EXPERIMENT 1

Aim - To implement E-R diagram of an enterprise.

Theory- Database management system is the software system that allows users to define, create and maintain a database and provides controlled access to the data.

A Database Management System (DBMS) is basically a collection of programs that enables users to store, modify, and extract information from a database as per the requirements. DBMS is an intermediate layer between programs and the data. Programs access the DBMS, which then accesses the data. There are different types of DBMS ranging from small systems that run on personal computers to huge systems that run on mainframes.

There are two techniques used for the purpose of data base designing from the system requirements. These are:

• Top down Approach known as Entity-Relationship Modelling

• Bottom Up approach known as Normalization.

Here we studied only ER diagram.

ER Diagram-

In Entity-Relationship model a database is modelled as a collection of entities and relationship among entities. The ER model views the real world as a construct of entities and association between entities.

The symbols used for the basic ER constructs are:

Entities

An entity is an object whose information is stored in the database. It is distinguishable from other objects. For example: specific person, company, event, plant. In other words, anything that may 'have an independent existence and about which we intend to collect data is known as Entity. It is also known as Entity type. Entities are represented by labeled rectangles. The label is the name of the entity. Entity names should be singular nouns.

Relationships

A Relationship represents an association between two or more entities. Relationships are classified in terms of degree, connectivity, cardinality, and existence. An example of a relationship would be:

• Employees are assigned to projects.

• Projects have subtasks

• Departments manage one or more projects

A solid line connecting two entities represents relationships. The name of the relationship is written above the line. Relationship names should be verbs and diamonds sign is used to represent relationship sets.

Attributes

Attributes describe the properties of the entity of which they are associated. A particular instance of an attribute is a value. For example, "Ram" is one value of the attribute Name.The domain of an attribute is the collection of all possible values an attribute can have. The domain of Name is a character string. Attributes are represented by Ellipses.

Attribute

We can classify attributes as following types:

• Simple

• Composite

• Single-values

• Multi-values

• Derived

Connectivity and Cardinality

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many". The cardinality of a relationship is the actual number of related occurrences for each of the two entities.

The basic types of connectivity for relations are:

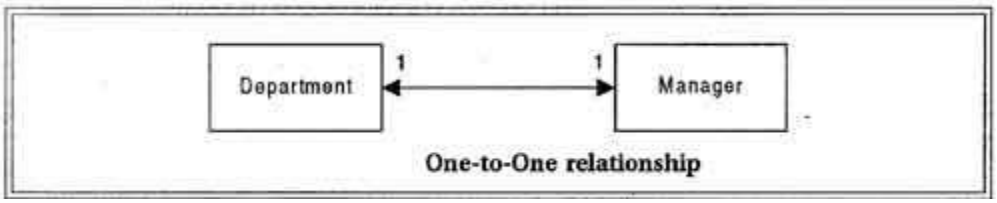
• One to One (1: 1).

• One to Many (1:M)

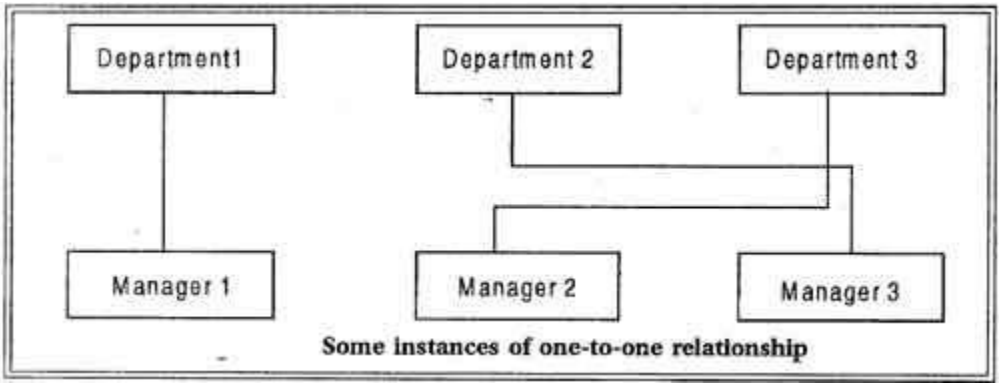
• Many to One (M:l)

• Many to Many (M:M)

1. **A one-to-one (1:1)** relationship is when at most one instance of an entity A is associated with one instance of entity B. For example, "employees in the company are each assigned their own office. For each employee there exists a unique office and for each office there exists a unique employee.



One-to-One relationship

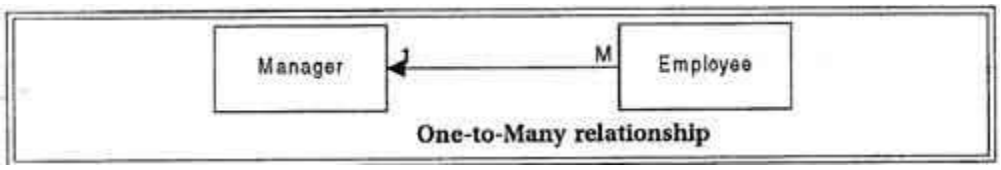


Some instances of one-to-one relationship

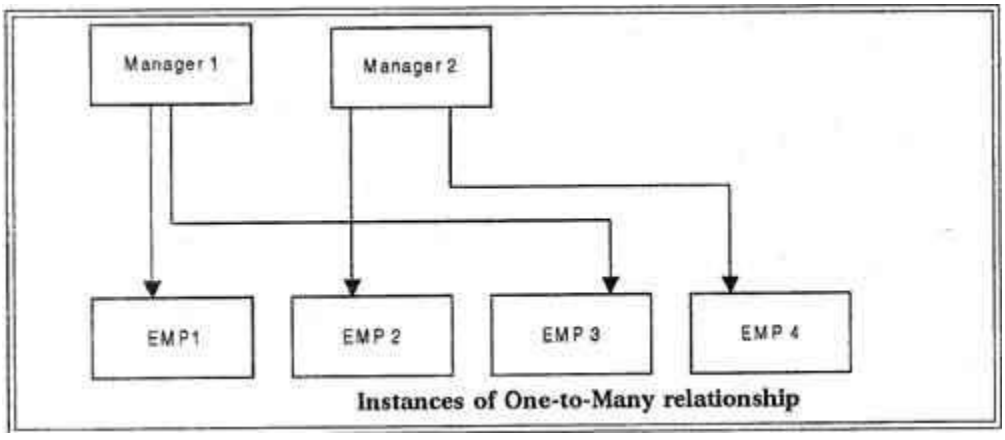
2. **A one-to-many (1:M)** relationships is when for one instance of entity A, there are zero, one, or many instances of entity B, but for one instance of entity B, there is only one instance of entity A. Examples of l:M relationships are:

   • A department has many employees.

   • Each employee is assigned to one department.



One-to-Many relationship



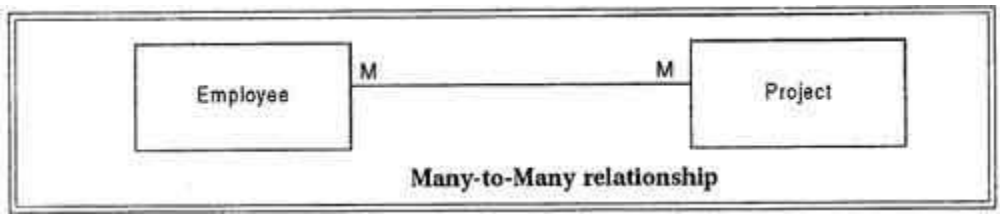
Instances of One-to-Many relationship

3. **A many-to-one (M: 1)** relationships is when for one instance of entity A is associated with at most one instances of entity B, but for one instance of entity B, there may be any number of instances of entity A. Examples of M: 1 relationships is

   • Many employees one department.

