Lab 2
Data Definition and Data Manipulation

CSE 4308

DATABASE MANAGEMENT SYSTEMS LAB

AUGUST 22, 2023

1 Data Definition

1.1 Creating a Table

The general syntax for creating a table is as follows:

```
CREATE TABLE table_name
(
attribute1 datatype [ NULL | NOT NULL ],
attribute2 datatype [ NULL | NOT NULL ],
...
);
```

There exist different data types in Oracle. Some of them are as follows:

- `char(n)`: value contains exactly *n* alpha-numeric characters
- `varchar2(n)`: value contains at most *n* alpha-numeric characters
- `number`: any integer or real numbers
- `date`: DD-MON-YY format, like '20-JAN-22'

Assume that you want to create a table named 'TEST' with 3 attributes:

1. `national_id`: `number` type
2. `name`: `varchar2` type
3. `birth_date`: `date` type

To create this table, you have to write the following query:

```
CREATE TABLE CITIZEN
(
  NATIONAL_ID NUMBER NOT NULL,
  NAME VARCHAR2(50) NOT NULL,
  BIRTH_DATE DATE
);
```

To create a table with constraints, the syntax is given below:

```
CREATE TABLE table_name
(
attribute1 datatype [ NULL | NOT NULL ],
attribute2 datatype [ NULL | NOT NULL ],
...,
[CONSTRAINT constraint_name] PRIMARY KEY (primary_attribute1, ...),
[CONSTRAINT constraint_name] CHECK condition
);
```

Primary key is a special column that is able to uniquely identify each record. For example, if we want to create a table called `CITIZEN`, it might have the `NATIONAL_ID` as the primary key.

```
CREATE TABLE CITIZEN
(
  NATIONAL_ID NUMBER NOT NULL,
  NAME VARCHAR2(50) NOT NULL,
  AGE INT,
  COUNTRY VARCHAR2(20),
  CONSTRAINT PK_COURSE PRIMARY KEY(NATIONAL_ID),
  CONSTRAINT VALIDITY_CHECK CHECK (AGE > 17 AND COUNTRY= '
   BANGLADESH')
);
```

1.2 Droping Tables

To delete the schema, we use:

```
DROP TABLE table_name;
```

It will also delete the information stored in the table.
To delete the table structure with constraints, we generally follow this command:

```
DROP TABLE table_name CASCADE CONSTRAINTS;
```

```
DROP TABLE CITIZEN CASCADE CONSTRAINTS;
```

1.3 Altering Tables

To add a new attribute to the table, we use:

```
ALTER TABLE table_name ADD attribute_name datatype;
```

We can even add multiple attributes at the same time:

```
ALTER TABLE table_name ADD (attribute1 datatype, ...);
```

```
ALTER TABLE CITIZEN ADD PHONE_NO VARCHAR2(12);
```

To delete an attribute from a table, we use:

```
ALTER TABLE table_name DROP COLUMN attribute_name;
```

We can even delete multiple attributes at the same time:

```
ALTER TABLE table_name DROP COLUMN (attribute1, ...);
```

```
ALTER TABLE CITIZEN DROP COLUMN PHONE_NO;
```

To modify the data type of an attribute, we need to ensure that the column is empty. Then we can execute:

```
ALTER TABLE table_name MODIFY attribute_name new_datatype;
```

```
ALTER TABLE CITIZEN MODIFY AGE NUMBER(2, 1);
```

To rename an attribute, we use:

```
ALTER TABLE table_name RENAME COLUMN old_attribute_name TO new_attribute_name;
```

To rename a table, we use:

```
ALTER TABLE table_name RENAME TO new_table_name;
```

2 Data Manipulation

2.1 Inserting Records/Rows into a Table

The general format for inserting a new record is:

```
INSERT INTO TABLE_NAME VALUES (..., ..., ...);
```

We can type the following command to insert new records into our 'TEST' table using positional notation:

```
INSERT INTO CITIZEN VALUES (2015001, 'W', 19, 'Bangladesh');
```

We can insert new records into 'TEST' table following named notation:

```
INSERT INTO CITIZEN (NATIONAL_ID, NAME, AGE, COUNTRY) VALUES
    (2015002, 'X', 23, 'Bangladesh');
```

2.2 Retrieval of Information

The basic SQL syntax of a query is as follows:

```
SELECT ATTRIBUTE1, ATTRIBUTE2
FROM TABLE_NAME
WHERE <CONDITIONAL CLAUSE>;
```

For example, to see the 'national_id' of all citizens from the 'CITIZEN' table:

```
SELECT NATIONAL_ID FROM CITIZEN;
```

To find the information of a citizen following a condition:

```
SELECT NATIONAL_ID FROM CITIZEN WHERE ID = 2015001;
```

To find the information of a citizen following multiple conditions:

```
SELECT NATIONAL_ID FROM CITIZEN WHERE AGE>21 AND COUNTRY='
    Bangladesh';
```

```
SELECT NATIONAL_ID FROM CITIZEN WHERE AGE>21 OR NAME='W';
```

To find all records and their information:

```
SELECT * FROM CITIZEN;
```

3 Executing SQL Script in SQLPlus

You can create a file with .sql extension that contains your sql statements. Then you can execute it from the SQLPlus command line directly.
Suppose, you have written your SQL statements in a file 'a.sql' saved under 'd:\sample' directory. To execute that script, you have to type:

```
@"d:\sample\a.sql"
```

4 Lab Task

You have to write all SQL statements in an editor first and save them with .sql extention. Then execute the SQL script.

1. Create a user with username = <C_student_id> and password = cse4308 and grant necessary privileges to log in and execute DDL and DML statements. Then log in as that user.

2. Write SQL statement to create a table 'PRODUCT_INFO' which has 4 attributes:
   - P_ID (assign it as Primary Key)
   - MODEL
   - MANUFACTURER
   - PRICE (ensure that PRICE is greater than 0)

3. Write SQL statements to insert the following records into 'PRODUCT_INFO' table:

P_ID	MODEL	MANUFACTURER	PRICE
10203	X515EA	ASUS	51900
20301	Latitude 14 3420	DELL	82500
20311	Inspiron 15 3525	DELL	58900
20114	P2451FA	ASUS	58500
20122	X415EA	ASUS	57000
30301	Vostro 14 3400	DELL	83000
00788	15s-eq3619AU	HP	73200
30583	P1412CEA	ASUS	63000
30543	15s-fq5486TU	HP	58500
10766	IdeaPad Slim 3i	Lenovo	56900
00821	K14	Lenovo	66500
30345	SModern 15 B11M	MSI	74000

4. Write SQL statements to perform the following queries:
   (a) Display all records of 'PRODUCT_INFO' table.
   (b) Show the MODELs and MANUFACTURERs only.
   (c) Find the MODELs and MANUFACTURERs that have price of less than 70000.
   (d) Find MODELs and MANUFACTURERs that have prices in between 75000 and 55000 (inclusive).
   (e) Find P_IDs and Models of the laptop of ASUS.
   (f) Find MODELs and Prices of the laptop of DELL.
   (g) Find P_IDs and MODELs of laptops that cost more than 55000 or of HP brand.
   (h) Find the name of the MANUFACTURERs.

5. Drop the 'PRODUCT_INFO' table with all its constraints.