



Estd.2001

Sri Indu

College of Engineering & Technology

UGC Autonomous Institution

Recognized under 2(f) & 12(B) of UGC Act 1956,

NAAC, Approved by AICTE &

Permanently Affiliated to JNTUH



NAAC

NATIONAL ASSESSMENT AND
ACCREDITATION COUNCIL



DBMS LAB MANUAL

II Year CSE- Semester II

DEPARTMENT OF INFORMATION TECHNOLOGY

ACADEMIC YEAR 2022-23



SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC, New Delhi)

Recognized under 2(f) and 12(B) of UGC Act 1956

NBA Accredited, Approved by AICTE and Permanently affiliated to JNTUH

Sheriguda (V), Ibrahimpatnam, R.R.Dist, Hyderabad - 501 510

DEPARTMENT OF
INFORMATION TECHNOLOGY
LAB MANUAL

Branch:CSE
Subject: Database Management Systems Lab
Academic Year: 2022-23
Core/Elective/H&S: Core

Class: B.Tech- II Year-I sem
Code: R20CSE22L2
Regulation: R20
Credits:1.5

Prepared By
Name:

Verified By
Head of the Department:

DEPARTMENT OF INFORMATION TECHNOLOGY

LAB MANUAL- INDEX

S. No	Contents
1	Vision, Mission, PEOs, POs, PSOs & COs
2	Syllabus
3	Institution Academic Calendar
4	Lab Structure
5	Time Table
6	Index
7	Programs
8	Viva Questions
9	Additional Programs



SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY

B. TECH –INFORMATION TECHNOLOGY

INSTITUTION VISION

To be a premier Institution in Engineering & Technology and Management with competency, values and social consciousness.

INSTITUTION MISSION

- IM₁** Provide high quality academic programs, training activities and research facilities.
- IM₂** Promote Continuous Industry-Institute Interaction for Employability, Entrepreneurship, Leadership and Research aptitude among stakeholders.
- IM₃** Contribute to the Economical and technological development of the region, state and nation.

DEPARTMENT VISION

To be a recognized knowledge center in the field of Information Technology with self-motivated, employable engineers to society.

DEPARTMENT MISSION

The Department has following Missions:

- DM₁** To offer high quality student centric education in Information Technology.
- DM₂** To provide a conducive environment towards innovation and skills.
- DM₃** To involve in activities that provide social and professional solutions.
- DM₄** To impart training on emerging technologies namely cloud computing and IOT with involvement of stake holders.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

- PEO 1: Higher Studies:** Graduates with an ability to apply knowledge of Basic sciences and programming skills in their career and higher education.
- PEO 2: Lifelong Learning:** Graduates with an ability to adopt new technologies for ever changing IT industry needs through Self-Study, Critical thinking and Problem solving skills.
- PEO 3: Professional skills:** Graduates will be ready to work in projects related to complex problems involving multi-disciplinary projects with effective analytical skills.
- PEO 4: Engineering Citizenship:** Graduates with an ability to communicate well and exhibit social, technical and ethical responsibility in process or product.

PROGRAM OUTCOMES (POs) & PROGRAM SPECIFIC OUTCOMES (PSOs)

PO	Description
PO 1	Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO 2	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.
PO 3	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.
PO 4	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.
PO 5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	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.
PO 7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO 8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice
PO 9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO 10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO 11	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.
PO 12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological Change.
Program Specific Outcomes	
PSO 1	Software Development: To apply the knowledge of Software Engineering, Data Communication, Web Technology and Operating Systems for building IOT and Cloud Computing applications.
PSO 2	Industrial Skills Ability: Design, develop and test software systems for world-wide network of computers to provide solutions to real world problems.
PSO 3	Project implementation: Analyze and recommend the appropriate IT Infrastructure required for the implementation of a project.

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE OUTCOMES (CO'S)

COURSE NAME: Database Management System

Course Name	Course outcomes
C22L2.1	Design database schema for a given application and apply normalization.
C22L2.2	Acquires skills in using SQL commands for data definition and data manipulation.
C22L2.3	Develop solutions for database applications using procedures, cursors and triggers.

COURSE ARTICULATION MATRIX

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C22L2.1	3	3	3	2	-	2	-	-	2	-	-	2	2	3	-
C22L2..2	3	2	2	-	-	-	-	-	-	-	-	2	2	2	-
C22L2.3	3	3	2	2	-	-	-	-	-	-	-	2	2	2	1
C22L2	3	2.6	2.3	2	-	0.6	-	-	0.6	-	-	2	2	2.3	0.3

Database Management Systems

SRI INDUCOLLEGE OF ENGINEERING & TECHNOLOGY

(An Autonomous Institution under UGC, New Delhi)

B.Tech.-II Year – II Semester

L	T	P	C
0	0	3	1.5

(R20CSE22L2) Database Management Systems Lab

Co-requisites:

- Database Management Systems.

Objectives:

- Introduce ER data model, database design and normalization
- Learn SQL basics for data definition and data manipulation

Course Outcomes:

- Design database schema for a given application and apply normalization.
- Acquire skills in using SQL commands for data definition and data manipulation.
- Develop solutions for database applications using procedures, cursors and triggers.

List of Experiments:

Roadway Travels

"Roadway Travels" is in business since 1997 with several buses connecting different places in India. Its main office is located in Hyderabad.

The company wants to computerize its operations in the following areas:

- Reservation and Ticketing
- Cancellations

Reservation & Cancellation:

Reservations are directly handled by booking office. Reservations can be made 30 days in advance and tickets issued to passenger. One Passenger/person can book many tickets (to his/her family).

Cancellations are also directly handed at the booking office.

In the process of computerization of Roadway Travels you have to design and develop a Database which consists the data of Buses, Passengers, Tickets, and Reservation and cancellation details. You should also develop query's using SQL to retrieve the data from database.

