Counselling Code : VBIT

VIGNANA BHARATHI Institute of Technology

(A UGC Autonomous Institution, Approved by AICTE, Accredited by NBA & NAAC-A Grade, Affiliated to JNTUH)

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING

[ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING]

LAB STUDENT MANUAL

DATABASE MANAGEMENT SYSTEMS LAB

B.Tech II – II SEM

Prepared by,

A . Gopya sri

Asst. Professor,

CSE[AI&ML] Department.

GUIDELINES TO STUDENTS

1.Equipment in the lab for the use of student community. Students need to maintain a proper decorum in the computer lab. Students must use the equipment with care. Any damage is caused is punishable.

2.Students are instructed to come to lab in formal dresses only.

3.Students are supposed to occupy the systems allotted to them and are not supposed to talk or make noise in the lab.

4.Students are required to carry their observation book and lab records with completed exercises while entering the lab.

5.Lab records need to be submitted every week.

6.Students are not supposed to use pen drives in the lab.

DEPARTMENT VISION AND MISSION

Vision

To achieve global standards of quality in technical education with the help of advanced resources and automated tools to bridge the gap between industry and academia.

Mission

- Build the students technically competent on global arena through effective teaching learningprocess and world-class infrastructure.
- Inculcate professional ethics, societal concerns, technical skills and life-long learning tosucceed in multidisciplinary fields.
- Establish competency centre in the field of Artificial Intelligence and Machine Learning withthe collaboration of industry and innovative research.

Program Educational Objectives (PEOs):

PEO1: Domain knowledge: Impart strong foundation in basic sciences, Mathematics, Engineering and emerging areas by Advanced tools and Technologies

PEO2: Professional Employment: Develop Professional skills that prepare them for immediateemployment in industry, government, entrepreneurship and Research.

PEO3: Higher Degrees: Pursue higher studies and acquire masters and research.

PEO4: Engineering Citizenship: Communicate and work effectively, engage in team work, achieveprofessional advancement, exhibit leadership skills, and ethical attitude with a sense of socialresponsibility.

PEO5: Life Long Learning: Lead in their field and respond to the challenges of an ever-changingenvironment with the most current knowledge and technology

(A) PROGRAM OUTCOMES(POs)

PO1:Engineering knowledge: Apply the knowledge of mathematics, science, engineeringfundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2:Problem analysis: Identify, formulate, review research literature, and analyze complexengineering 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 anddesign 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 researchmethods 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 modernengineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6:The engineer and society: Apply reasoning informed by the contextual knowledge to assesssocietal, 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 solutionsin societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8:Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms ofthe engineering practice.

PO9:Individual and team work: Function effectively as an individual, and as a member or leader indiverse teams, and in multidisciplinary settings.

PO10:Communication: Communicate effectively on complex engineering activities with the engineeringcommunity 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.

PO11:Project management and finance: Demonstrate knowledge and understanding of theengineering 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 ability to engage inindependent and life-long learning in the broadest context of technological change.

CSE[AI&ML]-PSO's

PSO 1	Understand and Apply Multi-Disciplinary and core concepts with emerging technologies for sustaining (endorse) with the Dynamic Industry Challenges.
PSO 2	Design Automated Applications in Machine Learning, Deep Learning,Natural Language Processing and Relevant Emerging areas for visualizing, interpreting the datasets.
PSO 3	Develop Computational Knowledge, project and Interpersonal skills using innovative tools for finding an elucidated solution of the real world problems and societal needs

3. List of experiments

SYLLABUS

22CS2252: DATABASE MANAGEMENT SYSTEMS LAB

B.Tech. II Year II Sem.	**L T P C**
	0 0 2 1

Co-requisites: “Database Management Systems”
Course Objectives: The course will explain in depth:
1. Introduce ER data model, database design and normalization
2. Learn SQL basics for data definition and data manipulation.
3. Design database schema for a given application and apply normalization
4. Acquire skills in using SQL commands for data definition and data manipulation.
5. Develop solutions for database applications using procedures, cursors and triggers.

Course Outcomes: At the end of the course student will be able to:
1. Design database schema for a given application and apply normalization.
2. Acquire skills using SQL commands for data definition and data manipulation.
3. Apply the normalization techniques for development of application software to realistic problems.
4. Develop solutions for database applications using procedures, cursors and triggers
5. Applying PL/SQL for processing database

LIST OF EXPERIMENTS:

1. Concept design with E-R Model
2. Relational Model
3. Normalization
4. Practicing DDL commands
5. Practicing DML commands
6. A. Querying (using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.) B. Nested, Correlated sub-queries
7. Queries using Aggregate functions, GROUPBY, HAVING and Creation and dropping of Views.
8. Triggers (Creation of insert trigger, delete trigger, update trigger)
9. Procedures
10. Usage of Cursors

TEXT BOOKS:
1. Database Management Systems, Raghurama Krishnan, Johannes Gehrke, Tata McGrawHill,3rd Edition
2. Database System Concepts, Silberschatz, Korth, McGrawHill, V edition.

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

2. Fundamentals of Database Systems, Elma sri 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.

INDEX

SNO	NAME OF THE PROGRAM	COURSE OUTCOMES	PAGE NO.
1	Concept design with E-R Model	CO1	10-13
2	Relational Model	CO1	14-16
3	Normalization	CO3	17-20
4	Practicing DDL commands	CO2	21-27
5	Practicing DML commands	CO2	28-34
6	A. Querying (using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.) B. Nested, Correlated subqueries	CO4	35-39
7	Queries using Aggregate functions, GROUP BY, HAVING and Creation and dropping of Views.	CO5	40-44
8	Triggers (Creation of insert trigger, delete trigger, update trigger)	CO4	45-50
9	Procedures	CO4	51-52
10	Usage of Cursors	CO5	53-55

Course Outcome mapping with POs

Program Name:	22CS2252: DATABASE MANAGEMENT SYSTEMS LAB	AY	2023-24
Course Name:	B.Tech,CSE[AI&ML]	Class / Sem	II/II
Faculty Name:	A. Gopya sri	Regulation	R 22

COURSE OUTCOMES (COs):

Upon completion of the course, students will be able to:

CO#	**Course Outcomes:** By the end of this course, Students will be able to
CO1	Design database schema for a given application and apply normalization
CO2	Acquire skills using SQL commands for data definition and data manipulation.
CO3	Apply the normalization techniques for development of application software to realistic problems
CO4	Develop solutions for database applications using procedures, cursors and triggers
CO5	Applying PL/SQL for processing database

CO-PO/PSO MATRIX: (Level of Mapping- 3: High; 2:Moderate; 1-Low; -:Not mapped)

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	2	2	3	1	3	1		2	3	2	2	3	2	3	3
CO2	2	2	2	1	3	1		2	3	2	2	3	3	2	3
CO3															
CO4	3	3	3	3	3	1		2	3	2	2	3	2	3	3
CO5															
	3	3	3	3	3	1		2	3	2	2	3	2	3	3

Revised Mapping considering the gaps

	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	2	2	3	1	3	1		2	3	2	2	3	2	3	3
CO2	2	2	2	1	3	1		2	3	2	2	3	3	2	3
CO3	3	3	3	3	3	1		2	3	2	2	3	2	3	3
CO4	3	3	3	3	3	1		2	3	2	2	3	2	3	3
CO5	3	3	3	3	3	1		2	3	2	2	3	2	3	3
	3	3	3	3	3	1		2	3	2	2	3	2	3	3

EXPERIMENT NO – 1

Concept design with E-R Model

Description: E-R Model

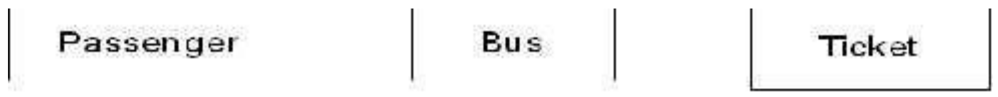
Analyze the problem carefully and come up with entities in it. Identify what date has to be persisted in the database. This contains the entities, attributes etc.

