NRI INSTITUTE OF INFORMATION SCIENCE & TECHNOLOGY BHOPAL

DEPARTMENT OF MASTER OF COMPUTER APPLICATION

LAB MANUAL

PROG. LAB. IN RDBMS (MCA-207)

MASTER OF COMPUTER APPLICATION (MCA)

Experiment - One

(a) Objective: Draw database & types of database using Smart Draw tool.

(b) Description:

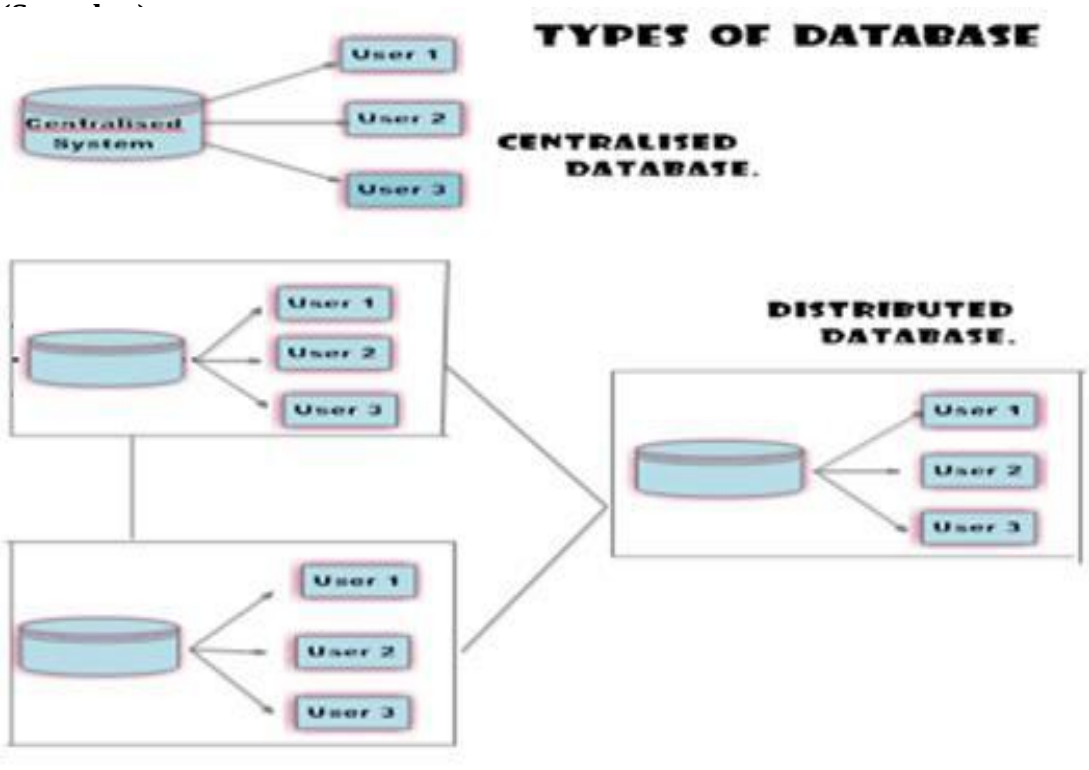
A **database** is an organized collection of data. The data is typically organized to model aspects of reality in a way that supports processes requiring information. Formally, a "database" refers to a set of related data and the way it is structured or organized. Access to this data is usually provided by a "database management system" (DBMS) consisting of an integrated set of computer software that allows users to interact with one or more databases and provides access to all of the data contained in the database (although restrictions may exist that limit access to particular data). The DBMS provides various functions that allow entry, storage and retrieval of large quantities of information as well as provide ways to manage how that information is organized

Large commercial databases may exist in two different Topologies.

- Centralised - where the database is physically in one location and users typically use an Internet connection to access it. Banks (such as ANZ) tend to use centralised databases.
- Distributed - Where the database is in many locations often where you have a national or international company and customers tend to regularly interact with a local branch. For example: Google uses Big-Table a distributed DBMS as searching tends to be by users in a particular region of the world.

In both cases the database "looks" like one database

(c) Result



Experiment - Two

(a) Objective: Draw 2 Tier- 3 Tier Architecture with Smart Draw tool.

(b) Description:

Two-Tier Architecture:

The two-tier architecture is like client server application. The direct communication takes place between client and server. There is no intermediate between client and server.

1. Database (Data tier)
2. Client Application (Client tier)

So, in client application the client writes the program for saving the record in SQL Server and thereby saving the data in the database.

Advantages**:**

1. Understanding and maintenances is easier.

Disadvantages:

1. Performance will be reduced when there are more users.

Three-Tier Architecture:

Three tier architecture having three layers. They are

1. Client layer
2. Business layer
3. Data layer

Client layer: Here we design the form using textbox, label etc.

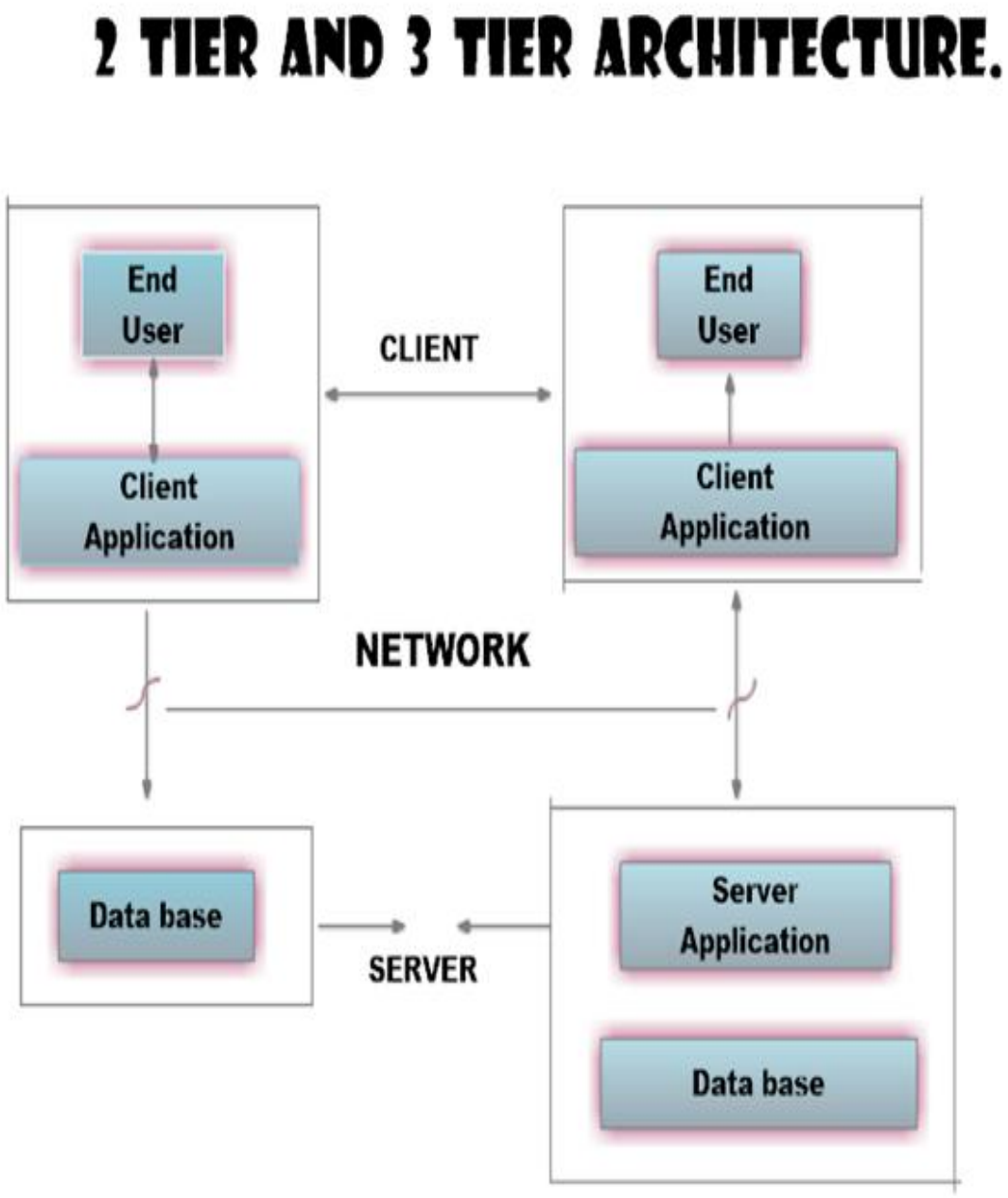
Business layer: It is the intermediate layer which has the functions for client layer and it is used to make communication faster between client and data layer. It provides the business processes logic and the data access.

