

GLOBAL INSTITUTE OF TECHNOLOGY

(Approved by AICTE and Affiliated to RTU, Kota)



LABORATORY MANUAL

(2019-2020)

DBMS LAB

II Year & IV Semester

Computer Science & Engineering

SYLLABUS

Objectives: At the end of the semester, the students should have clearly understood and implemented the following:

1. Stating a database design & application problem.
2. Preparing ER diagram
3. Finding the data fields to be used in the database.
4. Selecting fields for keys.
5. Normalizing the database including analysis of functional dependencies.
6. Installing and configuring the database server and the front end tools.
7. Designing database and writing applications for manipulation of data for a standalone and shared data base including concepts like concurrency control, transaction roll back, logging, report generation etc.
8. Get acquainted with SQL.

In order to achieve the above objectives, it is expected that each students will chose one problem. The implementation shall being with the statement of the objectives to be achieved, preparing ER diagram, designing of database, normalization and finally manipulation of the database including generation of reports, views etc. The problem may first be implemented for a standalone system to be used by a single user.

All the above steps may then be followed for development of a database application to be used by multiple users in a client server environment with access control. The application shall NOT use web techniques.

One exercise may be assigned on creation of table, manipulation of data and report generation using SQL.

Indicative List of exercises:

1. Student information system for your college.
2. Student grievance registration and redressal system.
3. A video library management system for a shop.
4. Inventory management system for a hardware/ sanitary item shop.
5. Inventory management system for your college.
6. Guarantee management system for the equipments in your college.

SOFTWARE AND HARDWARE REQUIREMENTS

Hardware Requirements:

1. Processor: Pentium IV
2. RAM: 256 MB
3. Hard Disk: 40 GB

Software Requirements:

1. OS: Windows 98/2000/ME/XP
2. Data Base Sever: MYSQL, ORACLE , MS Access, SQL Server 2005/2008

RATIONAL BEHIND DBMS LAB

Database management has evolved from a specialized computer application to a central component of a modern computing environment and as a result knowledge about database system has become an essential part of computer science. The course serves as a visual guide to the material presented during our lectures. The aim of this course is to provide an introduction to Database management system, with an emphasis on foundational material the fundamental concepts and algorithms covered are based on those used in existing commercial or experimental database systems. Our aim is to present these concepts and algorithms in general setting.

Objectives

Upon successful completion of this Lab the student will be able to:

- Creating database objects
- Modifying database objects
- Manipulating the data
- Retrieving the data from the database server
- Performing database operations (create, update, modify, retrieve, etc.,) using front-end tools.
- Design and Develop applications like banking, reservation system, etc.

INTRODUCTION TO DATABASE SYSTEMS

Need for data management systems

Today, the success of any organization depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use this data to analyze and guide its activities. Phrases such as the *information superhighway* have become ubiquitous, and information processing is a rapidly growing industry.

The amount of information available to us is increasing at an exploding rate, and the value of data as an organizational asset is widely recognized. Yet without the ability to manage this vast amount of data, and to quickly find the information that is relevant to a given scenario or interest, as the amount of information increases, it tends to become a distraction and a liability, rather than an asset. This paradox drives the need for increasingly powerful and flexible data management systems. To get the most out of their large and complex datasets, users must have tools that simplify the tasks of managing the data and extracting useful information in a timely fashion. Otherwise, data can become a liability, with the cost of acquiring it and managing it far exceeding the value that is derived from it.

Database

A *database* is a collection of related data, typically describing the activities of one or more related organizations.

For example, a *university database* might contain information about the following:

Entities such as students, faculty and courses.

Relationships between entities, such as students' enrollment in courses, and faculty teaching courses.

Database Management System (DBMS)

A **database management system (DBMS)** is software that enables users to create and maintain a database. The DBMS is hence a *general-purpose software system* that facilitates the processes of *defining, constructing, manipulating, and sharing* databases among various users and applications.

Applications of Database Systems

Databases are widely used. Some applications of database systems are given below:

1. **Banking:** For customer information, accounts, and loans, and banking transactions.
2. **Airlines:** For reservations and schedule information.
3. **Universities:** For student information, course registrations, and grades.
4. **Credit card transactions:** For purchases on credit cards and generation of monthly statements.
5. **Telecommunication:** For keeping records of calls made, generating monthly bills, maintaining balances on prepaid calling cards, and storing information about the communication networks.
6. **Finance:** For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds.
7. **Sales:** For customer, product, and purchase information.
8. **Manufacturing:** For management of supply chain and for tracking production of items in factories, inventories of items in warehouses/stores, and orders for items.
9. **Human resources:** For information about employees, salaries, payroll taxes and benefits, and for generation of paychecks.

