Bachelor of Technology (B.Tech)

Department of Computer Science and Engineering

II year II sem- Database Management System Laboratory Manual

SIDDHARTHA INSTITUTE OF TECHNOLOGY & SCIENCES

(Approved by AICTE, New Delhi & Affiliated to JNTUH,
Hyderabad)Accredited by NBA and NAAC with 'A+' Grade
Narapally, Korremula Road, Ghatkesar, Medchal- Malkajgiri (Dist)-501 301

Vision of the Institute

To be a reputed institute in technical education towards research, industrial and societal needs.

Mission of the Institute

Mission	Statement
IM_1	Provide state-of-the-art infrastructure, review, innovative and experiment teaching –learning methodologies.
IM_2	Promote training, research and consultancy through an integrated institute industry symbiosis
IM_3	Involve in activities to groom professional, ethical values and social responsibility

SIDDHARTHA INSTITUTE OF TECHNOLOGY AND SCIENCES

(*Approved by AICTE, New Delhi &Affiliated to JNTUH, Hyderabad*)
Narapally, Telangana – 500 088.

Department of Computer Science and Engineering

Vision of the Department

To be a recognized center of Computer Science education with values, and quality research

Mission of the Department

Mission	Statement
DM_1	Impart high quality professional training with an emphasis on basic principles of Computer Science and allied Engineering
DM_2	Imbibe social awareness and responsibility to serve the society
DM_3	Provide academic facilities, organize collaborated activities to enable overall development of stakeholders

Department of Computer Science and Engineering

Program Educational Objectives (PEOs)

PEO's	Statement
PEO1	Graduates will be able to solve Computer Science and allied Engineering problems, develop proficiency in computational tools.
PEO2	Graduates will be able to communicate and work efficiently in Multidisciplinary teams with a sense of professional and social responsibility.
PEO3	Graduates will be able to exhibit lifelong learning ability andpursue career as architects, software developers and entrepreneurs.

Programme Outcomes

PO1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	**Design/development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	**Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	**Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
PO6	**The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	**Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental context, and demonstrate the knowledge of, and need for sustainable development.
PO8	**Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	**Individual and team network:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	**Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	**Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	**Life-Long learning:** Recognize the need for, and have the preparation and able to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcomes:

PSO1	**Program Applications:** Able to develop programs modules for cloud based applications.
PSO2	**Development Tools:** Able to use tools such as Weka, Rational Rose Raspberry-Pi, Sql and advanced tools

SIDDHARTHA INSTITUTE OF TECHNOLOGY & SCIENCES
(UGC - AUTONOMOUS)

22X0581: DATABASE MANAGEMENT SYSTEMS LAB
(Common to CSE, AIML, DS, SE, CS, IOT)

B.Tech. II Year II Sem / III Year I sem

L	T	P	C
0	0	2	1

Co-requisites: "Database Management Systems"

Course Objectives:

- Introduce ER data model
- Understand the concepts of database design and normalization
- Learn SQL basics for data definition and data manipulation
- Learn various Aggregate functions in DBMS
- Understand the concepts of Triggers

Course Outcomes:

- Understand concepts of E-R model
- Design database schema for a given application and apply normalization
- Acquire skills in using SQL commands for data definition and data manipulation.
- Able to understand Queries using Aggregate functions
- Develop solutions for database applications using procedures, cursors and triggers

List of Experiments:

1. Concept design with E-R Model (Identifying entities, attributes, keys and relationships between entities, cardinalities, generalization, specialization etc.)
   i) Library Management System ii) Banking System
2. Relational Model
   Represent entities and relationships in Tabular form, Represent attributes as columns, identifying keys for the above applications
3. Normalization- Create database and remove the redundancies and anomalies in the above relational tables, Normalize up to Third Normal Form
4. Practicing DDL commands
   i) Creating Tables (along with Primary and Foreign keys), Altering Tables and Dropping Tables
5. Practicing DML commands(Insert, Select, Update, Delete)
6. A. Querying (using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.)
   B. Nested, Correlated subqueries
7. Queries using Aggregate functions, GROUP BY, HAVING and Creation and dropping of Views.

8. Triggers (Creation of insert trigger, delete trigger, update trigger)
9. Procedures (Creation of Stored Procedures, Execution of Procedure, and Modification of Procedure)
10. Usage of Cursors (Declaring Cursor, Opening Cursor, Fetching the data, closing the cursor)

TEXT BOOKS:

1. Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata Mc Graw Hill,3rd Edition
2. Database System Concepts, Silberschatz, Korth, McGraw Hill, V edition.

REFERENCE BOOKS:

1. Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7thEdition.
2. Fundamentals of Database Systems, Elmasri Navrate, Pearson Education
3. Introduction to Database Systems, C.J. Date, Pearson Education
4. Oracle for Professionals, The X Team, S. Shah and V. Shah, SPD.
5. Database Systems Using Oracle: A Simplified guide to SQL and PL/SQL, Shah, PHI.
6. Fundamentals of Database Management Systems, M. L. Gillenson, Wiley Student Edition.

EXPERIMENT- 1

CONCEPT DESIGN WITH E-R MODEL

AIM: To Relate the entities appropriately. Apply cardinalities for each relationship. Identify strong and weak entities. Indicate the type of relationships (total/partial). Incorporate generalization, aggregation and specialization etc wherever required.

E-R Model

Bus

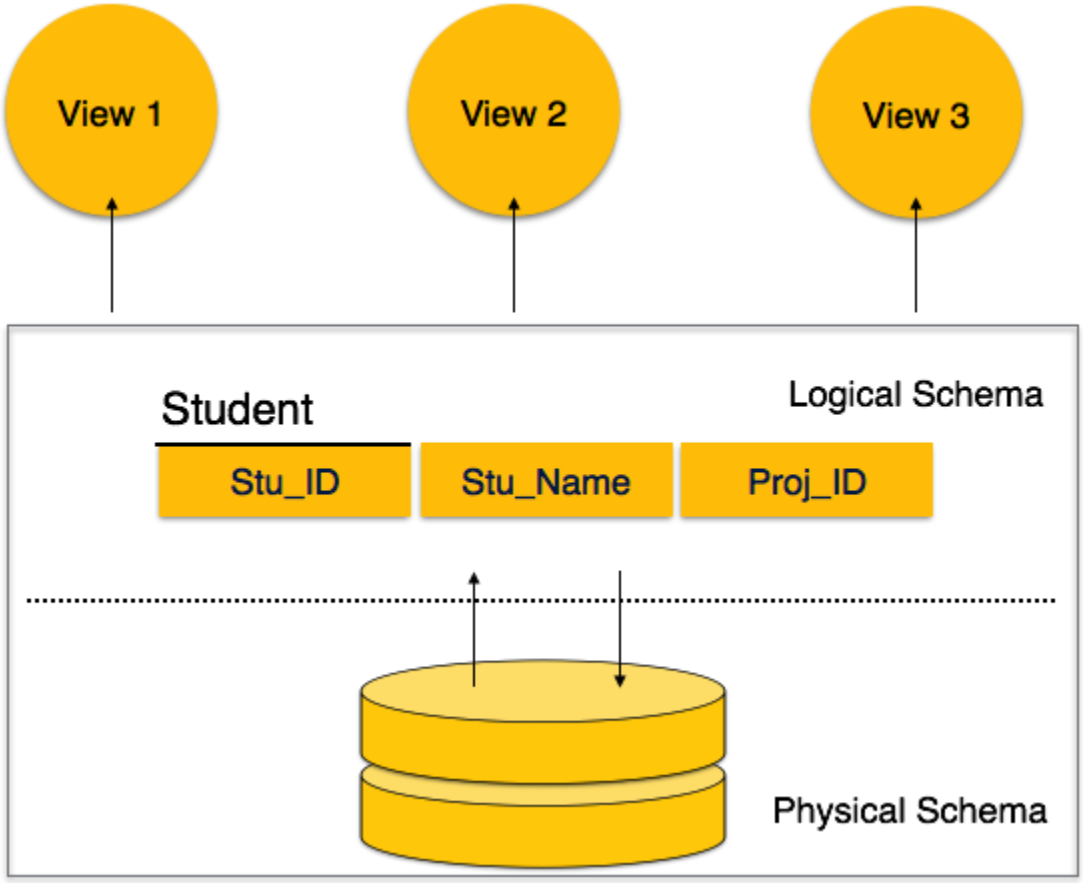
- BusNo
- Source
- Destination
- CoachType

SCHEMA

Database Schema

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

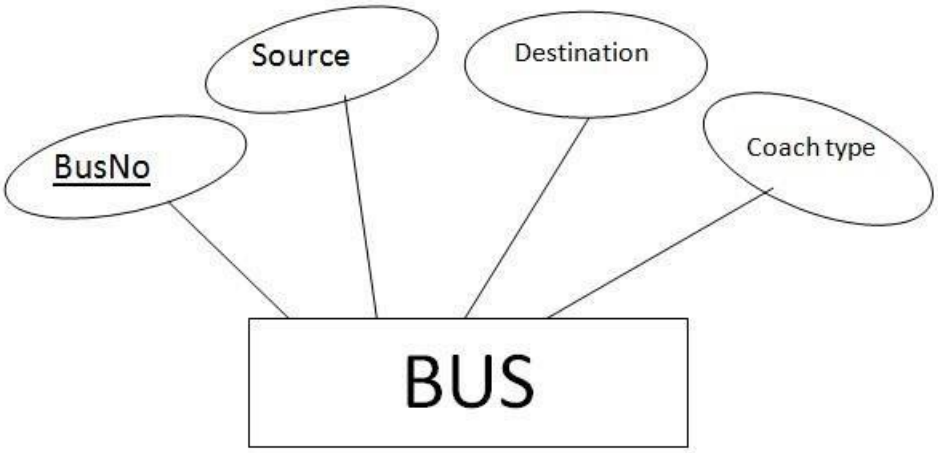
A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.



A database schema can be divided broadly into two categories −

- **Physical Database Schema** − This schema pertains to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- **Logical Database Schema** − This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Bus: Bus(BusNo :String ,Source : String, Destination: String, Coach Type: String)



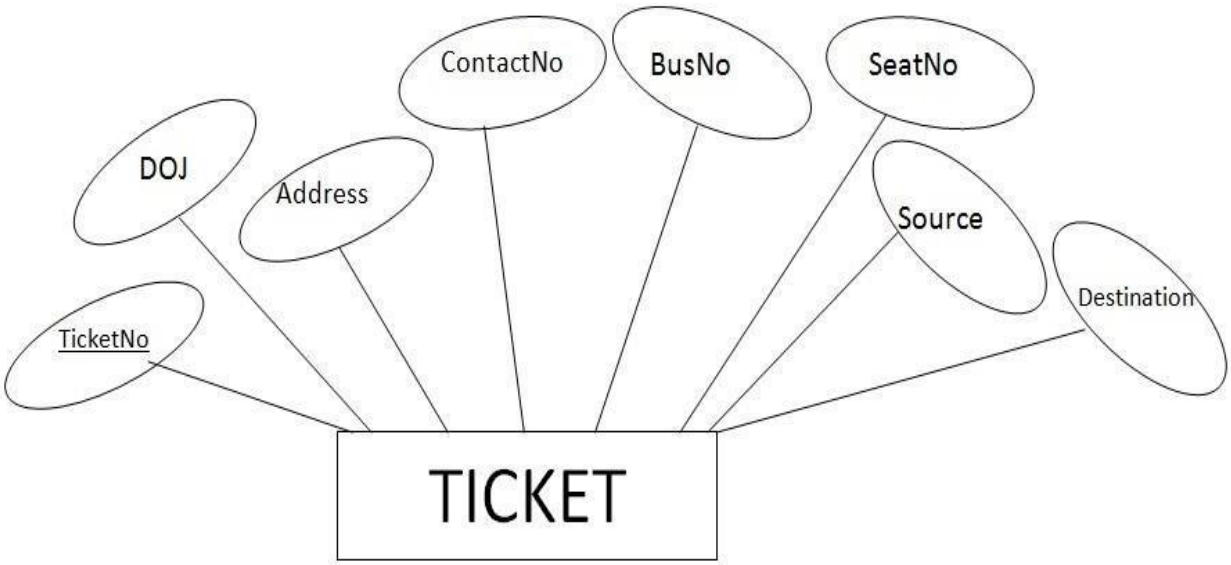
Ticket

- TicketNo
- DOJ
- Address
- ContactNo
- BusNo

- SeatNo
- Source
- Destination

SCHEMA

Ticket (TicketNo: string, DOJ: date, Address: string, ContactNo : string, BusNo:String SeatNo : Integer, Source: String, Destination: String)

