Lab 1: SQL2 Data Manipulation Language – Use DML operations to manage database tables

SECD2523 – 06 Database

Universiti Teknologi Malaysia

Objective:

1. To understand the purpose of data manipulation language (DML).

2. To identify the DDL operations needed to manage a database's table data:

    a. INSERT
    b. UPDATE
    c. DELETE

Name : Nur Hanisah Izzati Binti Abdul Haniff

Matric No : A22EC0097

Section : 06

EXERCISE 1: DATA MANIPULATION LANGUAGE

Use DML operations to manage database tables

In this exercise you will populate and work with the data that is stored in the database system tables.

Part 1: Running a script to populate the tables

You have to consider the order of the tables when populating them. A table that has a foreign key field cannot be populated before the related table with the primary key.

1. Use the table mapping document and list the order that you would use to populate the tables.

Entity in the provided script order:

- 'teams'
- 'inventory_list'
- 'sales_representatives'
- 'sales_rep_addresses'
- 'customers'
- 'customers_addresses'
- 'items'
- 'price_history'
- 'orders'
- 'ordered_items'

2. Open the “sports data.sql” and look at the order the data is being added there, does your list match? This file can be found in the Section 6 Lesson 4 interaction (sports data.zip) and must first be extracted.

Entity in ‘sports data.sql’ order:

- ‘inventory_list’
- ‘items’
- ‘price_history’
- ‘sales_representatives’
- ‘sales_rep_addresses’
- ‘teams’
- ‘customers’
- ‘customers_addresses
- ‘orders’
- ‘ordered_items’

The list in the ‘sports data.sql’ order does match with my list order in (1).

3. Run the “sports data.sql” script in APEX to populate your tables.

```
134  VALUES(5, 5, 'or0101250', 'im01101046');
135
136  INSERT INTO ordered_items (quantity_ordered, quantity_shipped, od
137  VALUES(5, 5, 'or0101350', 'im01101044');
138
139  INSERT INTO ordered_items (quantity_ordered, quantity_shipped, od
140  VALUES(18, 18, 'or0101425', 'im01101047');
141
142  INSERT INTO ordered_items (quantity_ordered, quantity_shipped, od
143  VALUES(10, 10, 'or0101681', 'im01101047');
144
145  INSERT INTO ordered_items (quantity_ordered, quantity_shipped, od
146  VALUES(1, 1, 'or0101750', 'im01101048');
```

```
1 row(s) inserted.
```

4. Check that no errors occurred when you ran the script.

No errors has occurred.

Part 2: Inserting rows to the system

1. Add a new team to the system.

id	name	Number_of_players	discount
t004	Jets	10	5

```
INSERT INTO teams (id, name, number_of_players, discount)
VALUES ('t004', 'Jets', 10, 5);
```

```
1 row(s) inserted.
```

2. Add a new Customer with the following details to the system.

ctr number	email	First name	Last name	Phone number	Current balance	Loyalty card number	tem id	sre id
c02001	brianrog@hootech.com	Brian	Rogers	01654564898	-5	lc4587		

```
INSERT INTO customers (
    ctr_number, email, first_name, last_name, phone_number,
    current_balance, loyalty_card_number, tem_id, sre_id
)
VALUES (
    'c02001', 'brianrog@hootech.com', 'Brian', 'Rogers', '01654564898',
    -5, 'lc4587', null, null
);
```

```
ORA-02290: check constraint (SQL_DFLBLDMJICMTVENWKDPWELAUR.CHECK_BALANCE) violated ORA-06512: at "SYS.DBMS_SQL", line 1721

More Details: https://docs.oracle.com/error-help/db/ora-02290
```

3. This information violates the check constraint that the current balance must not be less than zero. Change the current balance to 50 and rerun the query.

```
INSERT INTO customers (
    ctr_number, email, first_name, last_name, phone_number,
    current_balance, loyalty_card_number, tem_id, sre_id
)
VALUES (
    'c02001', 'brianrog@hootech.com', 'Brian', 'Rogers', '01654564898',
    50, 'lc4587', null, null
);
```

```
1 row(s) inserted.
```

EXERCISE 2: DATA MANIPULATION LANGUAGE

Use DML operations to manage database tables

In this exercise you will populate and work with the data that is stored in the database system.

