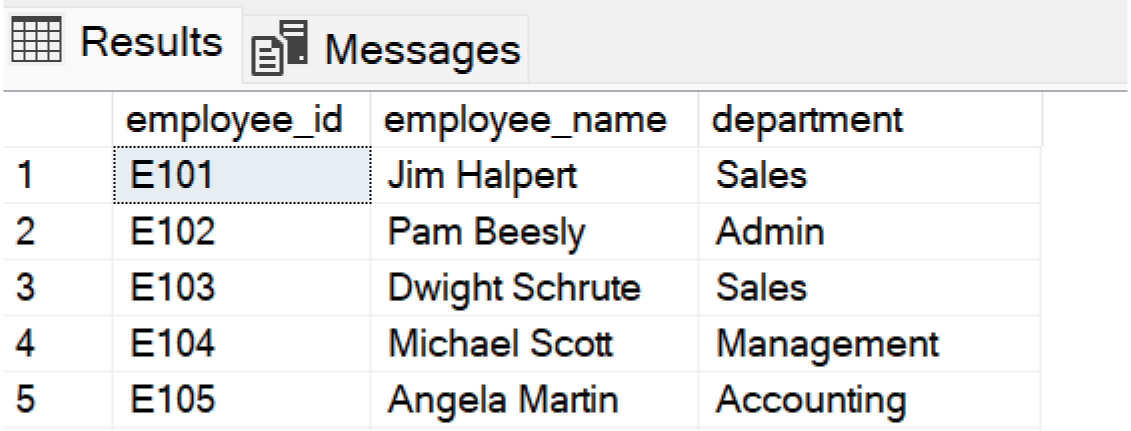
--- BASIC QUERIES ---

-- Can you provide a list of all employees working in the Scranton branch?"

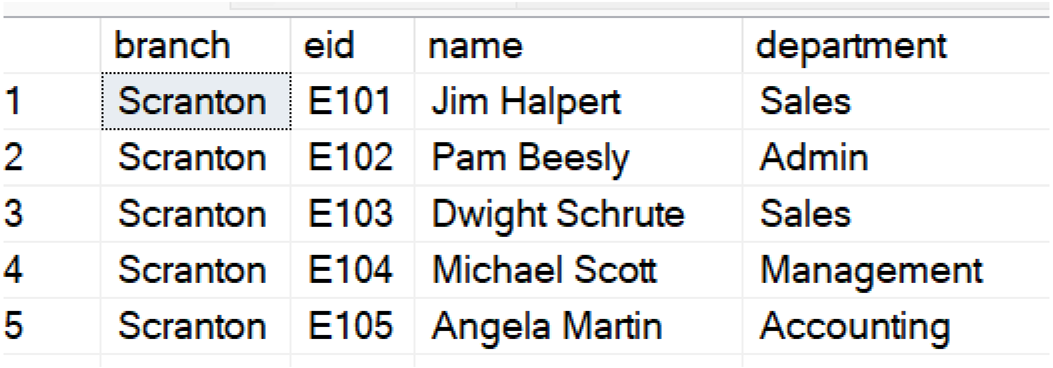
```
SELECT e.employee_id, e.employee_name, e.department

FROM employees e

WHERE branch_id = 'B101'; -- B101 represents the Scranton branch
```



	employee_id	employee_name	department
1	E101	Jim Halpert	Sales
2	E102	Pam Beesly	Admin
3	E103	Dwight Schrute	Sales
4	E104	Michael Scott	Management
5	E105	Angela Martin	Accounting

-- We need an employee list from all branches, including their respective branch names.