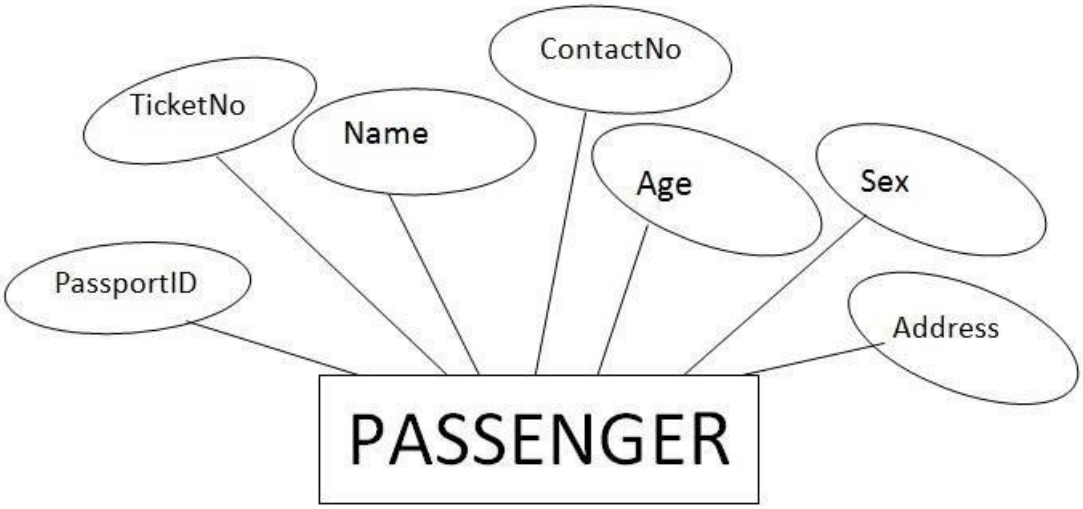


Passenger

- PassportID
- TicketNo
- Name
- ContactNo
- Age
- Sex
- Address

SCHEMA

Passenger (PassportID: String, TicketNo :string, Name: String, ContactNo: string, Age: integer, Sex: character, Address: String)

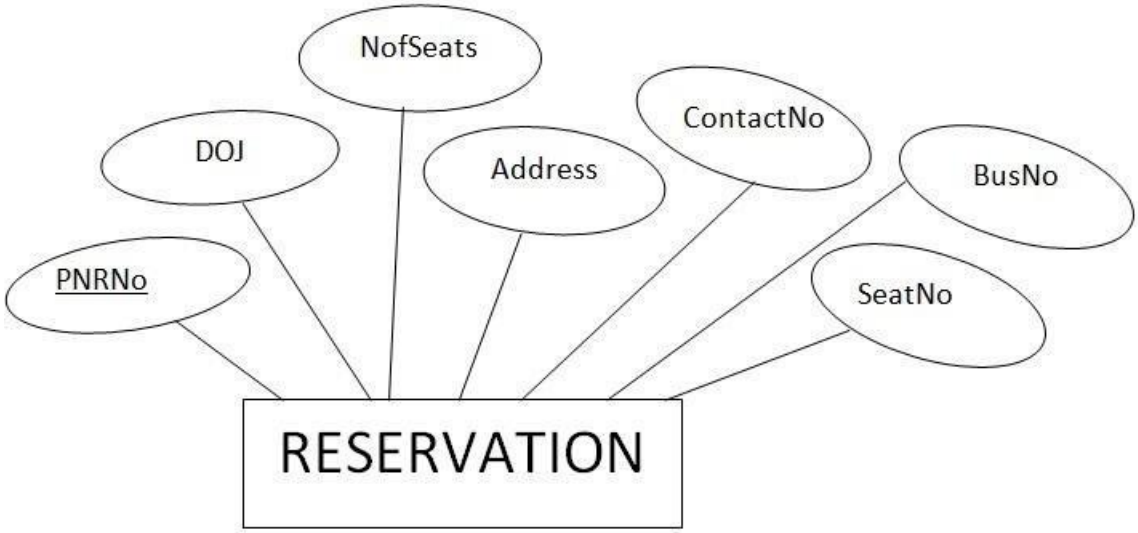


Reservation

- PNRNo
- DOJ
- No_of_seats
- Address
- ContactNo
- BusNo
- SeatNo

SCHEMA

Reservation(PNRNo: String, DOJ: Date, NoofSeats: integer , Address: String ,ContactNo: String, , BusNo: String,SeatNo:Integer)

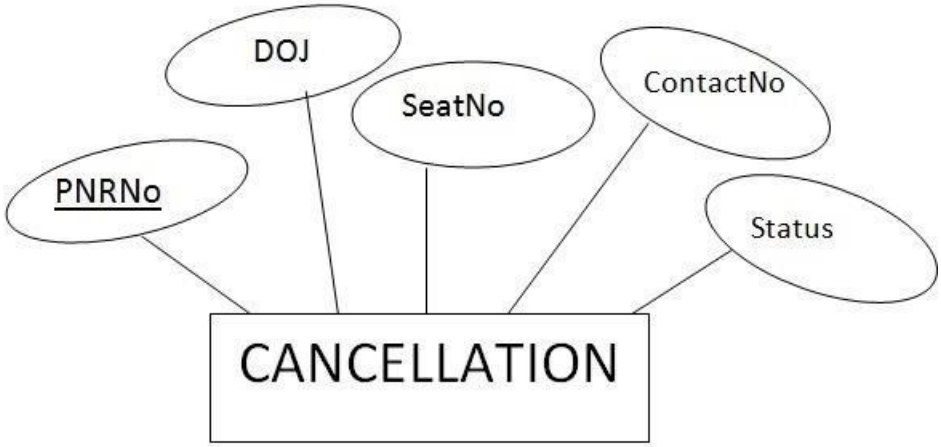


Cancellation

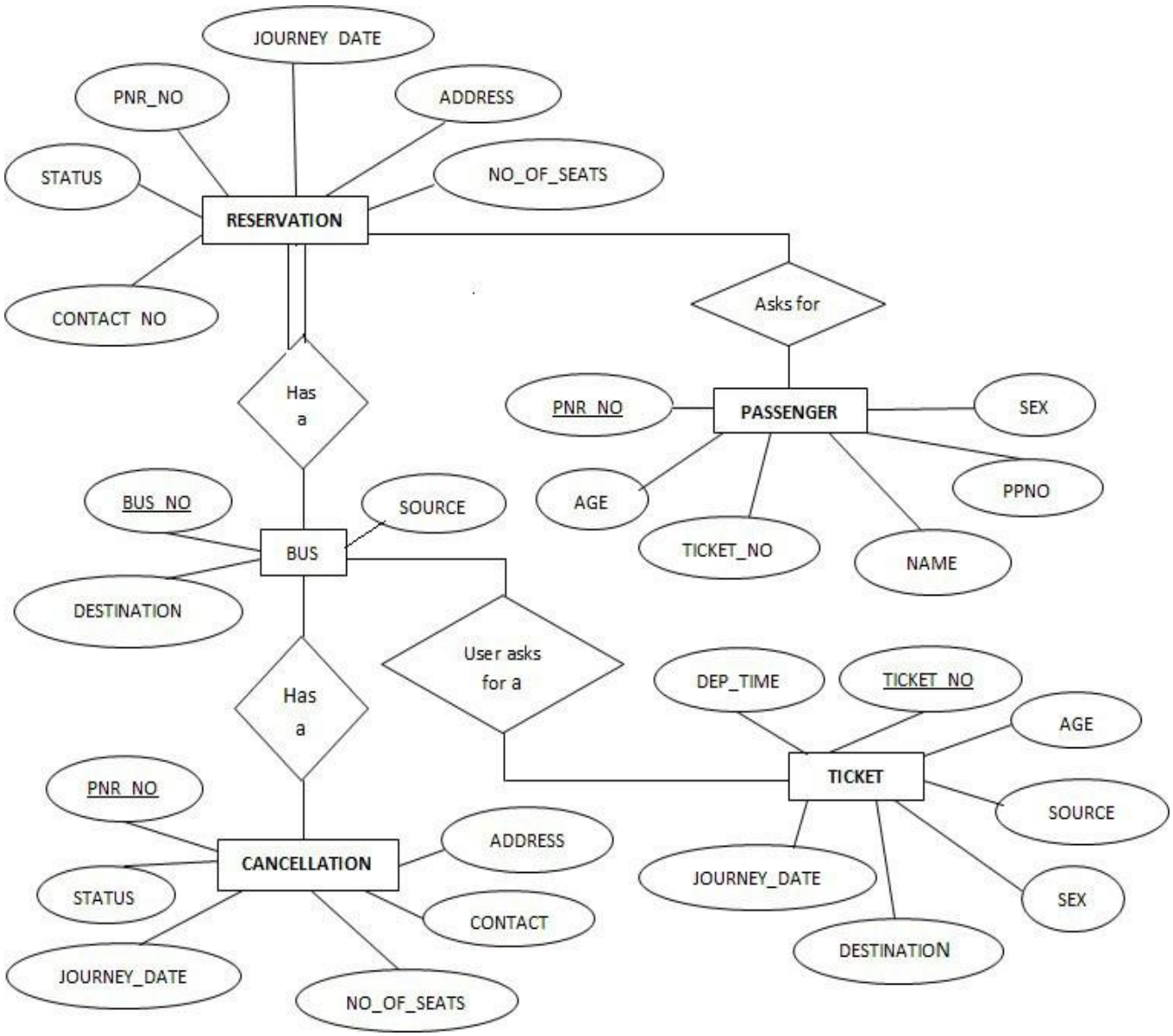
- PNRNo
- DOJ
- SeatNo
- ContactNo
- Status

SCHEMA

Cancellation (PNRNo: String, DOJ: Date, SeatNo: integer, ContactNo: String, Status: String)



CONCEPT DESIGN WITH E-R MODEL



EXPERIMENT – 2
RELATIONAL MODEL

AIM: To Represent all the entities (Strong, Weak) in tabular fashion. Represent relationships in a tabular fashion.

Entity in DBMS can be a real-world object with an existence, For example, in a **College** database, the entities can be Professor, Students, Courses, etc.

Types of DBMS Entities

The following are the types of entities in DBMS −

Strong Entity

The strong entity has a primary key. Weak entities are dependent on strong entity. Its existence is not dependent on any other entity.

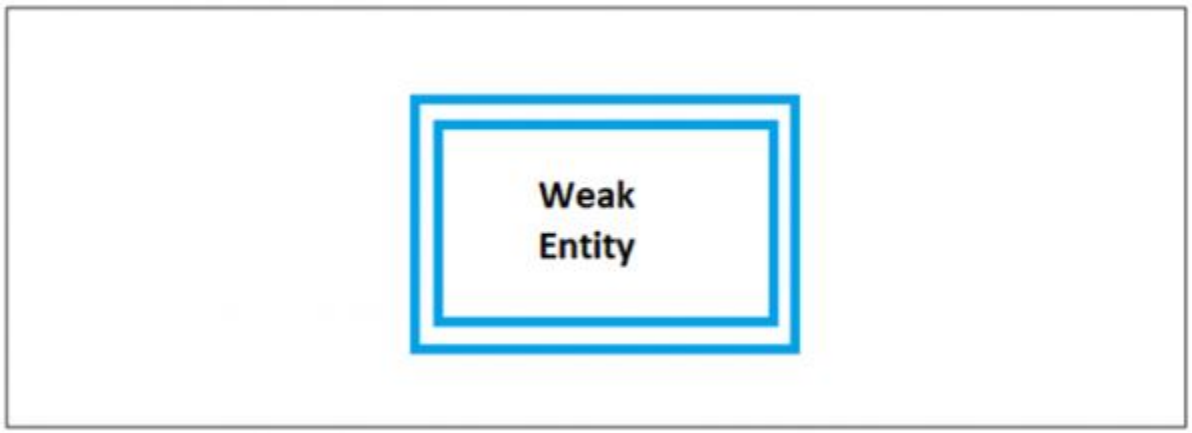
Strong Entity is represented by a single rectangle −

Continuing our previous example, **Professor** is a strong entity here, and the primary key is **Professor_ID.**

Weak Entity

The weak entity in DBMS do not have a primary key and are dependent on the parent entity. It mainly depends on other entities.

Weak Entity is represented by double rectangle −



Continuing our previous example, **Professor** is a strong entity, and the primary key is **Professor_ID**. However, another entity is **Professor_Dependents**, which is our Weak Entity.