Advantages of DBMS

Using a DBMS to manage data has following advantages:

1. **Data independence:** Application programs should be as independent as possible from details of data representation and storage. The DBMS can provide an abstract view of the data to insulate application code from such details.
2. **Efficient data access:** A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is stored on external storage devices.
3. **Data integrity and security:** If data is always accessed through the DBMS, the DBMS can enforce integrity constraints on the data. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, the DBMS can enforce *access controls* that govern what data is visible to different classes of users.

4. **Data administration:** When several users share the data, centralizing the administration of data can offer significant improvements. Experienced professionals, who understand the nature of the data being managed, and how different groups of users use it, can be responsible for organizing the data representation to minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient.
5. **Concurrent access and crash recovery:** A DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.
6. **Reduced application development time:** Clearly, the DBMS supports many important functions that are common to many applications accessing data stored in the DBMS. This, in conjunction with the high-level interface to the data, facilitates quick development of applications. Such applications are also likely to be more robust than applications developed from scratch because many important tasks are handled by the DBMS instead of being implemented by the application.

People dealing with databases

Below are categories of people who deal with databases:

1. **Database Implementers:** People in this category are associated with the creation and use of databases. These people build DBMS software. Database implementers work for vendors such as IBM or Oracle.
2. **End Users:** End users store and use data in a DBMS. End users come from a diverse and increasing number of fields. Many end users simply use applications written by database application programmers, and so require little technical knowledge about DBMS software. However, sophisticated users who make more extensive use of a DBMS, such as writing their own queries, require a deeper understanding of its features.
3. **Database application programmers:** These people develop packages that facilitate data access for end users using the host or data languages and software

tools that DBMS vendors provide. (Such tools include report writers, spreadsheets, etc.)

Database administrator (DBA): The DBA is responsible for authorizing access to the database, for coordinating and monitoring its use, and for acquiring software and hardware resources as needed. The DBA is accountable for problems such as breach of security or poor system response time.

ENTITY-RELATIONSHIP MODEL

A database can be modeled as:

- A collection of entities,
- Relationships among entities.

Entity: A real-world object that can be distinctly identified may represent some real physical object. May represent some conceptual idea E.g., SC304 is a course; Semester 1 2001/2002 is a semester An entity is an object that exists and is distinguishable from other objects.

Example: specific person, company, event, plant

Entity set: An entity set is a set of entities of the same type that share the same properties.

Example: set of all persons, companies, trees, holidays.

An Entity should be:

- An Object that will have many instances in the Database
- An Object that will have multiple attributes
- An Object that we are trying to model

An Entity should not be:

- User of the Database
- An output of the Database (eg. Report)

Attributes: An entity is represented by a set of attributes, that is, descriptive properties possessed by all members of an entity set. (value from corresponding entity)

Example: customer = (customer-name, social-security, customer-street, customer-city)

account = (account-number, balance)

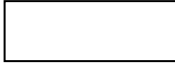
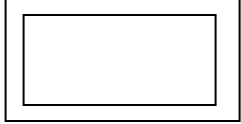


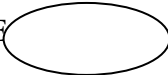
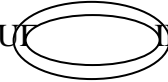
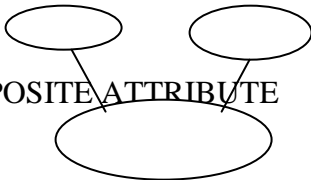


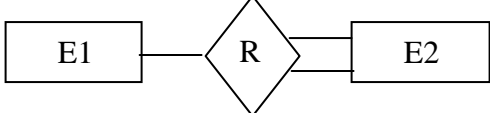
Domain: Domain is the set of permitted values for each attribute

Attribute types:

- Simple and composite attributes.
- Single-valued and multi-valued attributes
- Null attributes
- Derived attributes

