LAB MANUAL

STUDENT VERSION

for

DATABASE MANAGEMENT SYSTEM LAB

(THIRD SEMESTER, COURSE CODE: 3139)

Department Of Computer Engineering

GOVERNMENT POLYTECHNIC COLLEGE, PERUMBAVOOR

Koovappady P.O, Phone: 0484 2649251

email:gptcpbvr@gmail.com

This page is intentionally kept blank to preserve page format

INDEX

SL NO:	NAME OF EXPERIMENT	DATE	PAGE NO:
	VISION AND MISSION		
	PEO, PO AND PSOS OF THE PROGRAM		
	GENERAL INSTRUCTIONS		
1	SAFETY PROCEDURES		
2	IMPLEMENT DATA DEFINITION LANGUAGE COMMANDS		
3	IMPLEMENT CONSTRAINTS		
4	IMPLEMENT DATA MANIPULATION LANGUAGE COMMANDS		
5	COMPUTATION ON TABLES WITH BUILT IN FUNCTIONS		
6	CREATE NESTED QUERIES AND JOINS		
7	CREATE VIEWS		
8	CREATE PROCEDURES		
9	CREATE TRIGGERS		
10	NORMALIZE TABLES		
11	CREATE SMALL APPLICATION AND DO THE DATABASE CONNECTIVITY		

GENERAL REMARKS (*FOR OFFICE USE ONLY*)							
TEST 1:		TEST 2:		ASSIGN 1:		ASSIGN 2:	

VISION AND MISSION

Government Polytechnic College, Perumbavoor
Vision: Excel as a centre of skill education moulding professionals who sincerely strive for the betterment of society **Mission:** • To impart state of the art knowledge and skill to the graduate and moulding them to be competent, committed and responsible for the well-being of society • To apply technology in the traditional skills, thereby enhancing the living standard of the community

Department of Computer Engineering
Vision: Excel as a center of skill education in Computer Engineering moulding professionals who sincerely strive for the betterment of themselves and the society. **Mission:** • To impart state of the art knowledge, skill and attitude to the graduates and contribute to their sustainable development • To merge technologies in the field of computer engineering with occupational skills, thereby improving the quality of living

PEO, PO AND PSOs OF THE PROGRAM

PROGRAM OUTCOMES

PO1: Basic and Discipline specific knowledge: Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the engineering problems.

PO2: Problem analysis: Identify and analyse well-defined engineering problems using codified standard methods.

PO3: Design/ development of solutions: Design solutions for well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

PO4: Engineering Tools, Experimentation and Testing: Apply modern engineering tools and appropriate technique to conduct standard tests and measurements.

PO5: Engineering practices for society, sustainability and environment: Apply appropriate technology in context of society, sustainability, environment and ethical practices.

PO6: Project Management: Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

PO7: Life-long learning: Ability to analyse individual needs and engage in updating in the context of technological changes.

PROGRAM SPECIFIC OUTCOMES (PSOS)

PSO1: Apply concepts and knowledge in the field of software systems, hardware and networking with concern for the society.

PSO2: Generate ideas from the knowledge of engineering specialization leading to professional growth.

PSO3: Apply knowledge and understanding of engineering principles to initiate entrepreneurship ventures.

PROGRAM EDUCATIONAL OBJECTIVES (PEOS)

PEO1: Secure successful careers in hardware and software design, development, testing, maintenance and marketing.

PEO2: Acquire knowledge and competency in the domain to develop innovative, cost effective and socially acceptable solutions to engineering problems in a multi-disciplinary work environment.

PEO3: Develop strong fundamental knowledge that prepares them for professional careers/ higher studies with attitude for lifelong learning.

PEO4: Instill the attitude to be sensitive to ethical, societal and environmental issues while pursuing their professional duties.

PEO5: Possess leadership qualities and be effective communicator to work efficiently with diverse teams, promote and practice appropriate ethical practices.

GENERAL INSTRUCTIONS

Rough record and Fair record are needed to record the experiments conducted in the laboratory. Rough records are needed to be certified immediately on completion of the experiment. Fair records are due at the beginning of the next lab period. Fair records must be submitted as neat, legible, and complete.

