Experiment No - 01

Experiment Name: Study and implementation of DML commands of SQL with suitable Example

- Insert.
- Delete.
- Update.

Objectives:

(i) To understand and use data manipulation language (DML) to write query for database.

(ii) To understand the different issues in the design and implementation of a database system.

(iii) To understand and use data manipulation language to insert, update, delete and manage database.

Theory: SQL is structured query language, which is a computer language for storing, manipulating and retrieving data stored in a relational database. SQL is the standard language for relational database system. SQL uses certain commands like create, Drop, insert etc. to carry out the required tasks. These SQL commands are mainly categorized into four categories are:

(i) DDL - Data definition language.

(ii) DQL - Data query language.

(iii) DML - Data Manipulation language.

(iv) DCL - Data control language.

In this problem for specified DML type commands.

→ Data Manipulation language (DML): The SQL commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation language and this includes most of the SQL statements. it is the component of the SQL statement that controls access to data and to the database also the SQL DML provides the ability to query information from the database and to insert tuples into, delete tuples from and modify tuples in the database.

The functional capability of DML is organized in manipulation commands like update, insert and Delete etc. as described below:

- Insert: This command adds one or more records to a database table. the insert

command syntax is INSERT INTO [table Name] [column(s)] values [value(s)].

- Delete: This command removes one or more records from a table according to specified conditions. delete command syntax is.

```
DELETE FROM [table name] where [condition];
```

- Update: This command modifies data of one or more records. an update command.

Syntax is,

```
update [table name] SET [column name = value] where [condition];
```

Experiment No - 02

Experiment Name: Study and implementation of DDL commands of SQL with suitable example

- Create
- Alter
- Drop.

Objectives:

(i) To understand and use data definition language to write query for database.

(ii) To understand DDL to create, Alter, Drop and manage database.

(iii) To see how to drop any relation or database and how to change anything in database using alter and also see that how to create anything related database.

Theory: A data definition language (DDL) is a computer language used to create and modify the structure of database objects in a database. these database objects include views, index, tables etc.

this term also known as data description language in some contexts, as it describes the fields and records in a database table.

Commonly used DDL in ASQL querying are create, Alter, drop and truncate. below described.

Create: This command builds a new table and has a predefined syntax. the CREATE statement syntax is:

```
CREATE TABLE [tablename] ( [column definition]) [table
                                        parameters] ;
```

for example;

```
Create table employee
( employee id integer primary key,
  first name char (50) Null,
  Last name char (75) Not Null);
```

The mandatory semi-colon at the end of the statement is used to process every command before it. in this example, the string char is used to specify the data type other data types can be date, numbers or integer.

- Alter: An alter command modifies an existing database table. this command can add up additional column, drop existing columns and even change the data type of columns involved in a database table. an alter

command syntax is:

```
Alter object type objectname parameters;
```

for example:

```
alter table employee add primary key(employee_pk);
```

in this example, we added a unique primary key to the table to add a constraint and enforce a unique value. the constraint "employee_pk" is a primary key and is on the employee value.

- Drop: A drop command is used to delete objects such as a table, index or view. a drop statement cannot be rolled back, so once an object is destroyed there is no way to recover it. syntax is:

```
Drop object type objectname;
```

for example:

```
Drop table employee;
```

Experiment No - 03

Experiment Name: Study and implementation of DML commands of

- Select clause.
- From clause.
- Where clause.

Objectives:

(i) To understand and use data manipulation language (DML) to write query for database.

(ii) To see how to perform select clause on database.

(iii) To see how to perform from clause on database.

(iv) To see how to perform where clause on database.

Theory:

Select clause: The SQL select statement or clause used to select/retrieve data from a table in the database. the SQL select clause is most commonly used clause. The select clause query retrieve whole table data or some specific columns data from the table.

if we want to retrieve some data from table we need to write select statement query in SQL.

In SQL select statement we retrieve the data by columns wise.

The SQL select syntax categorize into two parts, first we retrive all columns means full table data, second we retrieve some specific

columns from a table.

Select all columns data from a table

```
Select * From table-name;
```

Select some columns data from a table

```
Select col1, col2, col3 From table-name;
```

- From clause: The SQL From clause is the source of rowset to be operated upon in a data manipulation language (DML) statement. From clauses are very common and will provide the rowset to be exposed through a select statement, the source of values in an update statement and the target rows to be deleted in a delete statement.

From is an SQL reserved word in the SQL Standard.

The From clauses is used in conjunction with SQL statements and taken the following general form:

```
SQL DML-statement from table-name
  where predicate
```

- Where clause: The SQL where clause is used to specify the condition while make action like select, update and delete operation on data-base table.

The where clause is used to tack action on only those records that fulfill a specified condition. the syntax is:

```
Select col1, col2 From table-name where [condition]
Select col1, col2 from table-name where
                                column = "value";
```

Experiment No – 04

Experiment Name: Study and implementation of DML commands of

- Group By & Having clause.
- Order By clause.
- Create view, indexing & Procedure clause

Objectives:

(i) To understand how to group by and having clause perform on the database.