Data layer: it has the database.

Advantages

1. Easy to modify with out affecting other modules
2. Fast communication
3. Performance will be good in three tier architecture.

(c) Result (Snapshot)



Experiment - Three

(a) Objective: Draw Schema Architecture with Smart Draw tool.

(b) Description:

Data are actually stored as bits, or numbers and strings, but it is difficult to work with data at this level. It is necessary to view data at different levels of abstraction.

Schema:

- Description of data at some level. Each level has its own schema.

We will be concerned with three forms of schemas:

- physical,
- conceptual, and
- external.

Physical Data Level

The **physical schema** describes details of how data is stored: files, indices, etc. on the random access disk system. It also typically describes the record layout of files and type of files (hash, b-tree, flat).

Conceptual Data Level

Also referred to as the Logical level

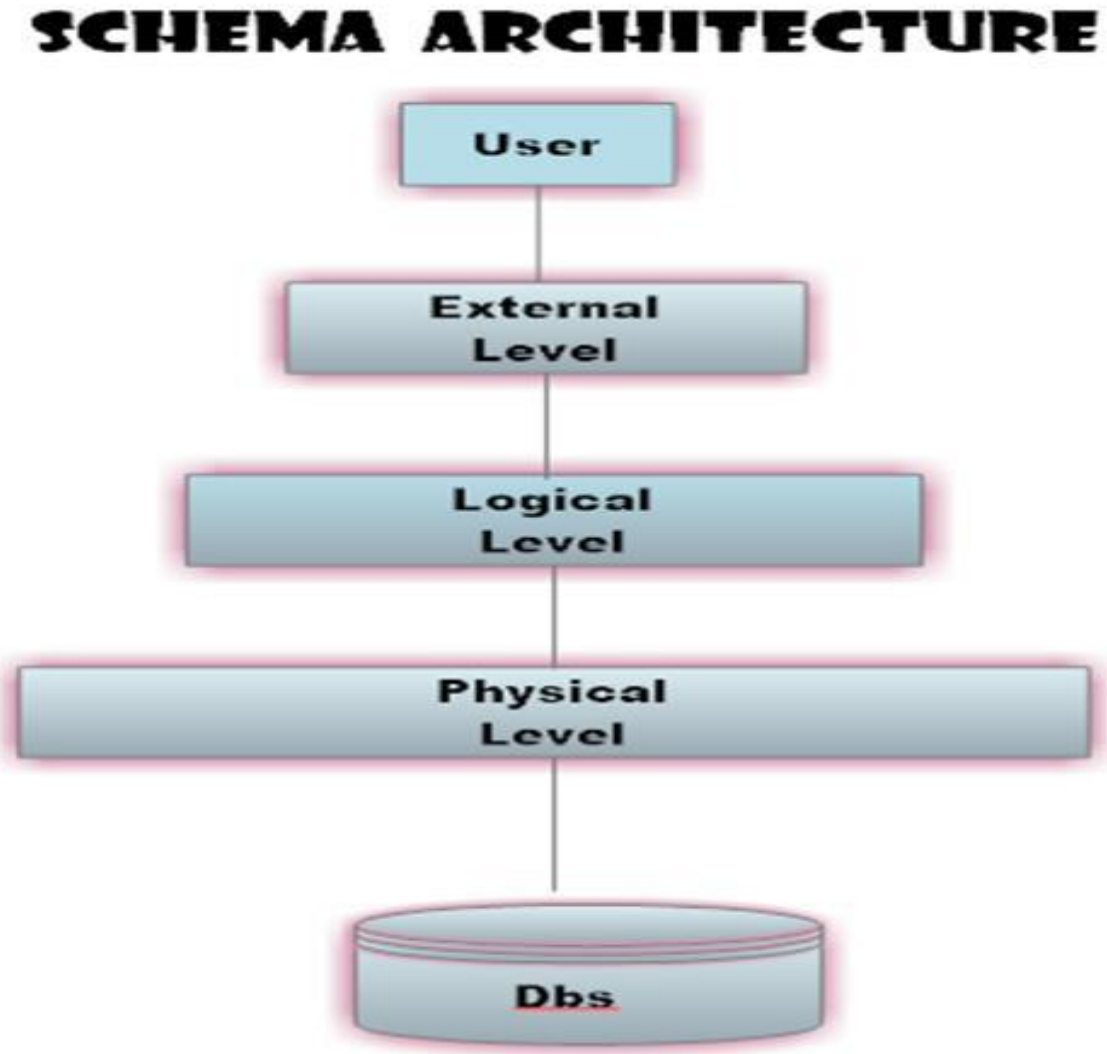
Hides details of the physical level.

- In the relational model, the conceptual schema presents data as a set of tables.

External Data Level

In the relational model, the **external schema** also presents data as a set of relations. An external schema specifies a **view** of the data in terms of the conceptual level. It is tailored to the needs of a particular category of users. Portions of stored data should not be seen by some users and begins to implement a level of security and simplifies the view for these users

(c) Result (Snapshot)



Experiment - Four

(a) Objective: Draw DBMS detailed architecture with Smart Draw tool

(b) Description:

A **database system** is partitioned into modules that deal with each of the responsibilities of the overall system.

The functional components of a database system can be broadly divided into the storage manager and the query processor components. Storage manager: The storage manager is important because database typically require a large amount of storage space. Corporate database range in size from hundreds of gigabytes to for the largest database terabytes of data. Since the main memory of computer cannot store this much information, the information is stored on disks. Data are moved between disk storage and main memory as needed. Since the movement of data to and from disk is slow relative to the speed of the central processing unit, it is imperative that the database system structure the data so as to minimize the need to move data between disk and main memory.

A storage manager is a program module that provides the interface between the low level data stored in the database and the application programs and queries submitted to the system. The storage manager is responsible for the interaction with the file manager. The raw data are stored on the disk using the file system, which is usually provided by a conventional operating system. The storage manager translates the various DML statements into low-level File system commands. Thus, the storage manager is responsible for storing, retrieving, and updating data in the data base.

The storage manager components include:

i) Authorization and integrity manager: Which tests for the satisfaction of integrity constraints and checks the authority of users to access data.

ii) Transaction manager: Which ensures that the database remains in a consistent (correct) state despite system failures, and that concurrent transaction executions proceed without conflicting.

iii) File manager: Which manages the allocation of space on disk- storage and the data structures used to represent information stored on disk.

iv) Buffer manager: Which is responsible for fetching data from disk storage into main memory and deciding what data to cache in main memory. The buffer manager is a critical part of the data base system, since it enables the data base to handle data sizes that are much larger than the size of main memory.