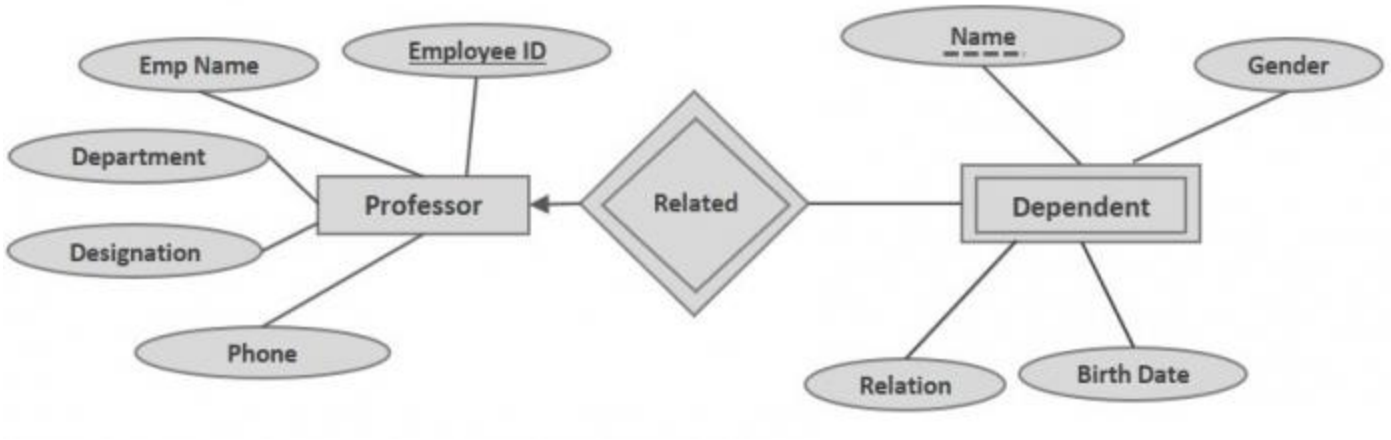
<Professor_Dependents>

Name	DOB	Relation

This is a weak entity since its existence is dependent on another entity **Professor**, which we saw above. A Professor has Dependents.

Example of Strong and Weak Entity

The example of a strong and weak entity can be understood by the below figure.



The Strong Entity is **Professor**, whereas **Dependent** is a Weak Entity.

ID is the primary key (represented with a line) and the Name in **Dependent** entity is called **Partial Key** (represented with a dotted line).

1. **Bus:** Bus(BusNo: String, Source: String, Destination: String, CoachType: String)

ColumnName	Datatype	Constraints	Type of Attributes
BusNo	**Varchar(10)**	**Primary key**	**Single-value**
Source	**Varchar(20)**		**Single-value**
Destination	**Varchar(20)**		**Simple**
CoachType	**Varchar(10)**		**Simple**

Mysql>create table Bus(BusNo varchar(10),source varchar(20),Destination varchar(20),coachType varchar(10),primary key(BusNo));

Mysql>desc Bus;

```
mysql> use cse;
Database changed
mysql> create table Bus(BusNo varchar(10),source varchar(20),Destination varchar(20),coachType varchar(10),primary key(BusNo));
Query OK, 0 rows affected (0.06 sec)

mysql> desc Bus;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| BusNo       | varchar(10) | NO   | PRI |         |       |
| source      | varchar(20) | YES  |     | NULL    |       |
| Destination | varchar(20) | YES  |     | NULL    |       |
| coachType   | varchar(10) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql>
```

Ticket:

Ticket(TicketNo: string, DOJ: date, Address:string,ContactNo: string, BusNo:String, SeatNo :Integer, Source: String, Destination: String)

ColumnName	Datatype	Constraints	Type of Attributes
TicketNo	Varchar(20)	Primary Key	Single-valued
DOJ	Date		Single-valued
Address	Varchar(20)		Composite
ContactNo	Integer		Multi-valued
BusNo	Varchar(10)	Foreign Key	Single-valued
SeatNo	Integer		Simple
Source	Varchar(10)		Simple
Destination	Varchar(10)		Simple

Mysql> create table ticket(ticketno varchar(20), doj date,address varchar(20),contactno int, busno varchar(20),seatno int,source varchar(10),destination varchar(10),primary key(ticketno,busno) foreign key(busno) references bus(busno);

Mysql>desc Ticket;

```
mysql> create table Ticket (TicketNo varchar(20),DOJ date,Address varchar(20),ContactNo varchar(15),BusNo varc
har(10),seatNo int,Source varchar(10),Destination varchar(10),primary key(TicketNo,BusNo),foreign key(BusNo)re
ferences Bus(BusNo));
Query OK, 0 rows affected (0.05 sec)

mysql> desc Ticket;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| TicketNo    | varchar(20) | NO   | PRI |         |       |
| DOJ         | date        | YES  |     | NULL    |       |
| Address     | varchar(20) | YES  |     | NULL    |       |
| ContactNo   | varchar(15) | YES  |     | NULL    |       |
| BusNo       | varchar(10) | NO   | PRI |         |       |
| seatNo      | int(11)     | YES  |     | NULL    |       |
| Source      | varchar(10) | YES  |     | NULL    |       |
| Destination | varchar(10) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

mysql>
```

Passenger:

Passenger(PassportID: String, TicketNo:string,Name: String, ContactNo:string,Age: integer, Sex: character, Address: String);

ColumnName	Datatype	Constraints	Type of Attributes
PassportID	Varchar(15)	Primary Key	Single-valued
TicketNo	Varchar(20)	Foreign Key	Single-valued

Name	**Varchar(20)**		**Composite**
ContactNo	**Varchar(20)**		**Multi-valued**
Age	**Integer**		**Single-valued**
Sex	**character**		**Simple**
Address	**Varchar(20)**		**Composite**

Mysql> Create table passenger(passportID varchar(15) ,TicketNo varchar(15),Name varchar(15),ContactNo varchar(20),Age integer, sex char(2),address varchar(20), primary key(passportID,TicketNo),foreign key(TicketNo) references Ticket(TicketNo));

Mysql> desc passenger;

```
mysql> use cse;
Database changed
mysql> create table passenger(passportid varchar(10),ticketno varchar(15),name varchar(15),contactno varchar(1
5),age integer,sex char(2),address varchar(20),primary key(passportid,ticketno),foreign key(ticketno) referenc
es ticket(ticketno));
Query OK, 0 rows affected (0.08 sec)

mysql> desc passenger;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportid | varchar(10) | NO   | PRI |         |       |
| ticketno   | varchar(15) | NO   | PRI |         |       |
| name       | varchar(15) | YES  |     | NULL    |       |
| contactno  | varchar(15) | YES  |     | NULL    |       |
| age        | int(11)     | YES  |     | NULL    |       |
| sex        | char(2)     | YES  |     | NULL    |       |
| address    | varchar(20) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
7 rows in set (0.03 sec)
```

Reservation:

Reservation(PNRNo: String, DOJ: Date, NoofSeats: integer , Address: String ,ContactNo: String, , BusNo: String,SeatNo:Integer)

ColumnName	Datatype	Constraints	Type of Attributes
PNRNo	**Varchar(20)**	**Primary Key**	**Single-valued**
DOJ	**date**		**Single-valued**
No_of_Seats	**Integer**		**Simple**
Address	**Varchar(20)**		**Composite**
ContactNo	**Varchar(10)**		**Multi-valued**

BusNo	**Varchar(10)**	**Foreign Key**	**Single-valued**
SeatNo	**Integer**		**Simple**

Mysql> Create table Resevation(PNRNo varchar(20),DOJ date,NoofSeates integer,Address varchar(20),ContactNo varchar(20),BusNo varchar(20),SeatNo integer, primary key(PNRNo,BusNo),foreign key(BusNo) references Bus(BusNo));

Mysql> desc reservation;

```
mysql> create table reservation(PNRNo varchar(20),DOJ date,NofSeats integer,Address varchar(20),ContactNo varc
har(20),BusNo varchar(20),SeatNo integer,primary key(PNRNo,BusNo),foreign key(BusNo) references Bus(BusNo));
Query OK, 0 rows affected (0.05 sec)

mysql> desc Reservation;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| PNRNo     | varchar(20) | NO   | PRI |         |       |
| DOJ       | date        | YES  |     | NULL    |       |
| NofSeats  | int(11)     | YES  |     | NULL    |       |
| Address   | varchar(20) | YES  |     | NULL    |       |
| ContactNo | varchar(20) | YES  |     | NULL    |       |
| BusNo     | varchar(20) | NO   | PRI |         |       |
| SeatNo    | int(11)     | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
7 rows in set (0.01 sec)
```

Cancellation:

Cancellation (PNRNo: String,DOJ: Date, SeatNo: integer,ContactNo: String,Status: String)

ColumnName	Datatype	Constraints	Type of Attributes
PNRNo	**Varchar(10)**	**Primary Key**	**Single-valued**
DOJ	**date**		**Single-valued**
SeatNo	**Integer**		**Simple**
ContactNo	**Varchar(15)**		**Multi-valued**
Status	**Varchar(10)**		**Simple**

Mysql> create table cancellation(PNRNo varchar(10),DOJ date,SeatNo integer, ContactNo varchar(15),Status varchar(10), primary key(PNRNo), foreign key(PNRNo) references reservation(PNRNo));

Mysql> desc cancellation;

```
mysql> create table cancellation(PNRNo varchar(10),DOJ date,SeatNo integer,ContactNo varchar(15),Status varcha
r(10),primary key(PNRNo),foreign key(PNRNo) references Reservation(PNRNo));
Query OK, 0 rows affected (0.05 sec)

mysql> desc cancellation;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| PNRNo     | varchar(10) | NO   | PRI |         |       |
| DOJ       | date        | YES  |     | NULL    |       |
| SeatNo    | int(11)     | YES  |     | NULL    |       |
| ContactNo | varchar(15) | YES  |     | NULL    |       |
| Status    | varchar(10) | YES  |     | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

EXPERIMENT – 3
NORMALIZATION

AIM: Apply the database Normalization techniques for designing relational database tables to minimize duplication of information like 1NF, 2NF, 3NF, BCNF.

Normalization is a process of converting a relation to be standard form by decomposition a larger relation into smaller efficient relation that depicts a good database design.

- 1NF: A Relation scheme is said to be in 1NF if the attribute values in the relation are atomic.i.e., Mutli –valued attributes are not permitted.

- 2NF: A Relation scheme is said to be in 2NF,iff and every Non-key attribute is fully functionally dependent on primary Key.

- 3NF: A Relation scheme is said to be in 3NF,iff and does not have transitivity dependencies. A Relation is said to be 3NF if every determinant is a key for each & every functional dependency.

- BCNF: A Relation scheme is said to be BCNF if the following statements are true for eacg FD P->Q in set F of FDs that holds for each FD. P->Q in set F of FD's that holds over R. Here P is the subset of attributes of R & Q is a single attribute of R.

The given FD is a trival

P is a super key.

Normalized tables are:-

Mysql> create table Bus2(BusNo varchar(20) primary key,Source varchar(20),Destination varchar(20));

Mysql>Create table passenger4(PPN varchar(15) Primary key,Name varchar(20),Age integer,Sex char,Address varchar(20));

Mysql> Create table PassengerTicket(PPN varchar(15) Primary key,TicketNo integer);

Mysql> Create table Reservation2(PNRNO integer Primary key, JourneyDate DateTime,NoofSeats int,Address varchar(20),ContactNo Integer);

Mysql> create table Cancellation2(PNRNO Integer primary key,JourneyDate DateTime,NoofSeats Integer,Address varchar(20),ContactNo Integer,foreign key(PNRNO) references Reservation2(PNRNO));

Mysql> Create table Ticket2(TicketNo Integer Primary key,JourneyDate DateTime, Age Int(4),Sex char(2),Source varchar(20),Destination varchar(20),DeptTime varchar(2));

EXPERIMENT – 4
PRACTICING DDL COMMANDS

AIM : Creating Tables and altering the Tables

Mysql>Create table passenger2(passportId Integer Primary Key,Name varchar(10) Not Null,Age Integer Not Null,Sex char,Address varchar(20) Not Null);

Mysql> desc passenger2;

```
mysql> create table passenger3(passportId integer primary key,name varchar(10) not null,Age Integer not null,
Sex char,Address varchar(20) not null);
Query OK, 0 rows affected (0.03 sec)

mysql> desc passenger3;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| name       | varchar(10) | NO   |     |         |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.02 sec)
```

USING ALTER COMMAND

Adding Extra column to Existing Table

Mysql>Alter table passenger3 add column TicketNo varchar(10);

```
mysql> Alter table passenger3  add column TicketNo  varchar(10);
Query OK, 0 rows affected (0.14 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc passenger3;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| name       | varchar(10) | NO   |     |         |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
| TicketNo   | varchar(10) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Mysql>Alter Table passenger3 add Foreign key(TicketNo) references Ticket(TicketNo);

```
C:\Program Files (x86)\MySQL\MySQL Server 5.0\bin\mysql.exe
mysql> alter table passenger3 add foreign key(TicketNo) references Ticket(TicketNo);
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc passenger3;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| name       | varchar(10) | NO   |     |         |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
| TicketNo   | varchar(10) | YES  | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)
```

Mysql>Alter Table passenger3 Modify column Name varchar(20);

```
C:\Program Files (x86)\MySQL\MySQL Server 5.0\bin\mysql.exe
mysql> Alter Table passenger3 Modify column Name varchar(20);
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc passenger3;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| Name       | varchar(20) | YES  |     | NULL    |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
| TicketNo   | varchar(10) | YES  | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Mysql>Alter table passenger drop foreign key fk1;

```
mysql> Alter table passenger2  add column TicketNo  varchar(10);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> alter table passenger2 add constraint fk1 foreign key(TicketNo) reference
s Ticket(TicketNo);
Query OK, 0 rows affected (0.07 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> Alter table passenger2 drop foreign key fk1;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc passenger2;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| name       | varchar(10) | NO   |     |         |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
| TicketNo   | varchar(10) | YES  | MUL | NULL    |       |
+------------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Mysql> Alter table passenger2 Drop column TicketNo;

```
mysql> Alter table passenger2 drop column ticketNo;
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc passenger2;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportId | int(11)     | NO   | PRI |         |       |
| name       | varchar(10) | NO   |     |         |       |
| Age        | int(11)     | NO   |     |         |       |
| Sex        | char(1)     | YES  |     | NULL    |       |
| Address    | varchar(20) | NO   |     |         |       |
+------------+-------------+------+-----+---------+-------+
5 rows in set (0.01 sec)
```

EXPERIMENT – 5
PRACTICING DML COMMANDS

AIM: Create a DML Commands are used to manage data within the scheme objects.

