Faculty of Computing

SECD2523-08 : DATABASE

Lab 2: DML 1

Lecturer: Dr. Noor Hidayah Binti Zakaria

Date:4/12/2023

Prepared by:

Name	Kevin Fachrezy
Matric Number	A21EC4048

Section 6 Lesson 4 Exercise 1: Data Manipulation Language

Part 1 : Running a script to populate the tables. You have to consider the order of the tables when populating them.

1. Use the table mapping document and list the order that you would use to populate the tables.
    a. inventory_list
    b. customers
    c. customers_address
    d. items
    e. orders_items
    f. orders
    g. price_history
    h. sales_representatives
    i. sales_rep_address
    j. teams

```
CREATE TABLE inventory_list (
    id      VARCHAR2(11) NOT NULL,
    cost    NUMBER(7,2) NOT NULL,
    units   NUMBER(4) NOT NULL,
    CONSTRAINT inventory_list_pk PRIMARY KEY ( id )
)


Table created.
```

```
CREATE TABLE items (
    itm_number    VARCHAR2(10) NOT NULL,
    name          VARCHAR2(20) NOT NULL,
    description   VARCHAR2(50) NOT NULL,
    category      VARCHAR2(25) NOT NULL,
    color         VARCHAR2(15),
    "Size"        CHAR(1),
    ilt_id        VARCHAR2(11) NOT NULL,
    CONSTRAINT item_pk PRIMARY KEY ( itm_number )
)


Table created.
```

```
CREATE TABLE price_history (
    start_date   DATE NOT NULL,
    start_time   DATE NOT NULL,
    price        NUMBER(7,2) NOT NULL,
    end_date     DATE,
    end_time     DATE,
    itm_number   VARCHAR2(10) NOT NULL,
    CONSTRAINT price_history_pk PRIMARY KEY ( itm_number, start_date, start_time ),
    CONSTRAINT price_history_items_fk FOREIGN KEY ( itm_number ) REFERENCES items ( itm_number )

)


Table created.
```

```
CREATE TABLE sales_representatives (
    id               VARCHAR2(4) NOT NULL,
    email            VARCHAR2(50) NOT NULL,
    first_name       VARCHAR2(20) NOT NULL,
    last_name        VARCHAR2(30) NOT NULL,
    phone_number     VARCHAR2(11) NOT NULL,
    commission_rate   NUMBER(2) NOT NULL,
    supervisor_id    VARCHAR2(4) NOT NULL,
    CONSTRAINT sales_representative_pk PRIMARY KEY ( id ),
    CONSTRAINT sre_email_uk UNIQUE (email)
)


Table created.
```

```
CREATE TABLE sales_rep_addresses (
    id               VARCHAR2(4) NOT NULL,
    address_line_1   VARCHAR2(30) NOT NULL,
    address_line_2   VARCHAR2(30),
    city             VARCHAR2(15) NOT NULL,
    zip_code         VARCHAR2(7) NOT NULL,
    CONSTRAINT sales_rep_address_pk PRIMARY KEY ( id )
)


Table created.
```

```
CREATE TABLE teams (
    id                  VARCHAR2(4) NOT NULL,
    name                VARCHAR2(20) NOT NULL,
    number_of_players   NUMBER(2) NOT NULL,
    discount            NUMBER(2),
    CONSTRAINT team_pk PRIMARY KEY ( id )
)


Table created.
```

```
CREATE TABLE customers_addresses (
    id               VARCHAR2(8) NOT NULL,
    address_line_1   VARCHAR2(30) NOT NULL,
    address_line_2   VARCHAR2(30),
    city             VARCHAR2(15) NOT NULL,
    zip_code         VARCHAR2(7) NOT NULL,
    ctr_number       VARCHAR2(6) NOT NULL,
    CONSTRAINT customer_address_pk PRIMARY KEY ( id )
)

Table created.
```

```
CREATE TABLE orders (
    id                VARCHAR2(9) NOT NULL,
    odr_date          DATE NOT NULL,
    odr_time          DATE NOT NULL,
    number_of_units   NUMBER(2) NOT NULL,
    ctr_number        VARCHAR2(6) NOT NULL,
    CONSTRAINT orders_pk PRIMARY KEY ( id )
)

Table created.
```

```
CREATE TABLE customers (
    ctr_number            VARCHAR2(6) NOT NULL,
    email                 VARCHAR2(50) NOT NULL,
    first_name            VARCHAR2(20) NOT NULL,
    last_name             VARCHAR2(30) NOT NULL,
    phone_number          VARCHAR2(11) NOT NULL,
    current_balance       NUMBER(6,2) NOT NULL,
    sre_id                VARCHAR2(4),
    tem_id                VARCHAR2(4),
    loyalty_card_number   VARCHAR2(6),
    CONSTRAINT customer_pk PRIMARY KEY ( ctr_number ),
    CONSTRAINT ctr_email_uk UNIQUE (email),
    CONSTRAINT ctr_lcn_uk UNIQUE (loyalty_card_number)
)

Table created.
```

```
CREATE TABLE ordered_items (
    quantity_ordered   NUMBER(3) NOT NULL,
    quantity_shipped   NUMBER(3) NOT NULL,
    itm_number         VARCHAR2(10) NOT NULL,
    odr_id             VARCHAR2(9) NOT NULL,
    CONSTRAINT ordered_item_pk PRIMARY KEY ( itm_number,odr_id )
)


Table created.
```

```
ALTER TABLE customers_addresses ADD CONSTRAINT customer_address_customer_fk FOREIGN KEY ( ctr_number )
    REFERENCES customers ( ctr_number )


Table altered.
```

```
ALTER TABLE customers ADD CONSTRAINT customer_sales_rep_fk FOREIGN KEY ( sre_id )
    REFERENCES sales_representatives ( id )


Table altered.
```

```
ALTER TABLE customers ADD CONSTRAINT customer_team_fk FOREIGN KEY ( tem_id )
    REFERENCES teams ( id )


Table altered.
```

```
ALTER TABLE items ADD CONSTRAINT item_inventory_list_fk FOREIGN KEY ( ilt_id )
    REFERENCES inventory_list ( id )


Table altered.
```

```
ALTER TABLE orders ADD CONSTRAINT order_customer_fk FOREIGN KEY ( ctr_number )
    REFERENCES customers ( ctr_number )

Table altered.
```

```
ALTER TABLE ordered_items ADD CONSTRAINT ordered_item_item_fk FOREIGN KEY ( itm_number )
    REFERENCES items ( itm_number )

Table altered.
```

```
ALTER TABLE ordered_items ADD CONSTRAINT ordered_item_order_fk FOREIGN KEY ( odr_id )
    REFERENCES orders ( id )

Table altered.
```

```
ALTER TABLE sales_rep_addresses ADD CONSTRAINT sales_rep_add_sales_rep_fk FOREIGN KEY ( id )
    REFERENCES sales_representatives ( id )

Table altered.
```

```
ALTER TABLE sales_representatives ADD CONSTRAINT sales_rep_sales_rep_fk FOREIGN KEY ( supervisor_id ) REFERENCES sales_representatives ( id )

Table altered.
```

```
CREATE OR REPLACE TRIGGER fkntm_orders BEFORE
    UPDATE OF ctr_number ON orders
BEGIN
    raise_application_error(
        -20225,
        'Non Transferable FK constraint  on table orders is violated'
    );
END;

Trigger created.
```