The storage manager implements several data structures as part of the physical system implementation.

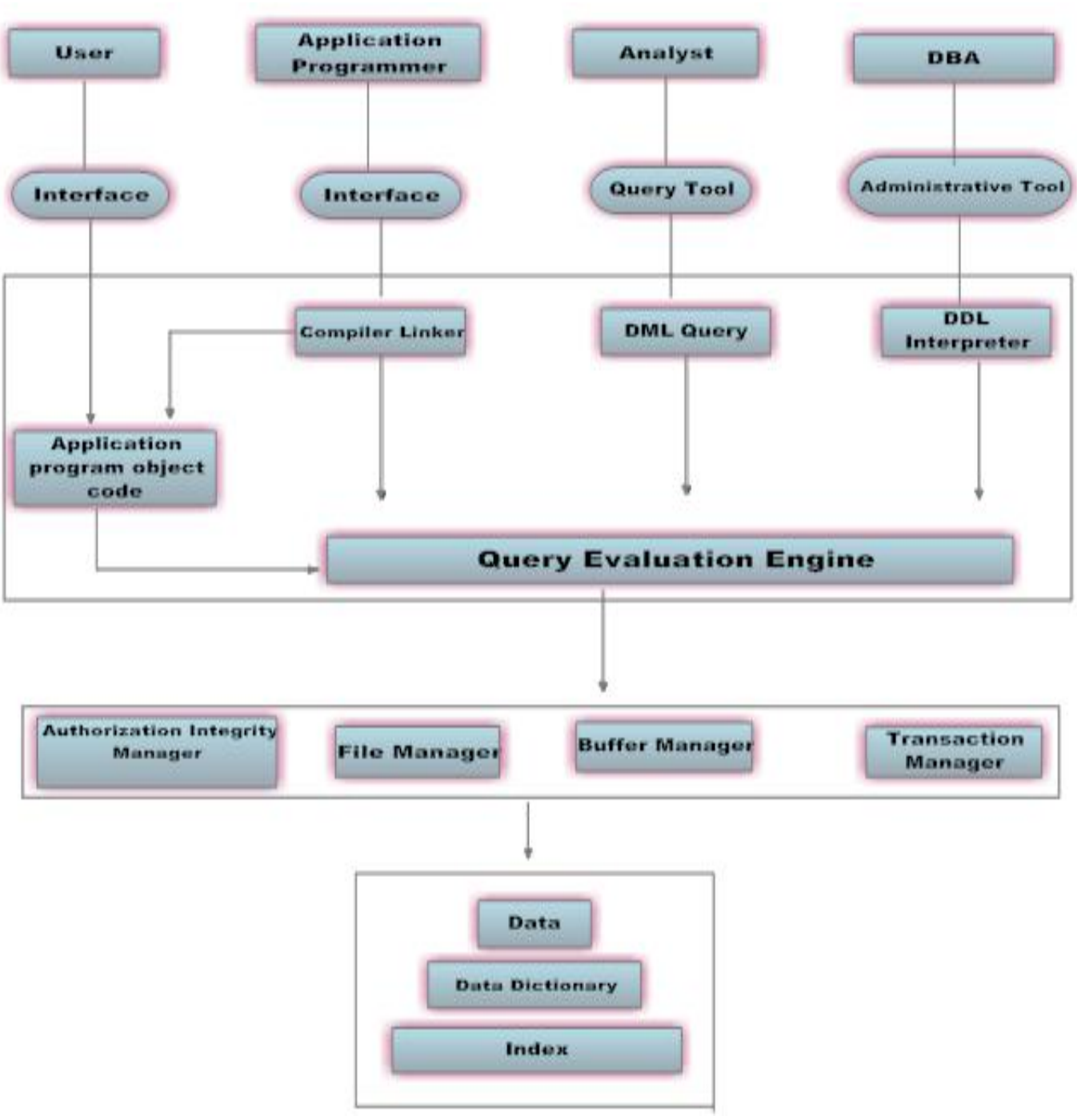
i) Data files which store the data base itself.

ii) Data dictionary which stores metadata about the structure of the database, in particular the schema of the data base.

iii) Indices which provide fast access to data items that hold particular values.

The query processor: Query processor helps the database system simplify and facilitate access to data. The job of the database system to translate update and queries written in a non procedural language, at the logical level, into an efficient sequence of operations at the physical level.

(c) Result (Snapshot)



Experiment - Five

(a) Objective: Draw Data Independence architecture with Smart Draw tool.

(b) Description:

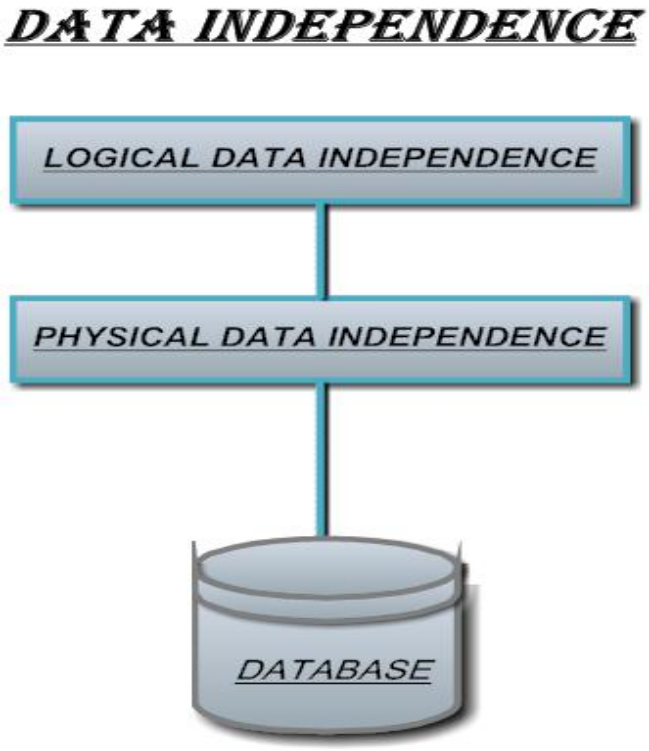
Data independence is the type of data transparency that matters for a centralized DBMS. It refers to the immunity of user applications to changes made in the definition and organization of data. Data independence can be explained as follows: Each higher level of the data architecture is immune to changes of the next lower level of the architecture.

Data Independence Types

The ability to modify schema definition in one level without affecting schema definition in the next higher level is called data independence. There are two levels of data independence, they are Physical data independence and Logical data independence.

1. Physical data independence is the ability to modify the physical schema without causing application programs to be rewritten. Modifications at the physical level are occasionally necessary to improve performance. It means we change the physical storage/level without affecting the conceptual or external view of the data. The new changes are absorbed by mapping techniques.
2. Logical data independence is the ability to modify the logical schema without causing application program to be rewritten. Modifications at the logical level are necessary whenever the logical structure of the database is altered (for example, when money-market accounts are added to banking system). Logical Data independence means if we add some new columns or remove some columns from table then the user view and programs should not change. It is called the logical independence. For example: consider two users A & B. Both are selecting the empno and ename. If user B add a new column salary in his view/table then it will not effect the external view user; user A, but internal view of database has been changed for both users A & B. Now user A can also print the salary. It means if we change in view then program which use this view need not to be changed.