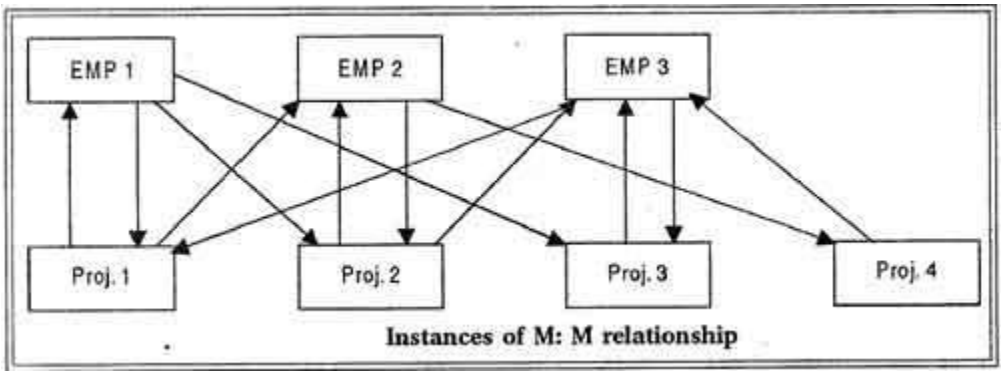
4. **A many-to-many (M: M)** relationship, sometimes called non-specific, is when for one instance of entity A, there are zero, one, or many instances of entity B and for one instance of entity B there are zero, one, or many instances of entity A. Examples are:

- Employees can be assigned to no more than two projects at the same time.
- Projects must have assigned at least three employees.
-



Many-to-Many relationship

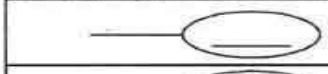
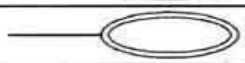
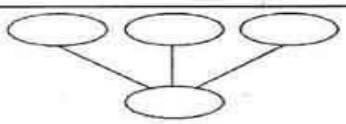
-



Instances of M: M relationship

The symbols used to design an ER diagram are shown.

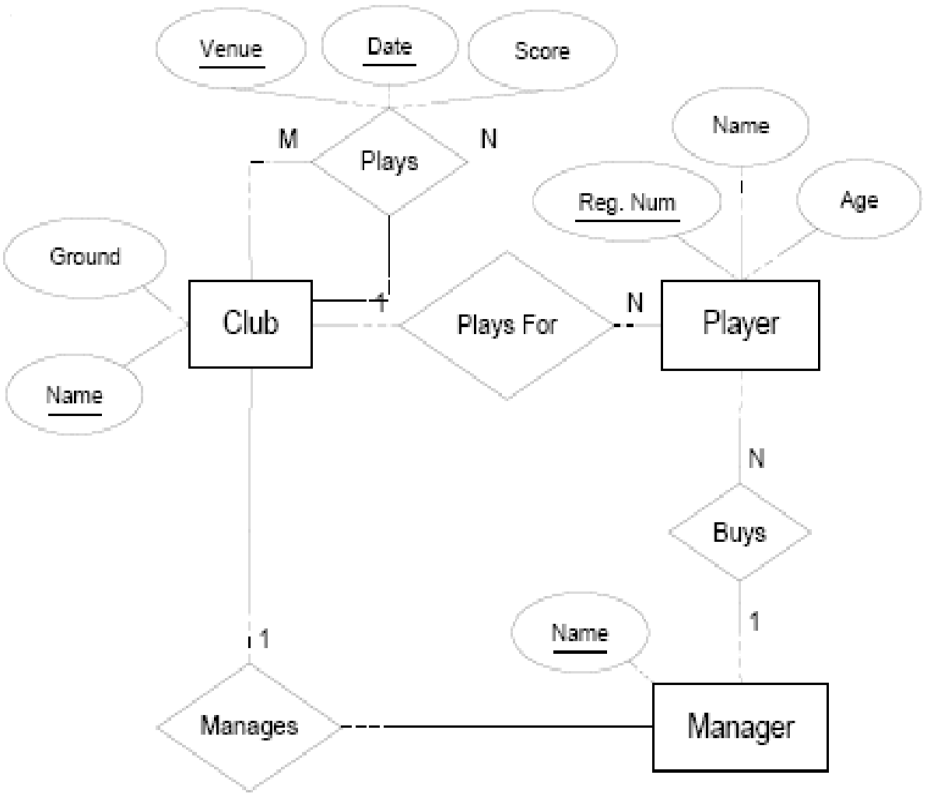
Symbol	Meaning
	Entity Type
	Weak Entity Type
	Relationship Type
	Identifying Relationship Type
	Attribute

Symbol	Meaning
	Key Attribute
	Multivalued Attribute
	Composite Attribute
	Derived Attribute
	Total Participation of E2 in R
	Cardinality Relation 1 : N for E1 : E2 in R

. Symbols used in E-R diagram

ER-EXAMPLE –Football Club

A football club has a name and a ground and is made up of players. A player can play foronly one club and a manager, represented by his name manages a club. A footballer has aregistration number, name and age. A club manager also buys players. Each club playsagainst each other club in the league and matches have a date, venue and score.



Conclusion:

Thus we have implemented E-R diagram

EXPERIMENT NO 1 : VIVA QUESTIONS

1. What is DBMS?
2. Advantage of database
3. What is data model?
4. What is object oriented model?
5. What is an entity?
6. What is an entity type?

7 A relational database consists of a collection of
   a) Tables
   b) Fields
   c) Records
   d) Keys

8. A _____ in a table represents a relationship among a set of values.
   a) Column
   b) Key
   c) Row
   d) Entry

9. The term _____ is used to refer to a row.
   a) Attribute
   b) Tuple
   c) Field
   d) Instance

10. The term attribute refers to a _____ of a table.
   a) Record
   b) Column
   c) Tuple
   d) Key

EXPERIMENT 2

Aim - Implementation of DDL commands.

Theory-

DBMS **A** database management system is the software system that allows users to define, create and maintain a database and provides controlled access to the data.

A Database Management System (DBMS) is basically a collection of programs that enables users to store, modify, and extract information from a database as per the requirements. DBMS is an intermediate layer between programs and the data. Programs access the DBMS, which then accesses the data. There are different types of DBMS ranging from small systems that run on personal computers to huge systems that run on mainframes.

To create a database and to retrieve the information from the database using SQL queries.

SQL statements are divided into two major categories: data definition language (DDL) and data manipulation language (DML).

Data Definition Language (DDL) statements are used to define the database structure or schema. Some examples:

COMMANDS –

1. **CREATE TABLE-**

The create table statement is used to create a new table. The CREATE TABLE statement defines a table. The definition must include its name and the names and attributes of its columns. The definition can include other attributes of the table, such as its primary key or check constraints. This statement comes under the DDL statement.

Syntax:

SQL> create table tablename

(Column1 data type [default exp],

Column2 data type [default exp] ,

Column3 data type [default exp] ,

);

To create a new table, enter the keywords create table followed by the table name, followed by an open parenthesis, followed by the first column name, followed by the data type for that

column, followed by any optional constraints, and followed by a closing parenthesis. It is important to make sure you use an open parenthesis before the beginning table and a closing parenthesis after the end of the last column definition. Make sure you separate each column definition with a comma. All SQL statements should end with a";".

Example:

SQL> create table employee

(Name varchar2 (15),

Eid number (6),

Age number (3),

City varchar2 (30),

State varchar2 (20)

);

Table created.

If you want to see the structure of the table then use the following command on SQL prompt.

SQL > DESC employee;

As you can see that this command displays the entire attribute names with their data type and if constraints are imposed then their detail also.

2.**SQL ALTER**

After the table(s) are created and begin to be used, requirements are likely to occur which requires modifications in the structure of the table, such as adding new columns to the table, modifying and existing column's definition etc. So these modifications can be performed using the ALTER table DDL statement. Tills statement changes the structure of the table and not its contents. There are many types of modifications that can be made to the structure of the tables. Using ALTER TABLE statement, you can make modifications such as

Add or drop one or more columns to *i* from existing table.

Increase or decrease the width of the existing column.

Change an existing column from mandatory to optional or vice versa.

The syntax is ALTER TABLE <tablename>

ADD (column_specification I , ...

column_specification);

Where tablename corresponds to the name of the table, column_specification corresponds to the valid specification for a column (columnname and datatype)

To add *address* column into the INSTRUCTOR table with column width 30 and data type varchar2 is shown below.

