

LAB 2

CHAN WEN KANG
A22EC0146

Section 6 Lesson 4 Exercise 1: Data Manipulation Language

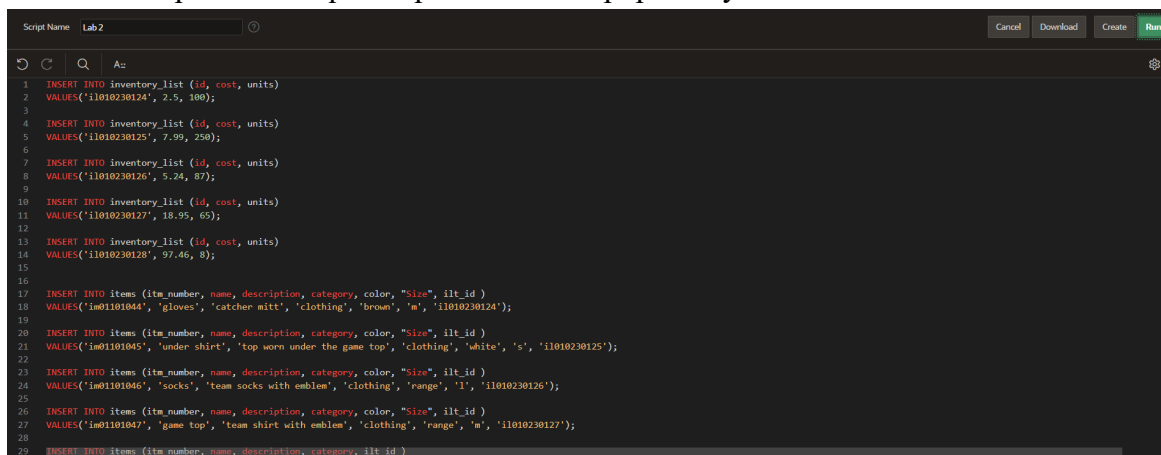
Use DML operations to manage database tables (S6L4 Objective 2)

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.
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



```
Script Name Lab2
Cancel Download Create Run

1 INSERT INTO inventory_list (id, cost, units)
2 VALUES('i1010230124', 2.5, 100);
3
4 INSERT INTO inventory_list (id, cost, units)
5 VALUES('i1010230125', 7.99, 250);
6
7 INSERT INTO inventory_list (id, cost, units)
8 VALUES('i1010230126', 5.24, 87);
9
10 INSERT INTO inventory_list (id, cost, units)
11 VALUES('i1010230127', 18.95, 65);
12
13 INSERT INTO inventory_list (id, cost, units)
14 VALUES('i1010230128', 97.46, 8);
15
16
17 INSERT INTO items (itm_number, name, description, category, color, "Size", ilt_id )
18 VALUES('i10101044', 'gloves', 'catcher mitt', 'clothing', 'brown', 'm', 'i1010230124');
19
20 INSERT INTO items (itm_number, name, description, category, color, "Size", ilt_id )
21 VALUES('i10101045', 'under shirt', 'top worn under the game top', 'clothing', 'white', 's', 'i1010230125');
22
23 INSERT INTO items (itm_number, name, description, category, color, "Size", ilt_id )
24 VALUES('i10101046', 'socks', 'team socks with emblem', 'clothing', 'range', 'l', 'i1010230126');
25
26 INSERT INTO items (itm_number, name, description, category, color, "Size", ilt_id )
27 VALUES('i10101047', 'game top', 'team shirt with emblem', 'clothing', 'range', 'm', 'i1010230127');
28
29 INSERT INTO items (itm_number, name, description, category, ilt_id )
```

