Data Extraction and Basic Analysis

1. Top 10 Customers by Total Payments

This query identifies the top 10 customers based on their total payments, which helps in understanding who the most valuable customers are.

--Query

```sql
--This query identifies the top 10 customers based on their total payments,
--which helps in understanding who the most valuable customers are.

SELECT TOP 10 c.customerName, SUM(p.amount) AS TotalPayments
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
GROUP BY c.customerName
ORDER BY TotalPayments DESC;
```

--Result

	customerName	TotalPayments
1	Robert Davis LLC	112620.57
2	William Smith LLC	107464.94
3	William Brown LLC	104487.29
4	Michael Smith LLC	99724.54
5	Robert Johnson LLC	98909.57
6	John Jones LLC	90283.74
7	John Williams LLC	89362.58
8	William Davis LLC	80707.52
9	Robert Brown LLC	77588.69
10	Robert Jones LLC	76754.21

2. Monthly Sales Trends

This query shows the total sales per month, helping to identify seasonal trends and peak sales periods.

--Query

```sql
--This query shows the total sales per month, helping to identify seasonal trends and peak sales periods.
SELECT FORMAT(o.orderDate, 'yyyy-MM') AS SalesMonth,
       SUM(od.quantityOrdered * od.priceEach) AS MonthlySales
FROM orders o
JOIN orderdetails od ON o.orderNumber = od.orderNumber
GROUP BY FORMAT(o.orderDate, 'yyyy-MM')
ORDER BY SalesMonth;
```

--Result

	SalesMonth	MonthlySales
1	2024-06	540510.48
2	2024-07	868894.97
3	2024-08	723842.65
4	2024-09	584517.64

---

3. Products with Low Stock Levels

This query lists products that are running low on stock, which is essential for inventory management.

--Query

```sql
--This query lists products that are running low on stock, which is essential for inventory management.
SELECT p.productCode, p.productName, p.quantityInStock, p.MSRP
FROM products p
WHERE p.quantityInStock < 60
ORDER BY p.quantityInStock ASC;
```

--Result

	productCode	productName	quantityInStock	MSRP
1	PRD00294	Product 294	52	1081.73
2	PRD00736	Product 736	52	1324.10
3	PRD00187	Product 187	54	1114.28
4	PRD00474	Product 474	54	796.42
5	PRD00302	Product 302	55	107.44
6	PRD00996	Product 996	57	1189.76
7	PRD00893	Product 893	58	1327.09
8	PRD00772	Product 772	59	753.07

4-Most Popular Products

This query identifies the most popular product based on the quantity ordered,

which can help with product marketing and stocking strategies.

--Query

```
--This query identifies the most popular product based on the quantity ordered,
--which can help with product marketing and stocking strategies.
WITH TopProduct AS (
    SELECT TOP 1 p.productCode
    FROM products p
    JOIN orderdetails od ON p.productCode = od.productCode
    GROUP BY p.productCode
    ORDER BY SUM(od.quantityOrdered) DESC)

SELECT p.productCode,
       p.productName,
       p.productLine,
       p.productScale,
       p.productVendor,
       p.productDescription,
       p.quantityInStock,
       p.buyPrice,
       p.MSRP,
       pl.textDescription AS ProductLineDescription,
       SUM(od.quantityOrdered) AS TotalQuantityOrdered,
       SUM(od.quantityOrdered * od.priceEach) AS TotalSalesValue
FROM products p
JOIN TopProduct tp ON p.productCode = tp.productCode
JOIN orderdetails od ON p.productCode = od.productCode
JOIN productlines pl ON p.productLine = pl.productLine
GROUP BY p.productCode, p.productName, p.productLine, p.productScale,
         p.productVendor, p.productDescription, p.quantityInStock,
         p.buyPrice, p.MSRP, pl.textDescription;
```

--Result

5. Employee Sales Performance

Identifies employees who have not made any sales, which could help in addressing performance issues or reassigning tasks to improve overall efficiency.