Result (Snapshot)



Experiment - Six

(a) Objective: Draw Data Models Category using Smart Draw tool

(b) Description:

A **database model** is a type of data model that determines the logical structure of a database and fundamentally determines in which manner data can be stored, organized, and manipulated. The most popular example of a database model is the relational model, which uses a table-based format. Common logical data models for databases include:

- Hierarchical database model
- Network model
- Relational model
- Entity–relationship model
    - Enhanced entity–relationship model

Hierarchical database model

A hierarchical database model is a data model in which the data is organized into a tree-like structure. The data is stored as records which are connected to one another through links. A record is a collection of fields, with each field containing only one value. The entity type of a record defines which fields the record contains.

Network model

The network model is a database model conceived as a flexible way of representing objects and their relationships. Its distinguishing feature is that the schema, viewed as a graph in which object types are nodes and relationship types are arcs, is not restricted to being a hierarchy or lattice.

Relational model

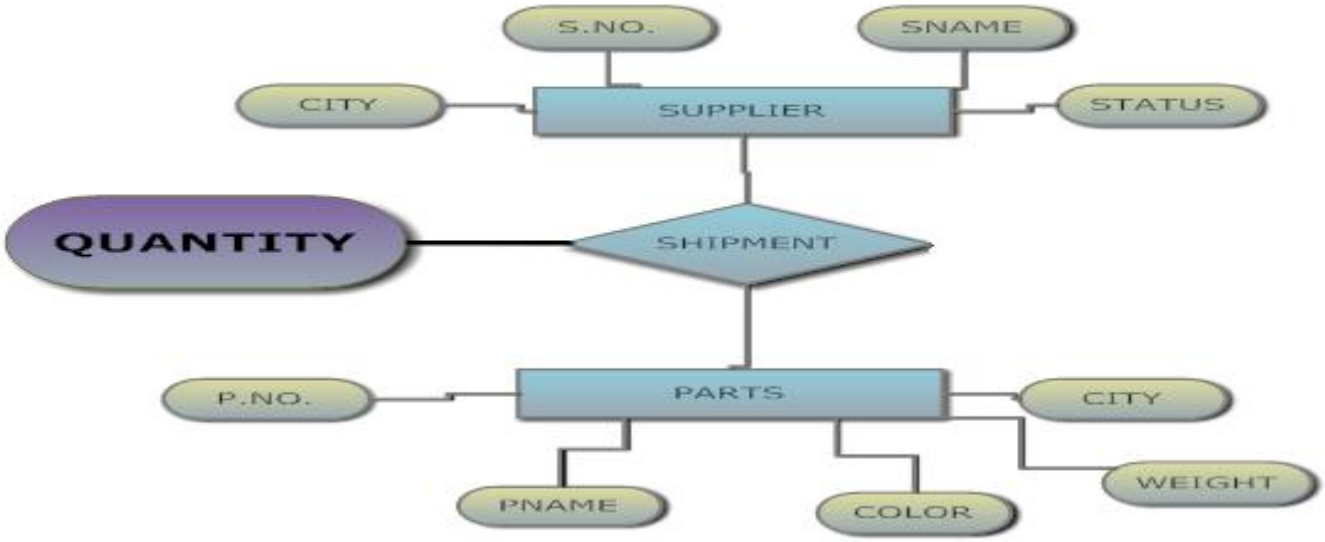
The relational model for database management is a database model based on first-order predicate logic, first formulated and proposed in 1969 by Edgar F. Codd. In the relational model of a database, all data is represented in terms of tuples, grouped into relations. A database organized in terms of the relational model is a relational database.

Entity–relationship model

In software engineering, an entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them.

(c) Result (Snapshot)

RELATIONAL MODEL



HIERARCHICAL DATA MODEL

S1	SMITH	20	PARIS

P1	NUT	RED	40	PARIS	300	
P2	BOLT	BLACK	10	NEW YORK	200	
P3	SCREW	GREY	20	ROME	400	

S2	RAM	10	NEW YORK

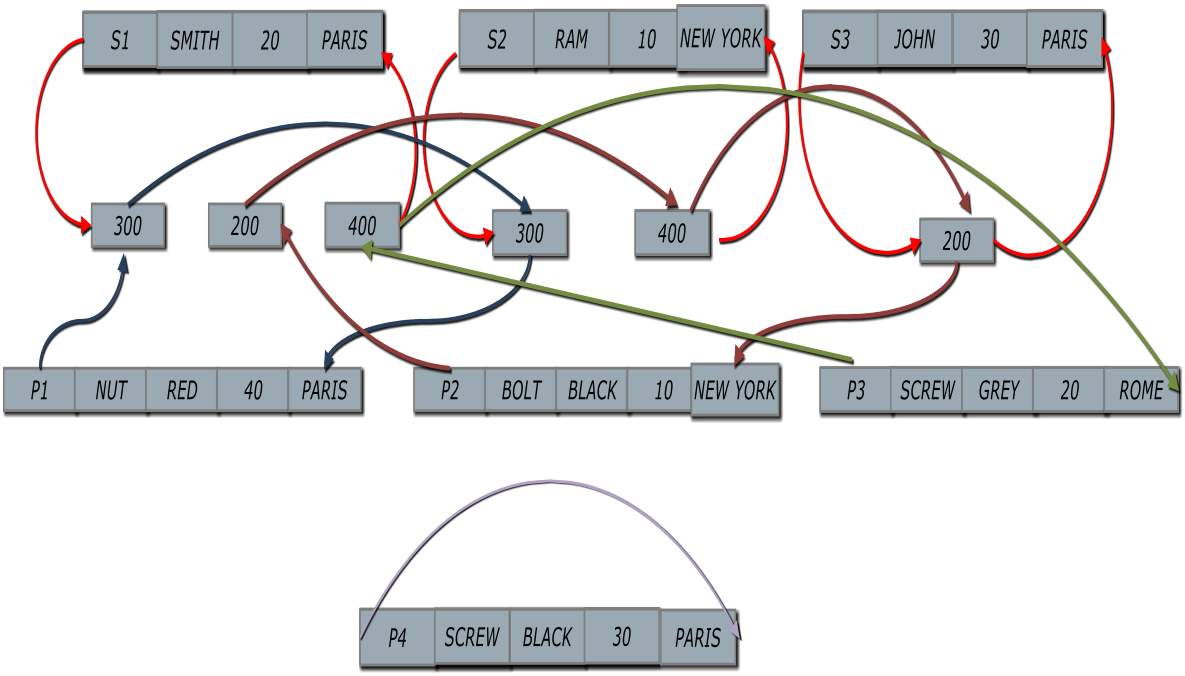
P1	NUT	RED	40	PARIS	300	
P2	BOLT	BLACK	10	NEW YORK	400	

S3	JOHN	30	PARIS

P2	BOLT	BLACK	10	NEW YORK	200

P4	SCREW	BLACK	30	PARIS

NETWORK MODEL



Experiment - Seven

(a) Objective: Draw Specialization Generalization & Aggregation with Smart Draw tool.

