Assignment No. 2

Date :-

Title :- Design and Develop SQL, DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym.

Objective :-
- To study syntax of DDL commands of SQL.
- To study and apply DDL commands on the database.

Outcomes :-
- Able to know syntax of DDL commands of SQL.
- Able to apply DDL commands on the database

Requirements :
1) Hardware :- Server and client
2) Software :- MySQL / oracle.

Theory :
SQL : Structured Query language
SQL: (Pronounced SEQUEL) is the standard language to access relational databases. SQL is an abbreviation for Structured Query language. I'll just add that SQL is composed of DML and DDL. DML are the keywords you use to access and manipulate data, hence the

name Data Manipulation language. DDL are the keywords you use to create objects such as views, tables and procedures, hence the name Data Definition language.

1) Tables :-

In relational database system (DBS) data are represented using tables (relations). A query issued against the DBS also results in a table. A table has following structure :-

Attributes

Column1 column2 ..... Column n

← Tuple (or Record)

....

A table is uniquely identified by its name and consists of rows that contain the stored information, each row containing exactly one tuple (or record). A table can have one or more Columns.

A column is made up of a column name and a data type, and it describes an attribute of the tuples. The structure of a table, also called relation schema, thus it is defined by its attributes. The type of information to be stored in a table is defined by the data types of the attributes of table

creation time. SQL uses the terms table, row, and column and relation, tuple and attribute respectively.

A table can have upto 254 columns which may have different or some data types and sets of values (domains), respectively. Possible domains are alphanumeric data (strings), numbers and date formats.

2) Data types :-

2.1) Number (p, s) :- Number having precision p and scale s.

2.2) Char (size) :- fixed length character data of length size bytes. This should be used for fixed length data such as codes A100, B102...

2.3) varchar2 (size) :- Variable length character string having maximum length size bytes. You must specify size.

2.4) ~~long~~ :- character data of variable length (A biggest version the VARCHAR2 data type).

2.5) Date :- Valid date range

2.6) Int :- A finite subset of integers. The full form is Integer.

2.1) Real, Double Precision :- Floating point and double-precision floating point numbers.

2.2) Float (n) :- A floating point number, with precision of at least n digits.

3) Data Definition in SQL.

3.1) Creating Tables :-

Syntax :- create table <table_name>
(column_name1 datatype size(),
column_name2 datatype size(),
:
column_name n datatype size()) ;

3.2) Constraints :-

Both the create table and Alter table SQL can be used to write SQL sentences that attach constraints. Basically constraints are of three types :-

1) Domain :-

- NOT NULL :- NOT NULL Constraint can be applied at column level only, we can define these constraints at the time of table creation.

Syntax :- create table < table name >
(< column_name > datatype (size) NOT NULL,
< column_name > datatype (size). ....);

After the table creation

After table < table name >
modify (< columnName > datatype (size) NOT NULL);

- check : It can be bound to column or table using create table or alter table command. checks are performed when write operation is performed Insert or update statement causes the relevant check constraint Ensure the integrity of the data in tables.

Check constraints at column level
Syntax :-
create table < tableName >
(< columnName > datatype (size) check (column Name condition), < columnname datatype (size));

create table < tableName >
(< columnName > datatype (size) constraint < constraint name > check (columnName condition). ....);

Check constraints at table level

Syntax

create table <tableName>
(<columnName> datatype (size),
<columnName> datatype (size), ...,
Constraint <constraint_name> check (columnName
condition) .....);

create table <tableName>
(<columnName> datatype(size),
<columnName> datatype (size), ...,
check (columnName condition));

2) Entity :-

- Primary key :- A primary key is one or more column in a table used to uniquely identify each row in the table. A table can have only one primary key. Cannot be left blank Data must be unique. Not allows null values. Not allows duplicate values. Unique index is created automatically if there is a primary key

primary key constraint defined at column level

Syntax: create table <tableName>
(<columnName 1><Data type>(<size>) Primary key.
<Columnname 2><Data type>(<size>), ...);

primary key constraint defined at Table level.
Syntax :

Create table <tableName>
(<columnName 1><Data type>(<size>), ..., primary key (<columnName 1><columnname 2>));

- Unique :- The unique column constraint permits multiple entries of NULL into the column. Unique key not allowed duplicate values, Unique index is automatically Created. Table can have more than one unique key.

unique constraint defined at column level
Syntax :-
Create table tablename (<columnname><data type> (<size> unique), <columnname> datatype (<size>) ...);

unique constraint defined at table level
Syntax :-
Create table tablename (<columnname><data type> (<size>), <columnname><datatype> (<size>), unique (<columnname>, <columnname>));

3) Referential :-
- Foreign key :- foreign key represents relationships between tables, A foregin key is a column whose values are derived from primary key of some other table.

Foreign key constraint defined at column level
Syntax
<ColumnName><DataType>(<size>) references
<tableName>[(<columnName>)] [ON DELETE CASCADE]

After the table creation
alter table <child tablename> ADD constraint <constraint name>
foreign key (<columnname in child table> references <parent
table name>;

foreign key constraint at table level
foreign key constraint is defined with ON DELETE CASCADE

foreign key(<ColumnName>[,<columnname>]) references
<TableName>[(<ColumnName>, <ColumnName>)] ON DELETE CASCADE

foreign key constraint defined with ON Delete set null
foreign key (<ColumnName>[,<columnname>]) references
<TableName>[(<ColumnName>, <ColumnName>) ON DELETE SET NULL

3.3] To view the constraint -
Syntax :- Select Constraint_name, constraint_type, search_condition
from user_constraints when table_name = <table name>;

Select constraint_name, column_name from user_cons_columns
where table_name = <tablename>;

3.4] To drop the Constraints :-

Syntax :-

Drop constraint constraintname;

3.5] Describe Commands :-

To view the structure of the table created use the DESCRIBE command. The command displays the column names with data types.