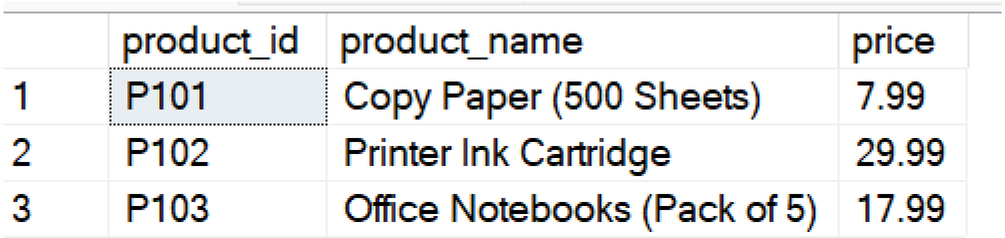
```
SELECT b.branch_name AS branch, e.employee_id AS eid, e.employee_name AS name,
e.department

FROM branches b

JOIN employees e ON b.branch_id = e.branch_id;

-- This query joins the branches and employees tables
```



	branch	eid	name	department
1	Scranton	E101	Jim Halpert	Sales
2	Scranton	E102	Pam Beesly	Admin
3	Scranton	E103	Dwight Schrute	Sales
4	Scranton	E104	Michael Scott	Management
5	Scranton	E105	Angela Martin	Accounting

--Please provide a list of all products we sell, along with their respective prices.

```
SELECT DISTINCT(product_id), product_name, price

FROM products;

-- DISTINCT ensures we don't have duplicate product entries
```

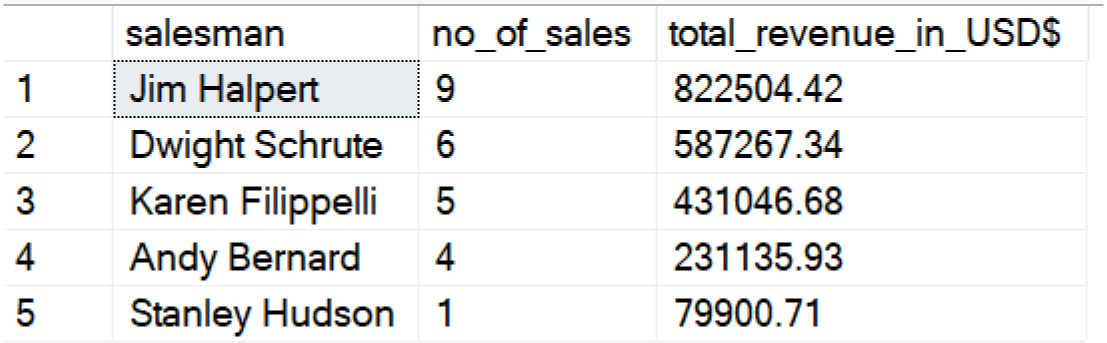


	product_id	product_name	price
1	P101	Copy Paper (500 Sheets)	7.99
2	P102	Printer Ink Cartridge	29.99
3	P103	Office Notebooks (Pack of 5)	17.99

--- SALES ANALYSIS ---

-- Can you provide a list of salespeople, the number of sales they've made, and the total revenue generated?

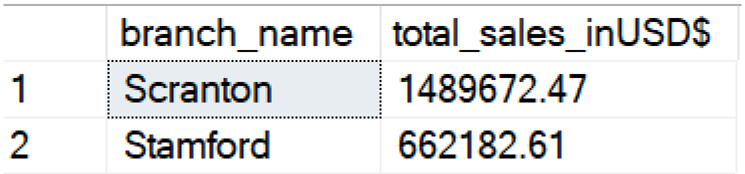
```
SELECT e.employee_name AS salesman,

    COUNT(s.sale_id) AS no_of_sales,

    SUM(s.sale_amount) AS total_revenue_in_USD$

FROM sales s

JOIN employees e ON s.employee_id = e.employee_id

GROUP BY e.employee_name

ORDER BY no_of_sales DESC;
```



	salesman	no_of_sales	total_revenue_in_USD$
1	Jim Halpert	9	822504.42
2	Dwight Schrute	6	587267.34
3	Karen Filippelli	5	431046.68
4	Andy Bernard	4	231135.93
5	Stanley Hudson	1	79900.71

-- We need to know the total sales amount for each branch. Can you provide a report for that?