(b) Description:

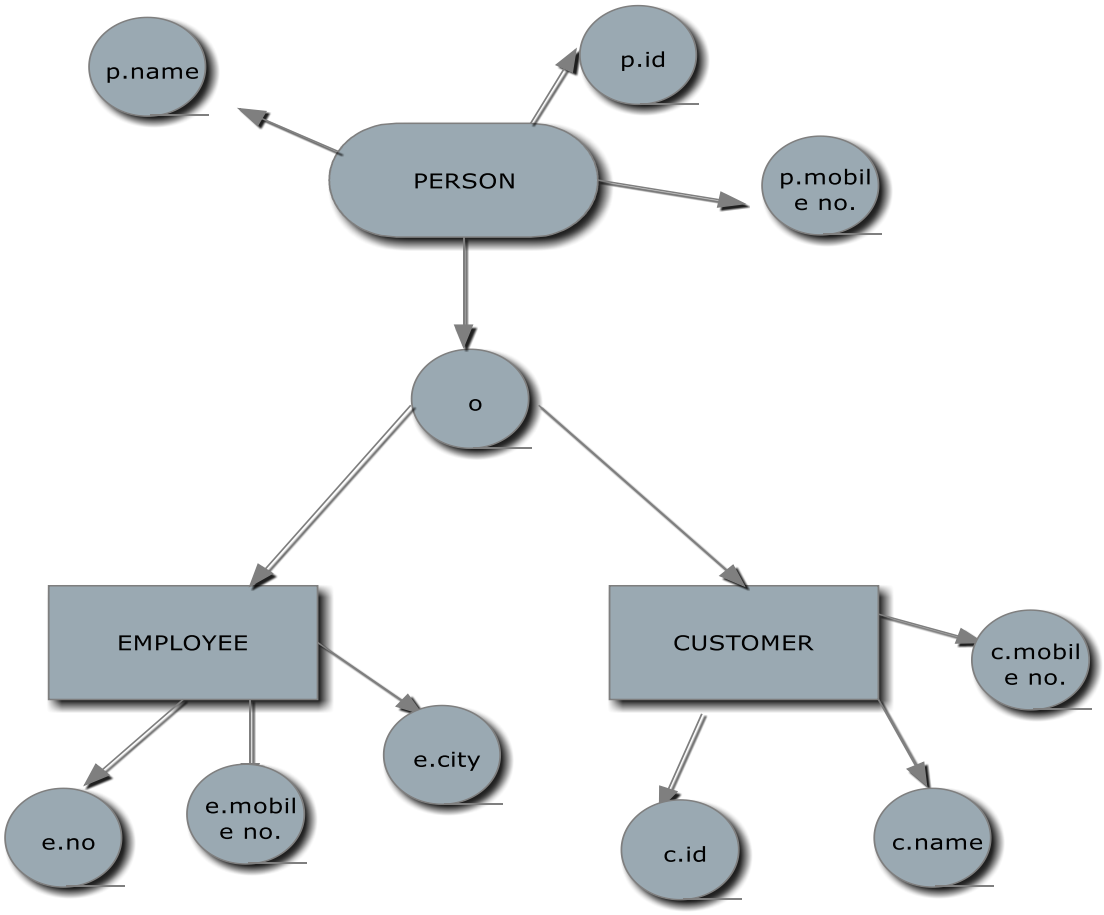
Specialization is top down manner, in which an entity set may include sub groupings of entities that are distinct in some way from other entities in the set. For instance, a subset of entities within an entity set may have attributes that are not shared by all the entities in the entity set.

Generalization is bottom-up manner, in which multiple entity sets are synthesized into a higher-level entity set on the basis of common features.

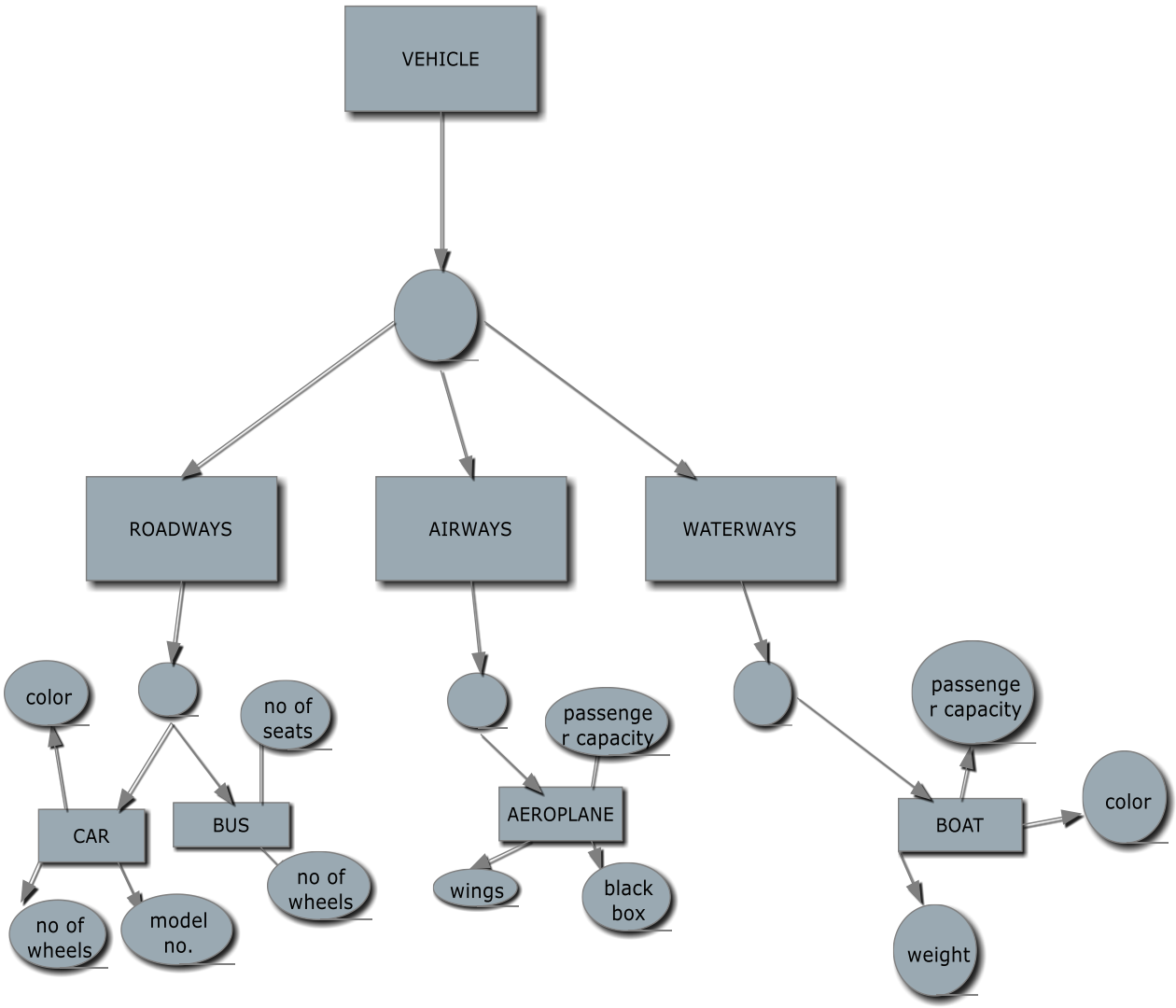
Aggregation is an abstraction in which relationship sets (along with their associated entity sets) are treated as higher-level entity sets, and can participate in relationships.

