Awarding Sales Bonuses Using SQL

The Business Question and Proposed Solution

- The salespeople at Northwind Traders with the top 5 sales amounts will be determined by using SQL and given bonuses.
- Data from multiple tables will be required to solve the business problem.

Task #1: Joining Tables Together in SQL to Obtain Data for Analysis

Three tables (orders, employees, and orderdetails) were joined to connect employees with sales.

```
SELECT lastname, firstname, orders.orderid, products.productID, quantity, price

    FROM employees

    INNER JOIN orders

    ON employees.employeeid = orders.employeeid

    INNER JOIN orderdetails

    ON orders.orderid = orderdetails.orderid

    INNER JOIN products

    ON orderdetails.productid = products.productid

    ORDER BY lastname, firstname;
```

Task #2: Calculate and Summarize Sales for each Order

- New, temporary fields were created for the result of a calculation in SQL.

```
SELECT lastname, firstname, orders.orderid, products.productID, quantity, price, quantity * price AS
salesamount

    FROM employees

    INNER JOIN orders

    ON employees.employeeid = orders.employeeid

    INNER JOIN orderdetails

    ON orders.orderid = orderdetails.orderid

    INNER JOIN products

    ON orderdetails.productid = products.productid

    ORDER BY lastname, firstname;
```

Awarding Sales Bonuses Using SQL

Task #3: Aggregate and group sales orders from highest to lowest

- Data was aggregated and grouped to make it more useful for decision making.
- In SQL, the SUM() function, together with the GROUP BY clause, was used to aggregate data.

```
SELECT lastname, firstname, orders.orderid, products.productID, quantity, price, SUM(quantity *
price) AS salesamount
        FROM employees
        INNER JOIN orders
        ON employees.employeeid = orders.employeeid
        INNER JOIN  orderdetails
        ON orders.orderid = orderdetails.orderid
        INNER JOIN products
        ON orderdetails.productid = products.productid
        GROUP BY orders.orderid
```

Task #4: Find the Solution

- Since we only needed the top 5 employee sales amounts, limiting the number of rows that displayed as the result of the query was accomplished using the LIMIT command.

```
SELECT lastname, firstname, orders.orderid, products.productID, quantity, price, SUM(quantity *
price) AS salesamount
        FROM employees
        INNER JOIN orders
        ON employees.employeeid = orders.employeeid
        INNER JOIN orderdetails
        ON orders.orderid = orderdetails.orderid
        INNER JOIN products
        ON orderdetails.productid = products.productid
        GROUP BY orders.orderid
        ORDER BY salesamount DESC
        LIMIT 5
```

Awarding Sales Bonuses Using SQL

- Since there was one salesperson whose two sales made the top 5, it would then be necessary to discuss what the scenario is for the bonuses. Does the manager want to award more than one bonus to that salesperson? Or do we need to go deeper to see what other salespeople had the highest sales? The HAVING command applies a filter after aggregation, anticipating the alternative type of business problem.

Code ADDED:

```
HAVING orders.orderid IN (10372, 10424, 10417, 10324, 10351) –These were the top 5 sales orders

ORDER BY salesamount DESC
```

Task #5: Additional Information Requested

The manager then wanted to know which customers had the most orders so that they would be awarded incentives as well.

Additional code used to find number of orders per customer using INNER JOIN:

```
SELECT customername, COUNT(orders.orderid) AS numberoforders

        FROM customers

        INNER JOIN orders

        ON customers.customerid = orders.customerid

        GROUP BY orders.customerid

        ORDER BY COUNT(orders.customerid) DESC
```