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

Number ↑	Elapsed	Statement	Feedback	Rows
1	0.05	INSERT INTO inventory_list (id, cost, units) VALUES('101023	1 row(s) inserted.	1
2	0.00	INSERT INTO inventory_list (id, cost, units) VALUES('101023	1 row(s) inserted.	1
3	0.00	INSERT INTO inventory_list (id, cost, units) VALUES('101023	1 row(s) inserted.	1
4	0.00	INSERT INTO inventory_list (id, cost, units) VALUES('101023	1 row(s) inserted.	1
5	0.00	INSERT INTO inventory_list (id, cost, units) VALUES('101023	1 row(s) inserted.	1
6	0.03	INSERT INTO items (itm_number, name, description, category,	1 row(s) inserted.	1
7	0.00	INSERT INTO items (itm_number, name, description, category,	1 row(s) inserted.	1
8	0.01	INSERT INTO items (itm_number, name, description, category,	1 row(s) inserted.	1
9	0.00	INSERT INTO items (itm_number, name, description, category,	1 row(s) inserted.	1
10	0.00	INSERT INTO items (itm_number, name, description, category,	1 row(s) inserted.	1
11	0.06	INSERT INTO price_history (start_date, start_time, price, it	1 row(s) inserted.	1
12	0.01	INSERT INTO price_history (start_date, start_time, price, en	1 row(s) inserted.	1
13	0.00	INSERT INTO price_history (start_date, start_time, price, en	1 row(s) inserted.	1
14	0.01	INSERT INTO price_history (start_date, start_time, price, it	1 row(s) inserted.	1
15	0.00	INSERT INTO price_history (start_date, start_time, price, it	1 row(s) inserted.	1

Download

row(s) 1 - 15 of 47 [Next](#)

47	47	0
Statements Processed	Successful	With Errors

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

```

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

```

▼

Results Explain Describe Saved SQL History

1 row(s) inserted.

0.01 seconds

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		

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

Results Explain Describe Saved SQL History

ORA-02290: check constraint (WKSP_MKCHAN.MIN_CURRENT_BALANCE) violated

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.

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

Results Explain Describe Saved SQL History

1 row(s) inserted.

Section 6 Lesson 4 Exercise 2: Data Manipulation Language

Use DML operations to manage database tables (S6L4 Objective 2)

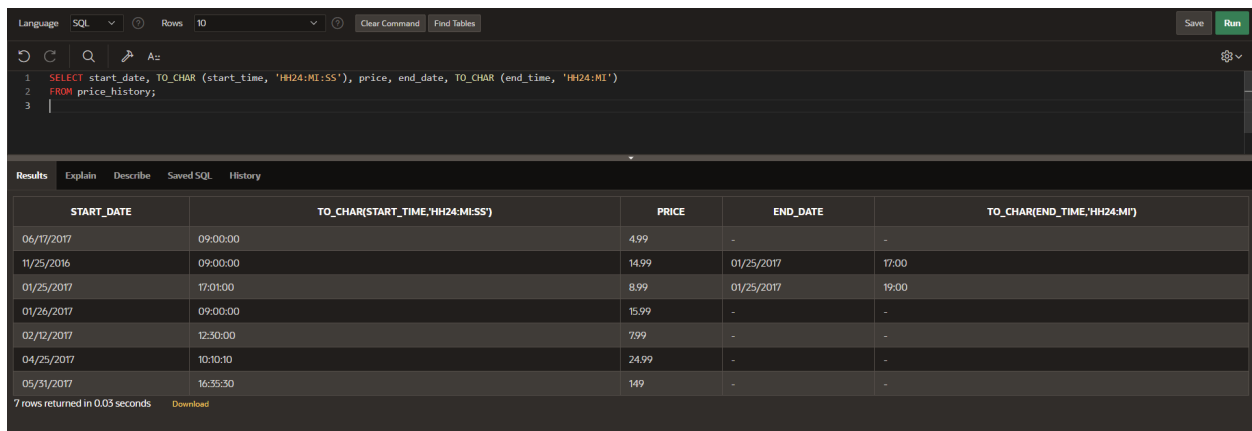
In this exercise you will populate and work with the data that is stored in the database system.