INSTRUCTIONS TO STUDENTS FOR WRITING THE FAIR RECORD

In the fair record, the index page should be filled properly by writing the corresponding experiment number, experiment name , date on which it was done and the page number.

On the right side page of the record following has to be written:

1. Title: The title of the experiment should be written in the page in capital letters.

2.Exp No: And Date: In the top margin, experiment number and date should be written.

3. Aim: The purpose of the experiment should be written clearly.

4.Principle/Theory: Simple algorithm should be written

5. Procedure: Steps for doing the experiment.

6. Program: Simple working of the algorithm should be written.

7. Results: The results of the experiment must be summarized in writing and should be fulfilling the aim.

On the Left side page of the record following has to be recorded:

1. **Input:** Input of the program given
2. **Output :** Output of the program
3. **Design:** The design of the output (if necessary).

Exp No : 01 **Date :** D D / M M / Y Y

SAFETY PROCEDURES

Problem Statement:

The safety instructions are presented to the attention of the students as a mean of preventing accidents while performing experiments and activities in Software lab of the department .The purpose is to draw attention to the risks involved in lab activities to prevent human suffering and damage to equipment.

Safety in the laboratory:

Working in the lab is not allowed without following electricity precautions displayed.
No individual work is allowed in the lab.
Laboratory in charge is responsible for the arrangements of your lab activities; Listen carefully to his/her instructions and follow them.

To do and not to do:

Inform the lab in charge about dangerous conditions and faults in the lab or nearby environment.
Do not do any action that may harm people or equipment in the lab.
Do not misuse any of the tools or instruments belong to the lab.
Strict discipline should be maintained in the laboratory.
Turn off cell phones before entering the lab.
At the end and beginning of laboratory, follow 5S procedures and leave the work table clean and tidy.

Electrical Safety:
Consult Electrical Engineering section available in the campus for electrical safety queries.
The lab equipment is powered from electrical sockets installed on the tables.
Do not use equipment that is powered from a damaged socket.
Do not use equipment that is powered from flexible cable with damaged insulation or if it's plug is not assembled properly.
Do not repair or disassemble electrical equipment including replacement of fuses installed in the equipment.

Do not open the main fuse box, unless it is an emergency and you need to switch off main circuit breaker.
Be sure to turn off the power and remove the power plug from all equipment before working repairing or assembling.
Do not plug in or remove equipment while the power is on.

Emergency Switches:
The laboratory has circuit breakers, which is located in the main panel. Identify the place.
In an emergency condition, switch off circuit breakers immediately.

Result:
Familiarization of safety precautions performed

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 02 **Date :** D D / M M / Y Y

IMPLEMENT DATA DEFINITION LANGUGE COMMANDS

AIM:

To execute and verify the Data Definition Language commands.

OBJECTVE:

To understand DDL commands

THEORY:

The commands used are:

- CREATE: - It is used to create a table.
- ALTER: - The structure of a table can be modified by using the ALTER TABLE command. This command is used to add a new column, modify the existing column definition and to include or drop integrity constraint.
- DROP: -It will delete the table structure provided the table should be empty.
- TRUNCATE: - If there is no further use of records stored in a table and the structure has to be retained, and then the records alone can be deleted.
- DESC : - This is used to view the structure of the table

PROCEDURE:

CREATION OF TABLE:

Syntax:

CREATE TABLE<table name> (column1 datatype, column2 datatype...);

(a)To Add column to existing Table

Syntax:

ALTER TABLE table-name ADD(column-name datatype);

(b)To Add Multiple columns to existing Table

Syntax:

ALTER TABLE table-name ADD(column-name1 datatype1, column-name2 datatype2, column-name3 datatype3);

(c) Dropping a Column from a Table

Syntax:

ALTER TABLE <Table Name>DROP COLUMN <CoumnName>;

(d) Modifying Existing Columns

Syntax:

ALTER TABLE <Table Name>MODIFY <CoumnName><Newdatatype>(<size>));

(e) To rename a column Using alter command we can rename an existing column

Syntax:

ALTER TABLE table-name RENAME old-column-name TO column-name;

RENAMING TABLES

Syntax:

RENAME<oldtable> TO <new table>;

TRUNCATE TABLE

Syntax:

TRUNCATE TABLE <table_name>;

DESTROYING TABLES

Syntax:

DROP TABLE <table_name>;

Syntax:

DESC <table_name>;

RESULT:

The DDL commands have been executed successfully

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 03 **Date :** D D / M M / Y Y

CONSTRAINTS

AIM:

To implement Data Constraints.