The above process involves many steps like

1. Analyzing the problem and identifying the Entities and Relationships,
2. E-R Model
3. Relational Model
4. Normalization
5. Creating the database

6. Querying. Students are supposed to work on these steps week wise and finally create a complete

Example: Entities:

- 1. BUS
- 2. Ticket
- 3. Passenger

Relationships:

- 1. Reservation
- 2. Cancellation

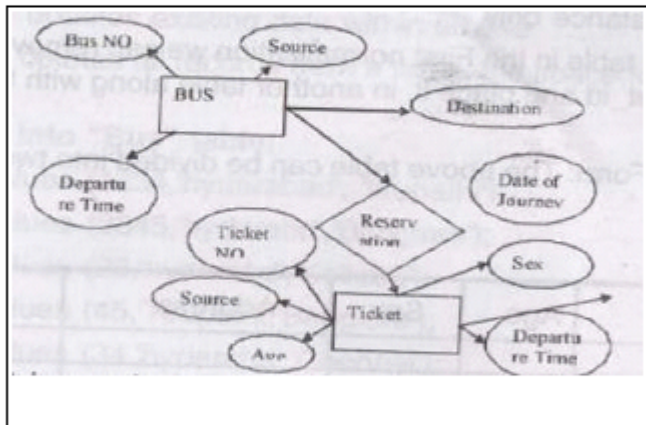
PRIMARYKEYATTRIBUTES:

- 1. TicketID (Ticket Entity)
- 2. PassportID (Passenger Entity)
- 3. Bus_No (BusEntity)

Experiment 1: Concept design with E-R Model

Relate the entities appropriately. Apply cardinalities for each relationship. Identify strong entities and weak entities (if any). Indicate the type of relationships (total/partial). Try to incorporate generalization, aggregation, specialization etc. wherever required.

Example: E-R diagram for bus



Experiment 2: Relational Model

Represent all the entities (Strong, Weak) in tabular fashion. Represent relationships in a tabular fashion. There are different ways of representing relationships as tables based on the requirement. Different types of attributes (Composite, Multi-valued and, and Derived) have different way of representation.

Example: The passenger tables look as below. This is an example. You can add more attributes based on E-R model. This is not a normalized table.

Passenger

Name	Age	Sex	Address	Ticket_id	PassportID

Note: The student is required to submit a document by Represent relationships in a tabular form

Experiment 3: Normalization

Database normalization is a technique for designing relational database tables to minimize duplication of information and, in so doing, to safeguard the database against certain types of logical or structural problems, namely data anomalies. For example, when multiple instances of a given piece of information occur in a table, the possibility exists that these instances will not be kept consistent when the data within the table is updated, leading to a loss of data integrity. A table that is sufficiently normalized is less vulnerable to problems of this kind, because its structure effects the basic assumptions for when multiple instances of the same information should be represented by a single instance only.

For the above table in the First normalization we can remove the multiple valued attribute Ticket_id and place it in another table along with the primary key of passenger.

Name	Age	Sex	Address	PassportID

PassportID	Ticket_id

You can do the second and third normal forms if required. Anyhow Normalized tables are given at the end.

Experiment 4: Practicing DDL commands

In this week you will learn Creating databases, How to create tables, altering the database, dropping tables and databases if not required. You will also try truncate, rename commands etc.

Example for creation of a normalized "Passenger" table.

```
CREATE TABLE Passenger(  
    Passport_id INTEGERPRIMARYKEY,  
    Name VARCHAR(50)NOTNULL,  
    Age INTEGER NOTNULL, Sex  
    CHAR,  
    Address VARCHAR(50)NOT NULL  
);
```

Similarly create all other tables.

Experiment 5: Practicing DML commands

DML commands are used for managing data within schema objects. Some examples:

- SELECT-retrieve data from the database
- INSERT-insert data in to a table
- UPDATE-updates existing data within a table
- DELETE-deletes all records from a table, the space for the records remain

Insert values into "Bus" table:

```
insert into Bus values (1234, 'hyderabad', 'tirupathi');insert into Bus values(2345, 'hyderabad', 'banglore');insert into Bus values(23, 'hyderabad','kolkata');insert into Bus values(45, 'tirupathi','bangalore');
```

insert values into "Passenger" table:

```
Insert into Passenger values (1,45,'ramesh',45,'M','abc123');  
insert into Passenger values(2, 78, 'geetha', 36, 'F', 'abc124');  
insert into Passenger values(45, 90,'ram', 30,'M', 'abc12');  
insert into Passenger values(67,89,'ravi',50,'M','abc14');  
insert into Passenger values(56,22,'seetha',32,'F','abc55');
```

Few more Examples of DML commands:

```
SELECT*FROM Bus; (selects all the attributes and displays) UPDATE Bus SET  
Bus_No=1WHEREBus_No= 2;
```

Experiment 6: Querying (using ANY, ALL, IN, EXISTS, NOTEXIST, UNION, INTERSECT, Constraints etc.

In this week you are going to practice queries (along with sub queries) using ANY, ALL, IN, EXISTS, NOTEXIST, UNION, INTERSECT, Constraints etc.

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. Findthenamesofpassengerswhoseageisbetween30 and 45.
6. Display all the passengers names beginning with 'A'
7. Display the sorted list of passenger's names

Experiment 7: Querying using Aggregate functions, GROUP BY, HAVING and Creation and dropping of Views.

You are going to practice queries using Aggregate functions GROUPBY, HAVING and Creation and dropping of VIEWS.

1. Write a Query to display the information present in the Passenger and cancellation tables.
Hint: Use UNION Operator.
2. Display the number of days in a week on which the 9W01 bus is available.
3. Find number of tickets booked for each PNR_ No using GROUP BY CLAUSE. Hint: Use GROUP BY on PNR_No.
4. Find the distinct PNR numbers that are present.
5. Find the number of tickets booked by a passenger where the number of seats is greater than 1.
Hint: Use GROUPBY, WHERE and HAVING CLAUSES.
6. Find the total number of cancelled sets.

Experiment 8: Triggers (Creation of insert trigger, delete trigger, update trigger)

In this week you are going to work on Triggers. Creation of insert trigger, delete trigger, update trigger. Practice triggers using the above database.

E.g.

```
CREATE TRIGGER update check BEFORE UPDATE ON passenger FOR EACH
ROW BEGIN
  IF New.TicketNO > 60 THEN
    SET New.TicketNO = TicketNo;
  ELSE
    SET New.TicketNo = 0;
  END IF;
END
```

Experiment 9: Procedures

In this session you are going to learn Creation of stored procedure, Execution of procedure and modification of procedure. Practice procedures using the database.