Syntax :-

Desc < table name >

e.g desc student.

Restrictions for creating a table :

- Table names and column names must begin with a letter.
- Table names and column names can be 1 to 30 characters long
- Table names must contain only the characters A-Z, a-z, 0-9, underscore _, $ and #
- Table name should not be same as the name of another database object.
- Table name must not be an ORACLE reserved word.
- Column names should not be duplicate within a table definition.

3.6] Alteration of Table :-

Syntax :-

Case 1 :- alter table < table_name >
Add (column_name 1 datatype size (),
column_name 2 datatype size (),
⋮
column_name n datatype size ());

Case 2 :- alter table < table_name >
modify (column_name 1 datatype size (),
column_name 2 datatype size (),
⋮
column_name n datatype size ());

Dropping a column from a table :-

Syntax :-

Alter table < tablename > Drop column < ColumnName >;

Drop table command :-

Syntax :-

Drop table < table_name >;

Drop table command removes the definitions of an Oracle table. when you drop a table, the database loses all the data in the table and all the indexes associated with it.

eg drop table Student;

Truncate table command :
Syntax : Truncate table <table name>;

The truncate table statement is used to remove all rows from a table and to release the storage space used by the table.
eg truncate table Student;

Rename table Command :
Syntax : Rename <old table name> to <new table name>

Rename Statement is used to rename a table, view, sequence or synonym.
eg rename student to stud;

Database Objects :-
Views : View is a logical representation of subsets of data from one or more tables. A view takes the output of a query and treats it as a table therefore view can be called as stored query or a virtual table. The tables upon which a view is based are called base tables. In oracle the SQL command to create a view (virtual table) has the form

Syntax :-
create [or replace] view <view name> [(<column(s)>)] as
<select statement> [with check option [constraint <name>]];

The optional clause or replace re-creats the view if it already exists. <column (s)> names the columns of the view If <column(s)> is not specified in the view definition, the columns of the view get the same names as the attributes listed in the select statement (if possible).

A view can be used in the same way as a table, that is, rows can be retrieved from a view (also respective rows are not physically stored, but derived on basis of the select statement in the view Definition), or rows can be even modified. A view is evaluated again each time it is accessed. In oracle SQL no insert, update or delete medications on views are allowed that use one of the following constructs in the view definition:

- Joins
- Aggregate function
- Set-valued sub. queries.
- Group by clause or distinct clause, ordered by

A view can be deleted using the commond delete <view name>.

Removing a view

Syntax : Drop view <view_name>
eg Drop view stud

Index : An index is a schema objects that can speed up retrieval of rows by using pointer. An index provides direct and fast access to rows in a table. Index can be created explicitly or automatically.

Automatically : A unique index is created automatically when you define a primary key or unique key constraint in a table definition.

Manually : users can create non unique indexes or columns to speed up access time to the rows.

Syntax : create index <index name>
on table (column[, column]));
ie table name = 'emp';

Removing an Index :
Syntax : Drop index <index_name>;
eg Drop index emp_name_idx;

Note :-
1) we cannot modify indexes.
2) To change an index, we must drop it and re-create it

Conclusion :-

I have understood the DDL commands, syntax of the DDL commands and how to apply DDL commands on the database.

17/8/23

...ored in tables with

Assignment Questions.

1) What do you mean by DBMS? What are its different types?

→ 1) DBMS stands for "Database Management System."
2) It is a software that enables users to interact with databases by providing an interface for creating, organizing, retrieving, updating and managing data
3) DBMS systems help ensure data integrity, security, and efficient data manipulation
4) Types of DBMS :-
   i) Relational DBMS (RDBMS).
   ii) NoSQL DBMS
   iii) object-oriented DBMS (OODBMS)
   iv) NewSQL DBMS
   v) In-Memory DBMS
   vi) Columnar DBMS
   vii) Time-Series DBMS.

2) What is RDBMS? How is it different from DBMS?

→ 1) RDBMS stands for Relational "Database Management System".
2) It is a DBMS that organizes and manages data in a structured manner using a relational model. In RDBMS, data is stored in tables with

rows and columns, and relationships between tables are defined using keys, such as primary keys and foreign keys.

3) The main difference between DBMS and RDBMS is that, In DBMS, the data is stored as a file, whereas in RDBMS, data is stored in the form of tables.

3) Explain DDL commands with syntax and example.

→ 1) create table :-

Syntax :- create table table_name (
column1 datatype, column2 datatype ...);

Example :- create table employee (EmployeeID int(5),
Name varchar(30));

2) Alter table :-

Syntax :- alter table table_name
ADD column_name datatype;

Example :- alter table Employee
ADD Email varchar(100);

3) Drop table :

Syntax :- Drop table table_name;

Example :- Drop table Employees;

4) create index :-

Syntax :- create index index_name
on table_name (column1, column2, ...);

Example :- create index idx_EmployeeLastName
on Employees (LastName);

5) Drop index :-

Syntax :- Drop index index_name
On table_name;

Example :- Drop index idx_EmployeeLastName
on Employees;

6) create view :-

Syntax :- create view view_name as
Select column1, column2, ...
from table_name
where condition;

Example :- create view EmployeeDetails as
select EmployeeID, FirstName, LastName, DeptID
from Employees
where DeptID = 101;

7) Drop view :-
Syntax :- Drop view view name;

Example :- Drop view EmployeeDetails;

17/8/23