(c) Result (Snapshot)

Specialization:-



Generalization:-



Experiment - Eight

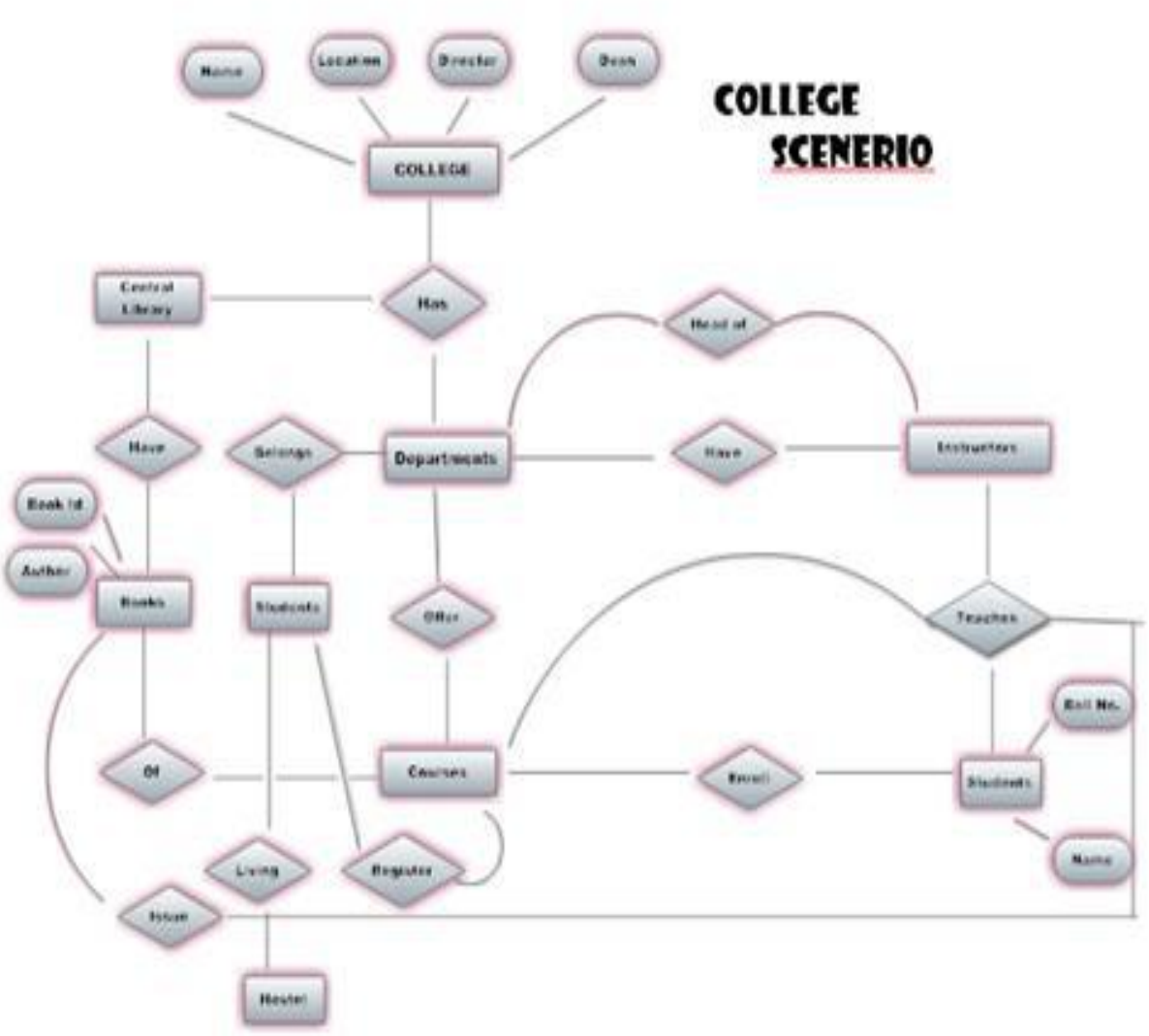
(a) Objective: Draw ER Diagram of College Database and Bank Database

(b) Description:

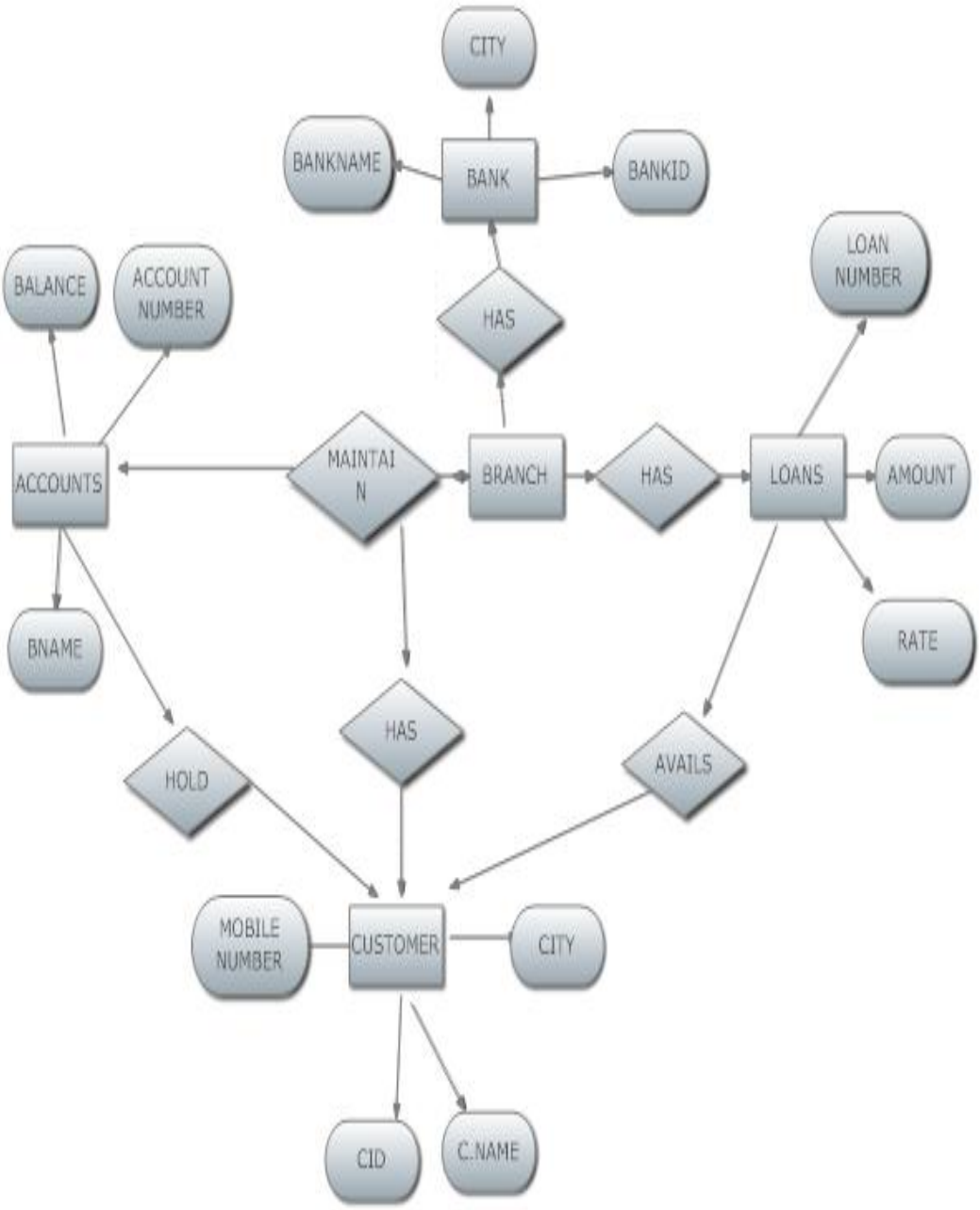
Assume all entities , attributes and relationship. Represent all cardinality and constraints for the following databases-

1) College Database
2) Bank Database

(c) Result (Snapshot)



Bank Scenerio:-



Experiment - Nine

(a) Objective: To study about the DDL commands in RDBMS.

(b) Coding & Result:

Table Creation:

Syntax: **Create table** table-name(A_1D_1,A_2D_2…….AnDn)

Description: To create a new table.

