Database Systems

Lab Manual (Lab 3)

****

Session: Fall 2024

LAB INSTRUCTOR: AYESHA MAJID ALI

Lab Manual: Creating DDL Commands in Oracle 10g

### Objective:

To learn how to create and use DDL commands (specifically the CREATE command) to define the structure of a relational database in Oracle 10g.

### Prerequisites:

* Basic knowledge of SQL.
* Access to Oracle 10g Database.

### Lab Environment Setup:

1. **Oracle Installation**: Ensure that Oracle 10g is installed and running.

### **Step 1: Opening oracle 10 g.**

Way 1:

Open Oracle 10g from the following path.

C:\oraclexe\app\oracle\product\10.2.0\server

Way 2:

Go to command prompt and find your port as following and insert that port in this link and paste it in browser ([http://127.0.0.1:<yourport>/apex/](http://127.0.0.1:%3cyourport%3e/apex/)) or ([http://localhost:<yourport>/apex/](http://localhost:%3cyourport%3e/apex/)) .

cd C:\oraclexe\app\oracle\product\10.2.0\server\BIN

lsnrctl status

In my case it is 8081 as you can see from the following snapshot.

A screenshot of a computer program

Description automatically generated

example: <http://127.0.0.1:8081/apex/>

### **Step 2: Sign in to your account.**

Username: System

Password: 1234

If you have set a different password enter that

A screenshot of a computer

Description automatically generated

### **Step 3: Click on sql>sql command this is where you are going to write all your sql commands.**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

### **Step 4: Create a Sample Table**

Create a table named Courses to store course information.

CREATE TABLE Courses (

CourseID NUMBER PRIMARY KEY,

CourseName VARCHAR2(100) NOT NULL,

Credits NUMBER CHECK (Credits > 0),

Department VARCHAR2(50)

);

### **Step 5: Verify Table Creation**

To verify that the tables were created successfully, run:

SELECT table\_name FROM user\_tables;

**SAMPLE EXAMPLES FOR SQL TABLE CREATION**

**Example 1: Create an Employees Table with Complex Constraints**:

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

FirstName VARCHAR2(50) NOT NULL,

LastName VARCHAR2(50) NOT NULL,

Email VARCHAR2(100) UNIQUE NOT NULL,

Phone VARCHAR2(15),

Salary NUMBER CHECK (Salary > 0),

DepartmentID NUMBER

);

**Example 2: Verifying email using regex.**

### Step 1: Create a Table

First, create a table that includes an email field.

CREATE TABLE Employees (

EmployeeID NUMBER PRIMARY KEY,

FirstName VARCHAR2(50) NOT NULL,

LastName VARCHAR2(50) NOT NULL,

Email VARCHAR2(100) UNIQUE NOT NULL

);

### Step 2: Create a Trigger to Validate Email

You can create a trigger that fires before an insert operation to validate the email format using a regular expression.

CREATE OR REPLACE TRIGGER trg\_validate\_email

BEFORE INSERT ON Employees

FOR EACH ROW

DECLARE

v\_invalid\_email EXCEPTION;

BEGIN

IF NOT REGEXP\_LIKE(:NEW.Email, '^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}$') THEN

RAISE v\_invalid\_email;

END IF;

EXCEPTION

WHEN v\_invalid\_email THEN

RAISE\_APPLICATION\_ERROR(-20001, 'Invalid email format: ' || :NEW.Email);

END;

### Explanation of the Trigger

* **BEFORE INSERT**: The trigger activates before a new row is inserted.
* **REGEXP\_LIKE**: This function checks if the new email matches the defined regex pattern for valid email formats.
* **RAISE\_APPLICATION\_ERROR**: If the email is invalid, this raises an error with a custom message.

### Step 3: Testing the Setup

#### Valid Email Insertion

You can now attempt to insert a valid email:

INSERT INTO Employees (EmployeeID, FirstName, LastName, Email)

VALUES (1, 'John', 'Doe', 'john.doe@example.com');

This should succeed.

#### Invalid Email Insertion

Try inserting an invalid email:

INSERT INTO Employees (EmployeeID, FirstName, LastName, Email)

VALUES (2, 'Jane', 'Smith', 'jane.smith@.com');

This should fail with an error message indicating that the email format is invalid.

**TIMESTAMP DATATYPE**

A **timestamp** is a data type used in databases to store date and time information. It typically represents a specific point in time, including both the date and the time down to fractions of a second.

### Key Features of Timestamp:

1. **Format**: Timestamps usually have a standard format, such as YYYY-MM-DD HH:MM:SS, which includes the year, month, day, hour, minute, and second.
2. **Time Zone Awareness**: Some databases support timezone-aware timestamps, allowing you to store and manipulate time data with respect to specific time zones.
3. **Automatic Management**: Many databases allow you to set a timestamp to automatically update to the current date and time when a record is created or modified. For example, using DEFAULT CURRENT\_TIMESTAMP in a table definition will set the timestamp field to the current time whenever a new record is inserted.
4. **Use Cases**: Timestamps are commonly used for:
   * Tracking when records were created or updated.
   * Logging events in applications.
   * Recording transaction times in e-commerce systems.

### Example in SQL:

Here’s a simple example of how you might define a timestamp column in a SQL table:

CREATE TABLE Events (

EventID INT AUTO\_INCREMENT PRIMARY KEY,

EventName VARCHAR(100),

EventDate TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

In this example, EventDate will automatically store the current date and time whenever a new event is added to the Events table.

