DSC301 Homework 9
Quinlan Spring 2021

Directions: Create stored queries and procedures.

---

1. The store manager suspects that someone in customer service is harvesting customer emails and selling them on the dark web. The manager tells you their suspicion. What do you suggest? Write some SQL to handle this situation, but that still permits agents to view the other customer information, including name and address.

   Solution:

```
CREATE VIEW CustNoEmail AS
    SELECT id, fname, lname, address1, address2, zip FROM Customers;
```

2. Create a view containing order information, in particular: date, customer name (first and last), address, phone number, email, product id and name, quantity, and MSRP. Hint: you will need to join tables.

   Solution:

```
USE dbsoln_Store;
CREATE OR REPLACE VIEW OrderInfo AS
SELECT
    O.id,
    O.date,
    C.fname,
    C.lname,
    C.address1,
    C.phone,
    C.email,
    P.product_id,
    P.name,
    D.qty,
    P.msrp
FROM
    Products P
        INNER JOIN
    OrderDetails D ON P.product_id = D.product
        INNER JOIN
    Orders O ON D.order = O.id
        INNER JOIN
    Customers C ON O.customer = C.id;
```

3. Write a statement that creates a summarized view of the average of products in each order. Include the order id.

   Solution:

```
CREATE VIEW AvgOfProducts AS
SELECT
    D.order, ROUND(AVG(P.msrp * D.qty), 2) AS ProductAvg
FROM
    Products P
        INNER JOIN
    OrderDetails D ON P.product_id = D.product
GROUP BY D.order;
```

4. Create a summarized view containing the average of products in each order, but base it on the view created in Questions 2.

   Solution:

```
CREATE VIEW AvgOfProds AS
SELECT
    ROUND(AVG(P.msrp * D.qty), 2) AS ProductAvg
FROM
    OrderInfo;
```

5. Create a view of all customers from West Virginia.

   Solution:

```
CREATE VIEW WVCustomers AS
SELECT
    id, fname, lname, phone, email
FROM
    Customers AS C
        INNER JOIN
    Zipcodes AS Z ON C.zip = Z.zip
WHERE
    Z.state = 'WV';
```

6. Create a view, `WVOrders`, with order details from West Virginia using the view created in Problem 5. Include order id, customer id, product id, and quantity ordered.

   Solution:

```
CREATE VIEW WVOrders AS
SELECT
    O.id, O.customer, D.product, D.qty
FROM
    OrderDetails D
        INNER JOIN
    Orders O ON D.order = O.id
        INNER JOIN
    WVCustomers W ON O.customer = W.id;
```

7. Create a stored procedure that returns order information (i.e., date, customer name, address, phone number, email, product id and name, quantity, and MSRP) for a given order number. Call your procedure on Order 1. Hint: See Problem 2.

   Solution:

```
DELIMITER $$
USE `dbsoln_Store`$$
CREATE PROCEDURE `orderInfo`
(
       IN oid   int
 )
   BEGIN

    SELECT * from OrderInfo O

WHERE
   O.id = oid;
END$$

DELIMITER ;
CALL orderInfo(1);
```

8. Update `Customers` table by setting `address2` to `NULL` if it is empty. Then write one (or more) stored procedure that accepts an order number (i.e., order id) and does the following.

   (a). Return a shipping label with customer information in the following format:

```
Joe Smith
123 Park Way
City, State Zip
```

   If there is a second address, that should be included as well. In particular,

```
Joe Smith
123 Park Way
Apt A.
City, State Zip
```

   (b). Return the order total including 8% sales tax and a $10.00 flat rate shipping fee.

   (c). Update the inventory levels by subtracting each product's quantity from the `Products` table.

   Solution:

```
# Set Address 2 to NULL if isempty
# May have to set safe_mode OFF
UPDATE dbsoln_Store.Customers
SET
    address2 = NULL;
```

```
SET @oid = 2;

# Shipping Label
SELECT
    CONCAT_WS(CHAR(13),
            CONCAT_WS(' ', fname, lname),
            CONCAT(address1,
                    IF(address2 IS NOT NULL OR address2 <> '',
                        CONCAT(CHAR(13), address2),
                        ' ')),
            CONCAT_WS(' ',
                    CONCAT_WS(', ', Z.city, Z.state),
                    C.zip)) AS 'Shipping Label'
FROM
    Orders O
        INNER JOIN
    Customers C ON O.customer = C.id
        INNER JOIN
    Zipcodes Z ON C.zip = Z.zip
WHERE
    O.id = @oid;


# Order Total
SELECT
    ROUND(1.08 * SUM(P.msrp * D.qty) + 10.00, 2) AS 'Order Total'
FROM
    Products P
        INNER JOIN
    OrderDetails D ON P.product_id = D.product
WHERE
    D.order = @oid;

# Update Inventory Levels
UPDATE Products
SET
    inventory = (SELECT
            qty
        FROM
            OrderDetails D
        WHERE
            D.`order` = oid);
```

9. Create, then call, a procedure that accepts two integers as input parameters and returns the sum and product.

   Solution:

```
DELIMITER //
DROP PROCEDURE IF EXISTS calculate //
CREATE PROCEDURE calculate
(
    IN x INT,
    IN y INT,
    OUT sum INT,
    OUT product INT
)
BEGIN
  SET sum = x + y;
  SET product = x * y;
END//

DELIMITER ;

# Execute stored procedure
CALL calculate(4,5,@s,@p);

# Display results
SELECT @s, @p;
```

10. Create indexes on the first and last name fields of the `Customers` table.

    Solution:

    ```
    CREATE INDEX idx_customer_name ON Customers (lname, fname);
    ```