DML Commands:

INSERT COMMAND ON BUS2 & PASSENGER2 RELATIONS

mysql> select * from Bus2; Empty set (0.00 sec)

mysql> insert into Bus2 values(1234,'Hyderabad','Tirupathi');

Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(2345,'Hyderabad','Banglore');

Query OK, 1 row affected (0.01 sec)

mysql> insert into Bus2 values(23,'Hyderabad','Kolkata');

Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(45,'Tirupathi','Banglore');

Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(34,'Hyderabad','Chennai');

Query OK, 1 row affected (0.03 sec)

mysql> select * from Bus2;

```
mysql> select * from Bus2;
Empty set (0.00 sec)

mysql> insert into Bus2 values(1234,'Hyderabad','Tirupathi');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(2345,'Hyderabad','Banglore');
Query OK, 1 row affected (0.01 sec)

mysql> insert into Bus2 values(23,'Hyderabad','Kolkata');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(45,'Tirupathi','Banglore');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Bus2 values(34,'Hyderabad','Chennai');
Query OK, 1 row affected (0.03 sec)

mysql> select * from Bus2;
+-------+-----------+-------------+
| BusNo | Source    | Destination |
+-------+-----------+-------------+
| 1234  | Hyderabad | Tirupathi   |
| 23    | Hyderabad | Kolkata     |
| 2345  | Hyderabad | Banglore    |
| 34    | Hyderabad | Chennai     |
| 45    | Tirupathi | Banglore    |
+-------+-----------+-------------+
5 rows in set (0.01 sec)
```

mysql> select * from Passenger2;

Empty set (0.00 sec)

mysql> insert into Passenger2 values(145,'Ramesh',45,'M','abc123');

Query OK, 1 row affected (0.05 sec)

mysql> insert into Passenger2 values(278,'Geetha',36,'F','abc124');

Query OK, 1 row affected (0.02 sec)

mysql> insert into Passenger2 values(4590,'Ram',30,'M','abc12');

Query OK, 1 row affected (0.03 sec)

mysql> insert into Passenger2 values(6789,'Ravi',50,'M','abc14');

Query OK, 1 row affected (0.03 sec)

mysql> insert into Passenger2 values(5622,'Seetha',32,'F','abc55');

Query OK, 1 row affected (0.03 sec)

mysql> select * from Passenger2;

```
mysql> select * from Passenger2;
Empty set (0.00 sec)

mysql> insert into Passenger2 values(145,'Ramesh',45,'M','abc123');
Query OK, 1 row affected (0.05 sec)

mysql> insert into Passenger2 values(278,'Geetha',36,'F','abc124');
Query OK, 1 row affected (0.02 sec)

mysql> insert into Passenger2 values(4590,'Ram',30,'M','abc12');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Passenger2 values(6789,'Ravi',50,'M','abc14');
Query OK, 1 row affected (0.03 sec)

mysql> insert into Passenger2 values(5622,'Seetha',32,'F','abc55');
Query OK, 1 row affected (0.03 sec)

mysql> select * from Passenger2;
+------------+--------+-----+------+---------+
| passportId | name   | Age | Sex  | Address |
+------------+--------+-----+------+---------+
|        145 | Ramesh |  45 | M    | abc123  |
|        278 | Geetha |  36 | F    | abc124  |
|       4590 | Ram    |  30 | M    | abc12   |
|       5622 | Seetha |  32 | F    | abc55   |
|       6789 | Ravi   |  50 | M    | abc14   |
+------------+--------+-----+------+---------+
5 rows in set (0.00 sec)
```

UPDATE COMMAND ON BUS2 RELATION

UPDATE Selected Rows & Multiple Rows

mysql> Update Bus2 SET Source='Secundrabad' where BusNo=1234; Query OK, 1 row affected (0.05 sec)

Rows matched: 1 Changed: 1 Warnings: 0

```
C:\Program Files (x86)\MySQL\MySQL Server 5.0\bin\mysql.exe
mysql> select * from Bus2;
+-------+-----------+-------------+
| BusNo | Source    | Destination |
+-------+-----------+-------------+
| 1234  | Hyderabad | Tirupathi   |
| 23    | Hyderabad | Kolkata     |
| 2345  | Hyderabad | Banglore    |
| 34    | Hyderabad | Chennai     |
| 45    | Tirupathi | Banglore    |
+-------+-----------+-------------+
5 rows in set (0.00 sec)

mysql> Update Bus2 SET Source='Secundrabad' where BusNo=1234;
Query OK, 1 row affected (0.05 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Bus2;
+-------+-------------+-------------+
| BusNo | Source      | Destination |
+-------+-------------+-------------+
| 1234  | Secundrabad | Tirupathi   |
| 23    | Hyderabad   | Kolkata     |
| 2345  | Hyderabad   | Banglore    |
| 34    | Hyderabad   | Chennai     |
| 45    | Tirupathi   | Banglore    |
+-------+-------------+-------------+
5 rows in set (0.00 sec)
```

DELETE COMMAND ON BUS2 RELATION

DELETES Selected Rows and Multiple Rows

mysql> Delete from Bus2 where BusNo=1234; Query OK, 1 row affected (0.05 sec)

mysql> select * from Bus2;

```
mysql> select * from Bus2;
+-------+-------------+-------------+
| BusNo | Source      | Destination |
+-------+-------------+-------------+
| 1234  | Secundrabad | Tirupathi   |
| 23    | Secundrabad | Kolkata     |
| 2345  | Secundrabad | Banglore    |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Banglore    |
+-------+-------------+-------------+
5 rows in set (0.00 sec)

mysql> Delete from Bus2 where BusNo=1234;
Query OK, 1 row affected (0.05 sec)

mysql> select * from Bus2;
+-------+-------------+-------------+
| BusNo | Source      | Destination |
+-------+-------------+-------------+
| 23    | Secundrabad | Kolkata     |
| 2345  | Secundrabad | Banglore    |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Banglore    |
+-------+-------------+-------------+
4 rows in set (0.00 sec)
```

mysql> Delete from Bus2 where Source=’Secundrabad’; Query OK, 1 row affected (0.05 sec)

mysql> select * from Bus2;

```
mysql> select * from Bus2;
+-------+-------------+-------------+
| BusNo | Source      | Destination |
+-------+-------------+-------------+
| 23    | Secundrabad | Kolkata     |
| 2345  | Secundrabad | Banglore    |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Banglore    |
+-------+-------------+-------------+
4 rows in set (0.00 sec)

mysql> Delete from Bus2 where Source='Secundrabad';
Query OK, 3 rows affected (0.03 sec)

mysql> select * from Bus2;
+-------+-----------+-------------+
| BusNo | Source    | Destination |
+-------+-----------+-------------+
| 45    | Tirupathi | Banglore    |
+-------+-----------+-------------+
1 row in set (0.00 sec)
```

EXPERIMENT – 6

Querying (using ANY, ALL, IN, Exists, NOT EXISTS, UNION, INTERSECT, Constraints etc.)

Aim: Practice the following Queries:

1. Display unique PNR_NO of all passengers
2. Display all the names of male passengers.
3. Display the ticket numbers and names of all the passengers.
4. Find the ticket numbers of the passengers whose name start with 'r' and ends with 'h'.
5. Find the names of Passengers whose age is between 30 and 45.
6. Display all the passengers names beginning with 'A'.
7. Display the sorted list of Passengers names

```
mysql> DESC RESERVATION2;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| PNRNO       | int(11)     | NO   | PRI |         |       |
| Journeydate | datetime    | YES  |     | NULL    |       |
| NoofSeats   | int(11)     | YES  |     | NULL    |       |
| Address     | varchar(20) | YES  |     | NULL    |       |
| CONTACTNO   | varchar(15) | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

mysql> insert into reservation2 values(10201,'2012-02-20 10:20:25',05,'HYD',9654
235242);
Query OK, 1 row affected (0.03 sec)

mysql> insert into reservation2 values(10202,'2012-02-22 10:22:25',05,'HYD',9654
232451);
Query OK, 1 row affected (0.02 sec)

mysql> insert into reservation2 values(10203,'2012-03-22 10:30:25',05,'DELHI',96
54587960);
Query OK, 1 row affected (0.01 sec)

mysql> insert into reservation2 values(10204,'2013-03-22 11:30:25',05,'CHENNAI',
9845761254);
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM RESERVATION2;
+-------+---------------------+-----------+---------+------------+
| PNRNO | Journeydate         | NoofSeats | Address | CONTACTNO  |
+-------+---------------------+-----------+---------+------------+
| 10201 | 2012-02-20 10:20:25 |         5 | HYD     | 9654235242 |
| 10202 | 2012-02-22 10:22:25 |         5 | HYD     | 9654232451 |
| 10203 | 2012-03-22 10:30:25 |         5 | DELHI   | 9654587960 |
| 10204 | 2013-03-22 11:30:25 |         5 | CHENNAI | 9845761254 |
+-------+---------------------+-----------+---------+------------+
4 rows in set (0.01 sec)
```

```
mysql> insert into passenger2 values(82302,'Smith',23,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82303,'Neha',23,'F','Hyderabad');
Query OK, 1 row affected (0.01 sec)

mysql> insert into passenger2 values(82304,'Neha',35,'F','Hyderabad');
Query OK, 1 row affected (0.03 sec)

mysql> insert into passenger2 values(82306,'Ramu',40,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82308,'Aakash',40,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82402,'Aravind',42,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82403,'Avinash',42,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82502,'Ramesh',23,'M','Hyderabad');
Query OK, 1 row affected (0.02 sec)

mysql> insert into passenger2 values(82602,'Rajesh',23,'M','Hyderabad');
 Query OK, 1 row affected (0.02 sec)
```

RESERVATION2

mysql> insert into reservation2 values(10201,'2012-02-20 10:20:25',05,'HYD',9654 235242);

Query OK, 1 row affected (0.03 sec)

mysql> insert into reservation2 values(10202,'2012-02-22 10:22:25',05,'HYD',9654 232451);

Query OK, 1 row affected (0.02 sec)

mysql> insert into reservation2 values(10203,'2012-03-22 10:30:25',05,'DELHI',96 54587960);

Query OK, 1 row affected (0.01 sec)

mysql> insert into reservation2 values(10204,'2013-03-22 11:30:25',05,'CHENNAI', 9845761254);

Query OK, 1 row affected (0.02 sec)

1. Display unique PNR_NO of all reservation Mysql>Select

   DISTINCT PNR_NO from Reservation;

PNR_No
10201
10202
10203
10204

```
mysql> SELECT DISTINCT PNRNO FROM RESERVATION2;
+-------+
| PNRNO |
+-------+
| 10201 |
| 10202 |
| 10203 |
| 10204 |
+-------+
4 rows in set (0.02 sec)
```

2. Display all the names of male passengers.

```
mysql> Select p.name from passenger2 p
       where    p.passportid IN (select p2.passportid from passenger2 p2
        where     p2.sex='M');
```

```
C:\Program Files (x86)\MySQL\MySQL Server 5.0\bin\mysql.exe
mysql> SELECT P.NAME FROM PASSENGER2 P
    -> WHERE P.PASSPORTID IN (SELECT P2.PASSPORTID FROM PASSENGER2 P2
    -> WHERE P2.SEX='M');
+---------+
| NAME    |
+---------+
| Ramesh  |
| Ram     |
| Ravi    |
| Smith   |
| Ramu    |
| Aakash  |
| Aravind |
| Avinash |
| Ramesh  |
| Rajesh  |
+---------+
10 rows in set (0.00 sec)
```

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
|       4590 | Ram     |  30 | M    | abc12     |
|       5622 | Seetha  |  32 | F    | abc55     |
|       6789 | Ravi    |  50 | M    | abc14     |
|      82302 | Smith   |  23 | M    | Hyderabad |
|      82303 | Neha    |  23 | F    | Hyderabad |
|      82304 | Neha    |  35 | F    | Hyderabad |
|      82306 | Ramu    |  40 | M    | Hyderabad |
|      82308 | Aakash  |  40 | M    | Hyderabad |
|      82402 | Aravind |  42 | M    | Hyderabad |
|      82403 | Avinash |  42 | M    | Hyderabad |
|      82502 | Ramesh  |  23 | M    | Hyderabad |
|      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT P.NAME FROM PASSENGER2 P
    -> WHERE P.PASSPORTID IN (SELECT P2.PASSPORTID
    -> FROM PASSENGER2 P2
    -> WHERE P2.SEX='M');
+---------+
| NAME    |
+---------+
| Ramesh  |
| Ram     |
| Ravi    |
| Smith   |
| Ramu    |
| Aakash  |
| Aravind |
| Avinash |
| Ramesh  |
| Rajesh  |
+---------+
10 rows in set (0.00 sec)
```

3. Display the ticket numbers and names of all the passengers.