Identify the primary keys for all the entities. Identify the other keys like candidate keys, partial keys, if any.

Definitions:

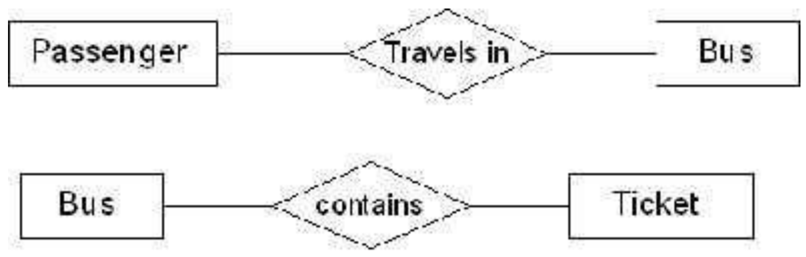
Entity: the object in the ER Model represents is an entity which is thing in the real world with an independent existence.

Eg:



ER-Model:

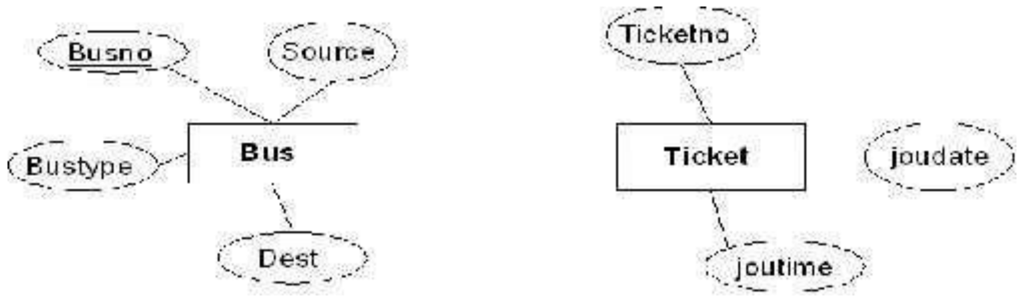
Describes data as entities, relationships and attributes .The ER-Model is important preliminary for its role in database design. ER Model is usually shown pictorially using entity relationship diagrams.



Attributes:

The properties that characterize an entity set are called its attributes. An attribute is referred to by the terms data items, data element, data field item.

Ex: attributes for bus entity and ticket entity.



Candidate key:

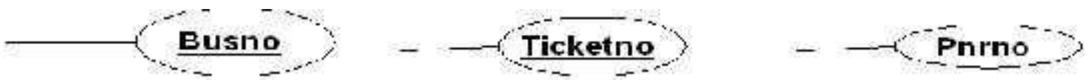
It can be defined as minimal super key or irreducible super key. In other words an attribute or combination of attributes that identifies the record uniquely but none of its proper subsets can identify the record uniquely.

Bus no	service no	source	destination	dep time	re time	bus type	no of seats

Bus no, service no-------------->candidate key

Primary key:

A candidate key that is used by the database designer for unique identification of each row in a table is known as primary key. A primary key consists of one or more attributes of the table.



Partial key:

A weak entity type normally has a partial key which is the set of attributes that can uniquely identify weak entity that are related to the same owner entity.

The entities in the “Roadway travels” is

1) Bus 2) Ticket 3) Passenger

Bus entity:

Attributes for the bus entity are

Bus no, service no, source, destination, dep time, re time, bus type, no.of.seats

Bus schema:

Busno	serviceno	source	destination	deptime	retime	bustype	noofseats

Busno,serviceno,source -----> super key

Busno,serviceno,bustype-----> super key

Busno,serviceno-------------->candidate key

Busno,serviceno--------------> primary key

Ticket entity:

Attributes for the ticket entity are

Ticketno, joudate, joutime, source, destination, seatno, amount, catcard

Ticket schema:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard

Ticketno, source, destination ------- >Super key

Ticketno, source, seatno -------------- > Super key

Ticketno, destination, seatno --------- > Super key

Ticketno----------- > candidate key

Ticketno----------- >primary key

Passenger entity:

Attributes for the Passenger entity are

Pnrno, pname, age, sex, ticketno, address, phno, catno

Passenger schema:

Pnrno	pname	age	sex	ticketno	address	phno	catno

Pnrno,pname-----
----- super key

Pnrno,ticketno---
------- super key

Pnrno,phno-------
--- super key

Pnrno -----
----- candidate key

primary key
Pnrno -----

Sample data for Bus entity:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

Sample data for Ticket entity:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	96	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	88	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	88	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	96	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	172	Yes

Sample data for Passenger entity:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

<u>EXPERIMENT NO –2</u>

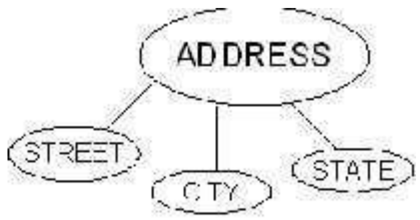
Relational Model

Description:

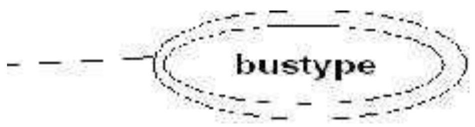
Represent all entities (strong, week) in tabular fashion. Represent relationships in a tabular fashion. There are different ways of representing as tables based on the cardinality. Represent attributes as columns in the tables or as tables based on the requirement. Different types of attributes (composite, multivalued and derived).

Definitions:

Composite attributes: can be divided into smaller sub parts which represent more basic attributes with independent meaning.

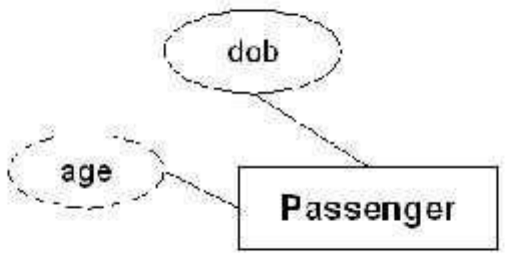


Multivalued attributes: for ex the attribute in the Bus entity Bustype can have different types of buses according that the Bustype attribute contains the values as Garuda, Luxury, Express, and Ordinary. This type of attribute is called multivalued attribute and may have lower and upper bounds to constrain the number of values allowed for each individual entity.