THEORY

Constraints are the business Rules which are enforced on the data being stored in a table are called Constraints

TYPES OF CONSTRAINTS:

1) Primary key
2) Foreign key/references
3) Check
4) Unique
5) Not null
6) Null
7) Default

PROCEDURE:

(a) The PRIMARY KEY

The PRIMARY KEY defined at column level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE CONSTRAINT <constraintname1> PRIMARY KEY,Columnname2 DATATYPE, columnname3 DATATYPE,.....);

The PRIMARY KEY defined at table level:

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, PRIMARY KEY (columnname1, columnname2));

(b)CHECK CONSTRAINT

The CHECK Constraint defined at column level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE CHECK (logical expression), columnname2 DATATYPE, columnname3 DATATYPE,...);
The CHECK Constraint defined at table level:

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, CHECK (logical expression1), CHECK (logical expression2));

(c) UNIQUE CONSTRAINT
The UNIQUE Constraint defined at the column level

Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE UNIQUE, columnname2 DATATYPE UNIQUE, columnname3 DATATYPE ...);

The UNIQUE Constraint defined at the the table level:
Syntax:

CREATE TABLE tablename (Columnname1 DATATYPE, columnname2 DATATYPE, columnname3 DATATYPE, UNIQUE (columnname1));

(d) Not Null

Syntax:

CREATE TABLE tablename(Columnname1 DATATYPE NOT NULL, columnname2 DATATYPE NOT NULL,columnname3 DATATYPE,...);

RESULT:
The Data Constraints are successfully implemented

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 04 **Date :** D D / M M / Y Y

DATA MANIPULATION LANGUAGE

AIM:

To execute the Data Manipulation Language (DML) commands.

OBJECTIVE

To understand Data Manipulation Language (DML) commands

THEORY

DML commands are the most frequently used SQL commands and is used to query and manipulate the existing database objects. Some of the commands are

- INSERT: - This is used to add one or more rows to a table. The values are separated by commas and the data types char and date are enclosed in apostrophes. The values must be entered in the same order as they are defined.
- SELECT: - It is used to retrieve information from the table.it is generally referred to as querying the table. We can either display all columns in a table or only specify column from the table.
- UPDATE: - It is used to alter the column values in a table. A single column may be updated or more than one column could be updated.
- DELETE: - After inserting row in a table we can also delete them if required. The delete command consists of a from clause followed by an optional where clause

PROCEDURE

INSERT COMMAND

(a)Inserting a single row into a table:

Syntax:

insert into <table name> values (<expression1>,<expression2>)

(b) Inserting more than one record using a single insert commands:

Syntax:

insert into <table name> values (&col1, &col2, ….)

(c) Skipping the fields while inserting:

INSERT INTO <tablename> (<column name1>, <column name3>) VALUES (<expression1>, <expression3>);

Other way is to give null while passing the values.

SELECT COMMAND

(a) View all rows and all columns

Syntax:

Select * from tablename;

Syntax:

Select <column1>,<column2> from tablename;

(c)Selected Columns And selected Rows

Syntax:
SELECt <column1>, <column2> FROM <tablename> WHERE <condition> ;

(c)Eliminating duplicate rows

Syntax:
SELECT DISTINCT <column1>, <column2> FROM <tablename>

UPDATE COMMAND

(b) updating all rows

Syntax:

Update tablename set columnname1>=<exprssion1>,<columnname2>=<exprssion2>;

(b) updating records conditionally

Syntax:

update tablename set field=values where condition;

(b)Removal of all rows

Syntax:

Delete from <table name> ;

(b)removal of specific rows

Syntax:

Delete from <table name> where <condition>;

RESULT

The DML commands are executed successfully.