```
mysql> desc passengerticket;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportid | varchar(15) | NO   | PRI |         |       |
| TicketNo   | int(11)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> insert into passengerticket values(145,100);
Query OK, 1 row affected (0.03 sec)

mysql> insert into passengerticket values(278,200);
Query OK, 1 row affected (0.03 sec)

mysql> insert into passengerticket values(6789,300);
Query OK, 1 row affected (0.03 sec)

mysql> insert into passengerticket values(82302,400);
Query OK, 1 row affected (0.03 sec)

mysql> insert into passengerticket values(82403,500);
Query OK, 1 row affected (0.03 sec)

mysql> insert into passengerticket values(82502,600);
Query OK, 1 row affected (0.02 sec)
```

mysql> select t.ticketno,p.name from passengerticket t,passenger2 p where t.passportid = p.passportid;

```
mysql> SELECT T.TICKETNO,P.NAME FROM PASSENGERTICKET T,PASSENGER2 P
    -> WHERE T.PASSPORTID=P.PASSPORTID;
+----------+---------+
| TICKETNO | NAME    |
+----------+---------+
|      100 | Ramesh  |
|      200 | Geetha  |
|      300 | Ravi    |
|      400 | Smith   |
|      500 | Avinash |
|      600 | Ramesh  |
+----------+---------+
6 rows in set (0.00 sec)
```

4. Find the ticket numbers of the passengers whose name start with ‘r’ and ends with ‘h’.

MySQL> SELECT Name FROM Passenger WHERE name LIKE ‘R%H’

Name
Rajesh
Ramesh
Ramesh

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
|       4590 | Ram     |  30 | M    | abc12     |
|       5622 | Seetha  |  32 | F    | abc55     |
|       6789 | Ravi    |  50 | M    | abc14     |
|      82302 | Smith   |  23 | M    | Hyderabad |
|      82303 | Neha    |  23 | F    | Hyderabad |
|      82304 | Neha    |  35 | F    | Hyderabad |
|      82306 | Ramu    |  40 | M    | Hyderabad |
|      82308 | Aakash  |  40 | M    | Hyderabad |
|      82402 | Aravind |  42 | M    | Hyderabad |
|      82403 | Avinash |  42 | M    | Hyderabad |
|      82502 | Ramesh  |  23 | M    | Hyderabad |
|      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT NAME FROM PASSENGER2 WHERE NAME LIKE 'R%H';
+--------+
| NAME   |
+--------+
| Ramesh |
| Ramesh |
| Rajesh |
+--------+
3 rows in set (0.00 sec)
```

5. Find the names of Passengers whose age is between 30 and 45.

MySQL> SELECT Name FROM PASSENGER WHERE AGE BETWEEN 30 AND 45

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
|       4590 | Ram     |  30 | M    | abc12     |
|       5622 | Seetha  |  32 | F    | abc55     |
|       6789 | Ravi    |  50 | M    | abc14     |
|      82302 | Smith   |  23 | M    | Hyderabad |
|      82303 | Neha    |  23 | F    | Hyderabad |
|      82304 | Neha    |  35 | F    | Hyderabad |
|      82306 | Ramu    |  40 | M    | Hyderabad |
|      82308 | Aakash  |  40 | M    | Hyderabad |
|      82402 | Aravind |  42 | M    | Hyderabad |
|      82403 | Avinash |  42 | M    | Hyderabad |
|      82502 | Ramesh  |  23 | M    | Hyderabad |
|      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT Name FROM PASSENGER2 WHERE AGE BETWEEN 30 AND 45;
+---------+
| Name    |
+---------+
| Ramesh  |
| Geetha  |
| Ram     |
| Seetha  |
| Neha    |
| Ramu    |
| Aakash  |
| Aravind |
| Avinash |
+---------+
9 rows in set (0.00 sec)
```

6. Display all the passengers names beginning with ‘A’.

MySQL> SELECT * FROM PASSENGER WHERE NAME LIKE ‘A%’;

Name
Akash
Arivind
Avinash

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
|       4590 | Ram     |  30 | M    | abc12     |
|       5622 | Seetha  |  32 | F    | abc55     |
|       6789 | Ravi    |  50 | M    | abc14     |
|      82302 | Smith   |  23 | M    | Hyderabad |
|      82303 | Neha    |  23 | F    | Hyderabad |
|      82304 | Neha    |  35 | F    | Hyderabad |
|      82306 | Ramu    |  40 | M    | Hyderabad |
|      82308 | Aakash  |  40 | M    | Hyderabad |
|      82402 | Aravind |  42 | M    | Hyderabad |
|      82403 | Avinash |  42 | M    | Hyderabad |
|      82502 | Ramesh  |  23 | M    | Hyderabad |
|      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT NAME FROM PASSENGER2 WHERE NAME LIKE 'A%';
+---------+
| NAME    |
+---------+
| Aakash  |
| Aravind |
| Avinash |
+---------+
3 rows in set (0.00 sec)
```

7. Display the sorted list of Passengers names

MySQL> SELECT NAME FROM PASSENGER ORDER BY NAME;

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
|       4590 | Ram     |  30 | M    | abc12     |
|       5622 | Seetha  |  32 | F    | abc55     |
|       6789 | Ravi    |  50 | M    | abc14     |
|      82302 | Smith   |  23 | M    | Hyderabad |
|      82303 | Neha    |  23 | F    | Hyderabad |
|      82304 | Neha    |  35 | F    | Hyderabad |
|      82306 | Ramu    |  40 | M    | Hyderabad |
|      82308 | Aakash  |  40 | M    | Hyderabad |
|      82402 | Aravind |  42 | M    | Hyderabad |
|      82403 | Avinash |  42 | M    | Hyderabad |
|      82502 | Ramesh  |  23 | M    | Hyderabad |
|      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT NAME FROM PASSENGER2 ORDER BY NAME;
+---------+
| NAME    |
+---------+
| Aakash  |
| Aravind |
| Avinash |
| Geetha  |
| Neha    |
| Neha    |
| Rajesh  |
| Ram     |
| Ramesh  |
| Ramesh  |
| Ramu    |
| Ravi    |
| Seetha  |
| Smith   |
+---------+
14 rows in set (0.02 sec)
```

EXPERIMENT – 7

Querying Aggregate Functions(COUNT,SUM,AVG,MAX and MIN)

Aim: To Practice Queries using Aggregate functions for the following

1. Write a Query to display the information present in the passenger and cancellation tables
2. Display the number of days in a week on which the AP123 bus is available
3. Find number of tickets booked for each PNR_No using GROUP BY CLAUSE
4. Find the distinct PNR Numbers that are present.

1. Write a Query to display the information present in the passenger and cancellation tables

```
MYSQL> CREATE TABLE CANCELLATION2(PNRNO INT PRIMARY KEY,JOURNEYDATE DATETIME,
NOOFSEATS INT,ADDRESS VARCHAR(20),CONTACTNO INT,STATUS VARCHAR(10),FOREIGN
KEY(PNRNO) REFERENCES RESERVATION2(PNRNO));
```

```
mysql> INSERT INTO CANCELLATION2 VALUES(10201,'2012-02-20
10:20:25',2,'HYD',9654235242,'CONFIRM');
```

```
mysql> INSERT INTO CANCELLATION2 VALUES(10202,'2012-02-22
10:22:25',2,'HYD',9654232451,'CONFIRM');
```

```
mysql> INSERT INTO CANCELLATION2 VALUES(10203,'2012-03-22
10:30:25',2,'DELHI',9654587960,'CONFIRM');
```

MySQL> SELECT * FROM RESERVATION UNION

SELECT * FROM CANCELLATION;

```
mysql> SELECT * FROM RESERVATION2
    -> UNION
    -> SELECT * FROM CANCELLATION2;
+-------+---------------------+-----------+---------+------------+---------+
| PNRNO | Journeydate         | NoofSeats | Address | CONTACTNO  | STATUS  |
+-------+---------------------+-----------+---------+------------+---------+
| 10201 | 2012-02-20 10:20:25 |         5 | HYD     | 9654235242 | NULL    |
| 10202 | 2012-02-22 10:22:25 |         5 | HYD     | 9654232451 | NULL    |
| 10203 | 2012-03-22 10:30:25 |         5 | DELHI   | 9654587960 | NULL    |
| 10204 | 2013-03-22 11:30:25 |         5 | CHENNAI | 9845761254 | NULL    |
| 10201 | 2012-02-20 10:20:25 |         2 | HYD     | 9654235242 | CONFIRM |
| 10202 | 2012-02-22 10:22:25 |         2 | HYD     | 9654232451 | CONFIRM |
| 10203 | 2012-03-22 10:30:25 |         2 | DELHI   | 9654587960 | CONFIRM |
+-------+---------------------+-----------+---------+------------+---------+
7 rows in set (0.01 sec)
```

2. Display the Minimum age of the Passenger

MySQL> SELECT MIN(AGE) as MINAGE FROM PASSENGER;

```
mysql> SELECT * FROM PASSENGER2;
+------------+---------+-----+------+-----------+
| passportId | name    | Age | Sex  | Address   |
+------------+---------+-----+------+-----------+
|        145 | Ramesh  |  45 | M    | abc123    |
|        278 | Geetha  |  36 | F    | abc124    |
       4590 | Ram     |  30 | M    | abc12     |
       5622 | Seetha  |  32 | F    | abc55     |
       6789 | Ravi    |  50 | M    | abc14     |
      82302 | Smith   |  23 | M    | Hyderabad |
      82303 | Neha    |  23 | F    | Hyderabad |
      82304 | Neha    |  35 | F    | Hyderabad |
      82306 | Ramu    |  40 | M    | Hyderabad |
      82308 | Aakash  |  40 | M    | Hyderabad |
      82402 | Aravind |  42 | M    | Hyderabad |
      82403 | Avinash |  42 | M    | Hyderabad |
      82502 | Ramesh  |  23 | M    | Hyderabad |
      82602 | Rajesh  |  23 | M    | Hyderabad |
+------------+---------+-----+------+-----------+
14 rows in set (0.00 sec)

mysql> SELECT MIN(AGE) as MINAGE FROM PASSENGER2;
+--------+
| MINAGE |
+--------+
|     23 |
+--------+
1 row in set (0.03 sec)
```

3. Find number of tickets booked for each PNR_No using GROUP BY CLAUSE

MySQL> SELECT PNRNO,SUM(No_of_SEATS) AS SUM_OF_SEATS FROM RESERVATION2 GROUP BY PNRNO;

```
mysql> SELECT * FROM RESERVATION2;
+-------+---------------------+-----------+---------+------------+--------+
| PNRNO | Journeydate         | NoofSeats | Address | CONTACTNO  | STATUS |
+-------+---------------------+-----------+---------+------------+--------+
| 10201 | 2012-02-20 10:20:25 |         5 | HYD     | 9654235242 | NULL   |
| 10202 | 2012-02-22 10:22:25 |         5 | HYD     | 9654232451 | NULL   |
| 10203 | 2012-03-22 10:30:25 |         5 | DELHI   | 9654587960 | NULL   |
| 10204 | 2013-03-22 11:30:25 |         5 | CHENNAI | 9845761254 | NULL   |
+-------+---------------------+-----------+---------+------------+--------+
4 rows in set (0.00 sec)

mysql> SELECT PNRNO,SUM(NOOFSEATS) AS SUM_OF_SEATS FROM RESERVATION2   GROUP BY
PNRNO;
+-------+--------------+
| PNRNO | SUM_OF_SEATS |
+-------+--------------+
| 10201 |            5 |
| 10202 |            5 |
| 10203 |            5 |
| 10204 |            5 |
+-------+--------------+
4 rows in set (0.00 sec)
```

4 Find the distinct PNR Numbers that are present.

MySQL> SELECT DISTINCT PNR_NO FROM RESERVATION2;