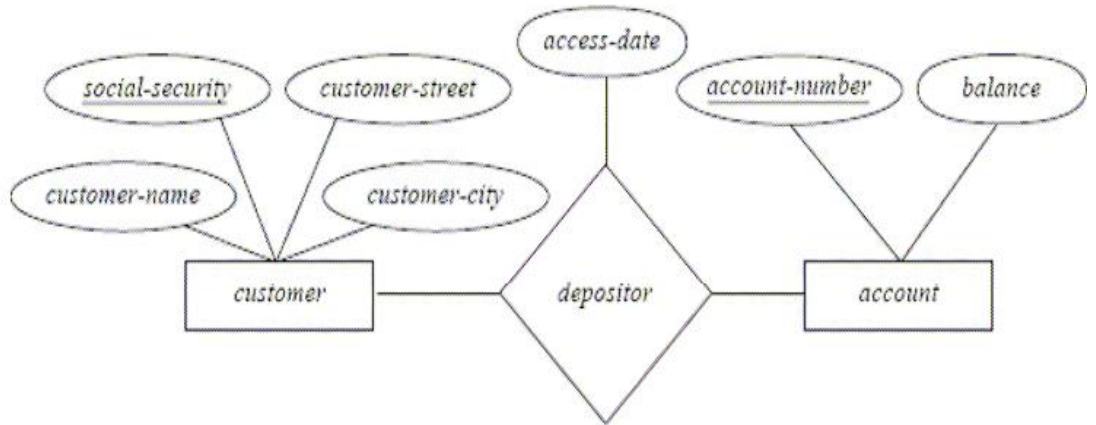
- Identifiers (Key) attributes

Symbols Used in ER Diagram:

Symbol	Meaning
	ENTITY TYPE
	WEAK ENTITY TYPE
RELATIONSHIP 	
IDENTIFYING RELATIONSHIP 	
ATTRIBUTE 	
MULTIVALUED ATTRIBUTE 	
COMPOSITE ATTRIBUTE 	
DERIVED ATTRIBUTE 	
KEY ATTRIBUTE 	
	TOTAL PARTICIPATION OF E2 IN R



Example of ER Diagram:



INTRODUCTION TO SQL

SQL stands for Structured Query Language. Oracle provides many extensions to ANSI SQL. SQL is standard language for interacting with a relational database.

A table is primary database object of SQL, which is used to store the data in the form of rows and columns. SQL developed by IBM is used as standard language to access data from database. In order to communicate with database SQL has been divided into four sub languages and each sub language consists of different commands, they are:

1. Data Definition Language (DDL): DDL commands are used to define the database structure or schema.

Different DDL commands are:

CREATE - to create objects in the database

ALTER - alters the structure of the database

DROP - delete objects from the database

TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed

RENAME - rename an object

2. Data Manipulation Language (DML): DML commands are used for managing data within schema objects.

Different DML commands are:

SELECT - retrieve data from the a database

INSERT - insert data into a table

UPDATE - updates existing data within a table

DELETE - deletes all records from a table, the space for the records remain

3. Data Control Language (DCL): DCL commands are used to control the access to the database objects.

Different DCL commands are:

GRANT - gives user's access privileges to database

REVOKE - withdraw access privileges given with the GRANT command

- 4. Transaction Control Language (TCL):** TCL commands are used to manage the changes made by DML statements. It allows statements to be grouped together into logical transactions.

Different TCL commands are:

COMMIT - save work done

SAVEPOINT - identify a point in a transaction to which you can later roll back

ROLLBACK - restore database to original since the last COMMIT

DDL COMMANDS

1. Creating Database Objects (Create Command)

a) Creating table (without constraints)

Syntax:

Create table<table name> (<col1><data type><size>,<col2><data type><size>);

Ex:

Create table emp(empnumber(4), ename char(10), address char(30), city char(10));

b) Creating view

Syntax:

Create view<view name>**As** (SELECT query);

Ex:

Create view empviewas(select empno, city from emp);

2. Modifying the structure of tables (Alter Command)

a) Add new columns

Syntax:

Alter table<table name>**add**(<new col><data type(size),<new col>data type(size));

Ex:

Alter table empadd(sal number(7,2));

b) Modifying existing columns

Syntax:

Alter table<table name>**modify**(<col><new data type(size));

Ex:

Alter table empmodify(ename varchar2(20));

c) Renaming existing columns

Syntax:

Alter table<table name> **rename column** <col> to <newcol>;

Ex:

Alter table emp rename column ename to empname;

d) Removing existing columns

Syntax:

Alter table<table name> drop column <col>;

Ex:

Alter table emp drop column sal;

3. Renaming the tables

Syntax:

Rename<old table> to <new table>;

Ex:

rename emp to emp1;

4. Truncating the tables

Syntax:

Truncate table<table name>;

Ex:

truncate table emp1;

5. Destroying tables

Syntax:

Drop table<table name>;

Ex:

Drop table emp1;

WORKING WITH CONSTRAINTS

Constraints are rules for a database that limit the acceptable data values for a table. They are the optional schema objects that depend on a table. The existence of a table without any constraint is possible, but the existence of a constraint without any table is not possible. Constraints enforce the business rules in a database. Constraints can be created along with the table in the CREATE TABLE statement. Addition and deletion of constraints can be done in the ALTER TABLE statement.