Derived attributes:

In some cases, two or more attribute values are related. With the help of one attribute we get the value of another attribute. Age and DOB attributes. With the DOB we get the age of the person to the current date.



Entity sets to tables:

Relational schema for Bus relation:

Bus(Busno:numeric, serviceno:numeric, source: varchar(10), destination :varchar(10), deptime:time, retime:time, bustype :varchar(10), noofseats :int)

Relational shema for Ticket relation:

Ticket(Ticketno:numeric, joudate:date, joutime:time, source:varchar(10), destination:varchar(10), seatno:int(4), amount:decimal(10,3), catcard:char(3))

Relational shema for Passenger relation:

Passenger(Pnrno:numeric, pname:varchar(15), age:int(4), sex:char(3), ticketno:numeric, address:varchar(50), phno:numeric(10), catno:varchar(10))

Relationship sets to tables:

Relational shema for reserve relation:

Rserves(pnrno:numeric, joudate:date, noofseats:int(4), address:varchar(50), contact_no:numeric(10) ,status:char(3))

Relational shema for Cancels relation:

Cancels(pnrno:numeric, joudate:date, noofseats:int(4), address:varchar(50 , contact_no:numeric(10),status:char(3))

TABLES:

Reserves:

Pnrno	Joudate	Noofseats	Address	Contact_no	Status
1001	2010-8-5	10	5-4,srpt	9492506282	Yes
1001	2010-8-15	5	5-4,srpt	9492506282	Yes
1002	2010-8-5	5	6-8,hyd	9949060540	Yes
1003	2010-8-15	6	h/7,vij	9704054050	Yes
1004	2010-8-18	8	8-9,hyd	9704613151	Yes
1005	2010-8-5	9	9-11,hyd	9848354941	No
1005	2010-8-6	5	9-11,hyd	9848354941	Yes

Cancels:

Pnrno	Joudate	Noofseats	Address	Contact_no	Status
1001	2010-8-5	5	5-4,srpt	9492506282	Yes
1002	2010-8-5	2	6-8,hyd	9949060540	No
1003	2010-8-15	2	h/7,vij	9704054050	Yes
1004	2010-8-18	5	8-9,hyd	9704613151	Yes
1005	2010-8-6	4	9-11,hyd	9848354941	Yes

Bus:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

Ticket:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	96	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	88	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	88	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	96	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	172	Yes

Passenger:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

EXPERIMENT NO–3

Normalization

Database normalization is a technique for designing relational database tables to minimize duplication of information and, in doing so , to safeguard the database against certain types of logical or structural problems namely data anomalies.

The normalization forms are:

1. First Normal Form: 1NF requires that the values in each column of a table are atomic. By atomic we mean that there are no sets of values within a column.
2. Second Normal Form: where the 1NF deals with atomicity of data, the 2NF deals with relationships between composite key columns and non-key columns. To achieve 2NF the tables should be in 1NF. The 2NF any non-key columns must depend on the entire primary key. In case of a composite primary key, this means that non-key column can't depend on only part of the composite key.
3. Third Normal Form: 3NF requires that all columns depend directly on the primary key. Tables violate the third normal form when one column depends an another column, which in turn depends on the primary key(transitive dependency). One way to identify transitive dependency is to look at your tables and see if any columns would require updating if another column in the table was updated. If such a column exists, it probably violates 3NF.

Let's normalize our entities:

Normalization of Passenger entity:

In the passenger entity there exists a passenger with two phone numbers, but atomic values should be there. So we normalize the relation as follows.

Passenger:

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282, 9848845985	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

Pnrno	pname	age	sex	ticketno	address	phno	Catno
1001	Subbu	31	M	1111	5-4,srpt	9492506282	Cap5112

1001	Subbu	31	M	1111	5-4,srpt	9848845985	Cap5112
1002	Achaith	22	M	2222	6-8,hyd	9949060540	Cap6900
1003	Padma	25	F	3333	h/7,vij	9704054050	Cap5772
1004	Ravi	23	M	4444	8-9,hyd	9704613151	Cap6132
1005	Satyam	42	F	5555	9-11,hyd	9848354941	Cap6732

The above relation is now in 1NF and the relation is 2NF as there are no partial functional dependencies and the relation is also in 3NF as there are no transitive dependencies.

Normalization of Bus entity:

Bus:

Busno	serviceno	source	destination	deptime	retime	bustype	Noofseats
Ap555	3889	Srpt	Hyd	9:00:00	19:15:00	Ac	36
Ap501	3891	Srpt	Hyd	10:00:15	20:15:00	Ac	36
Ap444	3601	Hyd	Srpt	9:00:00	19:30:00	Nonac	52
Ap891	3555	Hyd	Srpt	9:30:00	20:30:00	Nonac	52
Ap8830	3239	Hyd	Vij	9:00:00	22:30:00	Metro	45

In this relation the values in each column are atomic so it is already in 1NF.

In the Bus entity Busno+serviceno is the primary key.

There exists following partial dependencies.

Busno ----> Bustype,Noofseats

Serviceno---->Source,Dest

So the relation will be in 2NF as follows.

Busno	serviceno	deptime	retime
Ap555	3889	9:00:00	19:15:00
Ap501	3891	10:00:15	20:15:00
Ap444	3601	9:00:00	19:30:00
Ap891	3555	9:30:00	20:30:00
Ap8830	3239	9:00:00	22:30:00

Busno	bustype	Noofseats
Ap555	Ac	36
Ap501	Ac	36
Ap444	Nonac	52
Ap891	Nonac	52
Ap8830	Metro	45

serviceno	source	destination
3889	Srpt	Hyd
3891	Srpt	Hyd
3601	Hyd	Srpt
3555	Hyd	Srpt
3239	Hyd	Vij

The above relation is 2NF. And all columns directly depend on primary key. So there is no transitive dependency and the relation is 3NF.

Normalization of Ticket entity:

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Amount	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	96	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	88	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	88	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	96	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	172	Yes

In this relation the values in each column are atomic so it is already in 1NF.

In the above relation there are no partial functional dependencies so the relation is in 2NF. The ticket entity might face the following transitive dependency

Ticketno-----→ catcard

Catcard--------- >amount

So the relation is in 3NF.

Ticketno	Joudate	Joutime	Source	Destination	Seatno	Catcard
1111	2010-8-5	9:00:00	Srpt	Hyd	5	No
2222	2010-8-5	10:00:15	Srpt	Hyd	10	Yes
3333	2010-8-15	9:00:00	Hyd	Srpt	15	Yes
4444	2010-8-18	9:30:00	Hyd	Srpt	20	No
5555	2010-8-6	9:00:00	Hyd	Vij	18	Yes

Put the catcard and amount attributes in a separate table. Then the relation should be in 3NF.

Catcard	Amount
No	96
Yes	88
Yes	88
No	96
Yes	172

The above relation is 3NF as we have eliminated the transitive dependencies.

Finally all the tables are normalized and free from data redundancy, partial functional dependencies and transitive dependencies.