--Query

```sql
--5-Identifies employees who have not made any sales, which could help in addressing performance issues or reassigning tasks to improve overall efficiency.
SELECT e.employeeNumber, e.firstName + ' ' + e.lastName AS EmployeeName
FROM employees e
LEFT JOIN customers c ON e.employeeNumber = c.salesRepEmployeeNumber
LEFT JOIN orders o ON c.customerNumber = o.customerNumber
WHERE o.orderNumber IS NULL
GROUP BY e.employeeNumber, e.firstName, e.lastName;  -- Lists employees with no sales activity
```

--Result

	employeeNumber	EmployeeName
1	3	Robert Davis
2	4	William Johnson
3	5	James Williams
4	7	John Davis
5	8	Robert Jones
6	10	William Williams
7	16	William Davis
8	17	Michael Johnson
9	18	James Davis
10	19	Robert Williams
11	20	John Jones
12	21	David Williams
13	24	Robert Davis
14	27	James Williams
15	28	Robert Jones
16	30	James Johnson
17	31	David Jones
18	32	David Jones

6. Top 5 Most Ordered Products

Identifies the most popular products based on order quantity, helping the business focus on high-demand items, adjust marketing strategies, and manage inventory efficiently.

--Query

```sql
--This query identifies the most profitable products by calculating the total profit (sales minus cost)
--for each product, helping the business focus on high-margin items.
SELECT TOP 5 p.productName,
       SUM((od.priceEach - p.buyPrice) * od.quantityOrdered) AS TotalProfit
FROM products p
JOIN orderdetails od ON p.productCode = od.productCode
GROUP BY p.productName
ORDER BY TotalProfit DESC;  -- Orders by profit to highlight top-performing products
```

--Result

```sql
--7-This query analyzes payment behavior by comparing average payment amounts across different customers,
--which helps in identifying reliable payers and those who may need follow-up or credit adjustments.
SELECT c.customerName,
       COUNT(p.amount) AS NumberOfPayments,
       AVG(p.amount) AS AveragePaymentAmount,
       SUM(p.amount) AS TotalPayments
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
GROUP BY c.customerName
ORDER BY AveragePaymentAmount DESC;  -- Orders by average payment amount to highlight payment behavior
```

7. Customer Payment Behavior Analysis

This query analyzes payment behavior by comparing average payment amounts across different customers, which helps in identifying reliable payers and those who may need follow-up or credit adjustments.

--Query

```sql
--7-This query analyzes payment behavior by comparing average payment amounts across different customers,
--which helps in identifying reliable payers and those who may need follow-up or credit adjustments.
SELECT c.customerName,
       COUNT(p.amount) AS NumberOfPayments,
       AVG(p.amount) AS AveragePaymentAmount,
       SUM(p.amount) AS TotalPayments
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
GROUP BY c.customerName
ORDER BY AveragePaymentAmount DESC;  -- Orders by average payment amount to highlight payment behavior
```

--Result

```sql
--7-This query analyzes payment behavior by comparing average payment amounts across different customers,
--which helps in identifying reliable payers and those who may need follow-up or credit adjustments.
SELECT c.customerName,
       COUNT(p.amount) AS NumberOfPayments,
       AVG(p.amount) AS AveragePaymentAmount,
       SUM(p.amount) AS TotalPayments
FROM customers c
JOIN payments p ON c.customerNumber = p.customerNumber
GROUP BY c.customerName
ORDER BY AveragePaymentAmount DESC;  -- Orders by average payment amount to highlight payment behavior
```

	customerName	NumberOfPayments	AveragePaymentAmount	TotalPayments
1	James Johnson LLC	14	3056.595714	42792.34
2	Michael Smith LLC	33	3021.955757	99724.54
3	John Jones LLC	30	3009.458000	90283.74
4	James Brown LLC	20	2749.464500	54989.29
5	James Williams LLC	16	2728.454375	43655.27
6	William Davis LLC	30	2690.250666	80707.52
7	John Davis LLC	24	2664.606666	63950.56
8	David Brown LLC	27	2644.267037	71395.21
9	Robert Smith LLC	25	2632.600000	65815.00
10	John Williams LLC	34	2628.311176	89362.58
11	Michael Johnson ...	22	2608.297727	57382.55
12	David Davis LLC	26	2606.971538	67781.26
13	William Brown LLC	41	2548.470487	104487.29
14	James Jones LLC	30	2536.216666	76086.50
15	William Williams LLC	24	2519.747083	60473.93
16	William Johnson L...	23	2513.143043	57802.29

Thank You