The following types of constraints are available in Oracle Database:

NOT NULL

It enforces that a column, declared as not null, cannot have any NULL values. For example, if an employee's hire date is not known, then that employee may not be considered as a valid employee. If a protected column does not possess any value, then the INSERT and UPDATE statements on those columns will fail.

UNIQUE

It ensures that columns protected by this constraint cannot have duplicate values.

PRIMARY KEY

It is responsible for uniquely identifying a row in a table. A table can have only one PRIMARY KEY constraint. A PRIMARY KEY constraint completely includes both the NOT NULL and UNIQUE constraints. It is enforced with an index on all columns.

FOREIGN KEY

It is also known as referential integrity constraint. It enforces that values referenced in one table are defined in another table. It establishes a parent-child or reference-dependent relationship between the two tables.

CHECK

It enforces that columns must meet a specific condition that is evaluated to a Boolean

value. If the value evaluates to false, then the database will raise an exception, and not allow the INSERT and UPDATE statements to operate on columns.

Creating table with constraints

1. Not null constraint

a) Creating Table with not null constraint

Syntax:

Create table<table name> (<col1><data type><size>) not null,<col2><data type><size>);

Ex:

Create table empl (emponumber(4), ename char(10) not null,city char(10) not null);

2. Unique constraint

a) Creating Table with unique constraint at column level

Syntax:

Create table<table name> (<col1><data type><size>) unique,<col2><data type><size>);

Ex:

Create table empl (emponumber(4), ename char(10) unique,city char(10));

b) Creating Table with unique constraint at table level

Syntax:

Create table<table name> (<col1><data type><size>,<col2><data type><size>), unique(<col1>,<col2>);

Ex:

Create table empl (emponumber(4), ename char(10), city char(10), unique(ename,city));

3. Primary Key Constraints

a) Creating Table with primary key constraint at column level

Syntax:

Create table<table name>(<col1><data type>(<size>) primary key);

Ex:

Create table empl (empnumber(4) primary key, ename char(10), city char(10));

b) Creating Table with primary key constraint at table level

Syntax:

Create table<table name> (<col1><data type>(<size>),<col2><data type>(<size>), primary key(<col1>,<col2>);

Ex:

Create table prod (mfridchar(4), prodid char(4), pname char(10), price number, primary key(mfrid,prodid));

4. Foreign Key Constraints

a) Creating Table with foreign key constraint at column level

Syntax:

Create table<table name>(<col1><data type>(<size>) references <tablename>[<col>]);

Ex:

Create table mgr (mgrno number, deptno number references dept(deptno));

b) Creating Table with foreign key constraint at table level

Syntax:

Create table<table name> (<col1><data type>(<size>),<col2><data type>(<size>), foreign key(<col>[,<col>])references <table name>[(<col>,<col>)]);

Ex:

Create table orders (odrid number, mfridchar(4), prodid char(4), amount number, foreign key(mfrid,prodid) references prod(mfrid,prodid));

5. Check Constraints

a) Creating Table with check constraint at column level

Syntax:

Create table<table name>(<col1><data type>(<size>) check(<logical expression>);

Ex:

Create table mgr (mgrno number, deptno number, age number check(age between 1 and 100));

b) Creating Table with check constraint at table level

Syntax:

Create table<table name> (<col1><data type>(<size>),<col2><data type>(<size>), check(<logical expression>));

Ex:

Create table orders (odrid number, mfridchar(4), prodid char(4), amount number, check(ordered>0 and amount>2000));

DML COMMANDS

1. Inserting Data into Table (Insert Command)

a) When order of columns in the table is not known

Syntax:

```
insert into <table name> (<col1>,<col2>) values(<exp>,<exp>);
```

Ex:

```
insert into emp(empno,city,address,ename) values(1,'Jaipur','GIT  
Sitapura','Vijay');
```

b) When order of columns in the table is known

Syntax:

```
insert into <table name> values(<exp for col1>,<exp for col2>);
```

Ex:

```
insert into empvalues(1,'Vijay','GIT Sitapura','Ajmer');
```

c) Inserting data through user input in Oracle

Syntax:

```
insert into <table name> values ('&<col1>','&<col2>');
```

Ex:

```
insert into emp values('&empno','&ename','&address','&city');
```

2. Deleting Data from Table

a) Remove all rows

Syntax:

```
Delete from <table name>;
```

Ex:

```
Delete from emp;
```

b)Removalof specified rows

Syntax:

Delete from <table name> where <condition>;

Ex:

