Lab 02: DDL Queries

Objective(s) :

To learn Data Definition Language (DDL)

DDL

DDL is a standard subset of SQL that is used to define tables (database structure), and other metadata related things. The few basic commands include: CREATE TABLE, DROP TABLE, ALTER TABLE and TRUNCATE TABLE. There are many other statements, but those are the ones most commonly used.

Database Objects

An Oracle database can contain multiple data structures. Each structure should be outlined in the database design so that it can be created during the build stage of database development.

- Table: Stores data
- View: Subset of data from one or more tables
- Sequence: Numeric value generator
- Index: Improves the performance of some queries
- Synonym: Gives alternative names to objects

Oracle9*i* Table Structures

- Tables can be created at any time, even while users are using the database.
- You do not need to specify the size of any table. The size is ultimately defined by the amount of space allocated to the database as a whole. It is important, however, to estimate how much space a table will use over time.
- Table structure can be modified online.

Naming Rules

Name database tables and columns according to the standard rules for naming any Oracle database object:

- Table names and column names must begin with a letter and be 1–30 characters long.
- Names must contain only the characters A–Z, a–z, 0–9, _ (underscore), $, and # (legal characters, but their use is discouraged)
- Names must not duplicate the name of another object owned by the same Oracle server user.
- Names must not be an Oracle server reserved word.

Naming Guidelines

Use descriptive names for tables and other database objects.

Note: Names are case insensitive. For example, EMPLOYEES is treated as the same name as eMPloyees or eMpLOYEES.

CREATE TABLE

Probably the most common DDL statement is 'CREATE TABLE'. Intuitively enough, it is used to create tables. The general format is something along the lines of:

```
CREATE TABLE <table-name> (
...
);
```

The ... is where column definitions go. The general format for a column definition is the column name followed by column type. For example: PERSONID INT

Which defines a column name PERSONID, of type NUMBER. Column names have to be comma separated, i.e.:

```
CREATE TABLE PERSON (
PERSONID NUMBER,
LNAME VARCHAR2(20),
FNAME VARCHAR2(20) NOT NULL,
DOB DATE,
PRIMARY KEY(PERSONID)
);
```

The above creates a table named person, with person id, last name, first name, and date of birth. There is also the 'primary key' definition. A primary key is a column value that uniquely identifies a database record. So for example, we can have two 'person' records with the same last name and first name, but with different ids. Besides for primary key, there are many other flags we can specify for table columns. For example, in the above example, FNAME is marked as NOT NULL, which means it is not allowed to have NULL values3.

Many databases implement various extensions to the basics, and you should read the documentation to determine what features are present/absent, and how to use them.

DROP TABLE

Just like there is a 'create table' there is also a 'drop table', which simply removes the table. Note that it doesn't ask you for confirmation, and once you remove a table, it is gone forever.

```
DROP TABLE <table-name> ;
```

ALTER TABLE

There is a command to ‘alter’ tables after you create them. This is usually only useful if the table already has data, and you don’t want to drop it and recreate it (which is generally much simpler). Also, most databases have varying restrictions on what ‘alter table’ is allowed to do. For example, Oracle allows you do add a column, but not remove a column.

The general syntax to add a field is:

```
ALTER TABLE <table-name>
ADD (<field-name><data-type>)
```

The field declaration is pretty much exactly what it is in the ‘create table’ statement.

The general syntax to drop a field is:

```
ALTER TABLE <table-name>
DROP COLUMN <field-name>
```

Note that very few databases let you drop a field. The drop command is mostly present to allow for dropping of constraints (such as indexes, etc.) on the table.

The general syntax to modify a field (change its type, etc.) is:

```
ALTER TABLE <table-name>
MODIFY (<field-name><new-field-declaration>)
```

Note that you can only do this to a certain extent on most databases. Just as with ‘drop’, this is mostly useful for working with table constraints (changing ‘not null’ to ‘null’, etc.)

Exercises

Using EMP table, solve the following queries (1-5).

1. Create a replica of EMP table with all the records in it.

2. Add a column 'Address' in it.

3. Drop column 'Address' from it.

4. Add columns 'House No' character ,'Street No' numeric, 'Area' character ,'City' character in it with the respective data types.

5. Change the data type of 'House No' from character to numeric.

6. Create the Data Definitions for each of the relations shown below, using SQL DDL. Assume the following attributes and data types:
   FACULTY:
   FacultyID (integer, primary key)
   FacultyName (25 characters)

   COURSE:
   CourseID (8 characters, primary key)
   CourseName (15 characters)

   CLASS:
   ClassID (8 characters)
   CourseID (8 characters foreign key)
   SectionNo (integer)
   Semester (10 characters)

   STUDENT:
   StudentID (integer, primary key)
   StudentName (25 characters)
   FacultyID (integer foreign key)

7. How would you add an attribute, CLASS, to the STUDENT table?

8. How would you remove the FACULTY and STUDENT tables?