EXPERIMENT NO– 4

Practicing DDL commands

Creation of databases: mysql> show databases;

```
+--------------------+
| Database           |
+--------------------+

| information_schema |
| mysql              |
| test               |
+--------------------+
3 rows in set (0.09 sec) mysql> create database groupa;
Query OK, 1 row affected (0.01 sec) mysql> use groupa;
Database changed
```

Creation of tables:

```
mysql> create table groupamem(rollno numeric(10),name varchar(15),phone numeric(
10),branch varchar(10));
Query OK, 0 rows affected (0.42 sec)
mysql> desc
groupamem;
                                      +-
                                      ---
+--------+---------------     +------  -    +---------   +-------+
| Field  | Type               | Null | Key | Default | Extra |
                                      +-
                                      ---
+--------+---------------     +------  -    +---------   +-------+
| rollno | decimal(10,0) | YES     || NULL            |   |
| name  | varchar(15)      | YES   || NULL           |   |

| phone| decimal(10,0) | YES     || NULL           ||

| branch | varchar(10)     | YES   || NULL           |   |

                                     +-

                                     ---

+--------+---------------     +------  -    +---------   +-------+

4 rows in set (0.03 sec)
```

Altering the table:

```
mysql> alter table groupamem add gender char(3); Query OK, 0 rows affected (0.36 sec)

Records: 0 Duplicates: 0 Warnings: 0


mysql> desc groupamem;

                                     +-

                                     ---                  +-----

+--------+---------------     +-------     +---------      --   +

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

                                     +-

                                     ---                  +-----

+--------+---------------     +-------     +---------      --   +

| rollno | decimal(10,0) | YES   || NULL       |   |

| name  | varchar(15)  | YES      || NULL          ||

| phone| decimal(10,0) | YES    |     | NULL      ||

| branch | varchar(10)  | YES     || NULL          ||

| gender | char(3)        | YES|       | NULL  |      |
                                    +-
                                    ---                +-----
+--------+---------------    +-------    +---------     --  +
5 rows in set (0.00 sec)
```

Dropping the table:

```
mysql> create table dd(name varchar(10)); Query OK, 0 rows affected (0.09 sec)

mysql> show tables;

+-----------------         +

| Tables_in_groupa |

+-----------------         +

| dd              |

| groupamem         |

+-----------------         +

2 rows in set (0.00 sec)

mysql> drop table dd;

Query OK, 0 rows affected (0.06 sec)
```

Dropping the database:

```
mysql> create database dbl;

Query OK, 1 row affected (0.01 sec)

mysql> show databases;

+--------------------+

| Database          |

+--------------------+

| information_schema |

| dbl                |

| groupa             |
| mysql              |

| test               |

+------------------- +

6 rows in set (0.01 sec)

mysql> drop database dbl;

Query OK, 0 rows affected (0.00 sec)

mysql> show databases;

+--------------------+

| Database           |

+--------------------+

| information_schema |

| groupa             |

| mysql              |

| test               |

+------------------- +

5 rows in set (0.00 sec)
```

Rename the tables:

```
mysql> rename table groupamem to ga; Query OK, 0 rows affected (0.03 sec)

mysql> show tables;

+------------------+

| Tables_in_groupa |

+------------------+

| ga               |

+------------------+

1 row in set (0.00 sec)
```

Truncate the table:

```
mysql> insert into ga values(1111,'ram',9885321456,'mbbs','m'); Query OK, 1 row affected
(0.06 sec)


mysql> select * from ga;

+-
---
---
-      +------    +------------ +-------- +--------  +

  | rollno | name | phone   | branch | gender |

+-
---
---
-      +------    +------------ +-------- +--------  +

|   1111 | ram  | 9885321456 | mbbs  | m       |

+-
---
---
-      +------    +------------ +-------- +--------  +

1 row in set (0.01 sec)

mysql> truncate table ga;

Query OK, 1 row affected (0.09 sec) mysql> select * from ga;

Empty set (0.00 sec)
```

Creation of tables for Roadway Travels:

<u>Bus</u>

```
mysql> create table bus555(busno varchar(10),bustype varchar(10),primary key(bus no));

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

Ticket

```
mysql> create table ticket555(tic_no numeric(10),joudate date,source varchar(10) ,dest varchar(10),deptime time,reatime time,busnumber varchar(10),primary key(ti c_no));

Query OK, 0 rows affected (0.08 sec)


mysql> alter table ticket555 add constraint tic_fk foreign key(busnumber) refere nces bus555(busno);

Query OK, 0 rows affected (0.16 sec) Records: 0 Duplicates: 0 Warnings: 0
```

Passenger

```
mysql> create table passenger(pnrno numeric(10),ticnumber numeric(10),pname varc har(15),age int(4),sex char(10),ppno varchar(15),primary key(pnrno));

Query OK, 0 rows affected (0.06 sec)


mysql> alter table passenger add constraint pas_fk foreign key(ticnumber) refere nces ticket555(tic_no);

Query OK, 0 rows affected (0.14 sec)

Records: 0 Duplicates: 0 Warnings: 0
```

Reserve

```
mysql> create table reserve(pnrnumber numeric(10),noofseats int(8),address varch ar(50),phno numeric(10),status char(3));

Query OK, 0 rows affected (0.16 sec)


mysql> alter table reserve add constraint res_fk foreign key(pnrnumber) referenc es passenger(pnrno);

Query OK, 0 rows affected (0.17 sec) Records: 0 Duplicates: 0 Warnings: 0
```

Cancel

```
mysql> create table cancel(pnrnumber numeric(10),noofseats int(8),address varcha
r(50),phno numeric(10),status char(3));

Query OK, 0 rows affected (0.06 sec)


mysql> alter table cancel add constraint can_fk foreign key(pnrnumber) reference s
passenger(pnrno);

Query OK, 0 rows affected (0.14 sec) Records: 0 Duplicates: 0 Warnings: 0
```

EXPERIMENT NO – 5

Practicing DML commands

DML commands are used to for managing data within the schema objects.

Use of insert command:

Inserting values into *Bus* table:

```
mysql> insert into bus555 values('ap555','ac'); Query OK, 1 row affected (0.03 sec)


mysql> insert into bus555 values('ap501','ac'); Query OK, 1 row affected (0.03 sec)

mysql> insert into bus555 values('ap444','nonac'); Query OK, 1 row affected (0.03 sec)


mysql> insert into bus555 values('ap891','nonac'); Query OK, 1 row affected (0.03 sec)


mysql> insert into bus555 values('ap8830','metro'); Query OK, 1 row affected (0.03 sec)
```

Inserting values into *Ticket* table:

```
mysql> insert into ticket555 values(1111,'2010-08-05','srpt','hyd','09:00:05','1
9:15:00','ap555');

Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(2222,'2010-08-05','srpt','hyd','10:00:05','2
0:15:00','ap501');

Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(3333,'2010-08-15','hyd','srpt','09:00:05','2
0:15:00','ap444');

Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(4444,'2010-08-18','hyd','srpt','09:30:05','2
0:15:00','ap891');
```