Delete from emp where ename='Ajay' and city='Jaipur';

3. Updating the contents of a Table.

a)Updating all rows

Syntax:

Update <table name> set <col>=<exp>,<col>=<exp>;

Ex:

Update emp set ename='Akash',city='Udaipur';

b)Updatingselected records

Syntax:

Update<tablename>set<col>=<exp>,<col>=<exp>where <condition>;

Ex:

Update emp set ename='Ram' where city='Udaipur';

SELECT COMMAND

1. Viewing all data from table

a) Using all column names

Syntax:

Select <col> to <col n> from tablename;

Ex:

Select empno,ename,address,city from emp;

b) Using select *

Syntax:

Select * from tablename;

Ex:

Select * from emp;

2. Filtering table data: While viewing data from a table, it is rare that all the data from table will be required each time. Hence, SQL must give us a method of filtering out data that is not required data.

a) View Selected Columns

Syntax:

Select <col1>,<col2> from <tablename>;

Ex:

Select empno,ename from emp;

b) List only the different (distinct) values in a table.

Syntax:

Select distinct <col> from <tablename>;

Ex:

Select distinct ename from emp;

c) Selecting records that fulfill a specified criterion (where clause)

Syntax:

Select */<col1>,<col2> from <tablename> where <condition>;

Ex:

Select * from emp where empno in (2,4,6,7);

d) Selecting records in a sorted order (order by clause)

Syntax:

Select */<col1>,<col2> from <tablename> order by <col> asc/desc;

Ex:

Select empno,ename from emp order by ename desc;

e) Selecting data across multiple records and group the results by one or more columns (group by clause)

Syntax:

Select <col1>,<col2>,<aggregate function on column> from <tablename> group by <col>;

Ex:

Select deptno,max(salary) from emp group by deptno ;

f) Selecting data across multiple records and group the results by one or more columns according to some conditions (having clause)

Syntax:

Select <col1>,<col2>,<aggregate function on column> from <tablename> group by <col> having <condition>;

Ex:

Select deptno,max(salary) from emp group by deptno having max(sal)>20000 ;

DCL COMMANDS

Oracle provides extensive feature in order to safeguard information stored in its tables from unauthorized viewing and damage. The rights that allow the user of some or all oracle resources on the server are called privileges.

1. Grant privileges using the GRANT statement

The grant statement provides various types of access to database objects such as tables, views and sequences and so on.

Syntax:

```
GRANT <object privileges>  
ON <objectname>  
TO <username>  
[WITH GRANT OPTION];
```

Ex:

```
GRANT SELECT, INSERT  
ON HR.employees  
TO git;
```

2. Revoke permissions using the REVOKE statement

The REVOKE statement is used to deny the Grant given on an object.

Syntax:

```
REVOKE <object privilege>  
ON <objectname>  
FROM <user name>;
```

Ex:

```
REVOKE SELECT, INSERT  
ON HR.employees  
FROM git;
```


TCL COMMANDS

A transaction is a set of SQL statements which Oracle treats as a Single Unit. i.e. all the statements should execute successfully or none of the statements should execute.

To control transactions Oracle does not made permanent any DML statements unless you commit it. If you don't commit the transaction and power goes off or system crashes then the transaction is roll backed.

Transaction control statements manage changes made by DML statements.

1. COMMIT

To make the changes done in transactions permanent issue the COMMIT statement.

Syntax:

```
COMMIT [WORK] [COMMENT 'your comment'];
```

WORK is optional.

COMMENT is also optional, specify this if you want to identify this transaction in data dictionary DBA_2PC_PENDING.

Ex:

```
insert into emp (empno,ename,sal) values (101,'Abid',2300);  
commit;
```

2. ROLLBACK

To rollback the changes done in a transaction give rollback statement. Rollback restore the state of the database to the last commit point.

Syntax:

```
Rollabck/ Rollback to savepoint<savepointname>;
```

Ex:

```
delete from emp;  
rollback;      /* undo the changes */
```

3. SAVEPOINT

Specify a point in a transaction to which later you can roll back.

Syntax:

Savepoint<savepointname>;

Ex:

insert into emp (empno,ename,sal) values (109,'Sami',3000);

savepoint a;

insert into dept values (10,'Sales','Hyd');

savepoint b;

insert into salgrade values ('III',9000,12000);

Now if you give

rollback to a;

Then row from salgrade table and dept will be roll backed. Now you can commit the row inserted into emp table or rollback the transaction.

If you give

rollback to b;

Then row inserted into salgrade table will be roll backed. Now you can commit the row inserted into dept table and emp table or rollback to savepoint a or completely roll backed the transaction.

