Lab 03: SQL Data Definition Language (DDL) and Insertion

Objective

The students should be able to:

1. Understand how to CREATE tables in the database
2. data types (char, varchar2, number, etc.)
3. Apply table constraints (not null, unique, primary key, foreign key, check)
4. Column-level and table level constraints
5. Table creation using subqueries
6. Insert data
7. Drop table

Submission Requirements

Save your script file and upload it to LMS.

Creating new user

We will create a new user for this lab as we will be creating a new schema. For students using the lab PCs, the steps are shared with the demo.

Connect the user to the database using the SQL Developer. Create a new connection, give a suitable schema name, enter your username and password and connect to the service.

Open a new script file and start writing SQL queries. If you face any issues with the system, Oracle or Sql developer, you can switch to LiveSQL for backup.

Queries

Write SQL commands for the following. Where data types are not provided, you may make assumptions that reflect a real-life scenario.

Part A: CREATE tables

1. Create a table `my_first_table` with the following.

Variable	Data type
Name	Varchar2(255)
Email	Varchar2(255)
Age	Int
Marks	Number (3,1)

   Execute `DESC my_first_table` to confirm the structure of the table.

2. Create a table for `Products` containing product_id, product_name, price, supplier_id, category_id. An example of product price is 10.50.

Q3 and 4 will prompt an error on Oracle but will work for other DBMS. You may type your query in your answers without executing.

3. Consider the situation where we only need to create the `Products` table if it doesn't already exist. The attributes will remain the same as in 1.
4. Create a table named `duplicate_Products` which is similar to the products table. Only the copy of table structure and not the data.
5. Create a table named `dup_data_Products` which is a duplicate copy of Products table including structure and data. Execute the DESC command for products and this new table to compare the components.
6. Create a table for `Suppliers` containing supplier_id (starts from 1,2,3...), supplier_name, address, city (3 characters e.g KHI). Make sure none of the attributes are allowed to have null values.
7. Create a table named `Order_Details` including order_id, product_id, quantity. Assume orders with quantity greater than 50 are not allowed. Execute the command "INSERT INTO Order_Details VALUES (1,1,100);". Then execute "SELECT * FROM Order_Details" to see the data being inserted.
8. Create another suppliers table and name it as `Supplier_2` with the same attributes and datatypes. Assume that the suppliers from Karachi (KHI), Islamabad (ISB) or Lahore (LHR) can be entered. Execute the following to test:
   a. INSERT INTO Supplier_2 (supplier_name, address, city)
      Values ('MySupplier1','MyAddress','LHR');
   b. INSERT INTO Supplier_2 (supplier_name, address, city)
      Values ('MySupplier2','MyAddress','RWP');
   c. SELECT * FROM Supplier_2

9. Create a table named `Orders` including order_id, order_date and ship_address. We need to ensure that the date entered follows the specific format like '--/--/----'. Hint: Use like operator with wildcards to specify the pattern for date as a string. Execute the following to test:
    a. INSERT INTO Orders VALUES (1,'23/05/2023','Address');
    b. INSERT INTO Orders VALUES (2, '2-5-2023','Address2');

    Modify the datatype for order date to DATE and record your observations after you insert another row. What do we need to ensure on insertion? *Hint: To_Date*

10. Alter the table `Products` and apply primary key constraint on product_id.
11. Drop the Suppliers table. Create the `Suppliers` table again. This time make sure to set the supplier_id as primary key. Also remember that it cannot be Null.
12. Create a table for `Categories` containing category_id (as Primary Key), category_name, description.
13. Drop the products table. Create the `Products` table again containing product_id (PK), product_name, price, supplier_id (foreign key referencing to suppliers table), category_id (foreign key referencing to categories table). Restrict duplicate records. Hint: use *unique* constraint for essential attributes. Make sure to give names for all constraints.
14. Drop the orders table. Create the `Orders` table again containing order_id, order_date and ship_address. Set the order_id as primary key which is unique and auto incremented. The default value for order_date is sysdate. Execute to verify default date:
    a. INSERT INTO Orders (ship_address) VALUES ('Address');
    b. SELECT * FROM Orders;
15. Drop the `Order_Details` table. Create the Order_Details table containing order_id, product_id, quantity. The product_id and order_id forms the composite primary key and are related to the Orders table and Products table respectively.

Part B: Insert Data

16. Insert into `Categories` the following data

Category_id	Category_Name	Description
1	Beverages	Soft drinks, coffees, teas
2	Dairy Products	Cheese
3	Condiments	Sweet and savory sauces, spreads and seasonings
4	Confections	Desserts and candies

17. Insert into `Suppliers`

Supplier_id	Supplier_name	Address	City
1	Alpha	205 A, Street 11, Gulshan-e-Iqbal	KHI
2	Bravo	100 B, Street 2, F-6/3	ISB

18. Insert into `Products`

Product_id	Product_name	Price	Supplier_id	Category_id
1	Chai	100.00	1	1
2	Cheddar Cheese	950.00	2	2
3	BBQ sauce	500.00	2	3
4	Coffee	200.00	1	1
5	Sprite	80.00	1	1
6	Mayo Garlic Sauce	450.00	2	3

Recalling older concepts

19. Display all products belonging to category_id =1 and supplier _id=1
20. Display the average price of products for each category_id.
21. Display all category names sorted in the alphabetical order.

Self Practice: Enrollment Database

You may attempt this for your own learning – no need to submit

Consider the ER Model for the enrollment database.

Write the necessary DDL commands to create student table and course table. You may ignore the Foreign key constraints in this case.