```
mysql> SELECT * FROM RESERVATION2;
+-------+---------------------+-----------+---------+------------+--------+
| PNRNO | Journeydate         | NoofSeats | Address | CONTACTNO  | STATUS |
+-------+---------------------+-----------+---------+------------+--------+
| 10201 | 2012-02-20 10:20:25 |         5 | HYD     | 9654235242 | NULL   |
| 10202 | 2012-02-22 10:22:25 |         5 | HYD     | 9654232451 | NULL   |
| 10203 | 2012-03-22 10:30:25 |         5 | DELHI   | 9654587960 | NULL   |
| 10204 | 2013-03-22 11:30:25 |         5 | CHENNAI | 9845761254 | NULL   |
+-------+---------------------+-----------+---------+------------+--------+
4 rows in set (0.00 sec)

mysql> SELECT DISTINCT PNRNO FROM RESERVATION2;
+-------+
| PNRNO |
+-------+
| 10201 |
| 10202 |
| 10203 |
| 10204 |
+-------+
4 rows in set (0.00 sec)
```

5 Mysql> select sum(Noofseats) from Cancellation2;

```
mysql> SELECT * FROM CANCELLATION2;
+-------+---------------------+-----------+---------+------------+---------+
| PNRNO | JOURNEYDATE         | NOOFSEATS | ADDRESS | CONTACTNO  | STATUS  |
+-------+---------------------+-----------+---------+------------+---------+
| 10201 | 2012-02-20 10:20:25 |         2 | HYD     | 9654235242 | CONFIRM |
| 10202 | 2012-02-22 10:22:25 |         2 | HYD     | 9654232451 | CONFIRM |
| 10203 | 2012-03-22 10:30:25 |         2 | DELHI   | 9654587960 | CONFIRM |
+-------+---------------------+-----------+---------+------------+---------+
3 rows in set (0.00 sec)

mysql> SELECT SUM(NOOFSEATS) FROM CANCELLATION2;
+----------------+
| SUM(NOOFSEATS) |
+----------------+
|              6 |
+----------------+
1 row in set (0.00 sec)
```

6 Find the total number of cancelled seats.

MySQL> select sum(noofseats) as canceled_seats from cancellation2;

```
mysql> SELECT * FROM CANCELLATION2;
+-------+---------------------+-----------+---------+------------+---------+
| PNRNO | JOURNEYDATE         | NOOFSEATS | ADDRESS | CONTACTNO  | STATUS  |
+-------+---------------------+-----------+---------+------------+---------+
| 10201 | 2012-02-20 10:20:25 |         2 | HYD     | 9654235242 | CONFIRM |
| 10202 | 2012-02-22 10:22:25 |         2 | HYD     | 9654232451 | CONFIRM |
| 10203 | 2012-03-22 10:30:25 |         2 | DELHI   | 9654587960 | CONFIRM |
+-------+---------------------+-----------+---------+------------+---------+
3 rows in set (0.00 sec)

mysql> select sum(noofseats) as canceled_seats from  cancellation2;
+----------------+
| canceled_seats |
+----------------+
|              6 |
+----------------+
1 row in set (0.00 sec)
```

Creation and Droping of Views

mysql> create table students(sid int primary key,name varchar(15),login varchar(15), age int,gpa real); mysql> create table Enrolled(sid int,cid int,grade varchar(5),primary key(sid,cid), foreign key(sid) references students(sid));

mysql>create view BStudents(name,sid,course) AS SELECT

s.name,s.sid,E.cid from students s,enrolled E where s.sid=e.sid AND

E.grade='B';

```
mysql> create view BStudents(name,sid,course) AS SELECT s.name,s.sid,E.cid from
students s,enrolled E where s.sid=e.sid AND E.grade='B';
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Bstudents;
+-------+-------+--------+
| name  | sid   | course |
+-------+-------+--------+
| jones | 53666 |      3 |
| Guldu | 53832 |      2 |
+-------+-------+--------+
2 rows in set (0.03 sec)
```

Syntax: Drop view viewname;

Mysql> Drop view Bstudents; Mysql> Drop view Goodstudents;

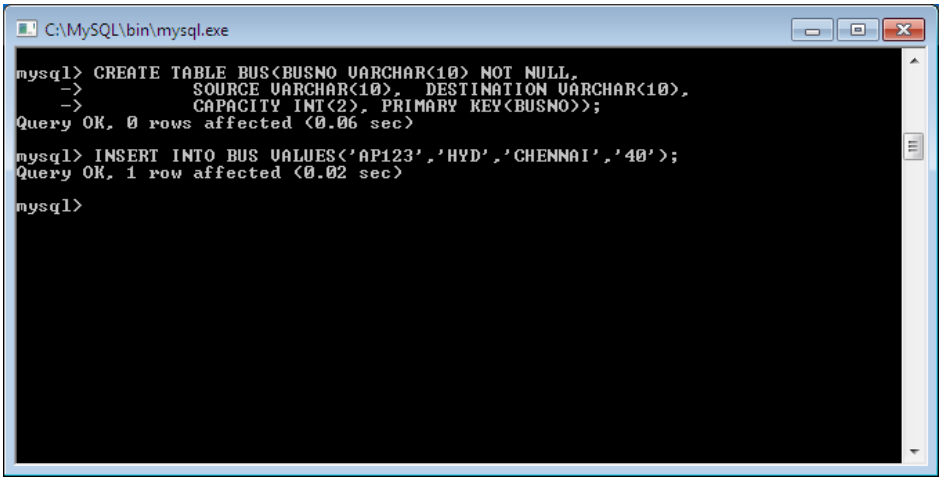
```
mysql> Drop view Bstudents;
Query OK, 0 rows affected (0.00 sec)

mysql> Drop view Goodstudents;
Query OK, 0 rows affected (0.00 sec)
```

EXPERIMENT – 8
TRIGGERS

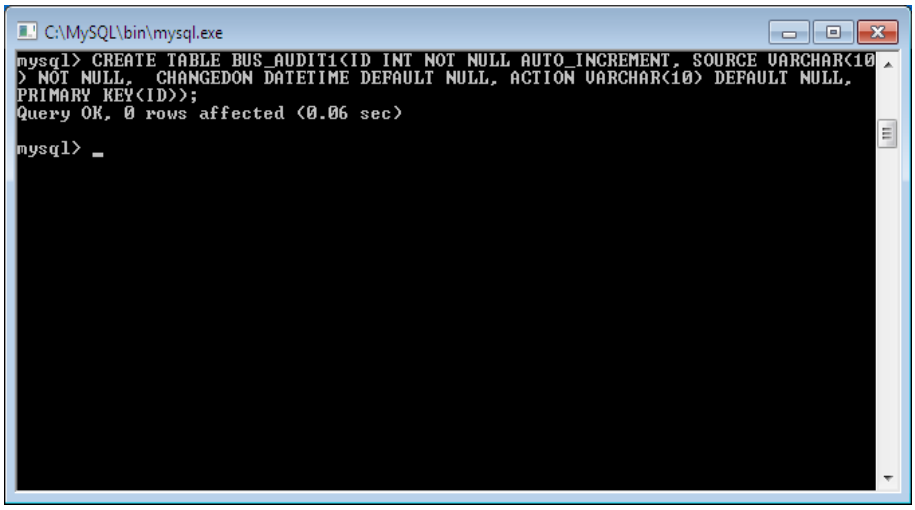
Aim: Creation of insert trigger, delete trigger and update trigger.

MySQL>CREATE TABLE BUS(BUSNO VARCHAR(10) NOT NULL, SOURCE VARCHAR(10), DESTINATION VARCHAR(10), CAPACITY INT(2), PRIMARY KEY(BUSNO));

MySQL>INSERT INTO BUS VALUES('AP123','HYD','CHENNAI','40');



CREATE TABLE BUS_AUDIT1(ID INT NOT NULL AUTO_INCREMENT, SOURCE VARCHAR(10) NOT NULL, CHANGEDON DATETIME DEFAULT NULL, ACTION VARCHAR(10) DEFAULT NULL, PRIMARY KEY(ID));

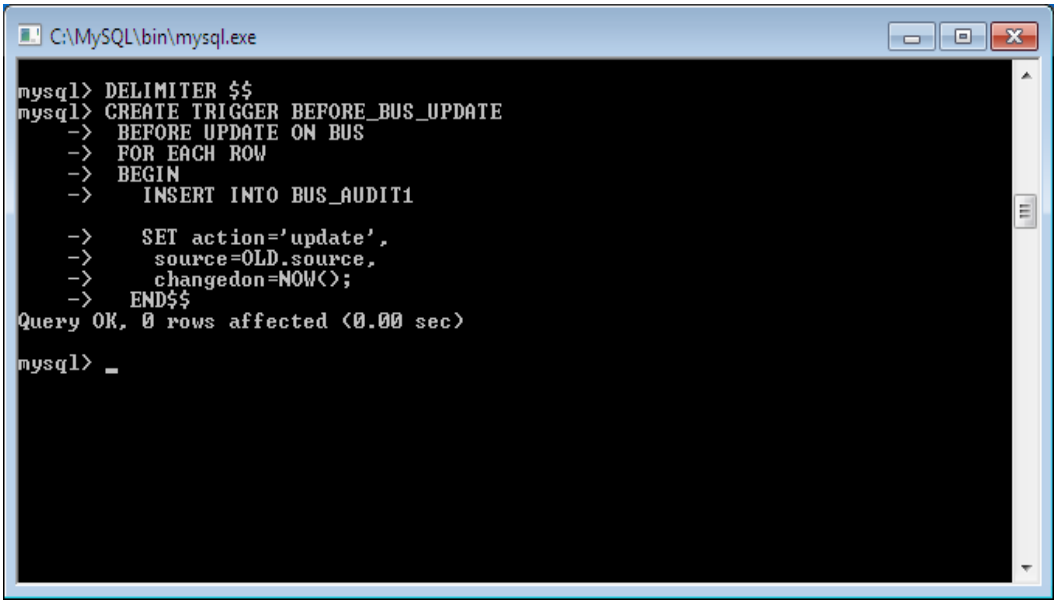


CREATE TRIGGER BEFORE_BUS_UPDATE BEFORE UPDATE ON BUS

FOR EACH ROW BEGIN

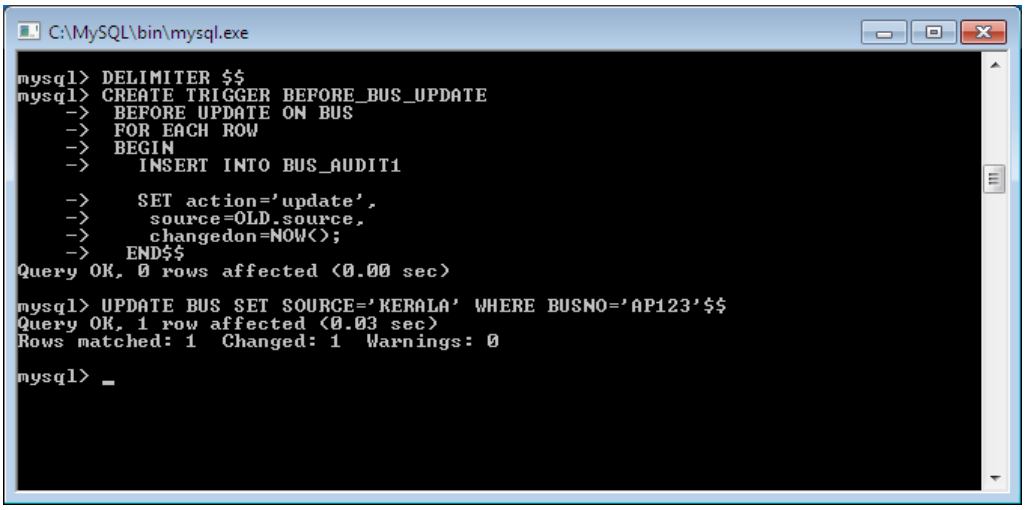
INSERT INTO BUS_AUDIT1

SET action='update', source=OLD.source, changedon=NOW(); END$$



UPDATE :

MySQL>UPDATE BUS SET SOURCE='KERALA' WHERE BUSNO='AP123'$$



SNo	Source	Changedon	Action
1	Banglore	2014:03:23 12:51:00	Insert
2	Kerela	2014:03:25:12:56:00	Update
3	Mumbai	2014:04:26:12:59:02	Delete

INSERT:

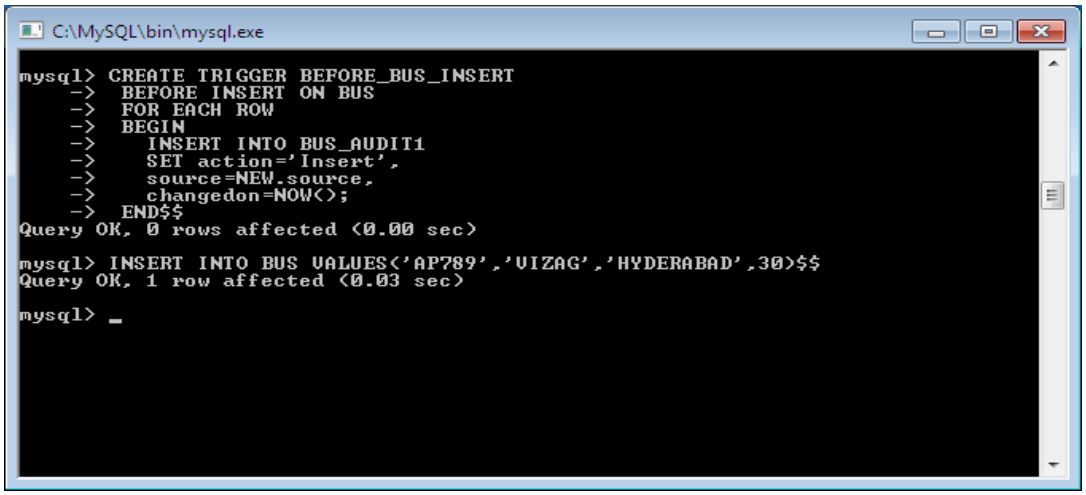
CREATE TRIGGER BEFORE_BUS_INSERT BEFORE INSERT ON BUS

FOR EACH ROW BEGIN

INSERT INTO BUS_AUDIT1

SET action='Insert', source=NEW.source, changedon=NOW(); END$$

MYSQL>INSERT INTO BUS VALUES('AP789','VIZAG','HYDERABAD',30)$$



SNo	Source	Changedon	Action
1	Banglore	2014:03:23 12:51:00	Insert
2	Kerela	2014:03:25:12:56:00	Update
3	Mumbai	2014:04:26:12:59:02	Delete

CREATE TRIGGER BEFORE_BUS_DELETE BEFORE DELETE ON BUS

FOR EACH ROW BEGIN

DELETE FROM BUS_AUDIT1

SET action='Insert', source=NEW.source, changedon=NOW(); END$$

DELETE FROM BUS WHERE SOURCE=’HYDERABAD’$$

SNo	Source	Changedon	Action
1	Banglore	2014:03:23 12:51:00	Insert
2	Kerela	2014:03:25:12:56:00	Update
3	Mumbai	2014:04:26:12:59:02	Delete

Examples

CREATE TRIGGER updcheck1 BEFORE UPDATE ON passengerticket FOR EACH ROW

BEGIN

IF NEW.TicketNO > 60 THEN

SET New.TicketNo = New.TicketNo; ELSE

SET New.TicketNo = 0; END IF;

END;