SQL> ALTER TABLE instructor

ADD (address VARCHAR2(30));

Table altered.

Instead of adding columns one by one we can add multiple columns in the table. To add address and salary columns into the INSTRUCTOR table, the statement is

SQL> ALTER TABLE instructor

ADD (address VARCHAR2 (30),

Salary NUMBER (10, 2));

Table altered.

3. DROP TABLE

The SQL DROP command is used to remove an object from the database. If you drop a table, all the rows in the table is deleted and the table structure is removed from the database. Once a table is dropped we cannot get it back, When a table is dropped all the references to the table will not be valid.

Syntax: SQL> Drop Table Tablename [cascade constraint];

Example:

SQL> DROP TABLE Persons;

Table dropped

Conclusion: Thus the creation of database and the SQL queries to retrieve information from the database has been implemented and the output was verified using create, alter, drop commands are used as DDL commands.

EXPERIMENT NO 2 : VIVA QUESTIONS

1. What is weak entity set?

2 .What is relationship?

3. What is DDL?

4.What is DML?

5. For each attribute of a relation, there is a set of permitted values, called the _____ of that attribute.
a) Domain
b) Relation
c) Set
d) Schema

6. Database _____ , which is the logical design of the database, and the database _____,which is a snapshot of the data in the database at a given instant in time.
a) Instance, Schema
b) Relation, Schema
c) Relation, Domain
d) Schema, Instance

7.Course(course_id,sec_id,semester)
Here the course_id,sec_id and semester are _____ and course is a _____ .
a) Relations, Attribute
b) Attributes, Relation
c) Tuple, Relation
d) Tuple, Attributes

8. Department (dept name, building, budget) and Employee (employee_id , name, dept name,salary)
Here the dept_name attribute appears in both the relations .Here using common attributes in relation schema is one way of relating _____ relations.
a) Attributes of common
b) Tuple of common
c) Tuple of distinct
d) Attributes of distinct

9. A domain is atomic if elements of the domain are considered to be _____ units.
a) Different
b) Indivisbile
c) Constant
d) Divisible

10. The tuples of the relations can be of _____ order.
a) Any
b) Same
c) Sorted
d) Constant

EXPERIMENT 3

Aim- Implementation of DML commands.

Theory-

SQL (Standard Query Language) is born as a result of the mathematical work of Codd, who founded the work of relational databases, three types of manipulations on the database:

1 The maintenance of tables: create, delete, and modify the table structure.
2 The manipulation of databases: Selecting, modifying, deleting records.
3 The management of access rights to tables: Data control: access rights, commit the changes.

Data Manipulation Language (DML) statements are used for managing data within schema objects. Some examples:

1. **SELECT-**

   A select statement is used to select information from one or more tables. SQL statements can be on one or more lines. SQL is not case sensitive. SELECT is the same as select.

   SQL SELECT Syntax

   SELECT * FROM table_name

   If all columns of a table should be selected, the asterisk symbol * can be used to denote all attributes. The query retrieves all tuples with all columns from the table STUD. To illustrate the usage of the SELECT command we are going to use the STUD table:

2. **SQL INSERT INTO -**

   The INSERT statement is used to add new row to a table. To insert new row(s) into a table, the table must be in your own schema or you must have INSERT privilegeon the table. Only one row is inserted at a time with this syntax.

   The Syntax is

   INSERT INTO <table_name> values (--------- <columnname1>,---------- <columnname2>, ...<columnnameN>);

   In order to insert data, let us first create an INSTRUCTOR table using CREATE TABLE command.

Example:

CREATION OF TABLE

SQL>create table stud (sname varchar2(30), sid varchar2(10), sage number(10), sarea varchar2(20), sdeptvarchar2(20));

Table created.

INSERTION OF VALUES INTO THE TABLE

SQL> insert into stud values ('ashwin',101,19,'anna nagar','aeronautical');

1 row created.

SQL> insert into stud values ('bhavesh',102,18,'nungambakkam','marine');

1 row created.

SQL> insert into stud values ('pruthvik',103,20,'anna nagar','aerospace');

1 row created.

SQL> insert into stud values ('charith',104,20,'kilpauk','mechanical');

1 row created.

SQL> select * from stud;

SNAME	SID	SAGE	SAREA	SDEPT
ashwin	101	19	anna nagar	aeronautical
bhavesh	102	18	nungambakkam	marine
pruthvik	103	20	anna nagar	aerospace
charith	104	20	kilpauk	mechanical

3. **UPDATE-**

Quite often it is required to make changes or modifications in the records of the table, so in order to make these changes, the UPDATE statement is used. With this statement, the user can modify the existing data stored in the table. It can update zero or more rows in a table. To update rows in table, it must be in your own schema or you must have update privilege on the table.

The syntax is

UPDATE <tablename>

SET <columnnamel> = <expression> [, <column narne2>= <expression>]

..... [,<columnnameN>=<expression>]

[WHERE condition];

Suppose that we insert a column SALARY (Number (8, 2)) into the INSTRUCTOR table and we want to set the salary of each instructor to be 10000. Then the update statement will be

Update INSTRUCTOR

SET SALARY = 10000;

5 ROW UPDATED

On execution this will modify the salary of each instructor to 10000. Now suppose that we want to increase the salary of each INSTRUCTOR with a post'READER' by 5000. The statement will be

Update INSTRUCTOR

SALARY = SALARY + 5000;

WHERE POST = 'READER';

Row Updated

4. **SQL DELETE**

The **DELETE** statement is used to delete rows in a table. To do so, we can use the DELETE FROM command. An **SQL DELETE** statement removes one or more records from a table. A subset may be defined for deletion using a condition, otherwise all records are removed.

Delete from <table> [where <condition>];

If the where clause is omitted, all tuples are deleted from the table. An alternative command for deleting all tuples from a table is the truncate table <table> command. However, in this case, the deletions cannot be undone.

SQL > DELETE FROM stu

WHERE ROLL=2214;

1 row deleted.

• Now we will delete all the records from the stu table.

SQL > DELETE FROM stu;

OR

SQL > DELETE stu;

RENAMING THE TABLE 'STUD'
SQL> rename stud to studs;
Table renamed.

Conclusion: Thus the creation of database and the SQL queries to retrieve information from the database has been implemented and the output was verified using SELECT, INSERT, UPDATE, DELETE commands are used as DML commands

EXPERIMENT NO 3 : VIVA QUESTIONS

1. A Database Management System (DBMS) is
A. Collection of interrelated data
B. Collection of programs to access data
C. Collection of data describing one particular enterprise
D. All of the above

2. Which of the following is not a level of data abstraction?
A. Physical Level
B. Critical Level
C. Logical Level
D. View Level

3. Disadvantages of File systems to store data is:
A. Data redundancy and inconsistency
B. Difficulty in accessing data
C. Data isolation
D. All of the above

4. In an Entity-Relationship Diagram Rectangles represents
A. Entity sets
B. Attributes
C. Database
D. Tables

5. Which of the following is not a Storage Manager Component?
A. Transaction Manager
B. Logical Manager
C. Buffer Manager
D. File Manager

6. Data Manipulation Language enables users to
A. Retrieval of information stored in database
B. Insertion of new information into the database
C. Deletion of information from the database
D. All of the above

7. Which of the following is not an Schema?
A. Database Schema
B. Physical Schema
C. Critical Schema
D. Logical Schema

8. Which of the following is Database Language?
A. Data Definition Language
B. Data Manipulation Language
C. Query Language
D. All of the above

9. Which of the following in not a function of DBA?
A. Network Maintenance
B. Routine Maintenance
C. Schema Definition
D. Authorization for data access

10. Which of the following is a Data Model?
A. Entity-Relationship model
B. Relational data model
C. Object-Based data model
D. All of the above

Experiment No: 04

Aim: Implementation of SQL Constraints.

Theory:

Constraints are used to limit the type of data that can go into a table.Constraints can be specified when a table is created (with the CREATE TABLE statement) or after the table is created (with the ALTER TABLE statement).