If you give

rollback;

Then the whole transaction is roll backed.

If you give

commit;

Then the whole transaction is committed and all savepoints are removed.

EXERCISE WITH SQL QUERIES

Exercise1: Consider the insurance database, where the primary keys are underlined. Construct the following SQL queries for this relational database.

person (driver-id, name, address)

car (license, model, year)

accident (report-number, date, location)

owns (driver-id, license)

participated (driver-id, car, report-number, damage-amount)

- (a) Find the total number of people who owned cars that were involved in accidents in 1989.
- (b) Find the number of accidents in which the cars belonging to “Ajay Singh” were involved.
- (c) Add a new accident to the database; assume any values for required attributes.
- (d) Delete the Mazda belonging to “Ajay Singh”.
- (e) Update the damage amount for the car with license number “AABB2000” in the accident with report number “AR2197” to Rs. 3000.
- (f) Add one more column named mobile_no to the person table.
- (g) Rename the column named mobile_no in the person table to contact_no.
- (h) Remove the column named contact_no from the person table.

Solutions-:

- (a) Find the total number of people who owned cars that were involved in accidents in 1989.

Note: this is not the same as the total number of accidents in 1989. We must count people with several accidents only once.

select count (distinct name)

from accident, participated, person

where accident.report-number = participated.report-number

and participated.driver-id = person.driver-id

and date between date '1989-00-00' and date '1989-12-31';

(b) Find the number of accidents in which the cars belonging to “Ajay Singh” were involved.

```
select count (distinct *)
from accident
where exists (select * from participated, person
where participated.driver-id = person.driver-id
and person.name = 'Ajay Singh'
and accident.report-number = participated.report-number);
```

(c) Add a new accident to the database; assume any values for required attributes.

```
insert into accident
values (4007, '2001-09-01', 'Berkeley');
```

(d) Delete the Mazda belonging to “Ajay Singh”.

```
delete from car
where model = 'Mazda' and license in
(select license
from person p, owns o
where p.name = 'Ajay Singh' and p.driver-id = o.driver-id);
```

(e) Update the damage amount for the car with license number “AABB2000” in the accident with report number “AR2197” to Rs. 3000.

```
update participated
set damage-amount = 3000
where report-number = "AR2197" and driver-id in
(select driver-id
from owns
where license = "AABB2000");
```

(f) Add one more column named mobile_no to the person table.

```
alter table person add (mobile_no number);
```

(g) Rename the column named `mobile_no` in the `person` table to `contact_no`.

```
alter table person
```

```
rename column mobile_no to contact_no;
```

(h) Remove the column named `contact_no` from the `person` table.

```
alter table person drop column contact_no;
```

Exercise2: Consider the employee database, where the primary keys are underlined. Give an expression in SQL for each of the following queries.

`employee` (employee-name, street, city)

`works` (employee-name, company-name, salary)

`company` (company-name, city)

`manages` (employee-name, manager-name)

(a) Find the names of all employees who work for First Bank Corporation.

(b) Find the names and cities of residence of all employees who work for first bank Corporation.

(c) Find the names, street addresses, and cities of residence of all employees who work for First Bank Corporation and earn more than Rs. 10,000.

(d) Find all employees in the database who live in the same cities as the companies for which they work.

(e) Find all employees in the database who live in the same cities and on the same streets as do their managers.

(f) Find all employees in the database who do not work for First Bank Corporation.

(g) Find all employees in the database who earn more than each employee of Small Bank Corporation.

(h) Assume that the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.

(i) Find all employees who earn more than the average salary of all employees of their company.

(j) Find the company that has the most employees.

(k) Find the company that has the smallest payroll.

(l) Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.

Solutions-:

(a) Find the names of all employees who work for First Bank Corporation.

```
select employee-name  
from works  
where company-name = 'First Bank Corporation';
```

(b) Find the names and cities of residence of all employees who work for First Bank Corporation.

```
selecte.employee-name, city  
from employee e, works w  
wherew.company-name = 'First Bank Corporation' and  
w.employee-name = e.employee-name;
```

(c) Find the names, street address, and cities of residence of all employees who work for First Bank Corporation and earn more than Rs. 10,000.

```
select *  
from employee  
where employee-name in  
(select employee-name  
from works  
where company-name = 'First Bank Corporation' and salary >  
10000);
```

(d) Find all employees in the database who live in the same cities as the companies for which they work.

```
selecte.employee-name  
from employee e, works w, company c  
wherew.employee-name = w.employee-name and e.city = c.city and
```

w.company -name = c.company-name;

(e) Find all employees in the database who live in the same cities and on the same streets as do their managers.