### Example 1: Specifying a Fixed Timestamp

You can insert a specific timestamp value into a table rather than using the current timestamp.

CREATE TABLE Appointments (

AppointmentID INT AUTO\_INCREMENT PRIMARY KEY,

AppointmentTime TIMESTAMP

);

INSERT INTO Appointments (AppointmentTime)

VALUES ('2024-11-01 15:30:00');

**ERROR WITH WRONG FORMAT:**

Inserting a timestamp in the wrong format will typically result in an error. Most relational databases have specific formats that they expect for timestamp values, and if the value doesn't conform to these formats, the database will reject the insertion and raise an error.

### Common Scenarios of Errors with Timestamps

1. **Invalid Format**: If you try to insert a timestamp using an unsupported format, you'll likely get an error. For example, if the database expects the format YYYY-MM-DD HH:MM:SS and you provide something like MM-DD-YYYY HH:MM:SS, it will fail.

INSERT INTO Events (EventTime) VALUES ('12-31-2024 15:30:00'); -- Wrong format

1. **Out of Range Values**: If you provide values that are out of the valid range (e.g., a month value greater than 12 or a day value greater than the maximum for the given month), the database will also return an error.

INSERT INTO Events (EventTime) VALUES ('2024-13-01 15:30:00'); -- Invalid month

**3.Invalid Data**: If the string contains non-numeric characters where numbers are expected, the database will raise an error.

**Example**:

INSERT INTO Events (EventTime) VALUES ('2024-11-xx 15:30:00'); -- Invalid day

4. **Database-Specific Behavior**: Different databases might have slightly different formats or behaviors. For example, PostgreSQL is stricter about formats compared to MySQL. Always refer to your database documentation for the specific format it expects.

**TIME STAMP FORMAT IN ORACLE 10 G**

In Oracle 10g, the **TIMESTAMP** data type is used to store date and time values, including fractional seconds. Here are the key points about timestamp format and usage in Oracle 10g:

### TIMESTAMP Format

1. **Basic Format**:
   * The standard format for a TIMESTAMP is YYYY-MM-DD HH24:MI:SS.FF, where:
     + YYYY - 4-digit year
     + MM - 2-digit month (01 to 12)
     + DD - 2-digit day (01 to 31)
     + HH24 - 2-digit hour in 24-hour format (00 to 23)
     + MI - 2-digit minute (00 to 59)
     + SS - 2-digit second (00 to 59)
     + FF - Fractional seconds (up to 9 digits for nanoseconds)

**Example of TIMESTAMP**:

TIMESTAMP '2024-10-23 15:30:45.123456789'

### Creating a Table with TIMESTAMP

You can create a table that includes a TIMESTAMP column as follow

CREATE TABLE Events (

EventID NUMBER PRIMARY KEY,

EventName VARCHAR2(100),

EventTime TIMESTAMP

);

### Inserting TIMESTAMP Values

When inserting TIMESTAMP values into the table, use the TIMESTAMP keyword:

INSERT INTO Events (EventID, EventName, EventTime)

VALUES (1, 'Meeting', TIMESTAMP '2024-10-23 15:30:00.123456');

### Using SYSDATE and SYSTIMESTAMP

* **SYSDATE**: Returns the current date and time but without fractional seconds. It is of the DATE type.
* **SYSTIMESTAMP**: Returns the current date and time, including fractional seconds and time zone information.

**Example**:

INSERT INTO Events (EventID, EventName, EventTime)

VALUES (2, 'Conference', SYSTIMESTAMP);

**TASKS:**

### **TASK 1:** Create a Database Table for Customer Information

**Objective:** Design and implement a table in the database to store customer information for a retail application.

 **Table Name:** Customers

 **Fields/Columns:**

* **CustomerID**:
  + Type: Integer
  + Properties: Primary Key
  + Description: Unique identifier for each customer.
* **FirstName**:
  + Type: Varchar(50)
  + Properties: Not Null
  + Description: Customer's first name.
* **LastName**:
  + Type: Varchar(50)
  + Properties: Not Null
  + Description: Customer's last name.
* **Email**:
  + Type: Varchar(100)
  + Properties: Not Null, Unique
  + Description: Customer's email address.
* **PhoneNumber**:
  + Type: Varchar(15)
  + Properties: Nullable
  + Description: Customer's phone number.
* **DateOfBirth**:
  + Type: Date
  + Properties: Nullable
  + Description: Customer's date of birth.

### Task 2: Create a Table for Products

**Objective:** Design and implement a table to store product information for a retail application.

#### Requirements:

1. **Table Name:** Products
2. **Fields/Columns:**
   * **ProductID:** Integer, Primary Key
   * **ProductName:** Varchar(100), Not Null
   * **Price:** Decimal(10, 2), Not Null
   * **StockQuantity:** Integer, Not Null
   * **Category:** Varchar(50), Nullable

### Task 3: Create a Table for Product Reviews

**Objective:** Design and implement a table to store product reviews provided by customers.

#### Requirements:

1. **Table Name:** ProductReviews
2. **Fields/Columns:**
   * **ReviewID:** Integer, Primary Key
   * **ProductID:** Integer, Foreign Key referencing Products.ProductID
   * **CustomerID:** Integer, Foreign Key referencing Customers.CustomerID
   * **Rating:** Integer (1 to 5), Not Null
   * **ReviewText:** Text, Nullable
   * **ReviewDate:** Timestamp, Default to current timestamp