DSC301 Homework 7
Quinlan Spring 2021

Directions: Use subqueries and joins. Deciding which columns you will need for each query will be your choice. For example, in questions that ask to return customers, I used customer ID and a concatenation of their first and last names. Of course, if the question involves location, you will need to use the `zip` as well. A basic syntax template that you can use for inner joins is provided below. A `WHERE` clause that specifies a condition may sometimes be needed and follows the `ON`.

```
SELECT
        Table1.column1, Table2.column2
FROM
        Table1 INNER JOIN Table2
ON
        Table1.somecol = Table2.samecol;
```

---

1. Get a (distinct) list of all products (i.e., just the product_id) that have been purchased. Hint: no subquery or join needed. You will use your answer to this question in the next problem.

   Solution:

```
SELECT DISTINCT
    (OrderDetails.product)
FROM
    OrderDetails;
```

2. Get a (distinct) list of all the <u>names</u> of products that have been purchased. Hint: use a subquery.

   Solution:

```
SELECT
    name
FROM
    Products
WHERE
    Products.product_id IN (SELECT DISTINCT
            (OrderDetails.product)
FROM
    OrderDetails);
```

3. Repeat Problem 2. using an inner join instead. This serves as evidence, and a reminder, that many subqueries can be written as joins and many joins can be written as subqueries. However, `JOIN`s are typically preferred.

Solution:

```
SELECT DISTINCT
    (P.name)
FROM
    Products AS P
        INNER JOIN
    OrderDetails AS D ON P.product_id = D.product;
```

4. List all orders (i.e., the order number) and the associated products (i.e., product name) purchased. Note: if asked for order number and product number (instead of product name), it would be unnecessary to join tables since the product id is contained within the order details table. Hint: Join `OrderDetails` and `Products` on product id number.

   Solution:

```
SELECT
    OrderDetails.order, Products.product_id, Products.name
FROM
    OrderDetails
        INNER JOIN
    Products ON OrderDetails.product = Products.product_id
ORDER BY OrderDetails.order;
```

5. Determine the total of the order, the most expensive product purchased, the least expensive product purchased, and the number of items purchased in order 1. Use MSRP in these calculations. Hints: use an inner join and four aggregate functions.

   Solution:

```
SELECT
    SUM(P.msrp * D.qty),
    MAX(P.msrp),
    MIN(P.msrp),
    COUNT(D.`order`)
FROM
    Products AS P
        INNER JOIN
    OrderDetails AS D ON P.product_id = D.product
WHERE
    D.`order` = @orderID;
```

6. Determine (list) the the average item's MSRP from each order. Sort largest to smallest. Hint: use an inner join and an aggregate function.

   Solution:

```
SELECT
    OrderDetails.`order`,
    ROUND(AVG(Products.msrp), 2) AS 'Average Price'
FROM
```

```
    Products
        INNER JOIN
    OrderDetails ON Products.product_id = OrderDetails.product
GROUP BY OrderDetails.`order`;
```

7. Use inner joins to yield the following results:

   (a). product name and manufacture name

   (b). product name and category name

   (c). product manufacturer and category name

   Solution:

   ```
   SELECT
       P.product_id, P.name, M.manufacturer, C.category
   FROM
       Manufacturers AS M
           INNER JOIN
       Products AS P ON M.manufacturer_id = P.manufacturer_id
           INNER JOIN
       Categories AS C ON P.category_id = C.category_id;
   ```

8. List order information (order number, product and quantity) and customer information (name, phone, and zip) of customers with zip codes from 4xxxxx (i.e., from any zip code that begins with a 4).

   Solution:

```
SELECT
    O.id,
    D.product,
    D.qty,
    CONCAT(C.fname, ' ', C.lname) AS Customer,
    C.phone,
    C.zip
FROM
    OrderDetails AS D
        INNER JOIN
    Orders AS O ON D.order = O.id
        INNER JOIN
    Customers AS C ON O.customer = C.id
WHERE
    C.zip between 40000 and 49999
ORDER BY O.id;
```

9. Use a left outer join to get all customers that have not placed an order. Use the `Customer.id` and `Order.customer` fields to join `ON`.

   Solution:

```
SELECT
    C.id, CONCAT(C.fname, ' ', lname) AS CustName
FROM
    Customers C
        LEFT OUTER JOIN
    Orders O ON C.id = O.customer
WHERE
    O.id IS NULL;
```

10. List order information of all customers from West Virginia. Hint: join four tables and use a subquery or (inner) join five tables using no subqueries.

    Solution:

```
SELECT
    D.order AS 'OrderID',
    C.id AS 'Customer Number',
    CONCAT(C.fname, ' ', lname) AS 'Customer',
    P.product_id AS 'Product Number',
    D.qty as 'Qty',
    P.name AS 'Product'
FROM
    Customers C
        INNER JOIN
    Orders O ON C.id = O.customer
        INNER JOIN
    OrderDetails D ON O.id = D.order
        INNER JOIN
    Products P ON D.product = P.product_id
WHERE
    C.zip IN (SELECT
            Z.zip
        FROM
            Zipcodes Z
        WHERE
            Z.state = 'WV');

# OR no subquery

SELECT
    D.order AS 'OrderID',
    C.id AS 'Customer Number',
    CONCAT(C.fname, ' ', lname) AS 'Customer',
    P.product_id AS 'Product Number',
    D.qty AS 'Qty',
    P.name AS 'Product'
FROM
    Zipcodes Z
        INNER JOIN
    Customers C ON Z.zip = C.zip
        INNER JOIN
    Orders O ON C.id = O.customer
        INNER JOIN
```

```sql
    OrderDetails D ON O.id = D.order
        INNER JOIN
    Products P ON D.product = P.product_id
WHERE
    Z.state = 'WV';
```