We will focus on the following constraints:

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

1. NOT NULL:

The NOT NULL constraint enforces a column to NOT accept NULL values.The NOT NULL constraint enforces a field to always contain a value. This means that you cannot insert a new record, or update a record without adding a value to this field.The following SQL enforces the "P_Id" column and the "LastName" column to not accept NULL values:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255)
);
```

2. UNIQUE:

The UNIQUE constraint uniquely identifies each record in a database table.The UNIQUE and PRIMARY KEY constraints both provide a guarantee for uniqueness for a column or set of columns.A PRIMARY KEY constraint automatically has a UNIQUE constraint defined on it.Note that you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.The following SQL creates a UNIQUE constraint on the "P_Id" column when the "Persons" table is created:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
```

Address varchar(255),
City varchar(255),
UNIQUE (P_Id)
);

3. PRIMARY KEY:

The PRIMARY KEY constraint uniquely identifies each record in a database table.Primary keys must contain unique values.A primary key column cannot contain NULL values.Each table should have a primary key, and each table can have only ONE primary key.The following SQL creates a PRIMARY KEY on the "P_Id" column when the "Persons" table is created:

CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
PRIMARY KEY (P_Id)
);

4. FOREIGN KEY:

A FOREIGN KEY in one table points to a PRIMARY KEY in another table.Let's illustrate the foreign key with an example. Look at the following two tables:

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	2
4	24562	1

Note that the "P_Id" column in the "Orders" table points to the "P_Id" column in the "Persons" table. The "P_Id" column in the "Persons" table is the PRIMARY KEY in the "Persons" table. The "P_Id" column in the "Orders" table is a FOREIGN KEY in the "Orders" table. The

FOREIGN KEY constraint is used to prevent actions that would destroy links between tables. The FOREIGN KEY constraint also prevents that invalid data form being inserted into the foreign key column, because it has to be one of the values contained in the table it points to.

The following SQL creates a FOREIGN KEY on the "P_Id" column when the "Orders" table is created:

```
CREATE TABLE Orders
(
O_Id int NOT NULL,
OrderNo int NOT NULL,
P_Id int,
PRIMARY KEY (O_Id),
FOREIGN KEY (P_Id) REFERENCES Persons(P_Id)
);
```

5. CHECK:

The CHECK constraint is used to limit the value range that can be placed in a column.If you define a CHECK constraint on a single column it allows only certain values for this column.If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

SQL CHECK Constraint on CREATE TABLE

The following SQL creates a CHECK constraint on the "P_Id" column when the "Persons" table is created. The CHECK constraint specifies that the column "P_Id" must only include integers greater than 0.

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255),
CHECK (P_Id>0)
);
```

SQL CHECK Constraint on ALTER TABLE

To create a CHECK constraint on the "P_Id" column when the table is already created, use the following SQL:

```
ALTER TABLE Persons
ADD CHECK (P_Id>0);
```

6. DEFAULT:

The DEFAULT constraint is used to insert a default value into a column.The default value will be added to all new records, if no other value is specified.

SQL DEFAULT Constraint on CREATE TABLE

The following SQL creates a DEFAULT constraint on the "City" column when the “Persons" table is created:

```
CREATE TABLE Persons
(
P_Id int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Address varchar(255),
City varchar(255) DEFAULT 'Sandnes'
);
```

SQL DEFAULT Constraint on ALTER TABLE

To create a DEFAULT constraint on the "City" column when the table is already reated, use the following SQL:

```
ALTER TABLE Persons
ALTER City SET DEFAULT 'SANDNES';
```

To DROP a DEFAULT Constraint

To drop a DEFAULT constraint, use the following SQL:

```
ALTER TABLE Persons
ALTER City DROP DEFAULT;
```

Conclusion:

Thus NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, DEFAULT are SQL constraints are implemented and the output was verified.

EXPERIMENT NO 04: VIVA QUESTIONS

1. Which of the following represents a relationship among a set of values.
A. A Row
B. A Table
C. A Field
D. A Column

2. Column header is refer as
A. Table
B. Relation
C. Attributes
D. Domain

3. A Relation is a
A. Subset of a Cartesian product of a list of attributes
B. Subset of a Cartesian product of a list of domains
C. Subset of a Cartesian product of a list of tuple
D. Subset of a Cartesian product of a list of relations

4. In mathematical term Table is referred as
A. Relation
B. Attribute
C. Tuple
D. Domain

5. In mathematical term Row is referred as
A. Relation
B. Attribute
C. Tuple
D. Domain

6. _____ allows us to identify uniquely a tuple in the relation.
A. Superkey
B. Domain
C. Attribute
D. Schema

7. Minimal Superkeys are called
A. Schema keys
B. Candidate keys
C. Domain keys
D. Attribute keys

8. Which of the following is not Modification of the Database
A. Deletion
B. Insertion
C. Sorting
D. Updating

9. Which of the following is Relation-algebra Operation
A. Select
B. Union

C. Rename
D. All of the above

10. Which of the following in not Outer join?
A. Left outer join
B. Right outer join
C. Full outer join
D. All of the above

Experiment No: 05

Aim: Implementation of SQL Logical operators.

Theory:

There are three Logical Operators namely, AND, OR, and NOT. These operators compare two conditions at a time to determine whether a row can be selected for the output.
When retrieving data using a SELECT statement, you can use logical operators in the WHERE clause, which allows you to combine more than one condition.

1. OR Logical Operator:

If you want to select rows that satisfy at least one of the given conditions, you can use the logical operator, OR.
For example: if you want to find the names of students who are studying either Maths or Science, the query would be like

```
SELECT first_name, last_name, subject
FROM student_details
WHERE subject = 'Maths' OR subject = 'Science';
```

2. AND Logical Operator:

If you want to select rows that must satisfy all the given conditions, you can use the logical operator, AND.

For Example: To find the names of the students between the age 10 to 15 years, the query would be like:

```
SELECT first_name, last_name, age
FROM student_details
WHERE age >= 10 AND age <= 15;
```

3. NOT Logical Operator:

If you want to find rows that do not satisfy a condition, you can use the logical operator, NOT. NOT results in the reverse of a condition. That is, if a condition is satisfied, then the row is not returned.

For example: If you want to find out the names of the students who do not play football, the query would be like:

```
SELECT first_name, last_name, games
FROM student_details
WHERE NOT games = 'Football';
```

Conclusion: Thus AND, OR, NOT are the logical operators in SQL are implemented and the output was verified.

EXPERIMENT NO 5 : VIVA QUESTIONS

1. Who proposed the relational model?
A. Bill Gates
B. E.F. Codd
C. Herman Hollerith
D. Charles Babbage

2. Set of premitted values of each attribute is called
A. Domain
B. Tuple
C. Relation
D. Schema

3. Which of the following in true regarding Null Value?
A. Null = 0
B. Null < 0 C. Null > 0
D. Null <> 0

4. Logical design of database is called
A. Database Instance
B. Database Snapshot
C. Database Schema
D. All of the above

5. Snapshot of the dta in the database at a given instant of time is called
A. Database Schema
B. Database Instance
C. Database Snapshot
D. All of the above

6. Which of the following is not Unary operation?
A. Select
B. Project
C. Rename
D. Union

7. Which of the following is not binary operation?
A. Union
B. Project

C. Set Difference
D. Cartesian Product

8. Which of the following is correct regarding Aggregate functions?
A. it takes a list of values and return a single values as result
B. it takes a list of values and return a list of values as result
C. it takes a single value and returns a list of values as result
D. it takes a single value and returns a single value as result

9. The Primary key must be
A. Non Null
B. Unique
C. Option A or B
D. Option A and B

10. A command to remove a relation from an SQL database
A. Delete table <table name>
B. Drop table <table name>
C. Erase table <table name>
D. Alter table <table name>

Experiment No: 06

Aim: Implementation of Aggregate Functions

Theory:

SQL has many built-in functions for performing calculations on data. SQL aggregate functions return a single value, calculated from values in a column.

Useful aggregate functions:

- AVG() - Returns the average value
- COUNT() - Returns the number of rows
- FIRST() - Returns the first value
- LAST() - Returns the last value
- MAX() - Returns the largest value
- MIN() - Returns the smallest value
- SUM() - Returns the sum

1. **AVG() –**
   The AVG() function returns the average value of a numeric column.

   SQL AVG() Syntax:

   SELECT AVG(column_name) FROM table_name;

   Example:

   We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

   Now we want to find the average value of the "OrderPrice" fields.We use the following SQL statement:

   SELECT AVG(OrderPrice) AS Order Average FROM Orders;

2. **COUNT() –**
The COUNT() function returns the number of rows that matches a specified criteria.The COUNT(column_name) function returns the number of values (NULL values will not be counted) of the specified column:

SQL COUNT(column_name) Syntax:

SELECT COUNT(column_name) FROM table_name;

Example

The COUNT(*) function returns the number of records in a table:

SELECT COUNT(*) FROM table_name;

We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

Now we want to count the number of orders from "Customer Nilsen".We use the following SQL statement:
SELECT COUNT(Customer) AS CustomerNilsen FROM Orders
WHERE Customer='Nilsen';

3. **FIRST() –**
The FIRST() function returns the first value of the selected column.
SQL FIRST() Syntax:
SELECT FIRST (column_name) FROM table_name;
SQL FIRST() Example: We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

Now we want to find the first value of the "OrderPrice" column.We use the following SQL statement:

SELECT FIRST(OrderPrice) AS FirstOrderPrice FROM Orders;

4. **LAST() –**
   The LAST() function returns the last value of the selected column.
   SQL LAST() Syntax:
   SELECT LAST(column_name) FROM table_name;
   SQL LAST() Example:

   We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

Now we want to find the last value of the "OrderPrice" column.We use the following SQL statement:

SELECT LAST(OrderPrice) AS LastOrderPrice FROM Orders;

MAX() –

The MAX() function returns the largest value of the selected column.

SQL MAX() Syntax:
SELECT MAX(column_name) FROM table_name;
SQL MAX() Example:
We have the following "Orders" table:

Now we want to find the largest value of the "OrderPrice" column.
We use the following SQL statement:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

SELECT MAX(OrderPrice) AS LargestOrderPrice FROM Orders;

5. **MIN() –**

The MIN() function returns the smallest value of the selected column.

SQL MIN() Syntax:
SELECT MIN(column_name) FROM table_name;
SQL MIN() Example:

We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

Now we want to find the smallest value of the "OrderPrice" column.We use the following SQL statement:

SELECT MIN(OrderPrice) AS SmallestOrderPrice FROM Orders;

6. **SUM() –**

The SUM() function returns the total sum of a numeric column.

SQL SUM() Syntax:
SELECT SUM(column_name) FROM table_name;
SQL SUM() Example:

We have the following "Orders" table:

O_Id	OrderDate	OrderPrice	Customer
1	2008/11/12	1000	Hansen
2	2008/10/23	1600	Nilsen
3	2008/09/02	700	Hansen
4	2008/09/03	300	Hansen
5	2008/08/30	2000	Jensen
6	2008/10/04	100	Nilsen

Now we want to find the sum of all "OrderPrice" fields".We use the following SQL statement:

SELECT SUM(OrderPrice) AS OrderTotal FROM Orders;

Conclusion:

Thus we have implemented that AVG(), COUNT(), FIRST(), LAST(), MAX(), MIN(),SUM() are aggregate functions of SQL and the output was verified.

EXPERIMENT NO 6: VIVA QUESTIONS

1. which of the following is not an Aggregate function?
A. Min
B. Max
C. Select
D. Avg

2. The attribute that can be divided into other attributes is called
A. Simple Attribute
B. Composite Attribute
C. Multi-valued Attribute
D. Derived Attribute

3. In an Entity-Relationship Diagram "Ellipses" represents
A. Attributes
B. Weak entity set
C. Relationship sets
D. Multi-valued attributes

4. In an Entity-Relationship Diagram "Diamonds" represents
A. Attributes
B. Multi-valued attributes
C. Weak entity set
D. Relationship sets

5. What are ACID properties of Transactions?
A. Atomicity, Consistency, Isolation, Database
B. Atomicity, Consistency, Isolation, Durability
C. Atomicity, Consistency, Inconsistent, Durability
D. Automatically, Concurrency, Isolation, Durability

6. If every non-key attribute is functionally dependent on the primary key, the relation will be in
A. First Normal Form
B. Second Normal Form
C. Third Normal Form
D. Fourth Formal Form

7. Database locking concept is used to solve the problem of
A. Lost Update
B. Uncommitted Dependency
C. Inconsistent Data
D. All of the above

8. UML is stands for
A. Universal Modeling Language
B. Unified Modeling Language
C. United Modeling Language
D. Uni Modeling Language

9. Data Manipulation Language (DML) is not to
A. Create information table in the Database
B. Insertion of new information into the Database
C. Deletion of information in the Database
D. Modification of information in the Database

10. Which of the following in true regarding Referential Integrity?
A. Every primary-key value must match a primary-key value in an associated table
B. Every primary-key value must match a foreign-key value in an associated table
C. Every foreign-key value must match a primary-key value in an associated table
D. Every foreign-key value must match a foreign-key value in an associated table

Experiment No: 07

Aim: Implementation of SQL Joins.

Theory:

SQL joins are used to query data from two or more tables, based on a relationship between certain columns in these tables.
The JOIN keyword is used in an SQL statement to query data from two or more tables, based on a relationship between certain columns in these tables.Tables in a database are often related to each other with keys.
A primary key is a column (or a combination of columns) with a unique value for each row. Each primary key value must be unique within the table. The purpose is to bind data together, across tables, without repeating all of the data in every table.
Look at the "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

Note that the "P_Id" column is the primary key in the "Persons" table. This means that no two rows can have the same P_Id. The P_Id distinguishes two persons even if they have the same name.

Next, we have the "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

Note that the "O_Id" column is the primary key in the "Orders" table and that the "P_Id" column refers to the persons in the "Persons" table without using their names.Notice that the relationship between the two tables above is the "P_Id" column.

Different SQL JOINs:

Before we continue with examples, we will list the types of JOIN you can use, and thedifferences between them.

JOIN: Return rows when there is at least one match in both tables.

LEFT JOIN: Return all rows from the left table, even if there are no matches in the right table

RIGHT JOIN: Return all rows from the right table, even if there are no matches in the left table

FULL JOIN: Return rows when there is a match in one of the tables.

1. SQL INNER JOIN:

The INNER JOIN keyword return rows when there is at least one match in both tables.

SQL INNER JOIN Syntax:

```
SELECT column_name(s)
FROM table_name1
INNER JOIN table_name2
ON table_name1.column_name=table_name2.column_name;
```

SQL INNER JOIN Example:

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

Now we want to list all the persons with any orders.

We use the following SELECT statement:

```
SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
INNER JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName;
```

The INNER JOIN keyword return rows when there is at least one match in both tables. If there are rows in "Persons" that do not have matches in "Orders", those rows will NOT be listed.

2. SQL LEFT JOIN:

The LEFT JOIN keyword returns all rows from the left table (table_name1), even if there are no matches in the right table (table_name2).

SQL LEFT JOIN Syntax:

```
SELECT column_name(s)
FROM table_name1
LEFT JOIN table_name2
ON table_name1.column_name=table_name2.column_name;
```

SQL LEFT JOIN Example:

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

Now we want to list all the persons and their orders - if any, from the tables above.We use the following SELECT statement:

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
LEFT JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName;

The LEFT JOIN keyword returns all the rows from the left table (Persons), even if there are no matches in the right table (Orders).

3. SQL RIGHT JOIN:

The RIGHT JOIN keyword returns all the rows from the right table (table_name2), even if there are no matches in the left table (table_name1).

SQL RIGHT JOIN Syntax:

SELECT column_name(s)
FROM table_name1
RIGHT JOIN table_name2
ON table_name1.column_name=table_name2.column_name;

SQL RIGHT JOIN Example:

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id
1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

Now we want to list all the orders with containing persons - if any, from the tables above.

We use the following SELECT statement:

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
RIGHT JOIN Orders ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName;

The RIGHT JOIN keyword returns all the rows from the right table (Orders), even if there are no matches in the left table (Persons).

4.SQL FULL JOIN:

The FULL JOIN keyword return rows when there is a match in one of the tables.

SQL FULL JOIN Syntax:

SELECT column_name(s)
FROM table_name1
FULL JOIN table_name2
ON table_name1.column_name=table_name2.column_name;

SQL FULL JOIN Example:

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

The "Orders" table:

O_Id	OrderNo	P_Id

1	77895	3
2	44678	3
3	22456	1
4	24562	1
5	34764	15

Now we want to list all the persons and their orders, and all the orders with their persons.

We use the following SELECT statement:

SELECT Persons.LastName, Persons.FirstName, Orders.OrderNo
FROM Persons
FULL JOIN Orders
ON Persons.P_Id=Orders.P_Id
ORDER BY Persons.LastName;

The FULL JOIN keyword returns all the rows from the left table (Persons), and all the rows from the right table (Orders). If there are rows in "Persons" that do not have matches in "Orders", or if there are rows in "Orders" that do not have matches in "Persons", those rows will be listed as well.

Conclusion:

Thus we have implemented Different SQL joins in SQL and the output was verified.

EXPERIMENT NO 7 : VIVA QUESTIONS

1. Which of the following option is use to retrieval of data?
a. Stack
b. Data Structure
c. Linked list
d. Query

2. ODBC stands for _____
a. Offline database connection
b. Oriented database connection
c. Open database connection
d. None of above

3. Which algebra is widely used in DBMS?
a. Relational algebra
b. Arithmetic algebra
c. Both d. None

4. Which of the following is an unary operation?
a. Selection operation
b. Generalized selection
c. Primitive operation
d. Projection operation

5. Which SQL Query is use to remove a table and all its data from the database?
a. Create Table
b. Alter Table
c. Drop Table
d. None of these

6. In precedence of set operators the expression is evaluated from:
a. Left to Left
b. Left to Right
c. Right to Right
d. Right to Left

7. In DBMS FD stands for _____
a. Facilitate data
b. Functional data
c. Facilitate dependency
d. Functional dependency

8. How many types of keys in Database Design?
a. Candidate key
b. Primary key
c. Foreign key
d. All of these

9. Which of the following is based on Multi Valued Dependency?
a. First
b. Second
c. Third
d. Fourth

10. Which of the following is the structure of the Database?
a. Table
b. Schema
c. Relation
d. None of these

Experiment No: 08

Aim: Implementation Numeric Functions and String Functions

Theory:

NUMERIC FUNCTIONS

- ABS: It returns the absolute value of 'n'.
- POWER: It returns m raised to nth power. n must be an integer else an error is returned.
- ROUND: It returns n rounded to m places right of the decimal point. If m is omitted, n is rounded to zero places. m must be an integer.
- SQRT: It returns square root of n. n should be greater than zero.

We use the following Numeric Functions statement -

```
SQL> select abs(-20) result from dual;
RESULT
---------
20
SQL> select power (2,10) result from dual;
RESULT
---------
1024

SQL> select round(15.359,2) result from dual;
RESULT
15.36

SQL> select sqrt (36) result from dual;
RESULT
---------
6
```

STRING FUNCTIONS

- LOWER: It returns char with letters in lower case.
- INITCAP: It returns char with the first letter in upper case.
- UPPER: It returns char with all letters forced to upper case.
- SUBSTR: It returns a portion of char beginning at character m, exceeding up to n characters. If n is omitted result is written up to the end character. The 1stposition of char is one.
- LENGTH: It returns the length of char
- LTRIM: It removes characters from the left of char with initial characters removed up to the 1st character not in set.
- RTRIM: It returns char with final characters removed after the last character not in the set. Set is optional. It defaults to spaces.
- LPAD: It returns char1, left padded to length n with the sequence of characters in char2. char2 defaults to blanks.
- RPAD: It returns char1, right padded to length n with the characters in char2, replicated as many times as necessary. If char2 is omitted, it is padded with blanks.

We use the following String Functions statement -

STRING FUNCTIONS

```
SQL> select lower('ORACLE') result from dual;
RESULT
------
oracle

SQL> select upper('oracle') result from dual;
RESULT
------
ORACLE

SQL> select initcap('Oracle') result from dual;
RESULT
------
Oracle
```

```
SQL> select substr('oracle' ,2 ,5) result from dual;
RESULT
-----
racle
SQL> select lpad('oracle',10,'#') result from dual;
RESULT
----------
####oracle
SQL> select rpad ('oracle',10,'^') result from dual;
RESULT
----------
oracle^^^^
```

Conclusion:

Thus we have implemented Numeric Functions And String Functions in SQL and the output was verified.

EXPERIMENT NO 8 : VIVA QUESTIONS

1. The minimal set of super key is called
A. Primary key
B. Secondary key
C. Candidate key
D. Foreign key

2. A relation that has no partial dependencies is in which normal form
A. First
B. Second
C. Third
D. BCNF

3. A functional dependency between two or more non-key attributes is called
A. Transitive dependency
B. Partial transitive dependency
C. Functional dependency
D. Partial functional dependency

4. A logical description of some portion of database that is required by a user to perform task is called as
A. System View
B. User View
C. Logical View
D. Data View

5. _____ is a classical approach to database design?
A. Left – Right approach
B. Right – Left approach
C. Top – Down approach
D. Bottom – Up approach

6. _____ refers to the correctness and completeness of the data in a database?
A. Data security
B. Data integrity
C. Data constraint
D. Data independence

7. A table that displays data redundancies yields _____ anomalies
A. Insertion
B. Deletion
C. Update
D. All of the above

8. A lock that allows concurrent transactions to access different rows of the same table is known as a
A. Field-level lock
B. Row-level lock
C. Table-level lock
D. Database-level lock

9. A type of query that is placed within a WHERE or HAVING clause of another query is called
A. Super query
B. Sub query
C. Master query
D. Multi-query

10. A transaction completes its execution is said to be
A. Saved
B. Loaded
C. Rolled
D. Committed

Experiment No: 09

Aim: Implementation of SQL Views.

Theory:

In SQL, a view is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database. You can add SQL functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

- **SQL CREATE VIEW Syntax:**

CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition;

SQL CREATE VIEW Examples:

If you have the Northwind database you can see that it has several views installed by default.The view "Current Product List" lists all active products (products that are not discontinued) from the "Products" table. The view is created with the following SQL:

CREATE VIEW [Current Product List] AS
SELECT ProductID,ProductName
FROM Products
WHERE Discontinued=No;

- **SQL Updating a View:**

You can update a view by using the following syntax

SQL CREATE OR REPLACE VIEW Syntax:

CREATE OR REPLACE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition:

Now we want to add the "Category" column to the "Current Product List" view. We will update the view with the following SQL:

CREATE VIEW [Current Product List] AS
SELECT ProductID,ProductName,Category
FROM Products
WHERE Discontinued=No;

- **SQL Dropping a View:**

You can delete a view with the DROP VIEW command.

SQL DROP VIEW Syntax:

DROP VIEW view_name;

Conclusion:

Thus we have implemented SQL view, Updating Views, dropping of views.

EXPERIMENT NO 09 : VIVA QUESTIONS

Name
Annie
Bob
Callie
Derek

Which of these query will display the the table given above ?
a) Select employee from name
b) Select name
c) Select name from employee
d) Select employee

2. Select _____ dept_name
from instructor;
Here which of the following displays the unique values of the column ?
a) All
b) From
c) Distinct
d) Name

3. The _____ clause allows us to select only those rows in the result relation of the ____ clause that satisfy a specified predicate.
a) Where, from

b) From, select
c) Select, from
d) From, where

4. Select ID, name, dept name, salary * 1.1
where instructor;
The query given below will not give an error. Which one of the following has to be replaced to get the desired output?
a) Salary*1.1
b) ID
c) Where
d) Instructor

5. The _____ clause is used to list the attributes desired in the result of a query.
a) Where
b) Select
c) From
d) Distinct

6. Select name, course_id
from instructor, teaches
where instructor_ID= teaches_ID;
This Query can be replaced by which one of the following ?
a) Select name,course_id from teaches,instructor where instructor_id=course_id;
b) Select name, course_id from instructor natural join teaches;
c) Select name ,course_id from instructor;
d) Select course_id from instructor join teaches;

7. Select * from employee where salary>10000 and dept_id=101;
Which of the following fields are displayed as output?
a) Salary, dept_id
b) Employee
c) Salary
d) All the field of employee relation

8.

Employee_id	**Name**	**Salary**
1001	Annie	6000
1009	Ross	4500
1018	Zeith	7000

This is Employee table.
Select * from employee where employee_id>1009;
Which of the following employee_id will be displayed?
a) 1009, 1001, 1018
b) 1009, 1018
c) 1001
d) 1018

9. Which of the following statements contains an error?
A) Select * from emp where empid = 10003;
B) Select empid from emp where empid = 10006;
C) Select empid from emp;
D) Select empid where empid = 1009 and lastname = ‘GELLER’;

10. Insert into employee _____ (1002,Joey,2000);
In the given query which of the keyword has to be inserted ?
a) Table
b) Values
c) Relation
d) Field

Experiment No: 10

Aim: Implementation of Complex and Nested queries.

Theory:

Sub query or Inner query or Nested query is a query in a query. A subquery is usually added in the WHERE Clause of the sql statement. Most of the time, a subquery is used when you know how to search for a value using a SELECT statement, but do not know the exact value.

Sub queries are an alternate way of returning data from multiple tables.Sub queries can be used with the following sql statements along with the comparision operators like =, <, >, >=, <= etc.

- SELECT
- INSERT
- UPDATE
- DELETE

For Example:

1) Usually, a subquery should return only one record, but sometimes it can also return multiple records when used with operators like IN, NOT IN in the where clause. The query would be like,