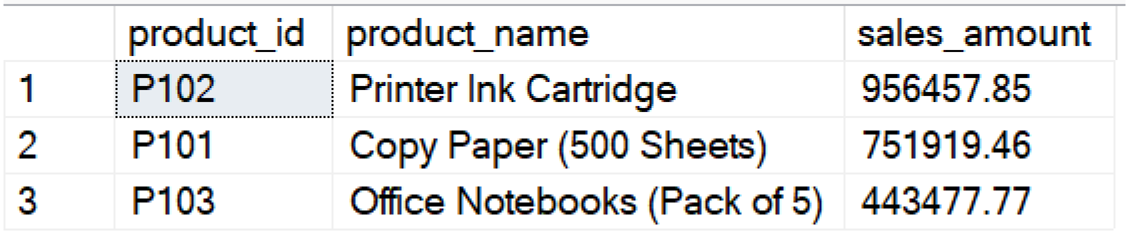
```
SELECT b.branch_name,

    SUM(s.sale_amount) AS total_sales_inUSD$

FROM sales s

JOIN branches b ON s.branch_id = b.branch_id

GROUP BY b.branch_name;
```



	branch_name	total_sales_inUSD$
1	Scranton	1489672.47
2	Stamford	662182.61

-- Show me the top-selling products across all branches.

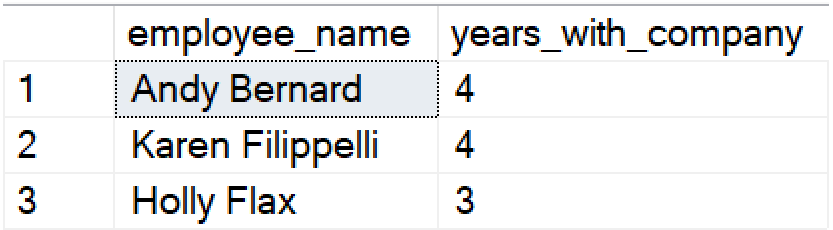
```
SELECT p.product_id, p.product_name,

    SUM(s.sale_amount) AS sales_amount

FROM products p

JOIN sales s ON p.product_id = s.product_id

GROUP BY p.product_id, p.product_name

ORDER BY sales_amount DESC;
```



	product_id	product_name	sales_amount
1	P102	Printer Ink Cartridge	956457.85
2	P101	Copy Paper (500 Sheets)	751919.46
3	P103	Office Notebooks (Pack of 5)	443477.77

--Can you tell me every employee's tenure with us up to 2010 for the Stamford branch?

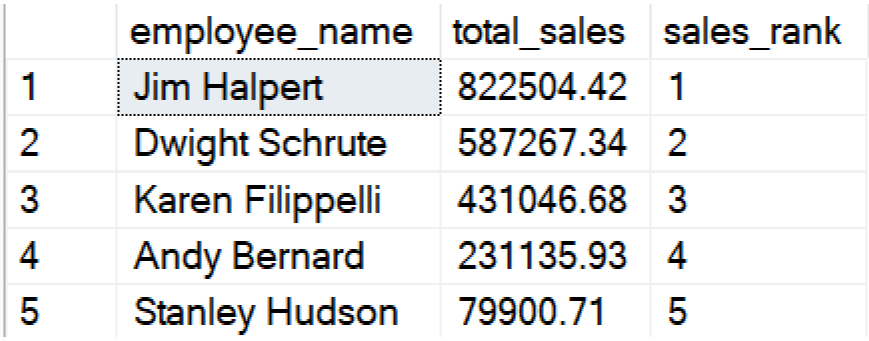
```
SELECT employee_name,

    DATEDIFF(YEAR, hire_date, '2010-12-31') AS years_with_company

FROM employees

WHERE branch_id LIKE '%201';
```



	employee_name	years_with_company
1	Andy Bernard	4
2	Karen Filippelli	4
3	Holly Flax	3

--Which employees from both branches are the top performers based on sales?