```
Query OK, 1 row affected (0.03 sec)

mysql> insert into ticket555 values(5555,'2010-08-8','hyd','vij','09:10:05','22:
15:00','ap8830');

Query OK, 1 row affected (0.03 sec)
```

Inserting values into *Passenger* table:

```
mysql> insert into passenger values(1001,1111,'subbu',31,'m','pp555'); Query OK, 1 row
affected (0.05 sec)


mysql> insert into passenger values(1002,2222,'achaith',22,'m','pp8830'); Query OK, 1 row
affected (0.03 sec)


mysql> insert into passenger values(1003,3333,'padma',25,'f','pp333'); Query OK, 1 row
affected (0.05 sec)


mysql> insert into passenger values(1004,4444,'ravi',23,'m','pp444'); Query OK, 1 row
affected (0.01 sec)


mysql> insert into passenger values(1005,5555,'nirma',42,'f','pp666'); Query OK, 1 row
affected (0.03 sec)
```

Inserting values into *reserve* table:

```
mysql> insert into reserve values(1001,10,'hno:5-4,srpt,nlg',9492506282,'yes'); Query OK, 1
row affected (0.05 sec)

mysql> insert into reserve values(1001,5,'hno:5-4,srpt,nlg',9492506282,'yes'); Query OK, 1
row affected (0.03 sec)

mysql> insert into reserve values(1002,5,'hno:15-4,lbngr,hyd',9491653714,'yes'); Query OK,
1 row affected (0.01 sec)
```

```
mysql> insert into reserve values(1003,6,'hno:151-4,dsnr,hyd',9704613151,'yes'); Query OK, 1 row affected (0.03 sec)

mysql> insert into reserve values(1004,8,'hno:11-4,dsnr,hyd',9704613111,'yes'); Query OK, 1 row affected (0.02 sec)

mysql> insert into reserve values(1005,8,'hno:41-4,dsnr,hyd',9989503111,'no'); Query OK, 1 row affected (0.03 sec)

mysql> insert into reserve values(1005,5,'hno:41-4,dsnr,hyd',9989503111,'yes'); Query OK, 1 row affected (0.02 sec)
```

Inserting values into *cancel* table:

```
mysql> insert into cancel values(1001,5,'hno:5-4,srpt,nlg',9492506282,'yes'); Query OK, 1 row affected (0.05 sec)

mysql> insert into cancel values(1001,2,'hno:5-4,srpt,nlg',9492506282,'yes'); Query OK, 1 row affected (0.03 sec)

mysql> insert into cancel values(1002,2,'hno:15-4,lbngr,hyd',9491653714,'no'); Query OK, 1 row affected (0.05 sec)

mysql> insert into cancel values(1003,2,'hno:151-4,dsnr,hyd',9704613151,'yes'); Query OK, 1 row affected (0.01 sec)

mysql> insert into cancel values(1004,5,'hno:11-4,dsnr,hyd',9704613111,'yes'); Query OK, 1 row affected (0.03 sec)

mysql> insert into cancel values(1005,4,'hno:41-4,dsnr,hyd',9989503111,'yes'); Query OK, 1 row affected (0.03 sec)
```

<u>Use of select command:</u>

```
mysql> select *from bus555;

+--------+--------- +

| busno  | bustype |

+--------+--------- +
| ap444  | nonac   |

| ap501  | ac          |

| ap555  | ac          |

| ap8830 | metro   |

| ap891  | nonac     |

+--------+--------- +

5 rows in set (0.00 sec)


mysql> select *from ticket555;

+-------- +------------      + --------+------ +---------- +---------- +----------- +

| tic_no | joudate            | source | dest | deptime             | reatime| busnumber |

+-

--

--                                                        +--

--                                                        ----            +-----

-                         +-----------  +--------    +------ ----             -----         +-----------   +

|  1111 | 2010-08-05 | srpt                  | hyd   | 09:00:05 | 19:15:00 | ap555      |

|  2222 | 2010-08-05 | srpt                  | hyd   | 10:00:05 | 20:15:00 | ap501      |

|  3333 | 2010-08-15 | hyd                   | srpt | 09:00:05 | 20:15:00 | ap444      |

|  4444 | 2010-08-18 | hyd                   | srpt | 09:30:05 | 20:15:00 | ap891      |

|  5555 | 2010-08-08 | hyd                    | vij  | 09:10:05 | 22:15:00 | ap8830    |

+-

--

--                                                        +--

--                                                        ----            +-----

-           +------------             +--------    +------ ----             -----         +-----------   +

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

```
mysql> select *from passenger;

+
--                                    +
--                                    --
--                                    --      +--
- +----------- ---------           +  --      ---- +--------      +

| pnrno | ticnumber | pname            | age | sex | ppno          |

+
--                                    +
--                                    --
--                                    --      +--
- +----------- ---------           +  --      ---- +--------      +

|            1001 |1111     | subbu   |  31 | m     | pp555 |
|            1002 |2222     | achaith |  22 | m    | pp8830 |
|            1003 |3333     | padma   |  25 | f     | pp333 |
|            1004 |4444     | ravi    |23 | m    | pp444   |
|            1005 |5555     | nirma   |  42 | f    | pp666   |

+
--                                    +
--                                    --
--                                    --      +--
- +----------- ---------           +  --      ---- +--------      +

5 rows in set (0.03 sec)

mysql> select *from reserve;

+
--
--
--                            +----------          +----------    +-------
--         -----------+       ----------                 -    -              +
--
-
| pnrnumber | noofseats | address                          | phno| status |
+
--
--
--
--
--                                   +----------             +-----------     +-------
-             -----------+           ----------                      -    -                    +
  |      1001 |10 | hno:5-4,srpt,nlg                 | 9492506282 | yes       |
  |      1001 |     5| hno:5-4,srpt,nlg  | 9492506282 | yes                    |
  |      1002 |     5| hno:15-4,lbngr,hyd | 9491653714 | yes                  |
  |      1003 |     6| hno:151-4,dsnr,hyd | 9704613151 | yes                  |
  |      1004 |     8| hno:11-4,dsnr,hyd               | 9704613111| yes      |
  |      1005 |     8| hno:41-4,dsnr,hyd               | 9989503111| no       |
  |      1005 |     5| hno:41-4,dsnr,hyd               | 9989503111| yes      |
+-----------          +-----------   +-------------------- +------------+--------  +
7 rows in set (0.02 sec)
mysql> select *from cancel;
+-----------          +-----------   +-------------------- +------------+--------  +
| pnrnumber | noofseats | address                           | phno| status |
+-----------          +-----------   +-------------------- +------------+--------  +
|            1001 | 5| hno:5-4,srpt,nlg                | 9492506282 | yes       |
|            1001 | 2| hno:5-4,srpt,nlg                | 9492506282 | yes       |
|            1002 | 2| hno:15-4,lbngr,hyd | 9491653714 | no                    |
|            1003 | 2| hno:151-4,dsnr,hyd | 9704613151 | yes                   |
|              1004 | 5| hno:11-4,dsnr,hyd               | 9704613111 | yes      |

|              1005 | 4| hno:41-4,dsnr,hyd               | 9989503111 | yes      |

+-----
------   +-----------   +-------------------                         +------------+--------   +
```

