AdventureWorks Sales Analysis Using SQL

Author: Simranjit Kaur\ **Tools Used:** SQLite Studio, SQL

Project Overview:

This project is a comprehensive SQL-based analysis of the AdventureWorks database. It involved answering various business questions using SQL queries focused on products, salespeople, performance metrics, currency conversion, and commission analysis.

Key Exercises & Queries:

Exercise 1 – Top Products by Average Review

- Joined `product` and `productreview`.
- Calculated average rating and number of reviews per product.

```
SELECT
    p.productid,
    p.NAME,
    ROUND(AVG(pr.rating), 2) AS avgrating,
    COUNT(pr.productreviewid) AS num_ratings
FROM
    product p
INNER JOIN
    productreview pr ON p.productid = pr.productid
GROUP BY
    p.productid, p.NAME
ORDER BY
    avgrating DESC;
```

Exercise 2 – Product Descriptions and Sales

- Found English product descriptions.
- Used CTE to find top-selling products by quantity sold.

```
WITH english_description AS (
    SELECT
        pmpdc.productmodelid,
        pd.description
    FROM
        productmodelproductdescriptionculture pmpdc
```

```
    JOIN
        productdescription pd ON pmpdc.productdescriptionid =
pd.productdescriptionid
    WHERE
        pmpdc.cultureid = 'en'
)

SELECT
    ed.productmodelid,
    ed.description,
    p.NAME,
    SUM(sod.orderqty) AS total_orders
FROM
    english_description ed
JOIN
    product p ON ed.productmodelid = p.productmodelid
JOIN
    salesorderdetail sod ON p.productid = sod.productid
GROUP BY
    ed.productmodelid, ed.description, p.NAME
ORDER BY
    total_orders DESC
LIMIT 10;
```

Exercise 3 – Sales by Subcategory and Price

- Calculated total quantity sold and average list price by subcategory using CTEs.

```
WITH product_sales AS (
    SELECT
        productid,
        SUM(orderqty) AS quantity
    FROM
        salesorderdetail
    GROUP BY
        productid
),
product_info AS (
    SELECT
        p.productid,
        pc.name AS category,
        psc.name AS subcategory,
        p.listprice
    FROM
        product p
```

```
    JOIN
        productsubcategory psc ON p.productsubcategoryid =
psc.productsubcategoryid
    JOIN
        productcategory pc ON psc.productcategoryid = pc.productcategoryid
)

SELECT
    pi.category,
    pi.subcategory,
    AVG(pi.listprice) AS average_price_in_subcategory,
    SUM(ps.quantity) AS total_items_sold_in_subcategory
FROM
    product_info pi
JOIN
    product_sales ps ON pi.productid = ps.productid
GROUP BY
    pi.category, pi.subcategory
ORDER BY
    pi.category, pi.subcategory;
```

Exercise 4 & 5 – Top Salespeople by YTD and Actual Sales

- Used `salesperson.salesytd` and compared it with real 2014 sales data from `salesorderheader`.

```
-- Exercise 4
SELECT
    businessentityid,
    salesytd
FROM
    salesperson
ORDER BY
    salesytd DESC
LIMIT 5;

-- Exercise 5
SELECT
    salespersonid,
    SUM(subtotal) AS totalsales
FROM
    salesorderheader
WHERE
    salespersonid IS NOT NULL
    AND salespersonid <> ''
```

```
    AND orderdate BETWEEN '2014-01-01' AND '2014-12-31'
GROUP BY
    salespersonid
ORDER BY
    totalsales DESC
LIMIT 5;
```

Exercise 6 – Manual Calculation of Sales Using Detail Table

- Used detailed `salesorderdetail` prices to manually calculate order totals.

```
WITH order_totals AS (
    SELECT
        salesorderid,
        SUM(unitprice * (1 - unitpricediscount) * orderqty) AS ordertotal
    FROM
        salesorderdetail
    GROUP BY
        salesorderid
),
order_salesperson AS (
    SELECT
        salesorderid,
        salespersonid
    FROM
        salesorderheader
    WHERE
        salespersonid IS NOT NULL
        AND salespersonid <> ''
        AND orderdate BETWEEN '2014-01-01' AND '2014-12-31'
)

SELECT
    os.salespersonid,
    SUM(ot.ordertotal) AS ordertotalsum
FROM
    order_totals ot
JOIN
    order_salesperson os ON ot.salesorderid = os.salesorderid
GROUP BY
    os.salespersonid
ORDER BY
    ordertotalsum DESC
LIMIT 5;
```