CS4750 – The Bank Database – from Database System Concepts

branch(branch-name, branch-city, assets)
customer(customer-name, customer-street, customer-city)
account(account-number, branch-name, balance)
loan(loan-number, branch-name, amount)
depositor(customer-name, account-number)
borrower(customer-name, loan-number)

Branch Table

branch-name	*branch-city*	*assets*
Brighton	Brooklyn	7100000
Downtown	Brooklyn	9000000
Mianus	Horseneck	400000
North Town	Rye	3700000
Perryridge	Horseneck	1700000
Pownal	Bennington	300000
Redwood	Palo Alto	2100000
Round Hill	Horseneck	8000000

Customer Table

customer-name	*customer-street*	*customer-city*
Adams	Spring	Pittsfield
Brooks	Senator	Brooklyn
Curry	North	Rye
Glenn	Sand Hill	Woodside
Green	Walnut	Stamford
Hayes	Main	Harrison
Johnson	Alma	Palo Alto
Jones	Main	Harrison
Lindsay	Park	Pittsfield
Smith	North	Rye
Turner	Putnam	Stamford
Williams	Nassau	Princeton

Account Table

account-number	*branch-name*	*balance*
A-101	Downtown	500
A-215	Mianus	700
A-102	Perryridge	400
A-305	Round Hill	350
A-201	Brighton	900
A-222	Redwood	700
A-217	Brighton	750

Loan Table

loan-number	*branch-name*	*amount*
L-11	Round Hill	900
L-14	Downtown	1500
L-15	Perryridge	1500
L-16	Perryridge	1300
L-17	Downtown	1000
L-23	Redwood	2000
L-93	Mianus	500

Depositor Table

customer-name	*account-number*
Hayes	A-102
Johnson	A-101
Johnson	A-201
Jones	A-217
Lindsay	A-222
Smith	A-215
Turner	A-305

Borrower Table

customer-name	*loan-number*
Adams	L-16
Curry	L-93
Hayes	L-15
Jackson	L-14
Jones	L-17
Smith	L-11
Smith	L-23
Williams	L-17

Table Description:

Syntax: **Desc** table-name.

Description: To display the structure of the table.

Table Deletion:

Syntax: **Drop table** table-name

Description: To delete the table.

Table Renaming:
Syntax: **Alter table** old-table-name **rename to** new-table-name
Description: To rename the table .

Rename Column:
Syntax: **Alter table** table-name **rename column** old-column-name to New-column-name
Description: To rename the column in the table.

Adding one Column:
Syntax: **Alter table** table-name **add** A D
Description: To alter the table by adding one column
Deleting One Column:
Syntax: **Alter table** table-name **drop column** A
Description: To delete one column from the table.

Datatype or Domaintype modification:
Syntax: **Alter table** table-name **modify**(A_1 new data-type(size),……..An new data-type(size))
Description: To change the datatype or domaintype.
Table Truncation:
Syntax: **Truncate table** table-name
Description: To delete all the records from the table.

Example:-
Creating the table:
Table created.
Description of the table:
Desc Table name
Output-(Snapshot)
Renaming the column:

Alter table student rename column course to degree

Table altered.
Desc Table_name

Adding one column:
Alter table Table_name add column_name datatype
Table altered.
Desc Table_name

Deleting one column:
Alter table Table_name drop column column_name
Table altered.
Desc Table_name

Domain type modification:

Desc Table_name

Table renaming:

Alter table Table_name rename to new_table_name

Table altered

Desc Table_name

Table truncation:

Truncate table class

Table truncated.

C**reate table with primary keys:**

Table deletion:

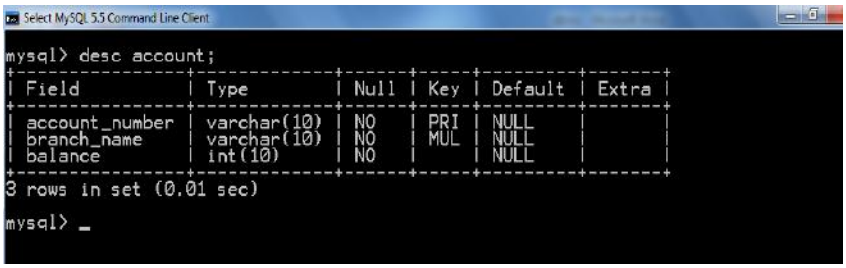
Drop table class

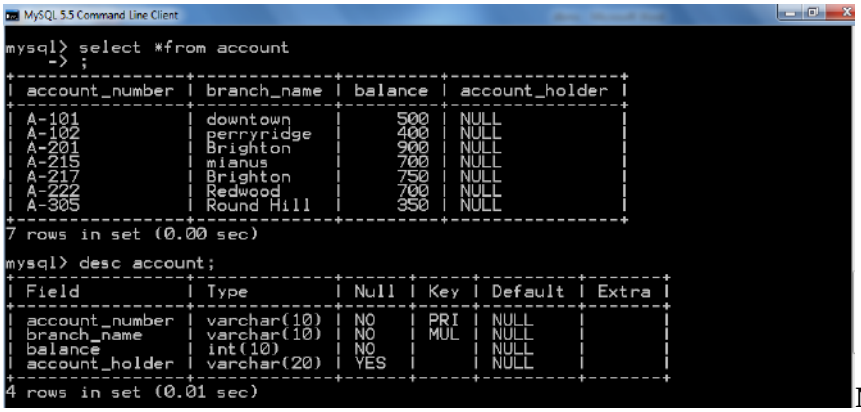
Table dropped.

Result(Snapshot)

Thus the DDL commands were executed.

Table created (table name - account)





New Column added to the table account

Column dropped from table account

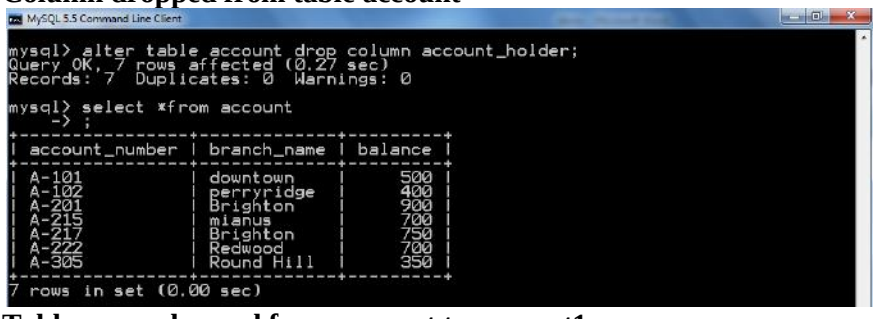


Table name changed from account to account1

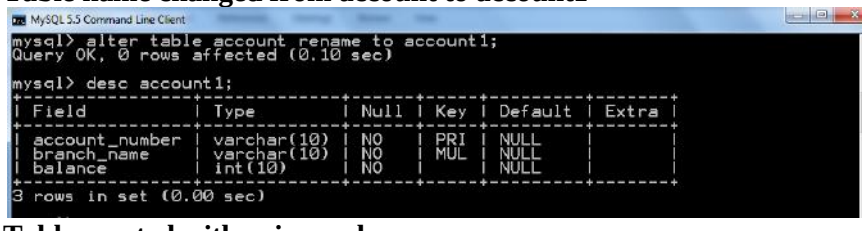
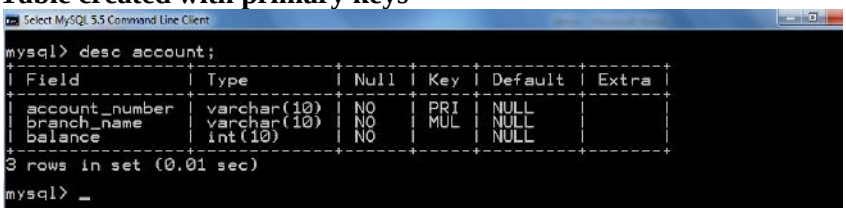


Table created with primary keys



Experiment - Ten

(a) Objective: To study about the DML commands.