6 rows in set (0.00 sec)

Use of update command:

```
mysql> update passenger set ppno=’pp888’ where pnrno=1001; Query OK, 1 row affected
(0.03 sec)

Rows matched: 1 changed: 1 warnings: 0
```

Use of DELETE command:

```
mysql>delete from cancel where status=’no’;

Query OK, 1 row affected (0.03 sec)

mysql> select *from cancel;

+----------- +-----------+--------------------+------------+-------- +

| pnrnumber | noofseats | address                  | phno| status |

+----------- +-----------+--------------------+------------+-------- +

|              1001 | 5| hno:5-4,srpt,nlg    | 9492506282 | yes        |

|              1001 | 2| hno:5-4,srpt,nlg    | 9492506282 | yes        |

|              1003 | 2| hno:151-4,dsnr,hyd | 9704613151 | yes      |

|              1004 | 5| hno:11-4,dsnr,hyd  | 9704613111 | yes      |

|              1005 | 4| hno:41-4,dsnr,hyd  | 9989503111 | yes      |

+-----
------   +-----------   +-------------------          +------------+--------   +

6 rows in set (0.00 sec)
```

EXPERIMENT NO – 6

A.Querying (using ANY, ALL, UNION, INTERSECT, JOIN, Constraints etc.)
B. Nested, Correlated subqueries

Practice the following queries:

1. Display unique PNR_no of all passengers.

```
mysql> select distinct pnrno from passenger; +-------+

| pnrno | +-------+ | 1001 | | 1002 | | 1003 | | 1004 | | 1005 | +-------+


5 rows in set (0.01 sec)
```

2. Display all the names of male passengers.

```
mysql> select pname from passenger where sex='m';

+--------- +

| pname   |

+--------- +

| subbu    |

| achaith |

| ravi  |

+--------- +

3 rows in set (0.00 sec)
```

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

```
mysql> select tic_no,pname from ticket555 t,passenger p where t.tic_no=p.ticnumber;
```

```
+-
--          +--------- +
 | tic_no| pname    |
+-
---
---
-           +--------- +
|    1111| subbu     |
|    2222| achaith |
|    3333| padma    |
|    4444| ravi  |
|    5555| nirma     |
+-
---
---
-           +--------- +
5 rows in set (0.00 sec)
```

4. Display the source and destination having journey time more than 10 hours.

```
mysql> select source,dest from ticket555 where hour(timediff(reatime,deptime))>10;
+-------- +------    +
| source | dest |
+-------- +------    +
| hyd      | srpt |
| hyd    | vij       |
+-------- +------    +
2 rows in set (0.00 sec)
```

5. Find the ticket numbers of passengers whose name starts with ‘A’ and ends with ‘H’.

```
mysql> select tic_no from ticket555 where tic_no=any (select ticnumber from passenger
where pname like 'a%h');

+-
---
---
-       +
 | tic_no |
+-
---
---
-       +
|    1111 |
|    2222 |
+--------+
2 rows in set (0.02 sec)
```

6. Find the name of passengers whose age is between 30 and 45.

```
mysql> select pname from passenger where age between 30 and 45; +-------+
| pname | +-------+ | subbu | | nirma | +-------+
2 rows in set (0.00 sec)
```

7. Display all the passengers names beginning with 'A'.

```
mysql> select all pname from passenger where pname like 'a%';
+--------- +
| pname   |
+--------- +
| subbu    |
| achaith |
+--------- +
2 rows in set (0.00 sec)
```

8. Display the sorted list of passengers names.

```
mysql> select pname from passenger order by pname;
+--------- +
| pname   |
 +---------+
| subbu    |
| achaith |
| nirma     |
| padma   |
| ravi  |
+--------- +
5 rows in set (0.02 sec)
```

9. Display the Bus numbers that travel on Sunday and Wednesday.

```
mysql> select busno from bus555 where busno in(select busnumber from ticket555 where
dayofweek(joudate)=1 or dayofweek(joudate)=4);
+--------+ | busno | +--------+ | ap444 | | ap8830 | | ap891 | +--------+
3 rows in set (0.00 sec)
```

10. Display the details of passengers who are traveling either in AC or NON_AC.

```
mysql> select pname,pnrno,age,sex from passenger where ticnumber in(select tic_no from
ticket555 where busnumber in(select busno from bus555 where bustype='ac' or
bustype='nonac'));

            +
            --
            --
            --              +----
+--------- -   +------     --        +
| pname   | pnrno | age    | sex |
            +
            --
            --
            --              +----
+--------- -   +------     --        +
| subbu   |    1001 | 31| m       |
| achaith |     1002 | 22 | m       |
| padma   |    1003 |  25 | f       |
| ravi  |   1004 |  23 | m           |
            +
            --
            --
            --              +----
+--------- -   +------     --        +
4 rows in set (0.00 sec)
```

EXPERIMENT NO– 7

Queries using Aggregate functions, GROUP BY, HAVING and Creation and dropping of Views.

1. Write a query to display the information present in the Passenger and Cancellation tables.

```
mysql> select pnrno from passenger union select pnrnumber from cancel; +-------+

| pnrno | +-------+ | 1001 | | 1002 | | 1003 | | 1004 | | 1005 | +-------+


5 rows in set (0.00 sec)
```

2. Write a query to display the busnumber with source and destination available in Roadway Travels.

```
mysql> select busno,source,dest from bus555,ticket555 where busno=busnumber grou p by
busnumber;

+--------+--------+------    +

| busno  | source | dest |

+--------+--------+------    +

| ap444  | hyd    | srpt |

| ap501  | srpt   | hyd  |

| ap555  | srpt   | hyd  |

| ap8830 | hyd  | vij    |

| ap891  | hyd    | srpt |

+--------+--------+------    +

5 rows in set (0.00 sec)
```

3. Display the number of days in a week on which AP444 bus is available.

```
mysql> select count(joudate) from ticket555 where busnumber in(select busno from bus555
where busno='ap444') and joudate between '2010-08-14' and '2010-08-20'; +----------------+

| count(joudate) |

+----------------+

|             2 |

+----------------+

1 row in set (0.00 sec)
```

4. Find the ticket numbers booked for each PNR_no using Group By clause.

```
mysql> select sum(noofseats) as reserved_seats,pnrnumber from reserve where stat us='yes'
group by pnrnumber;

+----------------+-----------+

| reserved_seats | pnrnumber |

+-----------
----                  +-----------+

|             15 |        1001 |

|              5 |        1002 |

|              6 |        1003 |

|              8 |        1004 |

|              5 |        1005 |

+-----------
----                  +-----------+

5 rows in set (0.33 sec)
```

5. Find the distinct PNR numbers that are present.

```
mysql> select distinct pnrnumber from reserve; +-----------+

| pnrnumber |

+-----
------          +

|        1001 |

|        1002 |

|        1003 |

|        1004 |

|        1005 |

+-----
------          +

5 rows in set (0.00 sec)
```