E.g:

```
CREATE PROCEDURE myproc()
BEGIN
  SELECT COUNT(Tickets)
  FROM Ticket
  WHERE age >= 40;
END;
```

Experiment 10: Usage of Cursors

In this week you need to do the following: 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

```
CREATE PROCEDURE myproc(in_customer_idINT) BEGIN
  DECLARE v_id INT;
  DECLARE v_name VARCHAR(30);

  DECLARE c1 CURSOR FOR
  SELECT stdid, stdFirstname FROM students WHERE stdid=in_customer_id;

  OPENc1;
  FETCHc1INTOv_id,v_name; CLOSE c1;
END;
```

Tables:

BUS

BusNo: VARCHAR : PK (primarykey)

Source: VARCHAR

Destination: VARCHAR

Passenger

PPNO: VARCHAR(15) : PK Name:

VARCHAR(15)

Age: INT(4)

Sex:CHAR(10) : Male/Female

Address: VARCHAR(20)

Passenger_Tickets

PPNO:VARCHAR(15): PK

Ticket_No:NUMERIC(9)

Reservation

PNR_No: NUMERIC(9) : FK

Journey_date: DATETIME(8)

No_of_seats: INT(8) Address:

VARCHRA(50)

Contact_No: NUMERIC(9)-->Should not less than 9 and Should not accept any other character other than interger

STATUS:CHAR(2): Yes/No

Cancellation

PNR_No: NUMERIC(9):FK

Journey_date: DATETIME(8)

No_of_seats : INT(8) Address:

VARCHRA(50)

Contact_No:NUMERIC(9)-->Should not less than 9 and Should not accept any other character other than interger

STATUS:CHAR(2):Yes/No

Ticket

Ticket_No: NUMERIC(9) : FK

Journey_date: DATETIME(8)

Age: INT(4)

Sex: CHAR(10): Male/Female

Source: VARCHAR

Destination: VARCHAR

Dep_time: VARCHAR

REFERENCES BOOKS:

1. Database Systems design, Implementation, and Management, Peter Rob & Carlos Coronel 7th Edition.
 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
-



SRI INDU COLLEGE OF ENGINEERING & TECHNOLOGY
 (An Autonomous Institution under UGC, New Delhi)
 Recognized under 2(f) and 12(B) of UGC Act 1956
 NBA Accredited, Approved by AICTE and Permanently affiliated to JNTUH
 Sheriguda (V), Ibrahimpatnam, R.R.Dist, Hyderabad - 501 510

D4

BR-20

LR.NO.SICET/AUTO/DAE/BR-20/ACADEMIC-CAL/421/2021

DATE: 09.10.2021

II B.TECH ACADEMIC CALENDAR
ACADEMIC YEAR : 2021-2022

Dr. G. SURESH,
Principal,

To,
All the HODs
Sir,

Sub: SICET (Autonomous) - Academic & Evaluation - Academic Calendar for
II B.Tech - I & II Semester for the academic year **2021-22** - Reg.

The approved Academic Calendar for **II B.Tech - I & II Semester** for the academic
 year **2021-22** is given below:

ACADEMIC CALENDAR - II B.TECH - I & II SEMESTER
ADMITTED BATCH - 2020-2021 of BR-20 Regulation

I SEMESTER

SNO	EVENT	PERIOD	DURATION
1.	Commencement of Class Work	21-10-2021	
2.	1 st Spell of Instructions for covering First Two and a half Units	21.10.2021 - 15.12.2021	8 Weeks
3.	I Mid Examinations	16.12.2021 - 18.12.2021	3 Days
4.	Submission of I Mid Term Examination Marks	24-12-2021	
5.	2 nd Spell of Instructions for covering Remaining Two and a half Units	20.12.2021 - 12.02.2022	8 Weeks
6.	II Mid Examinations	14.02.2022 - 16.02.2022	3 Days
7.	Submission of II Mid Term Examination Marks	22-02-2022	
8.	Preparation & Practical Examinations	17.02.2022 - 23.02.2022	1 Week
9.	I Semester End Examinations (Regular/Suppl.)	24.02.2022 - 09.03.2022	2 Weeks
10.	Supplementary Examinations for II Semester (BR-12, BR-14, BR-16 & BR-18 Regulations)	10.03.2022 - 23.03.2022	2 Weeks
Commencement of Class-Work for II B.Tech - II Semester 10.03.2022.			

II SEMESTER

SNO	EVENT	PERIOD	DURATION
1.	Commencement of II Sem Class Work	10.03.2022	
2.	1 st Spell of Instructions for covering First Two and a half Units	10.03.2022 - 04.05.2022	8 Weeks
3.	I Mid Examinations	05.05.2022 - 07.05.2022	3 Days
4.	Submission of I Mid Term Examination Marks	13-05-2022	
5.	2 nd Spell of Instructions for covering Remaining Two and a half Units	09.05.2022 - 02.07.2022	8 Weeks
6.	II Mid Examinations	04.07.2022 - 06.07.2022	3 Days
7.	Submission of II Mid Term Examination Marks	12-07-2022	
8.	Preparation & Practical Examinations	07.07.2022 - 13.07.2022	1 Week
9.	II Semester End Examinations (Regular/Suppl.)	14.07.2022 - 27.07.2022	2 Weeks
10.	Supplementary Examinations for I Semester (BR-12, BR-14, BR-16 & BR-18 Regulations)	28.07.2022 - 11.08.2022	2 Weeks
Commencement of Class Work for III B.Tech - I Semester - 01.08.2022			

[Signature]
ACE

[Signature]
CE 9/10/21

[Signature]
DEAN

[Signature]
PRINCIPAL
PRINCIPAL

Copy to
Copy to all the Heads of the Depts., A.O.