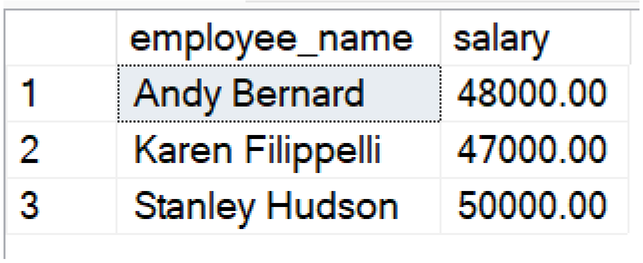
```
SELECT e.employee_name,

    SUM(s.sale_amount) AS total_sales,

    RANK() OVER (ORDER BY SUM(s.sale_amount) DESC) AS sales_rank

FROM employees e

JOIN sales s ON e.employee_id = s.employee_id

GROUP BY e.employee_name;
```



	employee_name	total_sales	sales_rank
1	Jim Halpert	822504.42	1
2	Dwight Schrute	587267.34	2
3	Karen Filippelli	431046.68	3
4	Andy Bernard	231135.93	4
5	Stanley Hudson	79900.71	5

--Can you find out if any sales employees are earning less than the average salary for their department?

```
SELECT employee_name, salary

FROM employees

WHERE department = 'Sales' AND salary < (

  SELECT AVG(salary)

  FROM employees

  WHERE department LIKE '%Sales%'

);
```

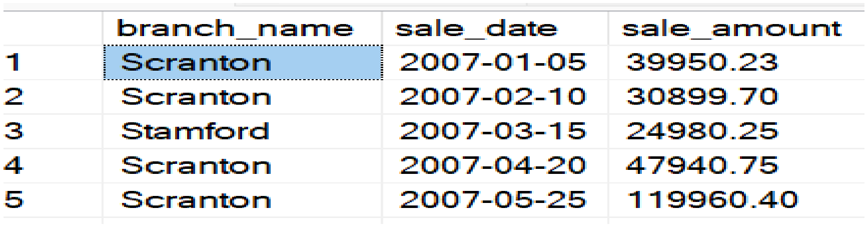


	employee_name	salary
1	Andy Bernard	48000.00
2	Karen Filippelli	47000.00
3	Stanley Hudson	50000.00

--- ADVANCED ANALYSIS ---

-- We need a report showing sales trends for each branch over time.

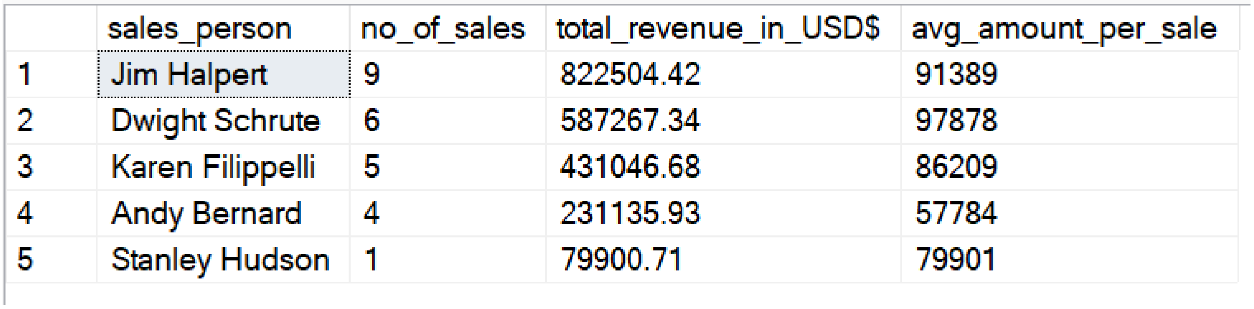
```
SELECT b.branch_name, s.sale_date, SUM(s.sale_amount) AS sale_amount

FROM sales s

JOIN branches b ON s.branch_id = b.branch_id

GROUP BY b.branch_name, s.sale_date

ORDER BY s.sale_date;
```



	branch_name	sale_date	sale_amount
1	Scranton	2007-01-05	39950.23
2	Scranton	2007-02-10	30899.70
3	Stamford	2007-03-15	24980.25
4	Scranton	2007-04-20	47940.75
5	Scranton	2007-05-25	119960.40

--Please create a view for frequent access to sales performance by each employee.