```
SELECT first_name, last_name, subject
FROM student_details
WHERE games NOT IN ('Cricket', 'Football');
```

The output would be similar to:

first_name	last_name	subject
Shekar	Gowda	Badminton
Priya	Chandra	Chess

2) Lets consider the student details table which we have used earlier. If you know the name of the students who are studying science subject, you can get their ids by using this query below,

```
SELECT id, first_name
FROM student_details
WHERE first_name IN ('Rahul', 'Stephen');
```

but, if you do not know their names, then to get their id's you need to write the query in this manner,

SELECT id, first_name

FROM student_details

WHERE first_name IN (SELECT first_name

FROM student_details

WHERE subject= 'Science');

Output:

id	first_name
100	Rahul
102	Stephen

In the above sql statement, first the inner query is processed first and then the outer query is processed.

3) Sub query can be used with INSERT statement to add rows of data from one or more tables to another table. Lets try to group all the students who study Maths in a table 'maths_group'.

INSERT INTO maths_group(id, name)
SELECT id, first_name || ' ' || last_name
FROM student_details WHERE subject= 'Maths';

4) A sub query can be used in the SELECT statement as follows. Let's use the product and order_items table defined in the sql_joins section.

select p.product_name, p.supplier_name, (select order_id from order_items where product_id = 101) as order_id from product p where p.product_id = 101;

product_name	supplier_name	order_id
Television	Onida	5103

Correlated Sub query:

A query is called correlated sub query when both the inner query and the outer query are interdependent. For every row processed by the inner query, the outer query is processed as well. The inner query depends on the outer query before it can be processed.

SELECT p.product_name FROM product p

WHERE p.product_id = (SELECT o.product_id FROM order_items o

WHERE o.product_id = p.product_id);

Conclusion:

Thus we have seen Nested queries, and complex queries, and we are concluded that we can nest as many queries as we want but it is recommended not to nest more than 16 sub queries in oracle.

EXPERIMENT NO 10: VIVA QUESTIONS

1. A _____ consists of a sequence of query and/or update statements.
a) Transaction
b) Commit
c) Rollback
d) Flashback

2. Which of the following makes the transaction permanent in the database ?
a) View
b) Commit
c) Rollback
d) Flashback

3. In order to undo the work of transaction after last commit which one should be used ?
a) View
b) Commit
c) Rollback
d) Flashback

4. Consider the following action:
Transaction…..
Commit;
Rollback;
What does Rollback do?
a) Undoes the transactions before commit
b) Clears all transactions
c) Redoes the transactions before commit
d) No action

5. In case of any shut down during transaction before commit which of the following statement is done automatically ?
a) View
b) Commit
c) Rollback
d) Flashback

6. In order to maintain the consistency during transactions database provides
a) Commit
b) Atomic
c) Flashback
d) Retain

7. Transaction processing is associated with everything below except
a) Conforming a action or triggering a response
b) Producing detail summary or exception report
c) Recording a business activity
d) Maintaining a data

8. A transaction completes its execution is said to be
a) Committed
b) Aborted
c) Rolled back
d) Failed

9. Which of the following is used to get back all the transactions back after rollback ?
a) Commit
b) Rollback
c) Flashback
d) Redo

10. _____ will undo all statements up to commit?
a) Transaction
b) Flashback
c) Rollback
d) Abort

Experiment No: 11

Aim: Write a program to implement program to satisfy some conditions by accepting input from the user using PL/SQL .

Theory: PL/SQL is a combination of SQL along with the procedural features of programming languages. It was developed by Oracle Corporation in the early 90's to enhance the capabilities of SQL.

PL/SQL is one of three key programming languages embedded in the Oracle Database, along with SQL itself and Java.

The PL/SQL programming language was developed by Oracle Corporation in the late 1980s as procedural extension language for SQL and the Oracle relational database. Following are notable facts about PL/SQL:

- PL/SQL is a completely portable, high-performance transaction-processing language.
- PL/SQL provides a built-in interpreted and OS independent programming environment.
- PL/SQL can also directly be called from the command-line SQL*Plus interface.
- Direct call can also be made from external programming language calls to database.
- PL/SQL's general syntax is based on that of ADA and Pascal programming language.
- Apart from Oracle, PL/SQL is available in TimesTen in-memory database and IBM DB2.

Features of PL/SQL

PL/SQL has the following features:

- PL/SQL is tightly integrated with SQL.
- It offers extensive error checking.
- It offers numerous data types.
- It offers a variety of programming structures.
- It supports structured programming through functions and procedures.
- It supports object-oriented programming.
- It supports developing web applications and server pages.

Advantages of PL/SQL

PL/SQL has the following advantages:

- SQL is the standard database language and PL/SQL is strongly integrated with SQL. PL/SQL supports both static and dynamic SQL. Static SQL supports DML operations and transaction control from PL/SQL block. Dynamic SQL is SQL allows embedding DDL statements in PL/SQL blocks.
- PL/SQL allows sending an entire block of statements to the database at one time. This reduces network traffic and provides high performance for the applications.

- PL/SQL gives high productivity to programmers as it can query, transform, and update data in a database.
- PL/SQL saves time on design and debugging by strong features, such as exception handling, encapsulation, data hiding, and object-oriented data types.
- Applications written in PL/SQL are fully portable.
- PL/SQL provides high security level.
- PL/SQL provides access to predefined SQL packages.
- PL/SQL provides support for Object-Oriented Programming.
- PL/SQL provides support for Developing Web Applications and Server Pages.

TO INPUT A VALUE FROM THE USER AND DISPLAY IT

```
SQL> set serveroutput on;
SQL> declare
2 a varchar2(20);
3 begin
4 a:=&a;
5 dbms_output.put_line(a);
6 end;
7 /
Enter value for a: 5
old 4: a:=&a;
new 4: a:=5;
5

PL/SQL procedure successfully completed.
```

1. **TO DISPLAY HELLO MESSAGE**

```
SQL> set serveroutput on;
SQL> declare
2 a varchar2(20);
3 begin
4 a:='Hello';
5 dbms_output.put_line(a);
6 end;
7 /
Hello
PL/SQL procedure successfully completed.
```

2. Insert the record into Sailors table by reading the values from the Keyboard.

```
SQL> create table sailors(sid number(10),sname varchar(10),rating number(10),age
number(10));

Table created.