CONTROLLER OF EXAMINATIONS

DEAN, ACADEMIC AFFAIRS & EVALUATION
 Sri Indu College of Engineering & Technology
 (An Autonomous Institution under JNTUH)

Sri Indu College of Engineering & Technology
 (An Autonomous Institution under JNTUH)

Lab Structure

1.DIMENSIONS OF THE LAB

Area of the lab in Sqmts : Sqm

2. CAPACITY OF THE LAB

:

3. EQUIPMENTS

Computer Systems (Clients) :

CPU :

Monitors :

Key Board :

Mouse :

4. SYSTEM CONFIGURATION

: DELLE1916HV

Speed : 3.10 GHz, 2GB RAM

Hard Disk : 500 GB

HCL LED Monitor Size : 18.5

5. SOFTWARE

: Turbo C++, MYSQL, Eclipse, Windows

OS,Open Office

6. AMBIENCE

Printers : 00

Projector : 01

Computer Tables :

Student Chairs :

Charts :

Photo Frames :

Switch/Hub :

White Boards :

A/C s :

Power Backup : UPS

TIME TABLE



Sri Indu College of Engineering & Technology

(An Autonomous Institution under UGC)

Sheriguda (V), Ibrahimpatnam (M), Ranga Reddy (Dist) – 501 510

DEPARTMENT OF INFORMATION TECHNOLOGY

LAB Time - Table

Class: II-II SEM

w.e.f:

Time	9:40-10:40	10:40-11:40	11:40-12:40	12:40 To 1:20	1:20-2:15	2:15-3:10	3:10-4:00
Days	1	2	3		4	5	6
Monday				L U N C H			
Tuesday							
Wednesday							
Thursday							
Friday							
Saturday							

GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis /program / experiment details.
3. Student should enter into the laboratory with:
 - a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
 - b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
 - c. Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation notebook, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out ; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department

Principal

INDEX

S.no	Topic	Page no	No. of sections	proposed date of handling	co
1.	Concept Design with E-R Model	1-5	3		2
2.	Relational Model	6-10	3		2
3.	Normalization	11-12	3		1
4.	Practicing DDL Commands	13-15	3		2
5.	Practicing DML Commands	16-21	3		2
6.	Querying using ANY, ALL, IN, EXISTS, UNION, INTERSECT, CONSTRAINTS etc.	22-31	3		2
7.	Querying using Aggregate functions, GROUP BY & HAVING and Creation and Dropping of Views.	32-36	3		2
8.	TRIGGERS USING Insert, Delete & Update.	37-41	3		3
9.	Procedures.	42-43	3		3
10.	Usage Cursors	44-46	3		3
11.	Additional Programs	47-56	3		

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.

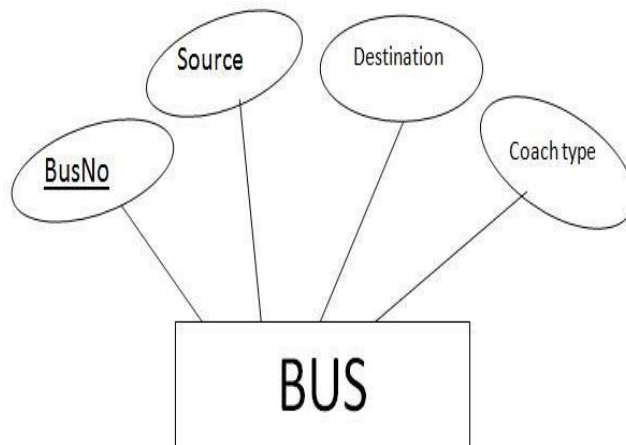
E-R Model

Bus

- BusNo
- Source
- Destination
- CoachType

SCHEMA

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



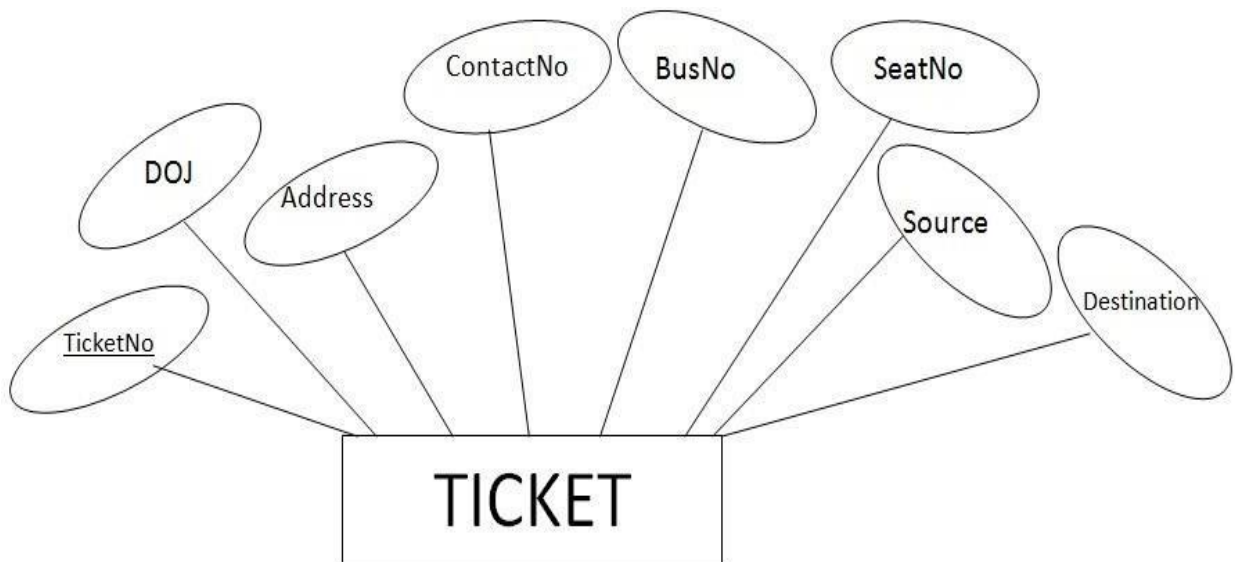
Ticket

- 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)