Exercise 7 – Correlation Between Sales and Commission

- Joined manual sales totals with commission percentages.

```sql
WITH order_totals AS (
    SELECT
        salesorderid,
        SUM(unitprice * (1 - unitpricediscount) * orderqty) AS ordertotal
    FROM
        salesorderdetail
    GROUP BY
        salesorderid
),
order_salesperson AS (
    SELECT
        salesorderid,
        salespersonid
    FROM
        salesorderheader
    WHERE
        salespersonid IS NOT NULL
        AND salespersonid <> ''
        AND orderdate BETWEEN '2014-01-01' AND '2014-12-31'
),
sales_summary AS (
    SELECT
        os.salespersonid,
        SUM(ot.ordertotal) AS ordertotalsum
    FROM
        order_totals ot
    JOIN
        order_salesperson os ON ot.salesorderid = os.salesorderid
    GROUP BY
        os.salespersonid
)

SELECT
    ss.salespersonid,
    ss.ordertotalsum,
    sp.commissionpct
FROM
    sales_summary ss
JOIN
    salesperson sp ON ss.salespersonid = sp.businessentityid;
```

Exercise 8 – Currency Analysis of Sales Orders

- Identified currency used per sales order (USD shown as 'Null').

```
SELECT
    soh.salespersonid,
    soh.salesorderid,
    CASE
        WHEN soh.currencyrateid IS NULL THEN 'Null'
        ELSE CAST(soh.currencyrateid AS TEXT)
    END AS currencyrateid,
    CASE
        WHEN cr.tocurrencycode IS NULL THEN 'Null'
        ELSE cr.tocurrencycode
    END AS tocurrencycode
FROM
    salesorderheader soh
LEFT JOIN
    currencyrate cr ON soh.currencyrateid = cr.currencyrateid
WHERE
    soh.salespersonid IS NOT NULL
    AND soh.salespersonid <> ''
    AND soh.orderdate BETWEEN '2014-01-01' AND '2014-12-31'
ORDER BY
    soh.salespersonid
LIMIT 10;
```

Exercise 9 – Sales by Currency and Commission Correlation

- Grouped sales totals by salesperson and currency code, joined with commission data.

```
WITH order_totals AS (
    SELECT
        salesorderid,
        SUM(unitprice * (1 - unitpricediscount) * orderqty) AS ordertotal
    FROM
        salesorderdetail
    GROUP BY
        salesorderid
),
order_currency AS (
    SELECT
        salesorderid,
        salespersonid,
        CASE
```

```
            WHEN cr.tocurrencycode IS NULL THEN 'USD'
            ELSE cr.tocurrencycode
        END AS tocurrencycode
    FROM
        salesorderheader soh
    LEFT JOIN
        currencyrate cr ON soh.currencyrateid = cr.currencyrateid
    WHERE
        soh.salespersonid IS NOT NULL
        AND soh.salespersonid <> ''
        AND soh.orderdate BETWEEN '2014-01-01' AND '2014-12-31'
),
sales_summary AS (
    SELECT
        oc.salespersonid,
        oc.tocurrencycode,
        SUM(ot.ordertotal) AS ordertotalsum
    FROM
        order_totals ot
    JOIN
        order_currency oc ON ot.salesorderid = oc.salesorderid
    GROUP BY
        oc.salespersonid, oc.tocurrencycode
)

SELECT
    ss.salespersonid,
    ss.tocurrencycode,
    ss.ordertotalsum,
    sp.commissionpct
FROM
    sales_summary ss
JOIN
    salesperson sp ON ss.salespersonid = sp.businessentityid
ORDER BY
    ss.tocurrencycode ASC,
    ss.ordertotalsum DESC;
```