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;
```



The screenshot shows a SQL IDE interface. The top bar includes 'Language SQL', 'Rows 10', and buttons for 'Clear Command' and 'Find Tables'. The main editor contains the following SQL query:

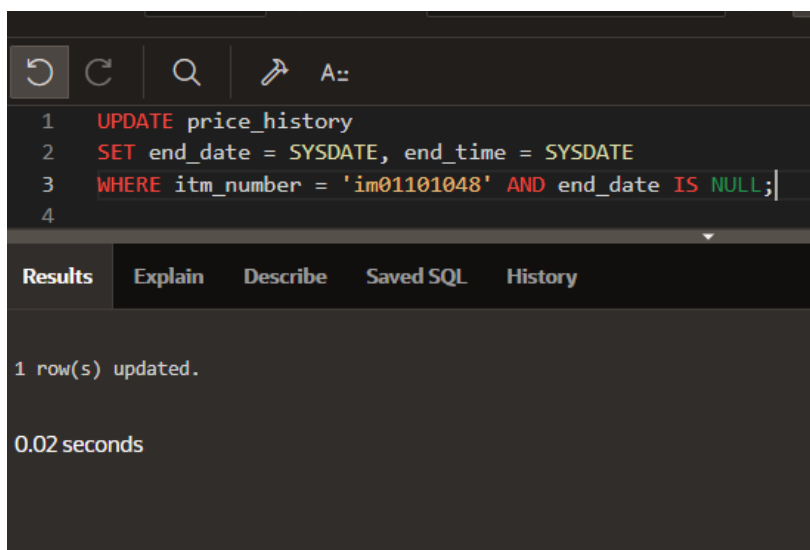
```
1 SELECT start_date, TO_CHAR (start_time, 'HH24:MI:SS'), price, end_date, TO_CHAR  
2 FROM price_history;  
3
```

Below the editor, the 'Results' tab is active, displaying a table with 7 rows. The table has the following columns: START_DATE, TO_CHAR(START_TIME, 'HH24:MI:SS'), PRICE, END_DATE, and TO_CHAR(END_TIME, 'HH24:MI').

START_DATE	TO_CHAR(START_TIME, 'HH24:MI:SS')	PRICE	END_DATE	TO_CHAR(END_TIME, 'HH24:MI')
06/17/2017	09:00:00	4.99	-	-
11/25/2016	09:00:00	14.99	01/25/2017	17:00
01/25/2017	17:01:00	8.99	01/25/2017	19:00
01/26/2017	09:00:00	15.99	-	-
02/12/2017	12:30:00	7.99	-	-
04/25/2017	10:10:10	24.99	-	-
05/31/2017	16:35:30	14.9	-	-

At the bottom of the results pane, it says '7 rows returned in 0.03 seconds' and there is a 'Download' button.

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.



The screenshot shows a SQL IDE interface. The main editor contains the following SQL query:

```
1 UPDATE price_history  
2 SET end_date = SYSDATE, end_time = SYSDATE  
3 WHERE itm_number = 'im01101048' AND end_date IS NULL;  
4
```





Below the editor, the 'Results' tab is active, displaying the message '1 row(s) updated.' and '0.02 seconds'.

3. 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')
06/17/2017	09:00:00	4.99	-	-
11/25/2016	09:00:00	14.99	01/25/2017	17:00
01/25/2017	17:01:00	8.99	01/25/2017	19:00
01/26/2017	09:00:00	15.99	-	-
02/12/2017	12:30:00	7.99	-	-
04/25/2017	10:10:10	24.99	-	-
05/31/2017	16:35:30	149	12/18/2023	09:36

7 rows returned in 0.00 seconds [Download](#)

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.

 A::

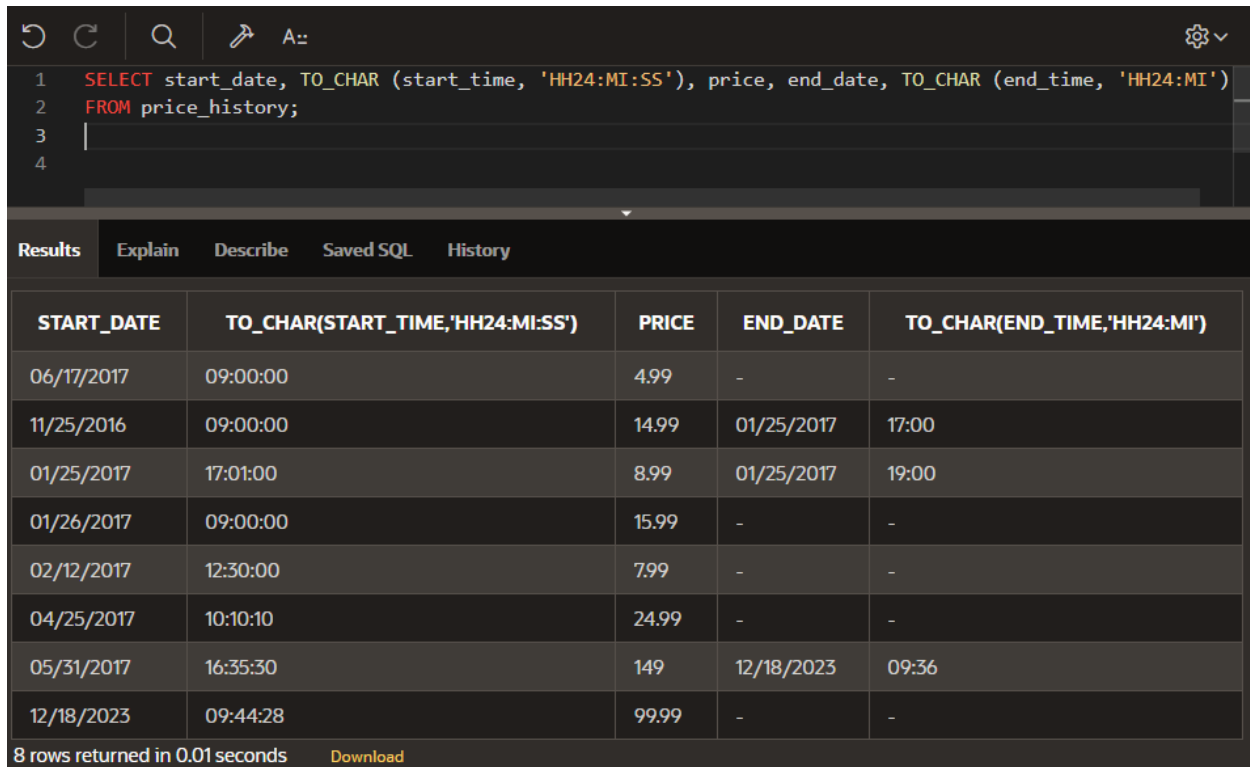
```
1  INSERT INTO price_history(start_date, start_time, price, itm_number)
2  VALUES (SYSDATE, SYSDATE, 99.99, 'im01101048')
3
```

Results [Explain](#) [Describe](#) [Saved SQL](#) [History](#)

1 row(s) inserted.

0.03 seconds

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



The screenshot shows a SQL IDE interface. At the top, there is a toolbar with icons for undo, redo, search, and a settings icon. Below the toolbar, a SQL query is entered in a text area:

```
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
4
```

Below the query editor, there is a tabbed interface with the following tabs: Results, Explain, Describe, Saved SQL, and History. The 'Results' tab is selected, displaying a table of query results:

START_DATE	TO_CHAR(START_TIME,'HH24:MI:SS')	PRICE	END_DATE	TO_CHAR(END_TIME,'HH24:MI')
06/17/2017	09:00:00	4.99	-	-
11/25/2016	09:00:00	14.99	01/25/2017	17:00
01/25/2017	17:01:00	8.99	01/25/2017	19:00
01/26/2017	09:00:00	15.99	-	-
02/12/2017	12:30:00	7.99	-	-
04/25/2017	10:10:10	24.99	-	-
05/31/2017	16:35:30	149	12/18/2023	09:36
12/18/2023	09:44:28	99.99	-	-

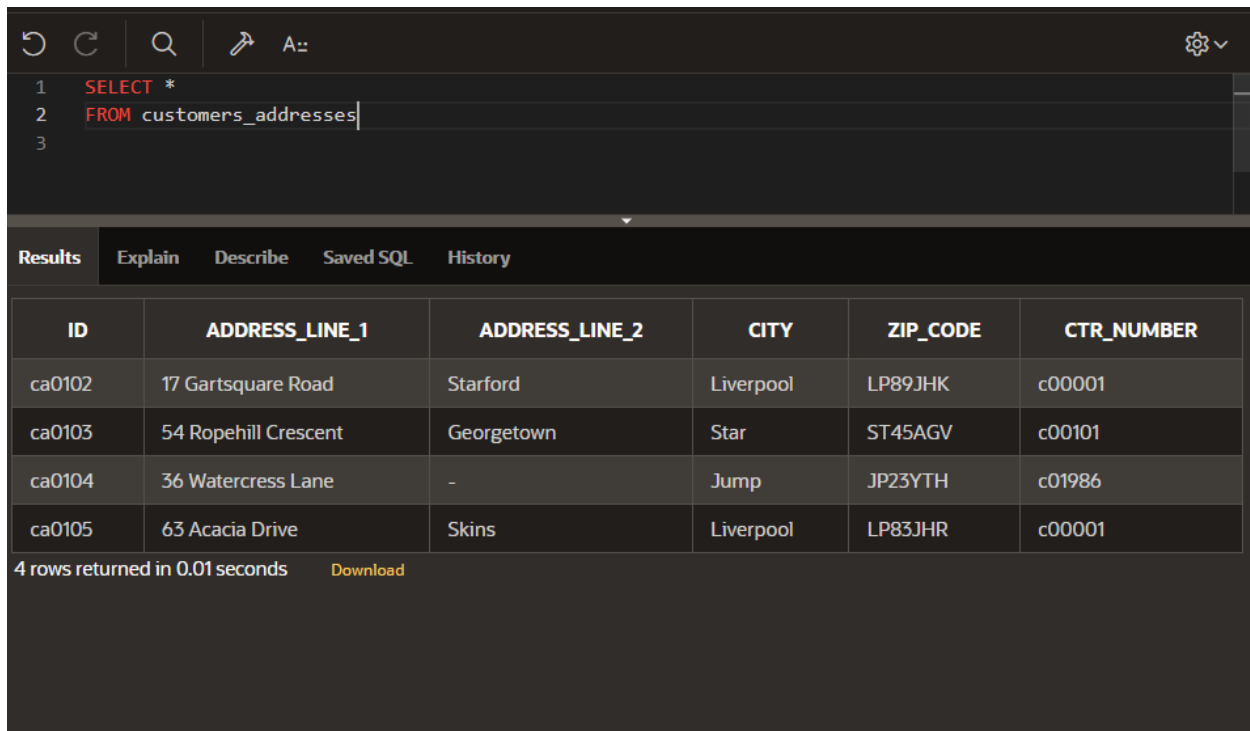
At the bottom of the Results tab, it says "8 rows returned in 0.01 seconds" and there is a "Download" link.

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 address_line_1 = '83 Barrhill Drive';
3
```

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



The screenshot shows a SQL IDE interface. The top toolbar includes icons for undo, redo, search, and a command prompt. The SQL editor contains the following query:

```
1 SELECT *
2 FROM customers_addresses
3
```

Below the editor, the 'Results' tab is active, displaying a table with 4 rows and 6 columns. The columns are labeled ID, ADDRESS_LINE_1, ADDRESS_LINE_2, CITY, ZIP_CODE, and CTR_NUMBER. The data rows are as follows:

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

At the bottom of the results pane, it states '4 rows returned in 0.01 seconds' and provides a 'Download' link.