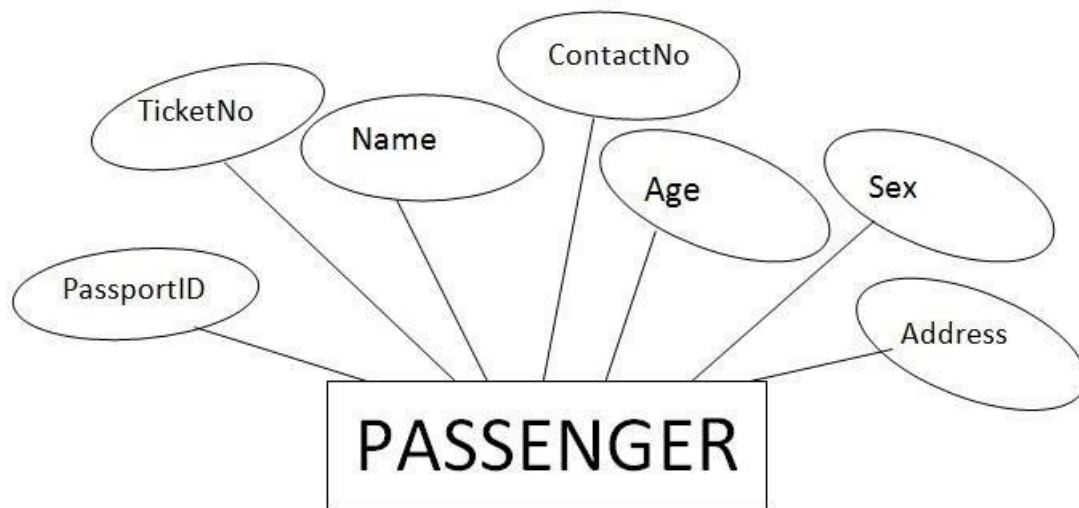


Passenger

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

SCHEMA

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

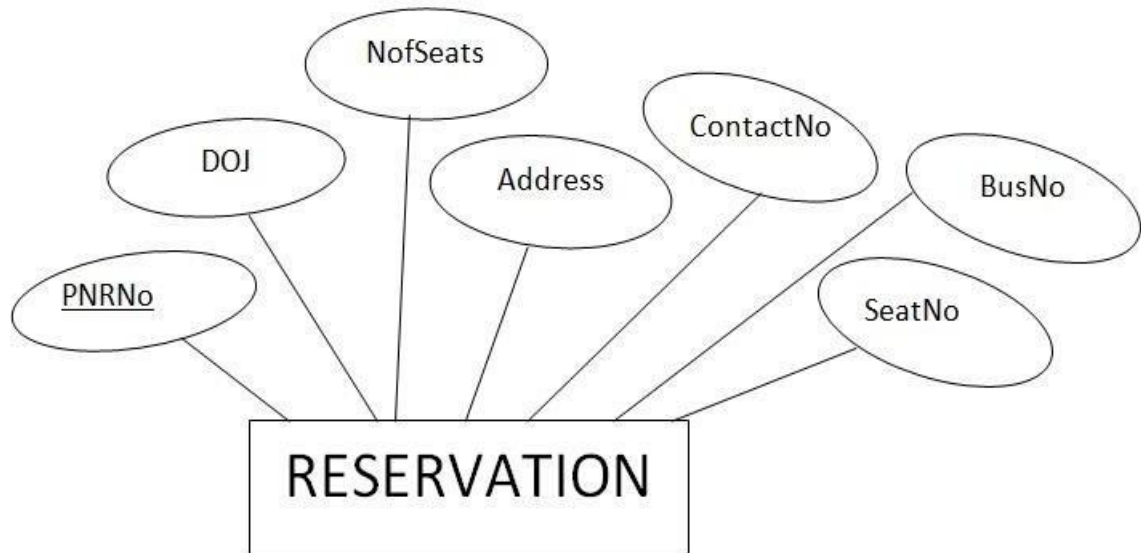


Reservation

- 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)

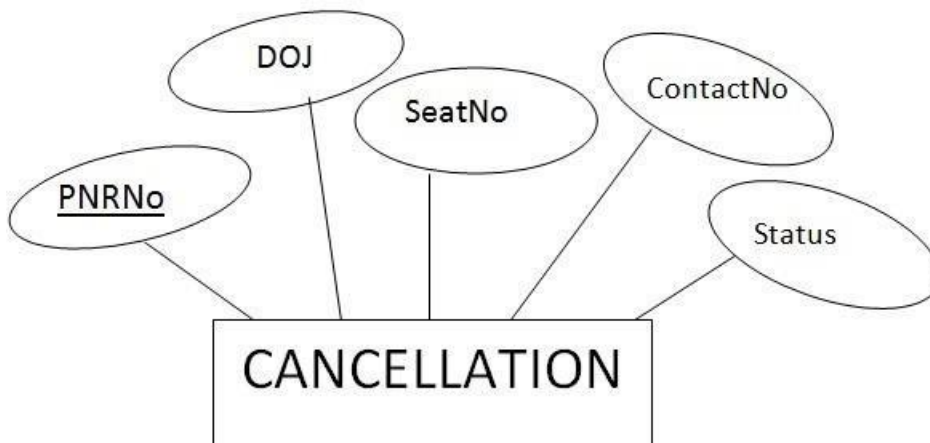


Cancellation

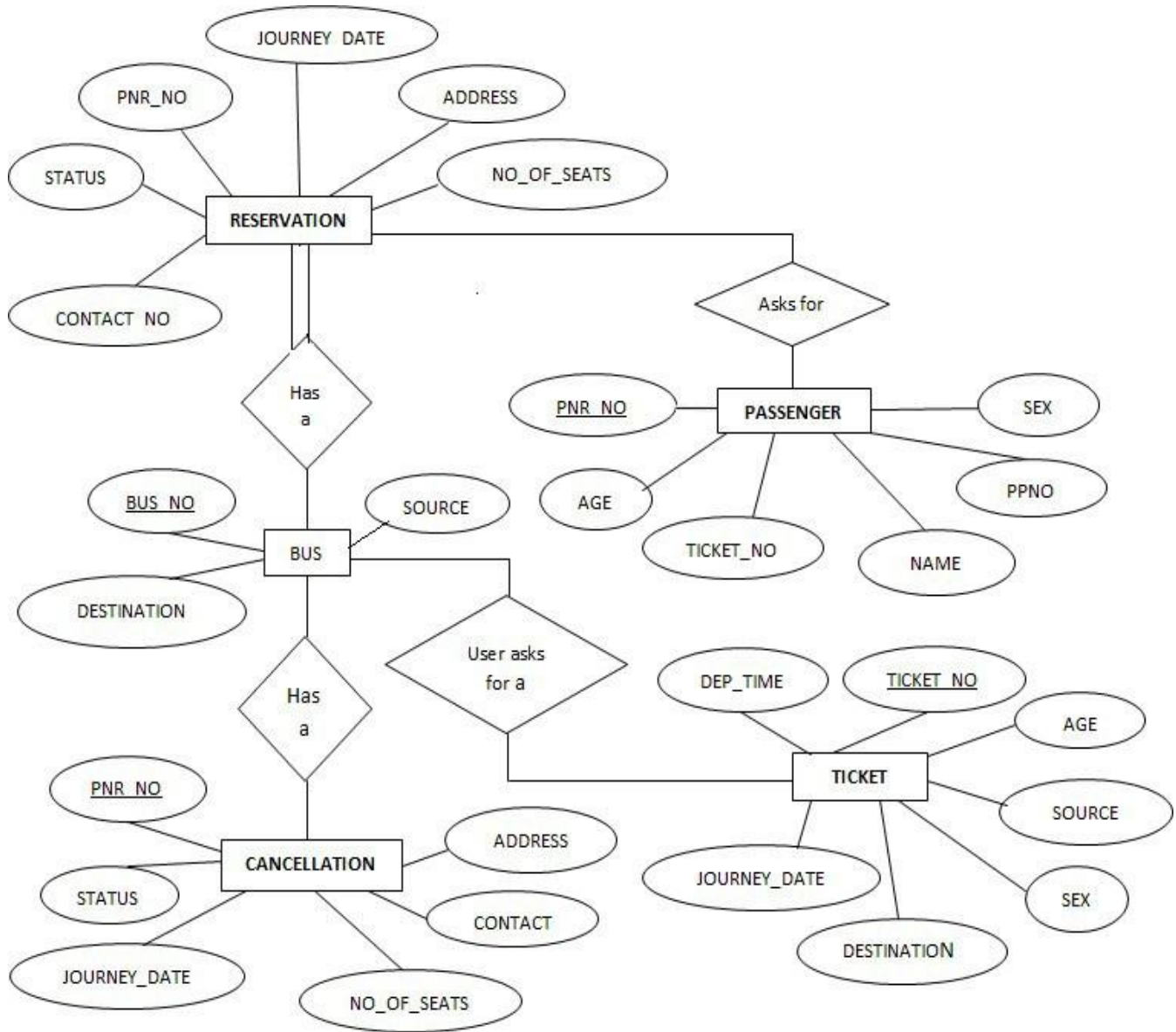
- PNRNo
- DOJ
- SeatNo
- ContactNo
- Status