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 05 **Date :** D D / M M / Y Y

COMPUTATIONS ON TABLE DATA WITH BUILT IN FUCTIONS

AIM

To implement computations done on data of the given table

OBJECTIVE

To understand computations done on data of the given table with built in functions

THEORY AND PROCEDURE

Group Functions/Aggregate functions:

A group function returns a result based on group of rows.

1. avg

Example:

select avg (total) from student;

2.max

Example:

select max (percentagel) from student;

3.min

Example: select min (marksl) from student;

4. sum

Example:

select sum(price) from product
Count Function

In order to count the number of rows, count function is used.

1. count(*) – It counts all, inclusive of duplicates and nulls.

Example:

select count(*) from student;

2. count(col_name)– It avoids null value.

Example:

select count(total) from order;

3. count(distinct col_name) – It avoids the repeated and null values.

Example: select count(distinct ordid) from order;

Special Clauses:

Group by clause

This allows us to use simultaneous column name and group functions.

Example:

Select max(percentage), deptname from student group by deptname;

Having clause

This is used to specify conditions on rows retrieved by using group by clause.

Example:

Select max(percentage), deptname from student group by deptname having count(*)>=5;

RESULT
Computations on data are implemented successfully

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 06 **Date :** D D / M M / Y Y

NESTED QUERIES/SUB QUERIES AND JOINS

AIM

To implement nested queries and joins on the given table .

OBJECTIVE

To understand nested queries and joins.

THEORY

NESTED QUERIES:

A sub query is a query within a query. Sub queries can be created within your SQL statements. These sub queries can reside in the WHERE clause, the FROM clause, or the SELECT clause.

JOINS:

Join is a query in which data is returned from two or more tables.

Natural join:

It returns the matching rows from the table that are being joined

Syntax:

select <attribute> from TN where TN1.attribute=TN2.attribute.

Inner join:

It returns the matching rows from the table that are being joined.

Syntax:

select <attribute> from TN1 innerjoin TN2 on TN1.attribute=TN2.attribute.

Left outer join:

It returns all the rows from the table1 even when they are unmatched.

Syntax:

select <attribute> from TN1 left outer join TN2 on TN1.attribute=TN2.attribute.
select <attribute> from TN where TN1.attribute(+)=TN2.attribute.

Right outer join:

It returns all the rows from the table2 even when they are unmatched.

Syntax:

select <attribute> from TN1 right outer join TN2 on TN1.attribute=TN2.attribute.
select <attribute> from TN where TN1.attribute=(+)TN2.attribute.

PROCEDURE

SQL>create table student(sid numeric(3), sname varchar(10), D_id numeric(5));
SQL>insert
SQL>select * from student;
SQL>create table department(D_id numeric(5),D_name varchar(10));
SQL>insert
SQL>select * from department
SQL>select * from student, department where student.D_id=department.D_id;
SQL>select * from student **left outer join** department on
student.D_id=department.D_id;
SQL>select * from student **right outer join** department on
student.D_id=department.D_id

NESTED QUERIES -

SQL> desc emp_det;
SQL> desc pro_det;
SQL> desc work_in;
SQL> select * from emp_det;
SQL> select * from Pro_det;

SQL> select * from work_in;

NESTED QUERIES

(i) SQL> select ename from emp_det where dno not in(select dno from emp_det where ename ='SaravanaKumar');
(ii)SQL> select ename, dno from emp_det where dno = (select dno from emp_det where ename ='RajKumar');
(iii)SQL> select ename from emp_det where eno in(select eno from work_in where pno = (select pno from pro_det where pname = 'DBMS')) order by ename;
(iv)SQL> select ename, basic_sal from emp_det where dno = 2 and basic_sal>(select max(basic_sal) from emp_det where dno = 10) order by ename;
(v)SQL> select pno,pname from pro_det where exists(select pno from work_in where work_in.pno =pro_det.pno);
(vi)SQL>select ename, job_status,basic_sal from emp_det where (dno,basic_sal) in (select dno,basic_sal from emp_det where ename ='RajKumar');
(vii)SQL>select * from emp_det where basic_sal=(select max(basic_sal) from emp_det);
(viii)SQL>select max(basic_sal) from emp_det where basic_sal< (select max(basic_sal) from emp_det);
(ix)SQL> select * from emp_det where basic_sal < (select avg(basic_sal) from emp_det);

RESULT:

Thus the nested queries and join operations are executed and verified in DBMS.