(ii) To understand how to order by clause perform on the database.

(iii) To understand how to create view, indexing and Procedure clause perform on the database.

(iv) To understand basic concept of DML.

Theory :

- **Group By clause:** The group by clause is often used with aggregate functions (MAX, SUM, AVG) to group the results by one or more columns. or in simple words we can say that the group by clause is used in collaboration with the select statement to arrange required data into groups

The group by statement groups rows that have the same values. this statement is used after the where clause. this statement is often used with some aggregate function like SUM, AVG, COUNT ate to group the results by one or more columns.

The group by clause syntax is;

```
select col1, aggregate-function (col2) from
table_name Group By col1.
```

- Having clause; Having clause is basically like the aggregate function with group by clause. the Having clause is used instead of where with aggregate functions. while the Group By clause groups rows that have the same values into summary rows. the having clause is used with the where clause in order to find rows with certain conditions the having clause is always used after the group by clause. the having clause syntax is;

```
select column_name(s) from tablename
where condition Group By column_name(s)
Having condition order By column_name(s)
```

- **Order By clause:** The order By clause is used to sort the records in ascending or descending order, based on one or more columns. Mostly all database server query fetch records by default in ascending order.

The syntax of order By clause:

For ascending order:

```
select col1, col2 from table-name order
                        By col1 Asc;
```

For descending order:

```
select col1, col2 from table-name order
                        By col1 desc;
```

- **Create View Clause:** 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

On more real tables in the databases.

add SQL statements and functions to a view and present the data as if the data were coming from one single table.

A view is created with the create view statement.

The syntax of create view is;

```
Create view view_name as
Select col1, col2,...
From table_name where condition;
```

• create index: the create index statement is used to create indexes in tables.

indexes are used to retrieve data from the database more quickly than otherwise, the users cannot see the indexes, they are just used to speed up searches | queries.

The syntax of create index is;

```
create index index-name
on table-name (column1, column2, ...);
```

The syntax of create unique index is:

```
create unique index index-name
on table-name (colum1, column2, ...);
```

• Procedure clause: A procedure is a named SQL block which performs one or more specific task. this is similar to a procedure in other programming language. a procedure has a header and a body.

the header consists of the name of the procedure and the parameters or variables passed to the procedure. the body consists of declaration

section, execution section and exception section similar to a general SQL block. A procedure is similar to an anonymous block but it is named for repeated usage. We can pass parameters to procedure in three ways:

In type: these types of parameters are used to send values to stored procedures.

Out type: These types of parameters are used to get values from stored procedures this is similar to a return type in functions.

IN out type: these types of parameters are used to send values and get values from stored procedures. a procedure may or may not return value.

The syntax of procedure is;

```
Create or Replace Procedure Proname (Argument{IN, OUT
                                    , <datatype>, ...)

IS
  Declaration section<variable, constant>;

BEGIN
   Execution section

Exception
   Exception section.

END.
```

There are two ways to execute a procedure:

(i) From the SQL prompt: EXECUTE [or EXEC]
procedure_name;

(ii) Simply use the procedure_name:

procedure_name.

Experiment No – 05

Experiment Name: Study and implementation of SQL commands of Join operations with Example

- Cartesian Product.
- Nutural Join.
- Left outer join.
- Right outer join.
- Full outer join.

objectives:

(i) To understand how to perform Carterian Product on the databases from many relation

(ii) To understand how to perform join clause perform specific task such as: Nutural join, Left outer join, Right outer join, Full outer join.

(iii) To understand the different issues in the design and implementation of a database system.

Theory:

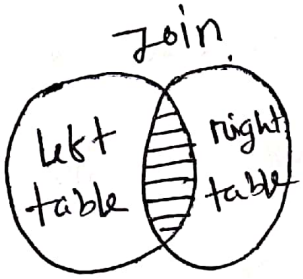
- **Cartesian Product:** Cartesian product in SQL is a term from the set theory of mathematics. a cartesian product of two sets X and Y, denoted X x Y, is the set of all ordered pairs where X is the X and Y is the Y.

in terms of SQL, the cartesian product is a new table formed of two tables. if those tables 3 and 4 lines respectively, the cartesian product table will have 3 x 4 lines. therefore, each row from the first table joins each row of the second table. get the multiplication result of two sets making all possible ordered pairs of the original set's of elements.

The syntax of cartesian product,

```
Select table1.colum1, table2.colum2 . . .
From table1, table2 [, table3].
```

• SQL Join: A join combines records from two tables. Join matches related column values in two tables. a query can contain zero, one or multiple join operations.



There are some types of join,

(i) Natural join.
(ii) Left outer join.
(iii) Right outer join.
(iv) Full outer join.

• Natural Join: Natural join is an SQL join operation that creates join on the base of the common columns in the tables. to perform natural join there must be one common attribute (column) between two tables natural join will retrieve from multiple relations.

Features of natural join,

(i) it will perform the Cartesian Product.

(ii) it finds consistent tuples and deletes inconsistent tuples.