SCHEMA

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



CONCEPT DESIGN WITH E-R MODEL



Viva Questions:

1. Define DBMS?
2. Define the terms i) Entity ii) Entity set iii) weak entity set iv) strong entity set?
3. What is ER-diagram?
4. What are the different types of entities?
5. List various types of attributes?

EXPERIMENT – 2
RELATIONAL MODEL

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

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 varchar(10),seatNo int,Source varchar(10),Destination varchar(10),primary key(TicketNo,BusNo),foreign key(BusNo) references 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(15),age integer,sex char(2),address varchar(20),primary key(passportid,ticketno),foreign key(ticketno) references 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,NoofSeates integer,Address varchar(20),ContactNo varchar(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 varchar(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)
```

Viva Questions:

- 1.What is relation schema and a relation?
- 2.What is Relation and Relationship?
- 3.Define Instance and Schema?
- 4.What is the difference between ER and Relational Model?
- 5.What is the degree of relation?

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 $P \rightarrow Q$ in set F of FDs that holds for each FD. $P \rightarrow Q$ 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));

Viva Questions:

1. Define Redundancy?
2. What is decomposition?
3. What is Normalization?
4. What is fully functional dependency?
5. List the different types of Normal forms?

EXPERIMENT – 4

PRACTICING DDL COMMANDS

AIM : Creating Tables and altering the Tables

Mysql>Create table passenger2(passportId Integer Primary Key, Name varchar(10) NotNull, 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  |     |          |       |
| 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  |     |          |       |
| Address    | varchar(20)   | NO   |     |          |       |
| TicketNo   | varchar(10)   | YES  |     |          |       |
+-----+-----+-----+-----+-----+-----+
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)

Viva Questions:

- 1.What is DDL?
- 2.What are the different types of commands used in DDL language?
- 3.Difference between drop and truncate?
- 4.What is the use of DDL language in DBMS?
- 5.Define Database?

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 | Bangalore   |
| 34    | Hyderabad | Chennai     |
| 45    | Tirupathi | Bangalore   |
+-----+-----+-----+
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   | Bangalore   |
| 34    | Hyderabad   | Chennai     |
| 45    | Tirupathi   | Bangalore   |
+-----+-----+-----+
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 | Bangalore   |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Bangalore   |
+-----+-----+-----+
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 | Bangalore   |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Bangalore   |
+-----+-----+-----+
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 | Bangalore   |
| 34    | Secundrabad | Chennai     |
| 45    | Tirupathi   | Bangalore   |
+-----+-----+-----+
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   | Bangalore   |
+-----+-----+-----+
1 row in set (0.00 sec)
```

Viva Questions:

- 1.What is DML?
- 2.What is the use of DML in DBMS?
- 3.What are the DML Commands?
- 4.What is the use of Alter command in DBMS?
- 5.Define view?

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',9654235242);
Query OK, 1 row affected (0.03 sec)

mysql> insert into reservation2 values(10202,'2012-02-22 10:22:25',05,'HYD',9654232451);
Query OK, 1 row affected (0.02 sec)

mysql> insert into reservation2 values(10203,'2012-03-22 10:30:25',05,'DELHI',9654587960);
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');
```



The screenshot shows a MySQL command prompt window with the following content:

```
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)
```

Viva Questions:

1. What are the Different types of set operators?
2. What are the aggregate operators?
3. What is the difference between ANY and OR operators?
4. What is NULL Values?
5. What is the use of EXISTS operator?

EXPERIMENT – 7

Querying Aggregate Functions (GROUP BY, HAVING and Creation and Dropping of Views)

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 Dropping 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 view name;

