QM-7093-01 ENTERPRISE DATA SYSTEMS
CASE STUDY (CS-3) – NOAH L. SCHRICK - 1492657

Instructor: Dr. Ismail Abdulrashid,

Instructions:

In this case study, **Cape_Codd** database needs to be used. You can find the Review Questions in the textbook: Exercises 2.40- 2.60

- Use both the INVENTORY and WAREHOUSE tables to answer Questions 2.40 through 2.55.
- Use both the CATALOG_SKU_2016 and CATALOG_SKU_2017 tables to answer Questions 2.56 through 2.60.
- Submit thru Harvey drop box
- Deliverable: You are expected to submit
    - A single SQL script file (.sql) prepared and saved in SQL Server Management Studio that includes your SQL statements that answer each of the questions in order.
    - This word file that you copied all of your SQL script (no result tables) from your SQL file.
- Each query should start with a comment line that looks like the following (last character corresponding to question number):
    - /* *** CS3-2.40*** */
    - This line should follow the SQL statement that is your answer to the particular question (e.g. 41, 42, 43, ...)
- You should include at least one line of white space between your answer SQL statements
- Do not include the result table unless it is said so!
- Check Harvey for the due date!

Your answer should look like this:

```
/* Your Names-Group Name */
/* *** CS1-2.17 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;

/* *** CS1-2.18 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;

/* *** CS1-2.19 *** */
SELECT      SKU, SKU_Description
FROM        INVENTORY;
```

Please write your solution below:

/* Noah L. Schrick */

```
/* *** CS3-2.40 *** */
SELECT i.SKU, i.SKU_Description, w.WarehouseID, w.WarehouseCity, w.WarehouseState
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                        AND
                                (w.WarehouseCity = 'Atlanta') OR
                                (w.WarehouseCity = 'Bangor') OR
                                (w.WarehouseCity = 'Chicago')
;

/* *** CS3-2.41 *** */
SELECT i.SKU, i.SKU_Description, w.WarehouseID, w.WarehouseCity, w.WarehouseState
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                        AND w.WarehouseCity IN  ('Atlanta', 'Bangor', 'Chicago')
;
```

```
/* *** CS3-2.42 *** */
SELECT i.SKU, i.SKU_Description, w.WarehouseID, w.WarehouseCity, w.WarehouseState
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                        AND
                                NOT (w.WarehouseCity = 'Atlanta') AND
                                NOT (w.WarehouseCity = 'Bangor') AND
                                NOT (w.WarehouseCity = 'Chicago')
;

/* *** CS3-2.43 *** */
SELECT i.SKU, i.SKU_Description, w.WarehouseID, w.WarehouseCity, w.WarehouseState
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                        AND w.WarehouseCity NOT IN  ('Atlanta', 'Bangor', 'Chicago')
;

/* *** CS3-2.44 *** */
SELECT (RTRIM(i.SKU_Description) + ' is located in ' + LTRIM(w.WarehouseCity)) AS
ItemLocation
        FROM INVENTORY i, WAREHOUSE w
;
```

```
/* *** CS3-2.45 *** */
SELECT SKU, SKU_Description, WarehouseID
        FROM INVENTORY
                WHERE WarehouseID IN
                        (SELECT WarehouseID
                                FROM WAREHOUSE
                                        WHERE Manager = 'Lucille Smith')
;

/* *** CS3-2.46 *** */
SELECT i.SKU, i.SKU_Description, w.WarehouseID
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                  AND w.Manager = 'Lucille Smith'
;

/* *** CS3-2.47 *** */
SELECT SKU, SKU_Description, w.WarehouseID
        FROM INVENTORY i
          JOIN WAREHOUSE w ON i.WarehouseID = w.WarehouseID
                WHERE Manager = 'Lucille Smith'
;
```

```
/* *** CS3-2.48 *** */
SELECT WarehouseID, AVG(QuantityOnHand) AS AVGQuantityOnHand
        FROM INVENTORY
                WHERE WarehouseID IN
                (SELECT WarehouseID
                        FROM WAREHOUSE
                                WHERE Manager = 'Lucille Smith'
                )
                GROUP BY WarehouseID
;

/* *** CS3-2.49 *** */
SELECT i.WarehouseID, AVG(i.QuantityOnHand) AS AVGQuantityOnHand
        FROM INVENTORY i, WAREHOUSE w
                WHERE i.WarehouseID = w.WarehouseID
                  AND w.Manager = 'Lucille Smith'
                        GROUP BY i.WarehouseID
;

/* *** CS3-2.50 *** */
SELECT i.WarehouseID, AVG(QuantityOnHand) AS AVGQuantityOnHand
        FROM INVENTORY i
          JOIN WAREHOUSE w ON i.WarehouseID = w.WarehouseID
```

```
                    WHERE Manager = 'Lucille Smith'
                        GROUP BY i.WarehouseID
;

/* *** CS3-2.51 *** */
SELECT w.WarehouseID, WarehouseCity, WarehouseState, Manager, SKU,
SKU_Description, QuantityOnHand
        FROM WAREHOUSE w
         JOIN INVENTORY i ON w.WarehouseID = i.WarehouseID
                WHERE Manager = 'Lucille Smith'
;

/* *** CS3-2.52 *** */
SELECT WarehouseID, SUM(QuantityOnOrder) AS TotalItemsOnOrder,
SUM(QuantityOnHand) AS TotalItemsOnHand
        FROM INVENTORY
                GROUP BY WarehouseID, QuantityOnOrder
;

/* *** CS3-2.53 *** */
-- All desired results are in the same table, however there are aggregate functions
involved.
-- SQL does not allow performing an aggregate function on expressions containing a
subquery.
-- Likewise, subqueries are not allowed in aggregate functions.

/* *** CS3-2.54 *** */
```

```
-- Subqueries only retrieve items from the top table, not any other table.
-- Subqueries return results "up", but do not combine or merge any tables.
-- Joins combine tables, and all columns are accessible since the tables are joined together.



/* *** CS3-2.55 *** */
SELECT *
        FROM WAREHOUSE w
         LEFT JOIN INVENTORY i ON w.WarehouseID = i.WarehouseID
;

/* *** CS3-2.56 *** */
SELECT SKU, SKU_Description, Department
        FROM CATALOG_SKU_2016
        UNION
         SELECT SKU, SKU_Description, Department
                FROM CATALOG_SKU_2017
;

/* *** CS3-2.57 *** */
SELECT SKU, SKU_Description, Department
        FROM CATALOG_SKU_2016
                WHERE CatalogPage IS NOT NULL
        UNION
         SELECT SKU, SKU_Description, Department
                FROM CATALOG_SKU_2017
```

```
                    WHERE CatalogPage IS NOT NULL
;

/* *** CS3-2.58 *** */
SELECT SKU, SKU_Description, Department
        FROM CATALOG_SKU_2016
        INTERSECT
          SELECT SKU, SKU_Description, Department
                FROM CATALOG_SKU_2017
;

/* *** CS3-2.59 *** */
SELECT SKU, SKU_Description, Department
        FROM CATALOG_SKU_2016
                WHERE CatalogPage IS NOT NULL
        INTERSECT
          SELECT SKU, SKU_Description, Department
                FROM CATALOG_SKU_2017
                        WHERE CatalogPage IS NOT NULL
;
```

```
/* *** CS3-2.60 *** */
SELECT SKU, SKU_Description, Department
        FROM CATALOG_SKU_2016
        EXCEPT
         SELECT SKU, SKU_Description, Department
                FROM CATALOG_SKU_2017
;

/* *** CS3-2.61 *** */
SELECT d.Buyer, d.SKU
        FROM SKU_DATA d
                WHERE d.Buyer IN
                        (SELECT t.Buyer
                                FROM SKU_DATA t
                                        WHERE d.Buyer = t.Buyer
                                         AND d.SKU <> t.SKU
                        )
;
-- Shows that changing the primary key to Buyer is not justifiable: Buyer is non-unique.
```