(b) Coding & Result :

1.Select:

Syntax:

Select $A_1,A_2,\ldots.An$

From $r_1,r_2,r_3\ldots.r_n$

Where **p**

Description: It selects the rows from the table.

2.Insert:

Syntax:

Insert into table-name **values**(value1,value2,…….valuen)

Description: It is used to insert values into specified table.

3.Delete:

Syntax:

Delete from table-name

Where P

Description: It is used to delete the values from specified table.

5.Group by:

Syntax:

Select <(columns)>

From <(tables)>

Where <(condition)>

Group by <groupcolumns>

Description: It is used to group the rows that have certain properties and then to apply an aggregate function on one column for each group separately.

6.OrderBy:

Syntax:

Select [distinct]<columns>

From <table>

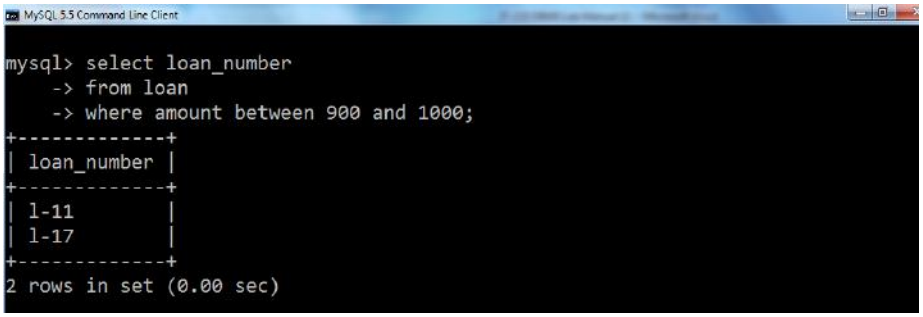
Where <condition>

[**order by** <columns[asc/desc]>]

Description: It is used to retrieve the information stored in database in a particular order.

Output:

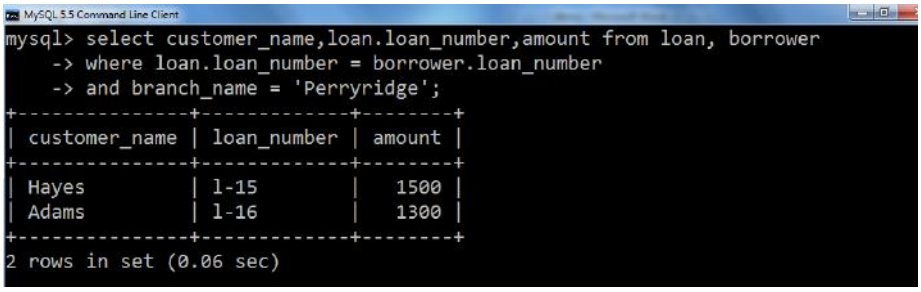
2. Select loannumber
   from loan
   where amount between 900 and 1000



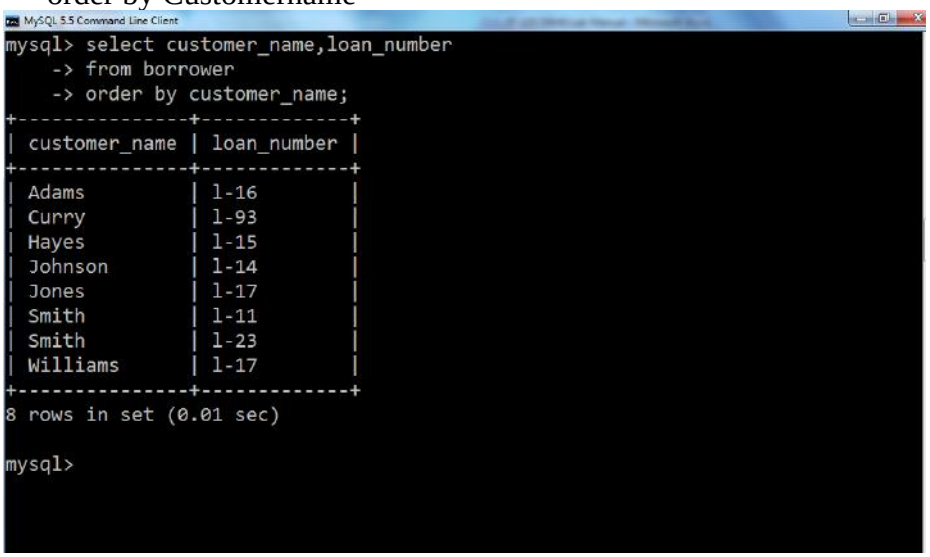
3. Select customername,loan.loannumber,amount

   From loan,Borrower

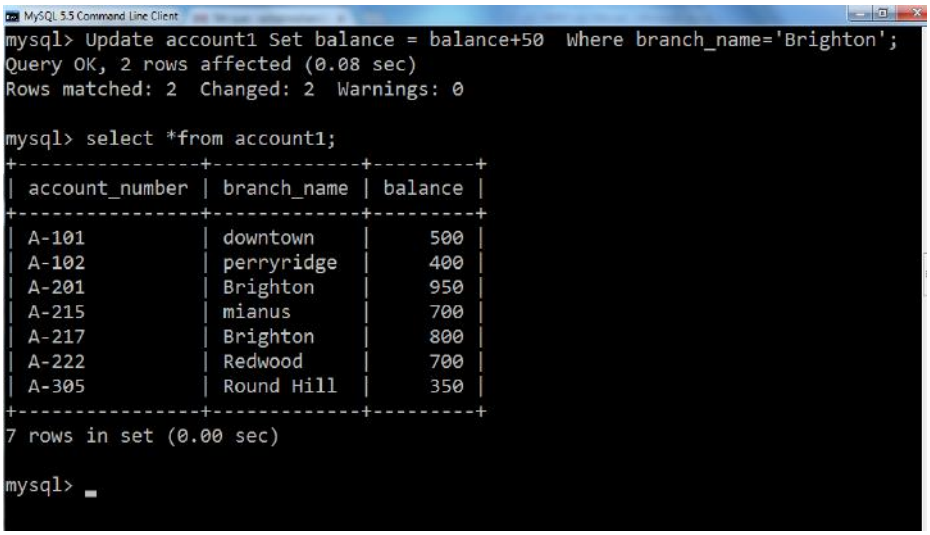
   Where loan.loannumber=Borrower.loannumber and
   Branchname = 'Perryridge'



4. select Customername,loannumber
   from Borrower
   order by Customername



5. Update account Set balance = balance+50 Where branchname='Brighton'



2 rows updated.

select * from account

6. Delete from loan

   Where branchname='Roundhill'

   1 row deleted.
   Select * from loan

```
mysql> delete from loan
    -> where branch_name = 'round hill';
Query OK, 1 row affected (0.02 sec)

mysql> select *from loan;
+-------------+-------------+--------+
| loan_number | branch_name | amount |
+-------------+-------------+--------+
| l-14        | downtown    |   1500 |
| l-15        | perryridge  |   1500 |
| l-16        | perryridge  |   1300 |
| l-17        | downtown    |   1000 |
| l-23        | redwood     |   2000 |
| l-93        | mianus      |    500 |
+-------------+-------------+--------+
6 rows in set (0.00 sec)
```

7. select customername
   from customer
   where customername like '_a%'