```
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)
```

Viva Questions:

- 1.What is Nested query?
- 2.What is the use of Group By Clause?
- 3.Define Join?List different types of joins?
- 4.What is difference between left outer join and right outer join?
- 5.What is Co-related nested query?

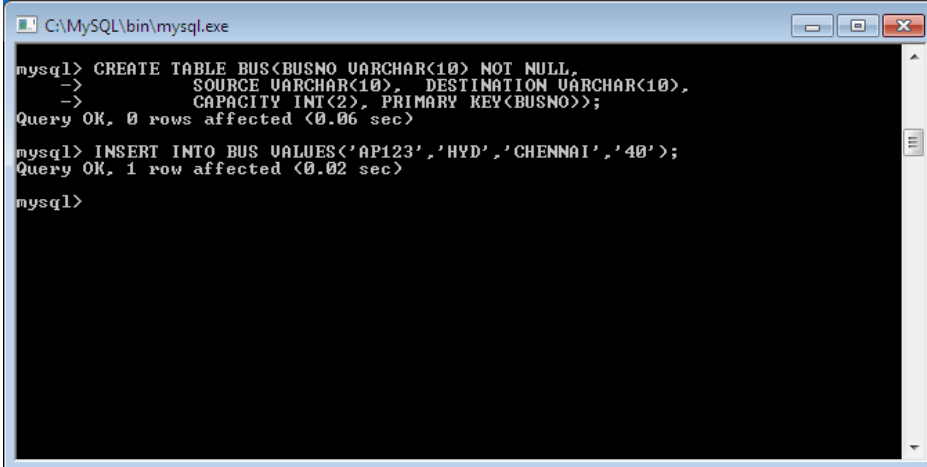
EXPERIMENT-8

Triggers

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

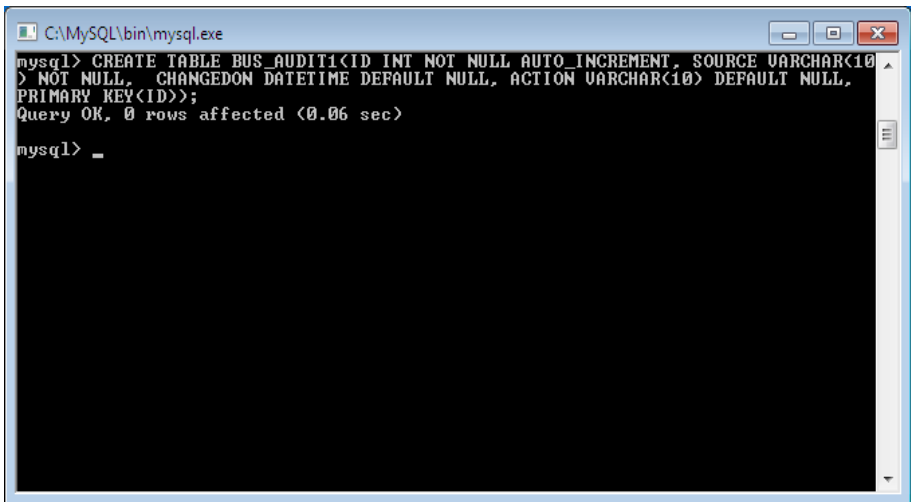
```
MySQL>CREATE TABLE BUS(BUSNO VARCHAR(10) NOT NULL, SOURCE  
VARCHAR(10), DESTINATION VARCHAR(10), CAPACITY INT(2), PRIMARY  
KEY(BUSNO));
```

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



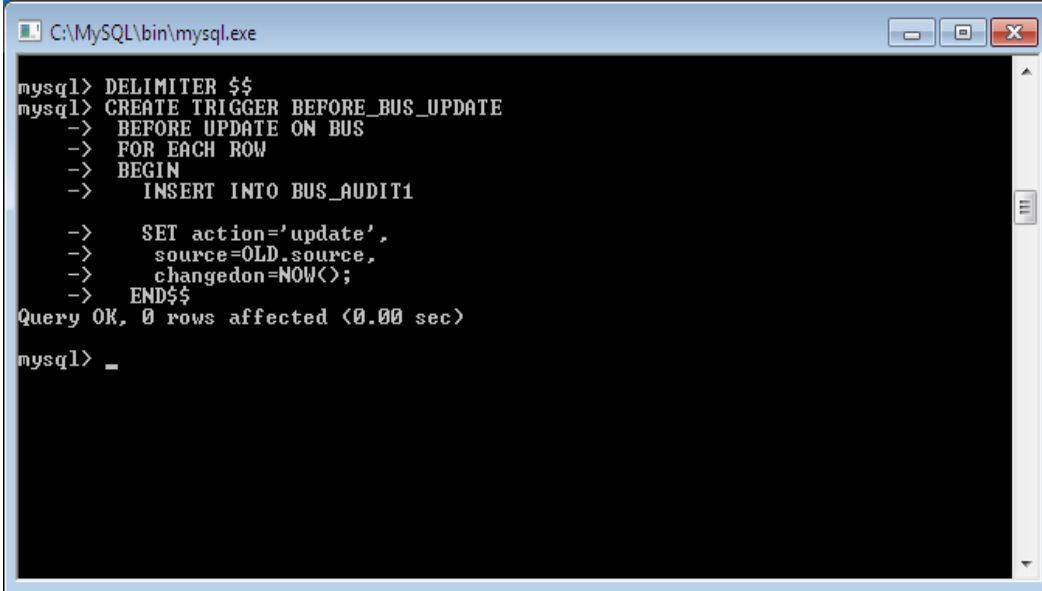
```
C:\MySQL\bin\mysql.exe  
mysql> CREATE TABLE BUS(BUSNO VARCHAR(10) NOT NULL,  
-> SOURCE VARCHAR(10), DESTINATION VARCHAR(10),  
-> CAPACITY INT(2), PRIMARY KEY(BUSNO));  
Query OK, 0 rows affected (0.06 sec)  
mysql> INSERT INTO BUS VALUES('AP123','HYD','CHENNAI','40');  
Query OK, 1 row affected (0.02 sec)  
mysql>
```

```
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));
```



```
C:\MySQL\bin\mysql.exe  
mysql> 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));  
Query OK, 0 rows affected (0.06 sec)  
mysql> _
```

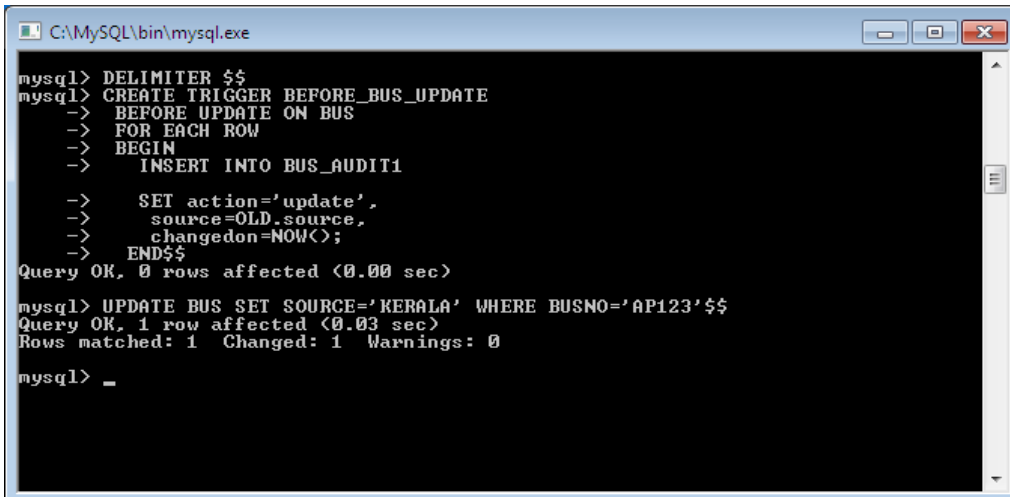
```
CREATE TRIGGER BEFORE_BUS_UPDATE BEFORE UPDATE ON BUS
FOR EACH ROW BEGIN
INSERT INTO BUS_AUDIT1
SET action='update', source=OLD.source, changedon=NOW(); END$$
```