*select P.employee-name
from employee P, employee R, manages M
where P.employee-name = M.employee-name and
M.manager-name = R.employee-name and
P.street = R.street and P.city = R.city;*

(f) Find all employees in the database who do not work for First Bank Corporation.

*select employee-name
from works
where company-name <> 'First Bank Corporation';*

(g) Find all employees in the database who earn more than every employee of Small Bank Corporation.

*select employee-name
from works
where salary > all
(select salary
from works
where company-name = 'Small Bank Corporation');*

(h) Assume that the companies may be located in several cities. Find all companies located in every city in which Small Bank Corporation is located.

*select T.company-name
from company T
where (select R.city
from company R
where R.company-name = T.company-name)*


```
select company-name  
from works  
group by company-name  
having avg (salary) > (select avg (salary)  
from works  
where company-name = 'First Bank Corporation');
```

REFERENCES

1. H.f.Korth and Silberschatz: Database Systems Concepts, McGraw Hill
2. Ramakrishnan and Gehrke: Database Management System, McGraw Hill
3. C.J. Date: Data Base Design, Addison Wesley
4. www.sourcepower.blogspot.com
5. www.filestube.com

Viva Questions:

1. What is database?

A database is a collection of information that is organized. So that it can easily be accessed, managed, and updated.

2. What is DBMS?

DBMS stands for Database Management System. It is a collection of programs that enables user to create and maintain a database.

3. What is a Database system?

The database and DBMS software together is called as Database system.

4. What are the advantages of DBMS?

- I. Redundancy is controlled.
- II. Providing multiple user interfaces.
- III. Providing backup and recovery
- IV. Unauthorized access is restricted.
- V. Enforcing integrity constraints.

5. What is normalization?

It is a process of analysing the given relation schemas based on their Functional Dependencies (FDs) and primary key to achieve the properties (1).Minimizing redundancy, (2). Minimizing insertion, deletion and update anomalies.

6. What is Data Model?

A collection of conceptual tools for describing data, data relationships data semantics and constraints.

7. What is E-R model?

This data model is based on real world that consists of basic objects called entities and of relationship among these objects. Entities are described in a database by a set of attributes.

8. What is Object Oriented model?

This model is based on collection of objects. An object contains values stored in instance variables with in the object. An object also contains bodies of code that operate on the object. These bodies of code are called methods. Objects that contain same types of values and the same methods are grouped together into classes.

9. What is an Entity?

An entity is a thing or object of importance about which data must be captured.

10. What is DDL (Data Definition Language)?

A data base schema is specifies by a set of definitions expressed by a special language called DDL.

11. What is DML (Data Manipulation Language)?

This language that enable user to access or manipulate data as organised by appropriate data model. Procedural DML or Low level: DML requires a user to specify what data are needed and how to get those data. Non-Procedural DML or High level: DML requires a user to specify what data are needed without specifying how to get those data

12. What is DML Compiler?

It translates DML statements in a query language into low-level instruction that the query evaluation engine can understand.

13. What is Query evaluation engine?

It executes low-level instruction generated by compiler.

14. What is Functional Dependency?

Functional Dependency is the starting point of normalization. Functional Dependency exists when a relation between two attributes allows you to uniquely determine the corresponding attribute's value.

15. What is 1 NF (Normal Form)?

The first normal form or 1NF is the first and the simplest type of normalization that can be implemented in a database. The main aims of 1NF are to:

1. Eliminate duplicative columns from the same table.
2. Create separate tables for each group of related data and identify each row with a unique column (the primary key).

16. What is Fully Functional dependency?

A functional dependency $X \twoheadrightarrow Y$ is full functional dependency if removal of any attribute A from X means that the dependency does not hold any more.

17. What is 2NF?

A relation schema R is in 2NF if it is in 1NF and every non-prime attribute A in R is fully functionally dependent on primary key.

18. What is 3NF?

A relation is in third normal form if it is in Second Normal Form and there are no functional (transitive) dependencies between two (or more) non-primary key attributes.

19. What is BCNF (Boyce-Codd Normal Form)?

A table is in Boyce-Codd normal form (BCNF) if and only if it is in 3NF and every determinant is a candidate key.

20. What is 4NF?

Fourth normal form requires that a table be BCNF and contain no multi-valued dependencies.

21. What is 5NF?

A table is in fifth normal form (5NF) or Project-Join Normal Form (PJNF) if it is in 4NF and it cannot have a lossless decomposition into any number of smaller tables.

22. What is a query?

A query with respect to DBMS relates to user commands that are used to interact with a data base.