SQL> set serveroutput on
```

```
SQL> declare
 2 sid number(10):=&sid;
3 sname varchar(10):='&sname';
4 rating number(10):=&rating;
5 age number(10):=&age;
6 begin
7 insert into sailors values(sid,sname,rating,age);
8 end;
9 /

Enter value for sid: 02
old 2: sid number(10):=&sid;
new 2: sid number(10):=02;
Enter value for sname: lavanya
old 3: sname varchar(10):='&sname';
new 3: sname varchar(10):='lavanya';
Enter value for rating: 01
old 4: rating number(10):=&rating;
new 4: rating number(10):=01;
Enter value for age: 25
old 5: age number(10):=&age;
new 5: age number(10):=25;

PL/SQL procedure successfully completed.

SQL> /

Enter value for sid: 03
old 2: sid number(10):=&sid;
new 2: sid number(10):=03;
Enter value for sname: vani
old 3: sname varchar(10):='&sname';
```

new 3: sname varchar(10):='vani';

Enter value for rating: 02

old 4: rating number(10):=&rating;

new 4: rating number(10):=02;

Enter value for age: 25

old 5: age number(10):=&age;

new 5: age number(10):=25;

PL/SQL procedure successfully completed.

SQL> select * from sailors;

SID	SNAME	RATING	AGE
2	lavanya	1	25
3	vani	2	25

CONCLUSION: Thus the PL/SQL block has been studied and implemented.

EXPERIMENT NO 11 : VIVA QUESTIONS

1. Student(ID, name, dept name, tot_cred)
   In this query which attribute form the primary key?
   a) Name
   b) Dept
   c) Tot_cred
   d) ID

2. Which one of the following is a procedural language ?
   a) Domain relational calculus
   b) Tuple relational calculus
   c) Relational algebra
   d) Query language

3. . The_____ operation allows the combining of two relations by merging pairs of tuples, one from each relation, into a single tuple.
   a) Select
   b) Join
   c) Union
   d) Intersection
4. The result which operation contains all pairs of tuples from the two relations, regardless of whether their attribute values match.
   a) Join
   b) Cartesian product
   c) Intersection
   d) Set difference

5. The _____operation performs a set union of two "similarly structured" tables
   a) Union
   b) Join
   c) Product
   d) Intersect

6. . The most commonly used operation in relational algebra for projecting a set of tuple from a relation is
   a) Join
   b) Projection
   c) Select
   d) Union

7. . The _____ operator takes the results of two queries and returns only rows that appear in both result sets.
   a) Union
   b) Intersect
   c) Difference
   d) Projection

8. . A _____ is a pictorial depiction of the schema of a database that shows the relations in the database, their attributes, and primary keys and foreign keys.
   a) Schema diagram b) Relational algebra
   c) Database diagram d) Schema flow

9.

9. The _____ provides a set of operations that take one or more relations as input and return a relation as an output.
a) Schematic representation b) Relational algebra
c) Scheme diagram d) Relation flow

10. Explain transaction.

Experiment No: 12

Aim: Write a program **t**o implement decision making and looping in PL/SQL

Theory:

PL/SQL PROGRAMMING

Procedural Language/Structured Query Language (PL/SQL) is an extension of SQL.

Basic Syntax of PL/SQL

```
DECLARE
/* Variables can be declared here */
BEGIN
/* Executable statements can be written here */
EXCEPTION
/* Error handlers can be written here. */
END;
```

Steps to Write & Execute PL/SQL

- As we want output of PL/SQL Program on screen, before Starting writing anything type (Only
  Once per session)
  SQL> SET SERVEROUTPUT ON
- To write program, use Notepad through Oracle using ED command.
  SQL> ED ProName
  Type the program Save & Exit.
- To Run the program
  SQL> @ProName

Decision making with IF statement:-

The general syntax for the using IF—ELSE statement is

```
IF(TEST_CONDITION) THEN
SET OF STATEMENTS
ELSE
SET OF STATEMENTS
END IF;
```

For Nested IF—ELSE Statement we can use IF--ELSIF—ELSE as follows

```
IF(TEST_CONDITION) THEN
SET OF STATEMENTS
ELSIF (CONDITION)
SET OF STATEMENTS
END IF;
```

LOOPING STATEMENTS:-
For executing the set of statements repeatedly we can use loops. The oracle supports number of looping statements like GOTO, FOR, WHILE & LOOP.
Here is the syntax of these all the types of looping statements.

GOTO STATEMENTS
<<LABEL>>
SET OF STATEMENTS
GOTO LABEL;

FOR LOOP
FOR <VAR> IN [REVERSE] <INI_VALUE>**..**<END_VALUE>
SET OF STATEMENTS
END LOOP;

WHILE LOOP
WHILE (CONDITION) LOOP
SET OF STATEMENTS
END LOOP;

LOOP STATEMENT
LOOP
SET OF STATEMENTS
IF (CONDITION) THEN
EXIT
SET OF STATEMENTS
END LOOP; While using LOOP statement, we have take care of EXIT condition, otherwise it may go into infinite loop.

GREATEST OF THREE NUMBERS

SQL> set serveroutput on;

SQL> declare

2 a number(7);

3 b number(7);

4 c number(7);

5 begin

6 a:=&a;

7 b:=&b;

8 c:=&c;

9 if(a>b and a>c) then

10 dbms_output.put_line (' The greatest of the three is ' || a);

11 else if (b>c) then

```
12 dbms_output.put_line (' The greatest of the three is ' || b);
13 else
14 dbms_output.put_line (' The greatest of the three is ' || c);
15 end if;
16 end if;
17 end;
18 /
Enter value for a: 5
old 6: a:=&a;
new 6: a:=5;
Enter value for b: 7
old 7: b:=&b;
new 7: b:=7;
Enter value for c: 1
old 8: c:=&c;
new 8: c:=1;
The greatest of the three is 7
PL/SQL procedure successfully completed.
```

PRINT NUMBERS FROM 1 TO 5 USING SIMPLE LOOP

```
SQL> set serveroutput on;
SQL> declare
2 a number:=1;
3 begin
4 loop
5 dbms_output.put_line (a);
6 a:=a+1;
7 exit when a>5;
8 end loop;
9 end;
10 /
1
2
 3
```

4

5

PL/SQL procedure successfully completed.

PRINT NUMBERS FROM 1 TO 4 USING WHILE LOOP

```
SQL> set serveroutput on;
SQL> declare
2 a number:=1;
3 begin
4 while(a<5)
5 loop
6 dbms_output.put_line (a);
7 a:=a+1;
8 end loop;
9 end;
10 /
1
2
3
4
```

PL/SQL procedure successfully completed.

PRINT NUMBERS FROM 1 TO 5 USING FOR LOOP

```
SQL> set serveroutput on;
SQL> declare
2 a number:=1;
3 begin
4 for a in 1..5
5 loop
6 dbms_output.put_line (a);
7 end loop;
8 end;
9 /
1
```

2

3

4

5

PL/SQL procedure successfully completed.

CONCLUSION: Thus we have implemented decision making and looping in PL/SQL

EXPERIMENT NO 12: VIVA QUESTIONS

1 What are different phases of transaction?

2 Give ACID properties of a database transaction

3 Define the "integrity rules"?

4 What is meant by query optimization?

5 What is durability in DBMS?

6 Define database views

7 Give different types of views

8 What is database Trigger?

9 Give shadow copy schema

10 Give different software packages for DBMS.

Question Bank

1. What is DBMS?
2. Advantage of database
3. What is data model?
4. What is object-oriented model?
5. What is an entity?
6. What is an entity type?
7. What is an entity set?
8. What is weak entity set?
9. What is relationship?
10. What is DDL?
11. What is DML?
12. What is normalization?
13. What is functional dependency?
14. What is 1st NF?
15. What is 2nd NF?
16. What is 3rd NF?
17. What is BCNF?
18. What is fully functional dependency?
19. What do you mean of aggregation, atomicity?
20. What are different phases of transaction?
21. What do you mean of flat file database?
22. What is query?
23. What are the disadvantages in File Processing System?
24. Describe the three levels of data abstraction?
25. Define the "integrity rules"?
26. What is meant by query optimization?
27. What is durability in DBMS?
28. What are the different phases of transaction?
29. What are the unary operations in Relational Algebra?
30. What is database Trigger?