```
mysql> select * from passengerticket;$$
+------------+----------+
| passportid | TicketNo |
+------------+----------+
| 145        |      100 |
| 278        |      200 |
| 6789       |      300 |
| 82302      |      400 |
| 82403      |      500 |
| 82502      |      600 |
+------------+----------+
6 rows in set (0.00 sec)

mysql> desc passengerticket;$$
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| passportid | varchar(15) | NO   | PRI |         |       |
| TicketNo   | int(11)     | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

```
mysql> CREATE TRIGGER updcheck BEFORE UPDATE ON passengerticket
    -> FOR EACH ROW
    -> BEGIN
    -> IF NEW.TicketNO > 60 THEN
    -> SET New.TicketNo = TicketNo;
    -> ELSE
    -> SET New.TicketNo = 0;
    -> END IF;
    -> END;
    -> $$
Query OK, 0 rows affected (0.00 sec)

mysql> update passengerticket set TicketNo=TicketNo-50 where passportid=145;$$
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from passengerticket;$$
+------------+----------+
| passportid | TicketNo |
+------------+----------+
| 145        |        0 |
| 278        |      200 |
| 6789       |      300 |
| 82302      |      400 |
| 82403      |      500 |
| 82502      |      600 |
+------------+----------+
6 rows in set (0.00 sec)
```

```
mysql> select * from passengerticket;$$
+------------+----------+
| passportid | TicketNo |
+------------+----------+
| 145        |        0 |
| 278        |      200 |
| 6789       |      300 |
| 82302      |      400 |
| 82403      |      500 |
| 82502      |      600 |
+------------+----------+
6 rows in set (0.00 sec)

mysql> CREATE TRIGGER updcheck BEFORE UPDATE ON passengerticket
    -> FOR EACH ROW
    -> BEGIN
    -> IF NEW.TicketNO>60 THEN
    -> SET New.TicketNo=New.TicketNo;
    -> ELSE
    -> SET New.TicketNo=0;
    -> END IF;
    -> END;
    -> $$
Query OK, 0 rows affected (0.00 sec)

mysql> update passengerticket set TicketNo=TicketNo+80 where passportid=145;$$
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from passengerticket;$$
+------------+----------+
| passportid | TicketNo |
+------------+----------+
| 145        |       80 |
| 278        |      200 |
| 6789       |      300 |
| 82302      |      400 |
| 82403      |      500 |
| 82502      |      600 |
+------------+----------+
6 rows in set (0.00 sec)
```

EXPERIMENT – 9
PROCEDURES

Aim: Creation of stored Procedures and Execution of Procedures and Modification of Procedures.

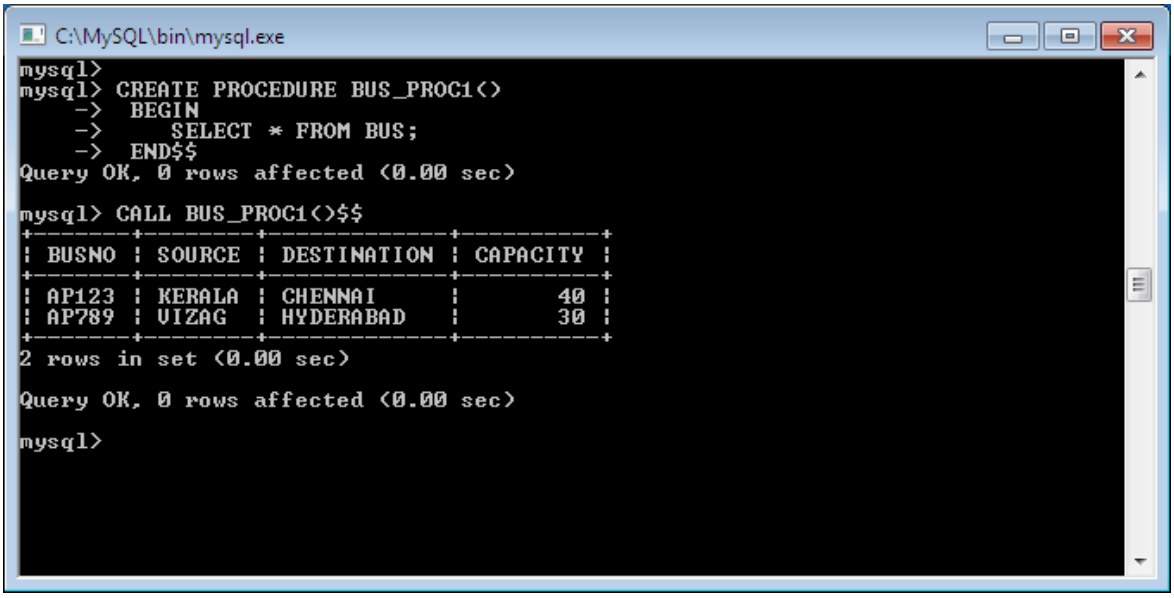
Ex1:

CREATE PROCEDURE BUS_PROC1() BEGIN

SELECT * FROM BUS;

END$$

CALL BUS_PROC1()$$



Ex2:

CREATE PROCEDURE SAMPLE2() BEGIN
DECLARE X INT(3); SET X=10;
SELECT X;

END$$
Mysql> CALL SAMPLE2()$$

```
C:\MySQL\bin\mysql.exe
mysql> CREATE PROCEDURE SAMPLE2()
    -> BEGIN
    ->    DECLARE X INT(3);
    ->    SET X=10;
    ->    SELECT X;
    -> END$$
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> CALL SAMPLE2()$$
+------+
| X    |
+------+
|   10 |
+------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> _
```

Ex3: CREATE PROCEDURE SIMPLE_PROC(OUT PARAM1 INT) BEGIN

SELECT COUNT(*) INTO PARAM1 FROM BUS;

END$$

Mysql> CALL SIMPLE_PROC(@a)$$ Mysql> select @a;

```
mysql> SELECT * FROM BUS2;
+-------+-----------+-------------+
| BusNo | Source    | Destination |
+-------+-----------+-------------+
| 35    | HYD       | CHENNAI     |
| 45    | Tirupathi | Banglore    |
| 55    | HYD       | MUMBAI      |
| 65    | DELHI     | KOLKATHA    |
+-------+-----------+-------------+
4 rows in set (0.00 sec)

mysql> DELIMITER $$
mysql> CREATE PROCEDURE SIMPLE_PROC(OUT PARAM1 INT)
    -> BEGIN
    -> SELECT COUNT(*) INTO PARAM1 FROM BUS2;
    -> END $$
Query OK, 0 rows affected (0.00 sec)

mysql> CALL SIMPLE_PROC(@a)$$
Query OK, 0 rows affected (0.03 sec)

mysql> SELECT @a$$
+------+
| @a   |
+------+
| 4    |
+------+
1 row in set (0.00 sec)
```

EXPERIMENT – 10

Cursors

Aim: Declare a cursor that defines a result set. Open the cursor to establish the result set. Fetch the data into local variables as needed from the cursor, one row at a time. Close the cursor when done.

Cursors

In MySQL, a cursor allows row-by-row processing of the result sets. A cursor is used for the result set and returned from a query. By using a cursor, you can iterate, or by step through the results of a query and perform certain operations on each row. The cursor allows you to iterate through the result set and then perform the additional processing only on the rows that require it.

In a cursor contains the data in a loop. Cursors may be different from SQL commands that operate on all the rows in the returned by a query at one time.

There are some steps we have to follow, given below :

- Declare a cursor
- Open a cursor statement
- Fetch the cursor
- Close the cursor

1 . Declaration of Cursor : To declare a cursor you must use the DECLARE statement. With the help of the variables, conditions and handlers we need to declare a cursor before we can use it. first of all we will give the cursor a name, this is how we will refer to it later in the procedure. We can have more than one cursor in a single procedure so its necessary to give it a name that will in some way tell us what its doing. We then need to specify the select statement we want to associate with the cursor. The SQL statement can be any valid SQL statement and it is possible to use a dynamic where clause using variable or parameters as we have seen previously.

Syntax : DECLARE *cursor_name* CURSOR FOR *select_statement;*

2 . Open a cursor statement : For open a cursor we must use the open statement.If we want to fetch rows from it you must open thecursor.

Syntax : OPEN cursor_name;

3 . Cursor fetch statement : When we have to retrieve the next row from the cursor and move the cursor to next row then you need to fetch the cursor.

Synatx : FETCH cursor_name INTO var_name;

If any row exists, then the above statement fetches the next row and cursor pointer moves ahead to the next row.

4 . Cursor close statement : By this statement closed the open cursor.

Syntax: CLOSE_name;

By this statement we can close the previously opened cursor. If it is not closed explicitly then a cursor is closed at the end of compound statement in which that was declared.