6. Find the number of tickets booked for each bus with bustype where the number of seats is greater than 1.

```
mysql> select busno,bustype,sum(noofseats) as booked_seats from bus555,reserve,t
icket555,passenger where busno=busnumber and tic_no=ticnumber and pnrno=pnrnumbe r
and status='yes'group by tic_no having count(*)>=1;

                                        -----
                                        -----
                                        ----
+--------+---------+                    +

| busno  | bustype | booked_seats |

+--------+---------+--------------   +

| ap555  | ac                       |15|

| ap501  | ac                       |5|

| ap444  | nonac                    |6   |

| ap891  | nonac               |      8 |
| ap8830 | metro     |      5 |
                  +-----------
+--------+---------          --       +
5 rows in set (0.00 sec)
```

7. Find the total number of cancelled seats.

```
mysql> select sum(noofseats) as cancelled_seats from cancel where status='yes'; +-------------
----+
| cancelled_seats |
+-----------------+
|            18 |
+-----------------+
1 row in set (0.00 sec)
```

8. Write a query to count the number of tickets for the buses which traveled after the date '2010-08-06'.

```
mysql> select busno,bustype,sum(noofseats) as booked_seats from bus555,reserve,t
icket555,passenger where busno=busnumber and tic_no=ticnumber and pnrno=pnrnumbe r
and status='yes'and joudate>'2010-8-6' group by tic_no having count(*)>=1;

                  +-----------
+--------+---------          --          +
| busno  | bustype | booked_seats |
                  +-----------
+--------+---------          --          +
| ap444  | nonac     |         6 |
| ap891  | nonac     |         8 |
| ap8830 | metro     |         5 |
                    +-----------
+--------+---------          --          +
```

3 rows in set (0.01 sec)

Creation of Views:

```
mysql> create view takes1 as
   -> select tic_no,pname from ticket555,passenger where tic_no=ticnumber; Query OK, 0
rows affected (0.44 sec)


mysql> select tic_no from takes1;
+-
---
---
-          +
 | tic_no |
+-
---
---
-          +
|    1111 |
|    2222 |
|    3333 |
|    4444 |
|    5555 |
+--------+
5 rows in set (0.05 sec)
```

Dropping of Views:

```
mysql> drop view takes1;
```

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

EXPERIMENT NO – 8

Triggers (Creation of insert trigger, delete trigger, update trigger)

A Trigger is a named database object which defines some action that the databaseshould take when some databases related event occurs. Triggers are executed when you issues a data manipulation command like INSERT, DELETE, UPDATE on a table for which the trigger has been created. They are automatically executed and also transparent to the user. But for creating the trigger the user must have the CREATE TRIGGER privilege. In this section we will describe you about the syntax to create and drop the triggers and describe you some examples of how to use them.

CREATE TRIGGER

The general syntax of CREATE TRIGGER is : CREATE TRIGGER trigger_name trigger_time trigger_event ON tbl_name FOR EACH

ROW trigger_statement

By using above statement we can create the new trigger. The trigger can associate only with the table name and that must be refer to a permanent table. Trigger_time means trigger action time. It can be BEFORE or AFTER. It is used to define that the trigger fires before or after the statement that executed it. Trigger_event specifies the statement that executes the trigger. The trigger_event can be any of the DML Statement : INSERT, UPDATE, DELETE.

We can not have the two trigger for a given table, which have the same trigger action time and event. For Instance : we cannot have two BEFORE INSERT triggers for same table. But we can have a BEFORE INSERT and BEFORE UPDATE trigger for a same table.

Trigger_statement have the statement that executes when the trigger fires but if you want to execute multiple statement the you have to use the BEGIN…END compound statement.

We can refer the columns of the table that associated with trigger by using the OLD and NEWkeyword. OLD.column_name is used to refer the column of an existing row before it is deleted or updated and NEW.column_name is used to refer the column of a new row that is inserted or after updated existing row.

In INSERT trigger we can use only NEW.column_name because there is no old row and in a DELETE trigger we can use only OLD.column_name because there is no new row. But in UPDATE trigger we can use both, OLD.column_name is used to refer the columns of a row before it is updated and NEW.Column_name is used to refer the column of the row after it is updated.

Update Trigger:

```
mysql> create trigger t1 before update on reserve -> for each row
    -> begin
    -> if new.noofseats>30 then
    -> set new.noofseats=old.noofseats; -> else
    -> set new.noofseats=new.noofseats; -> end if;
    -> end//
    Query OK, 0 rows affected (0.03 sec)
mysql> update reserve set noofseats=12 where pnrnumber=1004; -> //
    Query OK, 1 row affected (0.01 sec) Rows matched: 1 Changed: 1 Warnings: 0
    mysql> select *from reserve;
                      -> //
    +--
    ----
    ----
    -            +-----------+-------------------- +------------+-------- +
    | pnrnumber | noofseats | address                | phno| status |
    +--
    ----
    ----
    -            +-----------+-------------------- +-----------+-------- +
|            1001 |10 | hno:5-4,srpt,nlg   | 9492506282 | yes        |
|            1001 | 5| hno:5-4,srpt,nlg  | 9492506282 | yes  |
|            1002 |20 | hno:15-4,lbngr,hyd | 9491653714 | yes    |
|            1003 | 6| hno:151-4,dsnr,hyd | 9704613151 | yes     |
|            1004 |12 | hno:11-4,dsnr,hyd | 9704613111  | yes     |
|            1005 | 8| hno:41-4,dsnr,hyd    | 9989503111| no       |
|            1005 | 5| hno:41-4,dsnr,hyd    | 9989503111| yes     |
                                                              +-
                                                              ---
+-----                                                        ---
------  +-----------   +-------------------        +------------   -          +
7 rows in set (0.00 sec)
mysql> update reserve set noofseats=32 where pnrnumber=1004// Query OK, 0 rows affected
(0.00 sec)
Rows matched: 1 Changed: 0 Warnings: 0
mysql> select *from reserve//
+----------- +-----------+-------------------- +------------+-------- +
| pnrnumber | noofseats | address                  | phno| status |
+----------- +-----------+-------------------- +------------+-------- +
|            1001 |10 | hno:5-4,srpt,nlg   | 9492506282 | yes        |
|            1001 | 5| hno:5-4,srpt,nlg  | 9492506282 | yes  |
|            1002 |20 | hno:15-4,lbngr,hyd | 9491653714 | yes    |
|            1003 | 6| hno:151-4,dsnr,hyd | 9704613151 | yes     |
|            1004 |12 | hno:11-4,dsnr,hyd | 9704613111  | yes     |
|            1005 | 8| hno:41-4,dsnr,hyd    | 9989503111| no       |
|            1005 | 5| hno:41-4,dsnr,hyd    | 9989503111| yes     |
                                                                   +-
                                                                   ---
+-----                                                             ---
------  +-----------    +-------------------        +------------   -          +
```

