Ashok Enukonda

TASK 1 - DDL, DML, constraints and transaction processing

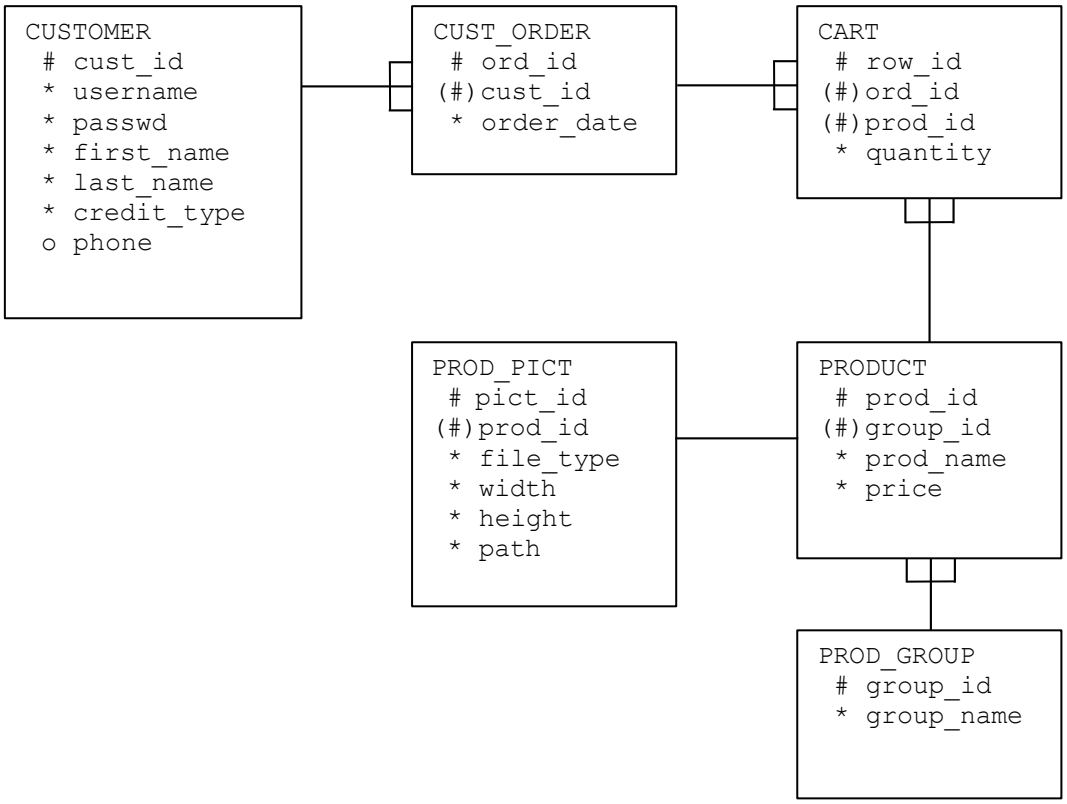
Task 1

Create a sequence object with the name **my_seq**. It should start with 1 and increase by 1. The sequence method NEXTVAL returns a numeric data type.

```
/*
Creating my_seq and other sequences to use as different Primary key,
    because using 1 sequence for multiple tables leads to inconsistency in
Keys.
*/
DROP SEQUENCE IF EXISTS my_seq;
CREATE SEQUENCE my_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for CUSTOMER table (cust_id)
DROP SEQUENCE IF EXISTS cust_seq;
CREATE SEQUENCE cust_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for CUST_ORDER table (order_id)
DROP SEQUENCE IF EXISTS order_seq;
CREATE SEQUENCE order_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for CART table (row_id)
DROP SEQUENCE IF EXISTS cart_seq;
CREATE SEQUENCE cart_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for PRODUCT table (product_id)
DROP SEQUENCE IF EXISTS prod_seq;
```

```
CREATE SEQUENCE prod_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for PROD_PICT table (pict_id)
DROP SEQUENCE IF EXISTS pict_seq;
CREATE SEQUENCE pict_seq
START WITH 1
INCREMENT BY 1;

-- Sequence for PROD_GROUP table (group_id)
DROP SEQUENCE IF EXISTS group_seq;
CREATE SEQUENCE group_seq
START WITH 1
INCREMENT BY 1;
```

Task 2
Create a table structure according to the drawing below:



Explanation of notation

```
 #  = Primary key
(#) = Foreign key
 *  = Mandatory (must contain a value => NOT NULL)
 o  = Optional (must not contain a value can be NULL)
```

customer.credit_type CHECK ('high','average','low')
prod_pict.file_type CHECK ('gif','jpg')
cust_order.ord_id (generated by the sequence my_seq)
cart.row_id (generated by the sequence my_seq)
cust_order.order_date (data type = DATE, DEFAULT SYSDATE)
customer.username (should be unique, constraint UNIQUE)
All Foreign Key columns should have the column constraint **NOT NULL**

Name all constraints except NOT NULL. Suggestion for a constraint naming convention: **table_column_constraint**, you can use the following abbreviations if you

like : **CK** = CHECK, **PK** = PRIMARY KEY, **FK** = FOREIGN KEY and finally **UQ** = UNIQUE, or whatever you like as long as you are consistently.

For the customer table above, a primary key constraint would be named: **customer_cust_id_pk**

```
-- Customer Table (Primary Key : cust_id)
DROP TABLE IF EXISTS customer CASCADE CONSTRAINTS;
CREATE TABLE customer (
cust_id       NUMBER(10)       NOT NULL,
username      VARCHAR2(50)     NOT NULL,
passwd        VARCHAR2(50)     NOT NULL,
first_name    VARCHAR2(50)     NOT NULL,
last_name     VARCHAR2(50)     NOT NULL,
credit_type   VARCHAR2(10)     NOT NULL,
phone         VARCHAR2(20)     NULL,

CONSTRAINT customer_cust_id_pk PRIMARY KEY (cust_id),
CONSTRAINT customer_username_uq UNIQUE (username),
CONSTRAINT customer_credit_type_ck CHECK (credit_type IN ('high', 'average',
'low'))
);

-- Customer Order Table (Primary Key : order_id)
DROP TABLE IF EXISTS cust_order CASCADE CONSTRAINTS;
CREATE TABLE cust_order (
    order_id       NUMBER(10)      NOT NULL,
    cust_id       NUMBER(10)      NOT NULL,
    order_date    DATE            DEFAULT SYSDATE NOT NULL,

    CONSTRAINT cust_order_order_id_pk PRIMARY KEY (order_id),
    CONSTRAINT cust_order_cust_id_fk FOREIGN KEY (cust_id) REFERENCES
customer(cust_id)
);

-- Product Group Table (Primary Key : group_id)
DROP TABLE IF EXISTS prod_group CASCADE CONSTRAINTS;
CREATE TABLE prod_group (
    group_id     NUMBER(10)       NOT NULL,
    group_name   VARCHAR2(100)    NOT NULL,

    CONSTRAINT prod_group_group_id_pk PRIMARY KEY (group_id)
);

-- Product Table (Primary Key : product_id)
DROP TABLE IF EXISTS "product" CASCADE CONSTRAINTS;
CREATE TABLE "product" (
    product_id     NUMBER(10)       NOT NULL,
    group_id    NUMBER(10)       NOT NULL,
    prod_name  VARCHAR2(100)    NOT NULL,
    price       NUMBER(10, 2)    NOT NULL,

    CONSTRAINT product_product_id_pk PRIMARY KEY (product_id),
    CONSTRAINT product_group_id_fk FOREIGN KEY (group_id) REFERENCES
prod_group(group_id)
);

-- Product Picture Table (Primary Key : pict_id)
DROP TABLE IF EXISTS prod_pict CASCADE CONSTRAINTS;
CREATE TABLE prod_pict (
```

```
    pict_id     NUMBER(10)      NOT NULL,
    product_id     NUMBER(10)      NOT NULL,
    file_type  VARCHAR2(10)     NOT NULL,
    width       NUMBER(10)      NOT NULL,
    height      NUMBER(10)      NOT NULL,
    path        VARCHAR2(255)  NOT NULL,

    CONSTRAINT prod_pict_pict_id_pk PRIMARY KEY (pict_id),
    CONSTRAINT prod_pict_product_id_fk FOREIGN KEY (product_id) REFERENCES
"product"(product_id),
    CONSTRAINT prod_pict_file_type_ck CHECK (file_type IN ('gif', 'jpg'))
);


-- Cart Table (Primary Key : pict_id)
DROP TABLE IF EXISTS cart CASCADE CONSTRAINTS;
CREATE TABLE cart (
    row_id      NUMBER(10)      NOT NULL,
    order_id      NUMBER(10)      NOT NULL,
    product_id     NUMBER(10)      NOT NULL,
    quantity    NUMBER(10)      NOT NULL,

    CONSTRAINT cart_row_id_pk PRIMARY KEY (row_id),
    CONSTRAINT cart_order_id_fk FOREIGN KEY (order_id) REFERENCES
cust_order(order_id),
    CONSTRAINT cart_product_id_fk FOREIGN KEY (product_id) REFERENCES
"product"(product_id)
);
```