Delimiter $$

```
Create procedure p1(in_customer_id int) begin
declare v_id int;
declare v_name varchar(20); declare v_finished integer default 0;
declare c1 cursor for select sid,sname from students where sid=in_customer_id; declare
continue handler for NOT FOUND set v_finished=1;
open c1; std:LOOP
fetch c1 into v_id,v_name; if v_finished=1 then
leave std; end if;
select concat(v_id,v_name); end LOOP std;
close c1; end;
```

```
mysql> select * from students;
+------+---------+------+-------+
| sid  | sname   | age  | marks |
+------+---------+------+-------+
| 1    | ravi    |   15 |    25 |
| 2    | ramu    |   20 |    30 |
| 2    | rahul   |   18 |    26 |
| 5    | kiran   |   19 |    28 |
| 6    | varun   |   21 |    32 |
| 8    | ramesh  |   22 |    33 |
| 8    | xyz     |   10 |    20 |
+------+---------+------+-------+
7 rows in set (0.00 sec)
```

```
mysql> delimiter $$
mysql> Create procedure p1(in_customer_id int)
    -> begin
    -> declare v_id int;
    -> declare v_name varchar(20);
    -> declare v_finished integer  default 0;
    -> declare c1 cursor for select sid,sname from students where sid=in_custome
r_id;
    -> declare continue handler for NOT FOUND set v_finished=1;
    -> open c1;
    -> std:LOOP
    -> fetch c1 into v_id,v_name;
    -> if v_finished=1 then
    -> leave std;
    -> end if;
    -> select concat(v_id,v_name);
    -> end LOOP std;
    -> close c1;
    -> end;$$
Query OK, 0 rows affected (0.01 sec)
```

ADDITIONAL PROGRAMMS

EMPLOYEES TABLE

mysql> create table Employees(ssn varchar(15),name varchar(20),lot int,PRIMARY KEY(ssn)); mysql> insert into Employees values('123-22-3666','Attishoo',48);

mysql> insert into Employees values('321-31-5368','Smiley',22); mysql> insert into Employees values('131-24-3650','Smethurst',35);

```
mysql> desc Employees;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| ssn   | varchar(15) | NO   | PRI |         |       |
| name  | varchar(20) | YES  |     | NULL    |       |
| lot   | int(11)     | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from Employees;
+-------------+-----------+------+
| ssn         | name      | lot  |
+-------------+-----------+------+
| 123-22-3666 | Attishoo  |   48 |
| 131-24-3650 | Smethurst |   35 |
| 321-31-5368 | Smiley    |   22 |
+-------------+-----------+------+
3 rows in set (0.02 sec)
```

DEPARTMENT TABLE

mysql> create table Departments(did int,dname varchar(10),budget real, PRIMARY KEY(did));

mysql> insert into Departments values(05,'CSE',500000);

mysql> insert into Departments values(04,'ECE',400000);

mysql> insert into Departments values(03,'ME',300000);

mysql> insert into Departments values(01,'CE',100000);

```
mysql> desc Departments;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| did    | int(11)     | NO   | PRI | 0       |       |
| dname  | varchar(10) | YES  |     | NULL    |       |
| budget | double      | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from Departments;
+-----+-------+--------+
| did | dname | budget |
+-----+-------+--------+
|   1 | CE    | 100000 |
|   3 | ME    | 300000 |
|   4 | ECE   | 400000 |
|   5 | CSE   | 500000 |
+-----+-------+--------+
4 rows in set (0.00 sec)
```

Sailors , Reserves , Boats Tables

Mysql> Create table Sailors(Sid integer PRIMARY KEY,sname varchar(15), rating int,age real); Mysql>Create table Reserves(Sid int,Bid int,Day Date);

Mysql>Create table Boats(Bid int,Bname varchar(15),Color varchar(15);

```
mysql> select * from sailors;
+-----+---------+--------+------+
| sid | sname   | rating | age  |
+-----+---------+--------+------+
|  22 | Dustin  |      7 |   45 |
|  29 | Brutus  |      1 |   33 |
|  31 | Lubber  |      8 | 55.5 |
|  32 | Andy    |      8 | 25.5 |
|  58 | Rusty   |     10 |   35 |
|  64 | Horatio |      7 |   35 |
|  71 | Zorba   |     10 |   16 |
|  74 | Horatio |      9 |   35 |
|  85 | Art     |      3 | 25.5 |
|  95 | Bob     |      3 | 63.5 |
+-----+---------+--------+------+
10 rows in set (0.00 sec)

mysql> select * from reserves;
+------+------+------------+
| sid  | bid  | day        |
+------+------+------------+
|   22 |  101 | 1998-10-10 |
|   22 |  102 | 1998-10-10 |
|   22 |  103 | 1998-08-10 |
|   22 |  104 | 1998-07-10 |
|   31 |  102 | 1998-10-11 |
|   31 |  103 | 1998-06-11 |
|   31 |  104 | 1998-12-11 |
|   64 |  101 | 1998-05-09 |
|   64 |  102 | 1998-08-09 |
|   44 |  103 | 1998-08-09 |
+------+------+------------+
10 rows in set (0.00 sec)

mysql> select * from boats;
+------+----------+-------+
| bid  | bname    | color |
+------+----------+-------+
|  101 | Interlake | blue  |
|  102 | Interlake | red   |
|  103 | Clipper   | green |
|  103 | Marine    | red   |
+------+----------+-------+
```

mysql> select S.sname from sailors S, reserves R where S.sid=R.sid AND R.bid=103;

```
mysql> select S.sname from sailors S, reserves R where S.sid=R.sid AND R.bid=103
;
+--------+
| sname  |
+--------+
| Dustin |
| Lubber |
+--------+
2 rows in set (0.00 sec)
```

mysql> select sname from sailors s,Reserves R where S.sid=R.sid AND bid=103; mysql> select R.sid from Boats B,Reserves R where B.bid=R.bid AND B.color='red';

```
mysql> select sname from sailors s,Reserves R where S.sid=R.sid AND bid=103;
+--------+
| sname  |
+--------+
| Dustin |
| Lubber |
+--------+
2 rows in set (0.00 sec)

mysql> select R.sid from Boats B,Reserves R where B.bid=R.bid AND B.color='red';

+------+
| sid  |
+------+
|   22 |
|   22 |
|   31 |
|   31 |
|   64 |
|   44 |
+------+
6 rows in set (0.00 sec)
```

mysql> select S.sname from sailors S,reserves R,Boats B where S.sid=R.sid AND R.bid=B.bid AND B.color='red';

mysql> select B.color from Sailors S,Reserves R,Boats B where S.sid=R.sid AND R.bid=B.bid AND S.sname='Lubber';

```
mysql> select S.sname from sailors S,reserves R,Boats B where S.sid=R.sid AND
R.bid=B.bid AND B.color='red';
+---------+
| sname   |
+---------+
| Dustin  |
| Dustin  |
| Lubber  |
| Lubber  |
| Horatio |
+---------+
5 rows in set (0.00 sec)


mysql> select B.color from Sailors S,Reserves R,Boats B where S.sid=R.sid AND
R.bid=B.bid AND S.sname='Lubber';
+-------+
| color |
+-------+
| red   |
| green |
| red   |
+-------+
3 rows in set (0.00 sec)
```

mysql> select S.sname,S.rating+1 AS rating from Sailors S,Reserves R1,Reserves R2 where S.sid=R1.sid AND S.sid=R2.sid AND R1.day=R2.day AND R1.bid<>R2.bid;

mysql> select S1.sname AS name1,S2.sname AS name2 from sailors S1,sailors S2 where 2*S1.rating=S2.rating-1;

```
mysql> select S.sname,S.rating+1 AS rating from Sailors S,Reserves R1,Reserves
R2 where S.sid=R1.sid AND S.sid=R2.sid AND R1.day=R2.day AND R1.bid<>R2.bid;
+--------+--------+
| sname  | rating |
+--------+--------+
| Dustin |      8 |
| Dustin |      8 |
+--------+--------+
2 rows in set (0.00 sec)

mysql> select S1.sname AS name1,S2.sname AS name2 from sailors S1,sailors S2
 where 2*S1.rating=S2.rating-1;
+--------+---------+
| name1  | name2   |
+--------+---------+
| Art    | Dustin  |
| Bob    | Dustin  |
| Art    | Horatio |
| Bob    | Horatio |
| Brutus | Art     |
| Brutus | Bob     |
+--------+---------+
6 rows in set (0.02 sec)
```

```
mysql> select S.age from sailors S where S.sname LIKE 'B_%B';
+------+
| age  |
+------+
| 63.5 |
+------+
1 row in set (0.00 sec)


mysql> select S.sname from sailors S where S.sname LIKE 'B_%B';
+-------+
| sname |
+-------+
| Bob   |
+-------+
1 row in set (0.00 sec)
```

USING UNION , INTERSECT , AND EXCEPT

1).Find the names of sailors who have reserved a red or a green boat.

```
mysql> SELECT S.SNAME FROM SAILORS S,RESERVES R,BOATS B
    -> WHERE S.SID=R.SID AND R.BID=B.BID
    -> AND(B.COLOR='red' OR B.COLOR='green');
+---------+
| SNAME   |
+---------+
| Dustin  |
| Dustin  |
| Dustin  |
| Lubber  |
| Lubber  |
| Lubber  |
| Horatio |
+---------+
7 rows in set (0.01 sec)
```

OR

```
mysql> SELECT S.SNAME
    -> FROM SAILORS S,RESERVES R,BOATS B
    -> WHERE S.SID=R.SID AND R.BID=B.BID AND B.COLOR='red'
    -> UNION
    -> SELECT S2.SNAME
    -> FROM SAILORS S2,BOATS B2,RESERVES R2
    -> WHERE S2.SID=R2.SID AND R2.BID=B2.BID AND B2.COLOR='green';
+---------+
| SNAME   |
+---------+
| Dustin  |
| Lubber  |
| Horatio |
+---------+
3 rows in set (0.02 sec)
```

2). Find the names of sailors who have reserved both a red and a green boat.

```
SELECT S.SNAME

FROM SAILORS S,RESERVES R,BOATS B

WHERE S.SID=R.SID AND R.BID=B.BID AND B.COLOR='red' INTERSECT

SELECT S2.SNAME

FROM SAILORS S2,RESERVES R2,BOATS B2

WHERE S2.SID=R2.SID AND R2.BID=B2.BID AND B2.COLOR='green';
```

NESTED QUERIES

1) Find the Names of sailors who have reserved boat 103

```
mysql> SELECT S.SNAME FROM SAILORS S
    -> WHERE S.SID IN (SELECT R.SID FROM RESERVES R
    -> WHERE R.BID=103)
    -> ;
+--------+
| SNAME  |
+--------+
| Dustin |
| Lubber |
+--------+
2 rows in set (0.00 sec)
```

2) Find the names of Sailors who have reserved a red Boat

```
mysql> SELECT S.SNAME FROM SAILORS S
    -> WHERE S.SID IN (SELECT R.SID FROM RESERVES R
    -> WHERE R.BID IN (SELECT B.BID FROM BOATS B
    -> WHERE B.COLOR='RED'));
+---------+
| SNAME   |
+---------+
| Dustin  |
| Lubber  |
| Horatio |
+---------+
3 rows in set (0.00 sec)
```

3) Find the names of Sailors who have NOT reserved a red Boat

```
mysql> select s.sname from sailors s
    -> where s.sid NOT IN (select r.sid from reserves r
    -> where r.bid IN (select b.bid from boats b
    -> where b.color='red'));
+---------+
| sname   |
+---------+
| Brutus  |
| Andy    |
| Rusty   |
| Zorba   |
| Horatio |
| Art     |
| Bob     |
+---------+
7 rows in set (0.00 sec)
```

Correlated Nested Queries:

1) Find the names of Sailors who have reserved a red Boat

```
mysql> select s.sname from sailors s
    -> where EXISTS ( select * from reserves r
    -> where r.bid=103 AND r.sid=s.sid);
+--------+
| sname  |
+--------+
| Dustin |
| Lubber |
+--------+
2 rows in set (0.00 sec)
```

Set Comparison Operators:

1) Find sailors whose rating is better than some sailor called Horatio

```
mysql> select s.sid from sailors s
    -> where s.rating > ANY ( select s2.rating from sailors s2
    -> where s2.sname='Horatio');
+-----+
| sid |
+-----+
|  31 |
|  32 |
|  58 |
|  71 |
|  74 |
+-----+
5 rows in set (0.00 sec)
```

2) Find the sailors with the highest rating.

mysql> SELECT S.sid FORM Sailors WHERE S.rating>=ALL(SELECT S2.rating FROM Sailors S2);

The GROUP BY and HAVING Clauses:

1) Find the age of the youngest sailor for each rating level.

```
mysql> SELECT S.rating , MIN(S.age)
    -> FROM Sailors S
    -> GROUP BY S.rating;
+--------+------------+
| rating | MIN(S.age) |
+--------+------------+
|      1 |         33 |
|      3 |       25.5 |
|      7 |         35 |
|      8 |       25.5 |
|      9 |         35 |
|     10 |         16 |
+--------+------------+
6 rows in set (0.01 sec)
```

2) Find the age of the youngest sailor who is eligible to vote for each rating level with at least two such sailors

```
mysql> SELECT S.rating , MIN(S.age) AS minage
    -> FROM  Sailors S
    -> WHERE S.age>=18
    -> GROUP BY S.rating
    -> HAVING COUNT(*)>1;
+--------+--------+
| rating | minage |
+--------+--------+
|      3 |   25.5 |
|      7 |     35 |
|      8 |   25.5 |
+--------+--------+
3 rows in set (0.00 sec)
```

3) For each red boat , find the number of reservations for this boat

```
mysql> SELECT B.BID,COUNT(*) AS SAILORCOUNT
    -> FROM BOATS B,RESERVES R
    -> WHERE R.BID=B.BID AND B.COLOR='RED'
    -> GROUP BY B.BID;
+------+-------------+
| BID  | SAILORCOUNT |
+------+-------------+
|  102 |           3 |
|  103 |           3 |
+------+-------------+
2 rows in set (0.00 sec)
```

4) Find the average age of sailors for each rating level that has at least two sailors

```
mysql> SELECT S.RATING, AVG(S.AGE) AS AVGAGE
    -> FROM SAILORS S
    -> GROUP BY S.RATING
    -> HAVING 1<(SELECT COUNT(*)
    -> FROM SAILORS S2
    -> WHERE S.RATING = S2.RATING);
+--------+--------+
| RATING | AVGAGE |
+--------+--------+
|      3 |   44.5 |
|      7 |     40 |
|      8 |   40.5 |
|     10 |   25.5 |
+--------+--------+
4 rows in set (0.01 sec)
```