23. What is meant by query optimization?

The phase that identifies an efficient execution plan for evaluating a query that has the least estimated cost is referred to as query optimization.

24. What is an attribute?

It is a particular property, which describes the entity.

25. What is RDBMS?

Relational Data Base Management Systems (RDBMS) are database management systems that maintain data records and indices in tables.

26. What's difference between DBMS and RDBMS?

DBMS provides a systematic and organized way of storing, managing and retrieving from collection of logically related information. RDBMS also provides what DBMS provides but above that it provides relationship integrity.

27. What is SQL?

SQL stands for Structured Query Language. SQL is an ANSI (American National Standards Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and update data in a database.

28. What is Stored Procedure?

A stored procedure is a named group of SQL statements that have been previously created and stored in the server database.

29. What is a view?

A view may be a subset of the database or it may contain virtual data that is derived from the database files but is not explicitly stored.

30. What is Trigger?

A trigger is a SQL procedure that initiates an action when an event (INSERT, DELETE or UPDATE) occurs.

31. What is Index?

An index is a physical structure containing pointers to the data.

32. What is extension and intension?

Extension -It is the number of tuples present in a table at any instance. This is time dependent.

Intension -It is a constant value that gives the name, structure of table and the constraints laid on it.

33. What do you mean by atomicity and aggregation?

Atomicity-Atomicity states that database modifications must follow an “all or nothing” rule. Each transaction is said to be “atomic.” If one part of the transaction fails, the entire transaction fails.

Aggregation - A feature of the entity relationship model that allows a relationship set to participate in another relationship set. This is indicated on an ER diagram by drawing a dashed box around the aggregation.

34. What is RDBMS KERNEL?

Two important pieces of RDBMS architecture are the kernel, which is the software, and the data dictionary, which consists of the system- level data structures used by the kernel to manage the database.

35. Name the sub-systems of a RDBMS?

I/O, Security, Language Processing, Process Control, Storage Management, Logging and Recovery, Distribution Control, Transaction Control, Memory Management, Lock Management.

36. How do you communicate with an RDBMS?

You communicate with an RDBMS using Structured Query Language (SQL)

37. Disadvantage in File Processing System?

- Data redundancy & inconsistency.
- Difficult in accessing data.
- Data isolation.
- Data integrity.
- Concurrent access is not possible.
- Security Problems.

38. What is VDL (View Definition Language)?

It specifies user views and their mappings to the conceptual schema.

39. What is SDL (Storage Definition Language)?

This language is to specify the internal schema. This language may Specify the mapping between two schemas.

40. Describe concurrency control?

Concurrency control is the process managing simultaneous operations against a database so that database integrity is no compromised. There are two approaches to concurrency control.

The pessimistic approach involves locking and the optimistic approach involves versioning.

41. Describe the difference between homogeneous and heterogeneous distributed database?

A homogenous database is one that uses the same DBMS at each node. A heterogeneous database is one that may have a different DBMS at each node.

42. What is a distributed database?

A distributed database is a single logical database that is spread across more than one node or locations that are all connected via some communication link.

43. Explain the difference between two and three-tier architectures?

Three-tier architecture includes a client and two server layers.

The application code is stored on the application server and the database is stored on the database server. A two-tier architecture includes a client and one server layer. The database is stored on the database server.

44. Briefly describe the three types of SQL commands?

Data definition language commands are used to create, alter, and drop tables. Data manipulation commands are used to insert, modify, update, and query data in the database. Data control language commands help the DBA to control the database.

45. List some of the properties of a relation?

Relations in a database have a unique name and no multivalued attributes exist. Each row is unique and each attribute within a relation has a unique name. The sequence of both columns and rows is irrelevant.

46. Explain the differences between an intranet and an extranet?

An Internet database is accessible by everyone who has access to a Web site. An intranet database limits access to only people within a given organization.

47. What is SQL Deadlock?

Deadlock is a unique situation in a multi user system that causes two or more users to wait indefinitely for a locked resource.

48. What is a Catalog?

A catalog is a table that contains the information such as structure of each file, the type and storage format of each data item and various constraints on the data. The information stored in the catalog is called Metadata.

49. What is data ware housing & OLAP?

Data warehousing and OLAP (online analytical processing) systems are the techniques used in many companies to extract and analyze useful information from very large databases for decision making.

50. Describe the three levels of data abstraction?

Physical level: The lowest level of abstraction describes how data are stored.

Logical level: The next higher level of abstraction, describes what data are stored in database and what relationship among those data.

View level: The highest level of abstraction describes only part of entire database.