Task 3

Insert three rows in the **customer** table.

```
INSERT INTO CUSTOMER (cust_id, username, passwd, first_name, last_name,
credit_type)
VALUES (cust_seq.NEXTVAL, 'rash', 'password1', 'Rashmi', 'Kumari', 'high');

INSERT INTO CUSTOMER (cust_id, username, passwd, first_name, last_name,
credit_type)
VALUES (cust_seq.NEXTVAL, 'ask', 'password2', 'Ashok', 'LN2', 'average');

INSERT INTO CUSTOMER (cust_id, username, passwd, first_name, last_name,
credit_type)
VALUES (cust_seq.NEXTVAL, 'tst', 'password3', 'Test', 'Test', 'low');
```

Task 4

Insert two rows in the **prod_group** table.

```
INSERT INTO prod_group (group_id, group_name)
VALUES (group_seq.NEXTVAL, 'electronics');

INSERT INTO prod_group (group_id, group_name)
VALUES (group_seq.NEXTVAL, 'kitchen');
```

Task 5
Insert two rows in the **product** table.

```
INSERT INTO "product" (product_id, group_id, prod_name, price)
VALUES (prod_seq.NEXTVAL, 1, 'Smartphone Model X', 799.99);
INSERT INTO "product" (product_id, group_id, prod_name, price)
VALUES (prod_seq.NEXTVAL, 2, 'Spoon', 199.50);
```

Task 6
Perform a sale by creating **one row** in the **cust_order** table and **two rows** in the **cart** table. **Remember** to use the sequence to generate primary key in the tables.

```
DECLARE
    -- Variable to temporarily store the generated order ID
    v_order_id NUMBER;
    -- Defining customer ID to use for the order
    v_customer_id NUMBER := 1;
BEGIN
    -- 1. Creating 1 row in the CUST_ORDER table
    INSERT INTO CUST_ORDER (order_id, cust_id, order_date)
    VALUES (order_seq.NEXTVAL, v_customer_id, SYSDATE)
    RETURNING order_id INTO v_order_id;

    -- 2. Creating 2 rows in the CART table
    INSERT INTO CART (row_id, order_id, product_id, quantity)
    VALUES (cart_seq.NEXTVAL, v_order_id, 1, 1);

    INSERT INTO CART (row_id, order_id, product_id, quantity)
    VALUES (cart_seq.NEXTVAL, v_order_id, 2, 2);

    COMMIT;

    DBMS_OUTPUT.PUT_LINE('Sale completed. New Order ID: ' || v_order_id);

EXCEPTION
    WHEN OTHERS THEN
        ROLLBACK;
        DBMS_OUTPUT.PUT_LINE('An error occurred. Transaction rolled back.');
        RAISE;
END;
```

NOTE that when you have created the cust_order you must check what value the sequence put in the ord_id column (i.e. the Primary Key value). Then take that number and use it in the insert on the cart table FK-column. **DO NOT USE** the sequence to generate a number to the foreign key ord_id in the cart table!

Task 7
Increase the price on all articles by 12%.

```
UPDATE "product"
SET price = price * 1.12;
COMMIT;
```

Task 8
Update the phone number for an optional customer.

```
UPDATE CUSTOMER
SET phone = '555-9012'
WHERE cust_id = 2;
COMMIT;

/*
```

Task 9
Delete all rows from the cust_order table, by using DML. **What happens and why!**

```
DELETE FROM CUST_ORDER;


/*
The DELETE FROM CUST_ORDER statement fails because the table CART contains
rows
that reference the same order_id through a foreign key.

Oracle does not allow deleting a parent row while child rows still exist,
since
this would break referential integrity. In this case, the CART table still has
items linked to the orders in CUST_ORDER, so the database prevents the delete
operation and returns ORA-02292(Error)(child record found).
*/
```