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 07 **Date :** D D / M M / Y Y

VIEWS

AIM

To create and drop View on the given table.

OBJECTIVE

To implement views

THEORY

A view is the tailored presentation of data contained in one or more table and can also be said as restricted view to the data‟s in the tables. A view is a “virtual table” or a “stored query” which takes the output of a query and treats it as a table. The table upon which a view is created is called as base table . A view is a logical table based on a table or another view. A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The tables on which a view is based are called base tables. The view is stored as a SELECT statement in the data dictionary .

Advantages of a view:

a. Additional level of table security.
b. Hides data complexity.
c. Simplifies the usage by combining multiple tables into a single table

Creating and dropping view:

Syntax:

Create or replace view view_name AS SELECT column_name(s) FROM table_name WHERE condition

Drop view <view name>;

PROCEDURE

1)create a table aa
`

2)describe the table aa

3) create table qq

4) describe table qq

5) create a view on qq

6) display the view

7) Update view statements

RESULT

Thus the view creation commands are executed successfully

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 08 **Date :** D D / M M / Y Y

PROCEDURE

AIM

To find factorial of a number using function .

OBJECTIVE

To write PL/SQL(Functions)and to understand stored procedures in SQL.

PROCEDURE:

create [or replace] procedure procedurename [parameter[in/out/in/in out]
datatype [:=/default expression] [(parameter)] is/as declaration
begin pl/sql codes [exception]
end

PROGRAM

……..
……..
……..

OUTPUT

SQL> mysql –u root –p;
SQL>use database student;
SQL>source filename.sql
SQL>call fact(5)
SQL>

RESULT:

Thus the functions and stored procedures are executed in SQL.

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 09 **Date :** D D / M M / Y Y

TRIGGER

AIM

Create a Trigger for EMP table it will update another table SALARY while inserting values.

OBJECTIVE

To develop and execute a Trigger Before and After update/Delete/Insert operations on a table
.

PROCEDURE

step 1: start
step 2: initialize the trigger with specific table id.
step 3:specify the operations (update, delete, insert) for which the trigger has to be executed.
step 4: execute the trigger procedure for both before and after sequences
step 5: carryout the operation on the table to check for trigger execution.
step 6: stop

PROGRAM
…….
…….
…….

RESULT:
The trigger procedure has been executed successfully for both before and after sequences

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 10 **Date :** D D / M M / Y Y

NORMALIZATION OF TABLES

AIM

Checking normalization of database tables

PROBLEM STATEMENT

An exercise to check whether the given database table is normalized or not. If yes find out the status of normalization and reasoning.

OBJECTIVE

To study the concept of various levels of normalization and understand how to convert into normalized forms.

.

THEORY

Design/Theory
Create a database table in SQL with a few no of rows and columns. Analyze the table and determine to which normal form it belongs to according to the rules and regulations of each normal forms.

PROCEDURE

Consider a book database table as given below.

Author Last Name	Author First Name	Book Title	Subject	Collection or Library	Building
Berdahl	Robert	The Politics of the Prussian Nobility	History	PCL General Stacks	Perry-Castañeda Library
Yudof	Mark	Child Abuse and Neglect	Legal Procedures	Law Library	Townes Hall
Harmon	Glynn	Human Memory and Knowledge	Cognitive Psychology	PCL General Stacks	Perry-Castañeda Library
Graves	Robert	The Golden Fleece	Greek Literature	Classics Library	Waggener Hall
Miksa	Francis	Charles Ammi Cutter	Library Biography	Library and Information Science Collection	Perry-Castañeda Library
Hunter	David	Music Publishing and Collecting	Music Literature	Fine Arts Library	Fine Arts Building
Graves	Robert	English and Scottish Ballads	Folksong	PCL General Stacks	Perry-Castañeda Library

By examining the table, we can infer that books dealing with history, cognitive psychology, and folksong are assigned to the PCL General Stacks collection; that books dealing with legal procedures are assigned to the Law Library; that books dealing with Greek literature are assigned to the Classics Library; that books dealing with library biography are assigned to the Library and Information Science Collection (LISC);and that books dealing with music literature are assigned to the Fine Arts Library. Moreover, we can infer that the PCL General Stacks collection and the LISC are both housed in the Perry-Castañeda Library (PCL) building; that the Classics Library is housed in Waggener Hall; and that the Law Library and Fine Arts Library are housed, respectively, in Townes Hall and the Fine Arts Building.