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.

   Yes, it matched

3. Run the “sports data.sql” script in APEX to populate your tables 4. Check that no errors occurred when you ran the script.

```
INSERT INTO inventory_list (id, cost, units)
VALUES('i1010230124', 2.5, 100)

1 row(s) inserted.
```

```
INSERT INTO inventory_list (id, cost, units)
VALUES('i1010230125', 7.99, 250)

1 row(s) inserted.
```

```
INSERT INTO inventory_list (id, cost, units)
VALUES('i1010230126', 5.24, 87)

1 row(s) inserted.
```

```
INSERT INTO inventory_list (id, cost, units)
VALUES('i1010230127', 18.95, 65)

1 row(s) inserted.
```

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

```
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', '-5', 'lc4587')
```

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

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.
    a. Information violates

```
1 ALTER TABLE customers
2     ADD CHECK (CURRENT_BALANCE >= 0)
```

```
ORA-02293: cannot validate (SQL_MSVVGSIQPZVLAIXTSUVPURTZB.) - check constraint violated

More Details: https://docs.oracle.com/error-help/db/ora-02293
```

b. Change current balance to 50

```
1 UPDATE customers
2     SET current_balance = 50
3     WHERE ctr_number = 'c02001'
```

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

c. Data updated

```
select
    "CTR_NUMBER",
    "EMAIL",
    "FIRST_NAME",
    "LAST_NAME",
    "PHONE_NUMBER",
    "CURRENT_BALANCE",
    "SRE_ID",
    "TEM_ID",
    "LOYALTY_CARD_NUMBER"
from "CUSTOMERS";
```

CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID	TEM_ID	LOYALTY_CARD_NUMBER
c00001	bob.thornberry@heatmail.com	Robert	Thornberry	01234567898	150	sr01	t001	-
c00012	Jjones@freemail.com	Jennifer	Jones	01505214598	0	-	-	lc1015
c00101	unknown@here.com	John	Doe	03216547808	987.5	sr01	t002	-
c00103	MurciaA@globaltech.com	Andrew	Murcia	07715246890	85	-	-	lc2341
c01986	margal87@delphiview.com	Maria	Galant	01442736589	125.65	sr03	t003	-
c02001	brianroe@hootech.com	Brian	Rogers	01654564898	50	-	-	lc4587

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- Update 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;

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

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	-	-

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;
```

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

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;
```

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	19-DEC-23	04:56
12-FEB-17	12:30:00	7.99	-	-

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.

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

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

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';
```

```
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
```

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