```
C:\MySQL\bin\mysql.exe
mysql> DELIMITER $$
mysql> CREATE TRIGGER BEFORE_BUS_UPDATE
-> BEFORE UPDATE ON BUS
-> FOR EACH ROW
-> BEGIN
-> INSERT INTO BUS_AUDIT1
-> SET action='update',
-> source=OLD.source,
-> changedon=NOW();
-> END$$
Query OK, 0 rows affected (0.00 sec)
mysql> _
```

UPDATE :

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



```
C:\MySQL\bin\mysql.exe
mysql> DELIMITER $$
mysql> CREATE TRIGGER BEFORE_BUS_UPDATE
-> BEFORE UPDATE ON BUS
-> FOR EACH ROW
-> BEGIN
-> INSERT INTO BUS_AUDIT1
-> SET action='update',
-> source=OLD.source,
-> changedon=NOW();
-> END$$
Query OK, 0 rows affected (0.00 sec)
mysql> UPDATE BUS SET SOURCE='KERALA' WHERE BUSNO='AP123'$$
Query OK, 1 row affected (0.03 sec)
Rows matched: 1 Changed: 1 Warnings: 0
mysql> _
```

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)\$\$

```

C:\MySQL\bin\mysql.exe
mysql> 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$$
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO BUS VALUES('AP789','VIZAG','HYDERABAD',30)$$
Query OK, 1 row affected (0.03 sec)

mysql> _

```

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	Chagedon	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)
```

Viva Questions:

1. Define Trigger?
2. What are the types of triggers?
3. What is the advantage of trigger in database
4. Define Active data bases?
5. When we apply trigger?

Experiment-9 Procedures

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

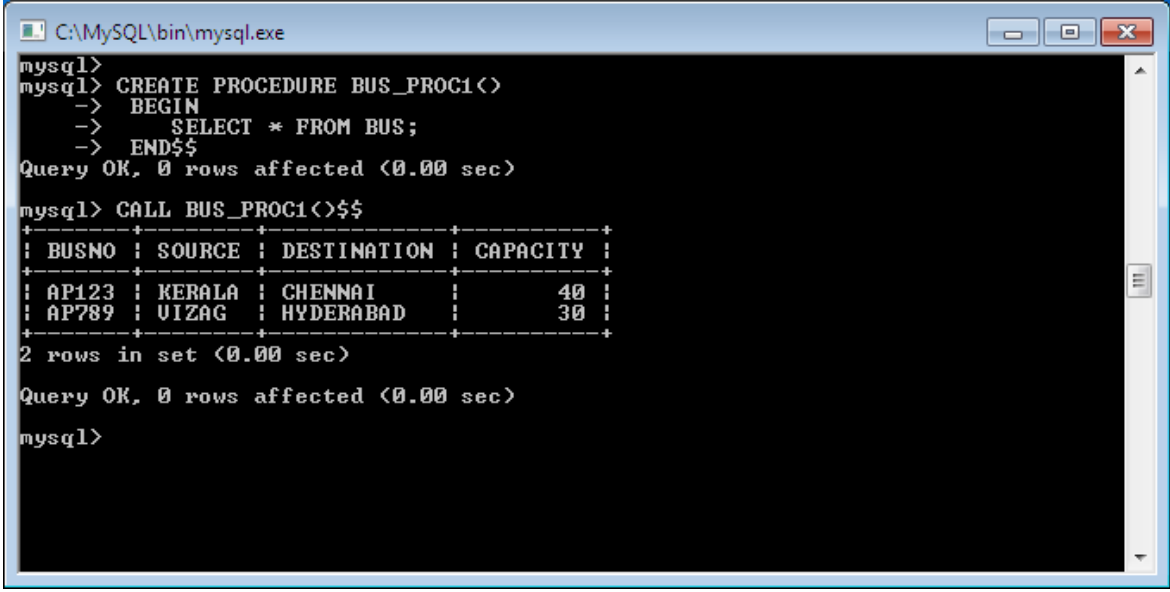
Ex1:

```
CREATE PROCEDURE BUS_PROC1() BEGIN
```

```
SELECT * FROM BUS;
```

```
END$$
```

```
CALL BUS_PROC1()$$
```



The screenshot shows a MySQL command prompt window with the following text:

```
C:\MySQL\bin\mysql.exe
mysql>
mysql> CREATE PROCEDURE BUS_PROC1(<)
-> BEGIN
->     SELECT * FROM BUS;
-> END$$
Query OK, 0 rows affected (0.00 sec)

mysql> CALL BUS_PROC1(<)$$
+-----+-----+-----+-----+
| BUSNO | SOURCE | DESTINATION | CAPACITY |
+-----+-----+-----+-----+
| AP123 | KERALA | CHENNAI    | 40       |
| AP789 | VIZAG  | HYDERABAD  | 30       |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql>
```

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 | Bangalore   |
| 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)

```

Viva Questions:

1. What do you mean by Procedure?
2. What is stored procedures used in databases?
3. What is stored procedure?
4. What are the advantage of using stored procedure?
5. List different types of keys in dbms?

EXPERIMENT-10 USAGE 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 the cursor.

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.

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

Viva Questions:

- 1.How Cursor used in Database?
- 2.How many types of Cursors in My Sql?
- 3.How to open and close the Cursor?
- 4.Differentiate Procedure and Cursor?
- 5.Can you pass a parameter to a Cursor?

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)
```

N . 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 FROM 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)
```