EXERCISE 1

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.

Inventory_list
Items
Price_history
Sales_representatives
Sales_representatives_addresses
Teams
Customers
Customer_addresses
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.

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

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

Part 2- Inserting rows to the system

1. Add a new team to the system

   INSERT INTO teams (id, name, number_of_players, discount)
   VALUES ('T004' , 'Jets' , 10 , 5);

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

   INSERT INTO customers (ctr_number, email, first_name, last_name, phone_number, current_balance, loyalty_card_number)
   VALUES ('c02001' , 'brianrog@hoote ch.com' , 'Brian' , 'Rogers' , ' 01654564898' , '-5' , 'lc4587');

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)
   VALUES ('c02001' , 'brianrog@hootech.com' , 'Brian' , 'Rogers' , '01654564898' , '50' , 'Lc4587');

EXERCISE 2

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;

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.

UPDATE price_history

SET end_date = SYSDATE, end_time = SYSDATE

WHERE itm_number = ‘im01101045’ AND end_date is null;

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

SELECT start_date, TO_CHAR (start_time, ‘HH24:MI:SS’), price, end_date, TO_CHAR (end_time, ‘HH24:MI’)

FROM price_history

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, ‘im011048’);

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

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.

```
DELETE FROM customers_addresses

WHERE id = 'ca0101';
```

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

```
SELECT * FROM customers_addresses
```