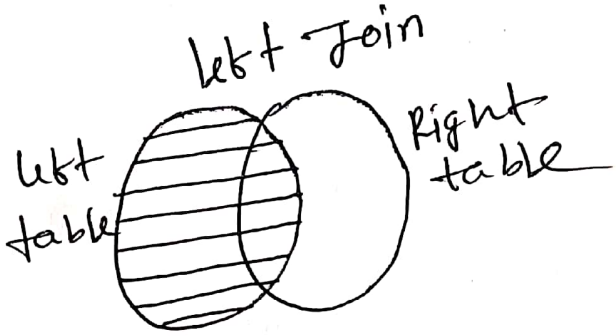
(iii) Then it deletes the duplicate attributes.

The syntax of natural join,

```
Select * From Table1 Natural
    join Table2
```

• Left outer join: a left outer join performs a join starting with the left table. then any matching records from the right table will be included. rows without a match will have Null columns values.

The figure see that.



The syntax of left outer join,

```
Select column_name(s)
From table1
Left outer join table2
ON table1.column_name=table2.column_name
```

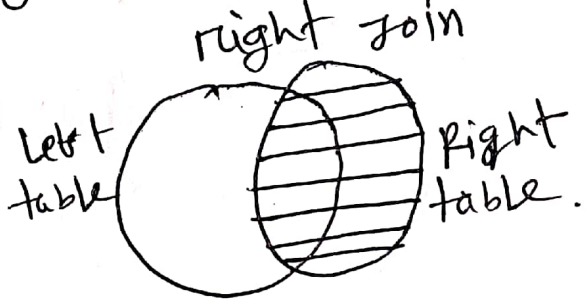
• Right outer join: Some important points given below,

(i) A right join performs a join starting with the right table.

(ii) Then Any matching records from the left table will be included.

(iii) Rows without a match will have Null column values.

The figure see that present.



The syntax of right outer join.

```
Select column_name(s)
From table1
Right outer join table2
ON table1.column_name = table2.column_name
```

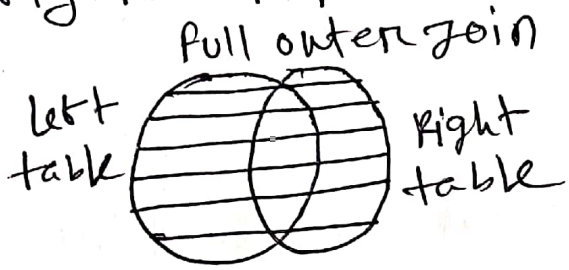
• Full outer join: Some important points present about on this topic.

(i) A full outer join returns all records from both tables.

(ii) This includes records that donot match.

(iii) Rows without a match will have NULL column values.

The figure represented by.



The syntax of full outer join.

```
Select column_name(s)
From table1
Full outer join table2
ON table1.column_name = table2.column_name
where condition;
```

Experiment No - 06

Experiment Name: Study and implementation of Aggregate function with Example.

- Count function.
- Max function.
- Min function.
- Avg function.

Objectives:

(i) To understand how to aggregate function perform on the database for specific task.

(ii) To understand the different issues in the design and implementation of a database system.

(iii) To understand and use (DML) to write query for database.

Theory:

Aggregate function: aggregate functions are those functions in the DBMS which takes the values of multiple rows of a single column and then form a single value by using a query. these functions allow the user to summarizing the data. these functions ignore the null values except the count function In database management systems, following are the five aggregate functions:

(i) Avg. function

(ii) Max function

(iii) Min function

(iv) count function.

(v) sum function.

• Count function: The count() function returns the number of rows that matches a specified criterion. the syntax is.

```
Select count (column-name)
From table-name
where condition;
```

• Max function: the Max() function returns the largest value of the selected column. The syntax is

```
select MAX (column-name)
From table-name
where condition;
```

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

The syntax is.

```
select MIN(column-name)
from table-name
where condition;
```

- Avg function: the AVG() function returns the average value of a numeric column.

the syntax is.

```
select AVG(column-name)
from table-name
where condition;
```

Experiment No – 07

Experiment Name: Study and implementation of Triggering system on database table using SQL commands with example.

Objectives:

(i) To see how to triggering system work on the database.

(ii) To understand and use data manipulation language to write query for database.

(iii) To understand the different issues in the design and implementation of a database system.

Theory:

Trigger in DBMS: A trigger is a procedure of SQL statements, which is automatically fired when the DML statements are executed on the table of the database. Triggers are the event-driven procedures, which are managed and stored by the database management system.

trigger helps in maintaining data integrity by changing the database data in a systematic way. triggers are always associated with the insert or update or delete command of the database table.

The syntax of trigger:

```
Create [or replace] Trigger trigger-name
{Before | After | Instead of}
{insert [or] | update [or] | Delete}
 [of name_of_column]
 ON name_of_the_table
Referencing [OLD as old | New As new]
[For each row | For each statement]
When (condition)
Declare
  [ declaration-section]
Begin
 --- sql statements
END;
```