Insert Trigger:

```
mysql> create trigger t3
-> before insert on passenger -> for each row
-> begin
-> if new.age>18 then
-> set new.ppno='pp8630'; -> else
-> set new.ppno=''; -> end if;
-> end//
Query OK, 0 rows affected (0.06 sec)
mysql> insert into passenger values(1009,5555,'mm','17','m','99')// Query OK, 1 row affected
(0.03 sec)
mysql> select * from passenger//
+
--
--                          +-
--                          ---            +------
-  +-----------     +---------  --   +------        --    +
| pnrno | ticnumber | pname   | age | sex    | ppno  |
                            +-
+
                            ---            +------
--
-- +-----------     +---------  --   +------        --    +
--
-
|       1001 | 1111| subbu   |   31 | m         | pp555 |
|       1002 | 2222| achaith |   22 | m       | pp8830 |
|       1003 | 3333| padma  |   25 | f      | pp333       |
|       1004 | 4444| ravi  |    23 | m        | pp444       |
|       1005 | 5555| nirma   |   42 | f      | pp666       |
|       1009 | 5555| mm      |   17 | m      |       |
+
--
--                                +-
--                                ---                 +-----
-  +-----------        +---------  --    +------             --     +
6 rows in set (0.00 sec)
```

<u>Delete Trigger:</u>

```
mysql> delimiter //
mysql> create trigger rc before delete on cancel -> for each row
-> begin
-> insert into reserve values(old.pnrnumber,old.noofseats,old.address,old.phno,old.status); ->
end//
Query OK, 0 rows affected (0.05 sec)
mysql> delete from cancel where pnrnumber=1003// Query OK, 1 row affected (0.08 sec)
mysql> select *from reserve//
+----------- +-----------+-------------------- +------------+-------- +
| pnrnumber | noofseats | address                  | phno| status |
+----------- +-----------+-------------------- +------------+-------- +
  |       1001 |10 | hno:5-4,srpt,nlg          | 9492506282 | yes    |
  |       1001 |      5    | hno:5-4,srpt,nlg  | 9492506282 | yes  |

  |       1002 |     20 | hno:15-4,lbngr,hyd | 9491653714 | yes    |

  |       1003 |      6  | hno:151-4,dsnr,hyd | 9704613151 | yes    |

  |       1004 |29 | hno:11-4,dsnr,hyd | 9704613111        | yes    |

  |       1005 |      8| hno:41-4,dsnr,hyd     | 9989503111| no       |

  |       1005 |      5| hno:41-4,dsnr,hyd     | 9989503111| yes      |

  |       1003 |      2| hno:151-4,dsnr,hyd | 9704613151    | yes    |

+-----
------   +-----------  +-------------------         +------------+--------+

8 rows in set (0.00 sec)


mysql> select *from cancel//

+-----
------   +-----------  +-------------------         +------------+--------  +

| pnrnumber | noofseats | address                  | phno    | status |

+-----
------   +-----------  +-------------------         +------------+--------  +

|              1001 | 5| hno:5-4,srpt,nlg        | 9492506282 | yes    |

|              1001 | 2| hno:5-4,srpt,nlg        | 9492506282 | yes    |

|              1004 | 5| hno:11-4,dsnr,hyd | 9704613111 | yes      |

|              1002 | 2| hno:15-4,lbnr,hyd | 9491653714 | no    |

|              1005 | 2| hno:41-4,dsnr,hyd | 9989503111 | yes      |

+-----
------   +-----------  +-------------------         +------------+--------  +

5 rows in set (0.00 sec)
```

EXPERIMENT NO – 9

Procedures

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

A stored procedure is a procedure (like a subprogram in a regular computing language) that is stored (in the database). Correctly speaking, MySQL supports "routines" and there are two kinds of routines: stored procedures which you call, or functions whose return values you use in other SQL statements the same way that you use pre-installed MySQL functions like pi(). I'll use the word "stored procedures" more frequently than "routines" because it's what we've used in the past, and what people expect us to use.

```
mysql> create procedure p2(p_age int) -> begin

 -> select pname,ticnumber,sex from passenger where age>p_age; -> end//

 Query OK, 0 rows affected (0.00 sec)

 mysql> call p2(30)//

         +------
 +-------   -----     +------    +

 | pname | ticnumber | sex  |

         +------
 +-------   -----     +------    +

 | subbu      |  1111 | m     |
 | nirma |          5555 | f      |

             +------
 +-------   -----      +------    +

 2 rows in set (0.00 sec)


 mysql> call p2(24)//

 +-------+------      +------    +

 | pname | ticnumber | sex  |

 | padma |        3333 | f      |

 | nirma |          5555 | f      |

             +------
 +-------   -----      +------    +

 3 rows in set (0.00 sec)

 Query OK, 0 rows
 affected (0.00 sec)

 /sql> create procedure
 p3() -> begin

 -> select source,dest
 from ticket555 where
 hour(timediff(reatime,de
 ptime))>1

 0;

 -> end//

 Query OK, 0 rows
 affected (0.02 sec)
 mysql> call p3()//

 +--------+------    +

 | source | dest |
 +-------- +------    +

 | hyd      | srpt |

 | hyd    | vij       |

 +-------- +------    +

 2 rows in set (0.00 sec)

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

EXPERIMENT NO – 10

Usage of Cursors

In this week you need to do the following: Declare the cursor that defines the 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 are used when the SQL Select statement is expected to return more than one row. Cursors are supported inside procedures and functions. Cursors must be declared and its definition contains the query. The cursor must be defined in the DECLARE section of the program. A cursor must be opened before processing and close after processing.

Syntax to declare the cursor:

DECLARE <cursor_name> CURSOR FOR <select_statement>

Multiple cursors can be declared in the procedures and functions but each cursor must have a unique name. And in defining the cursor the select_statement cannot have INTO clause.

Syntax to open the cursor :

OPEN <cursor_name>

By this statement we can open the previously declared cursor.

Syntax to store data in the cursor :

FETCH <cursor_name> INTO <var1>,<var2>…….

The above statement is used to fetch the next row if a row exists by using the defined open cursor.

Syntax to close the cursor : CLOSE <cursor_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.

```
mysql> create procedure mycur1(pa_id int) -> begin

-> declare v_id int;

-> declare v_name varchar(30);


-> declare c1 cursor for select pnrno,pname from passenger where pnrno=pa_id;

-> open c1;

-> fetch c1 into v_id,v_name; -> select v_id,v_name;

-> close c1; -> end//

Query OK, 0 rows affected (0.00 sec)


mysql> call mycur1(1001)// +------+--------+

| v_id | v_name | +------ +--------+ | 1001 | subbu | +------+--------+


1 row in set (0.01 sec)

Query OK, 0 rows affected (0.01 sec) mysql> create procedure mycur5(ti_id int)


-> begin

-> declare x_no int;

-> declare x_src varchar(30); -> declare x_dst varchar(30);
-> declare c2 cursor for select tic_no,source,dest from ticket555 where tic_ no=ti_id;

-> open c2;

-> fetch c2 into x_no,x_src,x_dst; -> select x_no,x_src,x_dst;


-> close c2; -> end//

Query OK, 0 rows affected (0.00 sec)


mysql> call mycur5(1111)// +------+-------+-------+

| x_no | x_src | x_dst | +------ +-------+-------+ | 1111 | srpt | hyd | +------+-------+-------+ 1 row
in set (0.00 sec)

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