 6 | Vipul | Diwan | 200000 | 2014-06-11 09:00:00 | Account |**

**| 7 | Satish | Kumar | 75000 | 2014-01-20 09:00:00 | Account |**

**| 8 | Geetika | Chauhan | 90000 | 2014-04-11 09:00:00 | Admin |**

**| 9 | JUNED | SHAIKH | 50000 | 2020-02-20 09:00:00 | HR |**

**+-----------+------------+-----------+--------+---------------------+------------+**

**9 rows in set (0.00 sec)**

**mysql> select \* from LOGS/**

**+-----------+--------------+---------------------+--------+**

**| WORKER\_ID | SALARYCHANGE | SDATE | ACTION |**

**+-----------+--------------+---------------------+--------+**

**| 2 | 0 | 2022-04-05 11:36:06 | NULL |**

**| 9 | 50000 | 2022-04-05 11:54:17 | INSERT |**

**+-----------+--------------+---------------------+--------+**

**2 rows in set (0.00 sec)**

**mysql> SHOW TRIGGERS/**

**+---------------+--------+--------+------------------------------------------**

**| Trigger | Event | Table | Statement | Timing | Created | sql\_mode | Definer | character\_set\_client | collation\_connection | Database Collation |**

**+---------------+--------+--------+------------------------------------------| BEFOREINSERT | INSERT | worker | BEGIN INSERT INTO LOGS VALUES (NEW.WORKER\_ID,NEW.SALARY,NOW(),'INSERT');END | BEFORE | 2022-04-05 11:47:54.31 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**| SALARYUPDATE1 | UPDATE | worker | INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); | AFTER | 2022-04-05 11:32:08.51 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci |**

**mysql> CREATE TRIGGER DELETEEMP BEFORE DELETE ON worker FOR EACH ROW BEGIN INSERT INTO LOGS VALUES (OLD,WORKER\_ID,OLD.SALARY,NOW(),'DELETE');END/**

**Query OK, 0 rows affected (0.20 sec)**

**mysql> SHOW TRIGGERS/**

**+---------------+--------+--------+------------------------------------------**

**| Trigger | Event | Table | Statement | Timing | Created | sql\_mode | Definer | character\_set\_client | collation\_connection | Database Collation |**

**+---------------+--------+--------+------------------------------------------| BEFOREINSERT | INSERT | worker | BEGIN INSERT INTO LOGS VALUES (NEW.WORKER\_ID,NEW.SALARY,NOW(),'INSERT');END | BEFORE | 2022-04-05 11:47:54.31 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**| SALARYUPDATE1 | UPDATE | worker | INSERT INTO LOGS VALUES (OLD.WORKER\_ID,NEW.SALARY,NOW()); | AFTER | 2022-04-05 11:32:08.51 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**| DELETEEMP | DELETE | worker | BEGIN INSERT INTO LOGS VALUES (OLD,WORKER\_ID,OLD.SALARY,NOW(),'DELETE');END | BEFORE | 2022-04-05 11:59:39.86 | ONLY\_FULL\_GROUP\_BY,STRICT\_TRANS\_TABLES,NO\_ZERO\_IN\_DATE,NO\_ZERO\_DATE,ERROR\_FOR\_DIVISION\_BY\_ZERO,NO\_ENGINE\_SUBSTITUTION | root@localhost | utf8mb4 | utf8mb4\_0900\_ai\_ci | utf8mb4\_0900\_ai\_ci |**

**+---------------+--------+--------+------------------------------------------**

**3 rows in set (0.00 sec)**

**mysql> SELECT \* FROM LOGS/**

**+-----------+--------------+---------------------+--------+**

**| WORKER\_ID | SALARYCHANGE | SDATE | ACTION |**

**+-----------+--------------+---------------------+--------+**

**| 2 | 0 | 2022-04-05 11:36:06 | NULL |**

**| 9 | 50000 | 2022-04-05 11:54:17 | INSERT |**

**+-----------+--------------+---------------------+--------+**

**2 rows in set (0.00 sec)**

**mysql>**

**Conclusion:**

| From the above experiment I have understood how to Perform Implementation of View and Triggers. |
| --- |