Part 1: Updating rows to the system

1. Run the following query to view the content of the price_history table:

```
SELECT start_date, TO_CHAR (start_time, 'HH24:MI:SS'),
price, end_date, TO_CHAR (end_time, 'HH24:MI')
FROM price_history;
```

```
1 SELECT start_date, TO_CHAR (start_time, 'HH24:MI:SS'), price, end_date, TO_CHAR (end_time, 'HH24:MI')
2 FROM price_history;
3
```

START_DATE	TO_CHAR(START_TIME,'HH24:MI:SS')	PRICE	END_DATE	TO_CHAR(END_TIME,'HH24:MI')
17-JUN-17	09:00:00	4.99	-	-
25-NOV-16	09:00:00	14.99	25-JAN-17	17:00
25-JAN-17	17:01:00	8.99	25-JAN-17	19:00
26-JAN-17	09:00:00	15.99	-	-
12-FEB-17	12:30:00	7.99	-	-
25-APR-17	10:10:10	24.99	-	-
31-MAY-17	16:35:30	149	-	-

2. Obl is going to update the price of the premium bat so you will need to write a query that will close off the current price by adding the system date values to the end_date and end_time fields. To run this query you will need to both match the item number and identify that the end date is null. This ensures that you are updating the latest price.

```
1  UPDATE price_history
2  SET end_date = SYSDATE, end_time = SYSDATE
3  WHERE itm_number = 'im01101048' AND end_date IS NULL;
```

```
1 row(s) updated.
```

3. Rerun the select statement on the price_history table to ensure that the statement has been executed.

```
1  SELECT start_date, TO_CHAR (start_time, 'HH24:MI:SS'), price, end_date, TO_CHAR (end_time, 'HH24:MI')
2  FROM price_history;
3
```

START_DATE	TO_CHAR(START_TIME,'HH24:MI:SS')	PRICE	END_DATE	TO_CHAR(END_TIME,'HH24:MI')
17-JUN-17	09:00:00	4.99	-	-
25-NOV-16	09:00:00	14.99	25-JAN-17	17:00
25-JAN-17	17:01:00	8.99	25-JAN-17	19:00
26-JAN-17	09:00:00	15.99	-	-
12-FEB-17	12:30:00	7.99	-	-
25-APR-17	10:10:10	24.99	-	-
31-MAY-17	16:35:30	149	10-NOV-23	07:33

4. Insert a new row that will use the current date and time to set the new price of the premium bat to be 99.99.

```
INSERT INTO price_history (start_date, start_time, price, itm_number)
VALUES (SYSDATE, SYSDATE, 99.99, 'im01101048');
```

```
1 row(s) inserted.
```

5. Rerun the select statement on the price_history table to ensure that the statement has been executed.

START_DATE	TO_CHAR(START_TIME,'HH24:MI:SS')	PRICE	END_DATE	TO_CHAR(END_TIME,'HH24:MI')
17-JUN-17	09:00:00	4.99	-	-
25-NOV-16	09:00:00	14.99	25-JAN-17	17:00
25-JAN-17	17:01:00	8.99	25-JAN-17	19:00
26-JAN-17	09:00:00	15.99	-	-
12-FEB-17	12:30:00	7.99	-	-
25-APR-17	10:10:10	24.99	-	-
31-MAY-17	16:35:30	149	10-NOV-23	07:33
10-NOV-23	07:38:10	99.99	-	-

Part 2: Deleting rows from the system

1. Bob Thornberry has contacted Obl to ask that the 83 Barrhill Drive address be removed from the system as he can longer receive parcels at this address. Write a SQL statement that will remove this address from the system.

```
1 DELETE FROM customers_addresses
2 WHERE id = 'ca0101';
3
```

```
1 row(s) deleted.
```

2. Run a select statement on the customers_addresses table to ensure that the statement has been executed.

```
1 SELECT * FROM customers_addresses;
2
```

ID	ADDRESS_LINE_1	ADDRESS_LINE_2	CITY	ZIP_CODE	CTR_NUMBER
ca0102	17 Gartsquare Road	Starford	Liverpool	LP89JHK	c00001
ca0103	54 Ropehill Crescent	Georgetown	Star	ST45AGV	c00101
ca0104	36 Watercress Lane	-	Jump	JP23YTH	c01986
ca0105	63 Acacia Drive	Skins	Liverpool	LP83JHR	c00001

Download CSV

4 rows selected.