It is seen that transitive dependency exists in the above table : any book that deals with history, cognitive psychology, or library biography will be physically housed in the PCL building (unless it is temporarily checked out to a borrower); any book dealing with legal procedures will be housed in Townes Hall; and so on. In short, if we know what subject a book deals with, we also know not only what library or collection it will be assigned to but also what building it is physically housed in.

A problem with transitive dependency is that, there is duplicated information: from three different rows we can see that the PCL General Stacks are in the PCL building. For another thing, we have possible deletion anomalies: if the Yudof book were lost

and its row removed from table, we would lose the information that books on legal procedures are assigned to the Law Library and also the information the Law Library is in Townes Hall. As a third problem, we have possible insertion anomalies: if we wanted to add a chemistry book to the table, we would find that the above table nowhere contains the fact that the Chemistry Library is in Robert A.Welch Hall. As a fourth problem, we have the chance of making errors in updating: a careless data-entry clerk might add a book to the LISC but mistakenly enter Townes Hall in the building column.
To solve this problem decompose the above table into three different tables as follows

Table A

Author Last Name	Author First Name	Book Title
Berdahl	Robert	The Politics of the Prussian Nobility
Yudof	Mark	Child Abuse and Neglect
Harmon	Glynn	Human Memory and Knowledge
Graves	Robert	The Golden Fleece
Miksa	Francis	Charles Ammi Cutter
Hunter	David	Music Publishing and Collecting
Graves	Robert	English and Scottish Ballads

Table B

Book Title	Subject
The Politics of the Prussian Nobility	History
Child Abuse and Neglect	Legal Procedures
Human Memory and Knowledge	Cognitive Psychology
The Golden Fleece	Greek Literature
Charles Ammi Cutter	Library Biography
Music Publishing and Collecting	Music Literature
English and Scottish Ballads	Folksong

Table C

Subject	Collection or Library
History	PCL General Stacks
Legal Procedures	Law Library
Cognitive Psychology	PCL General Stacks
Greek Literature	Classics Library
Library Biography	Library and Information Science Collection
Music Literature	Fine Arts Library
Folksong	PCL General Stacks

Table D

Collection or Library	Building
PCL General Stacks	Perry-Castañeda Library

RESULT

Normalization of tables are done

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		

Exp No : 11 **Date :** D D / M M / Y Y

CREATE A DATABASE APPLICATION AND DO THE DATABASE CONNECTIVITY

AIM

To create a PHP- mysql program to retrieve data using mysql

Problem statement

A PHP program that connects to a database table and retrieve values from table

Objective

To understand the connectivity to a database table using php and how to retrieve values from that table

Requirements

Mysql database software, LAMP server and Html

Program

Mysql.html

```
<html>
<form action='pp.php' method='post'>
First Name:<input type='text' name='firstname'><br>
LastName:<input type='text' name='lastname'><p>
<input type='submit' name='submit' value='submit'>
</form>
</html>
```

mysql1.php

```
<?php
require("connect.php");
if($_POST['submit'])
{
$f1=$_POST['firstname'];
$l1=$_POST['lastname'];
$w=mysql_query("select * from employee where first_name='$f1' and
last_name='$l1'");
$num=mysql_num_rows($w);
echo $num;
```

```
while($row=mysql_fetch_assoc($w))
{
$firstn=$row['first_name'];
$lastn=$row['last_name'];
echo "firstname=$firstn and lastname=$lastn";
}}?>
```

Connect.php

```
<?php
$connect=mysql_connect("localhost","root","mysql") or die
mysql_select_db("myclass") or die("error2");
?>
```

RESULT

PHP-mysql program to retrieve data is done successfully

	Signature of Lab in Charge	Remarks
Readiness to do experiment		
Completion of Experiment		