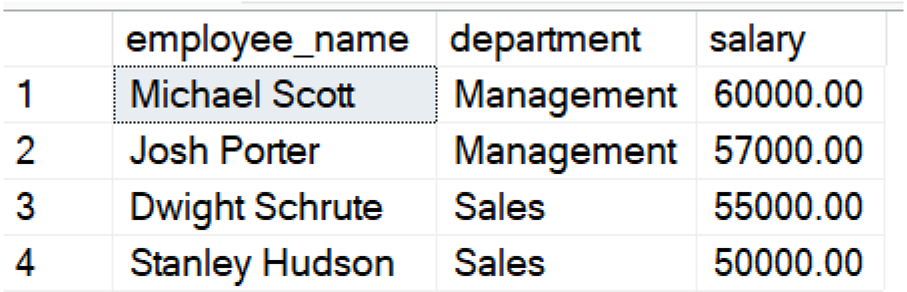
```
DROP VIEW IF EXISTS sales_performance;


CREATE VIEW sales_performance AS

SELECT e.employee_name AS sales_person,

    COUNT(s.sale_id) AS no_of_sales,

    SUM(s.sale_amount) AS total_revenue_in_USD$,

    CAST(ROUND((SUM(s.sale_amount)/COUNT(s.sale_id)),0) AS INT) AS avg_amount_per_sale

FROM sales s

JOIN employees e ON s.employee_id = e.employee_id

GROUP BY e.employee_name;


SELECT *

FROM sales_performance

ORDER BY total_revenue_in_USD$ DESC;

-- This creates a view for easy access to sales performance metrics
```



	sales_person	no_of_sales	total_revenue_in_USD$	avg_amount_per_sale
1	Jim Halpert	9	822504.42	91389
2	Dwight Schrute	6	587267.34	97878
3	Karen Filippelli	5	431046.68	86209
4	Andy Bernard	4	231135.93	57784
5	Stanley Hudson	1	79900.71	79901

--Please send us the list of employees with their respective departments whose salary ranges from 50,000 to 60,000

```
SELECT employee_name, department, salary

FROM employees

WHERE salary BETWEEN 50000 AND 60000

ORDER BY salary DESC;
```

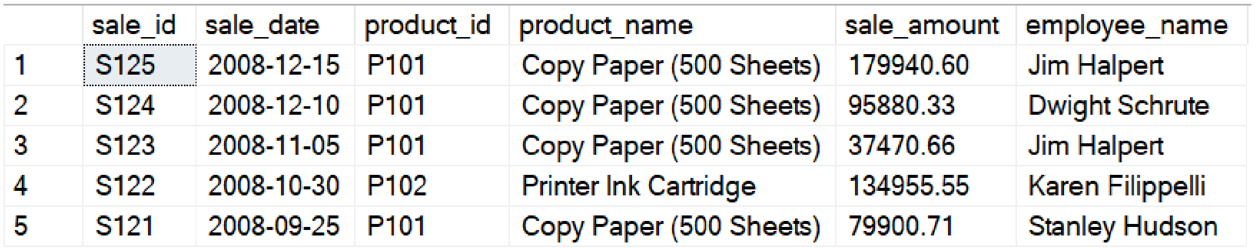


	employee_name	department	salary
1	Michael Scott	Management	60000.00
2	Josh Porter	Management	57000.00
3	Dwight Schrute	Sales	55000.00
4	Stanley Hudson	Sales	50000.00

--- DATA RETRIEVAL, GROUPING AND FILTERING ---

-- Retrieve the last 5 records of sales

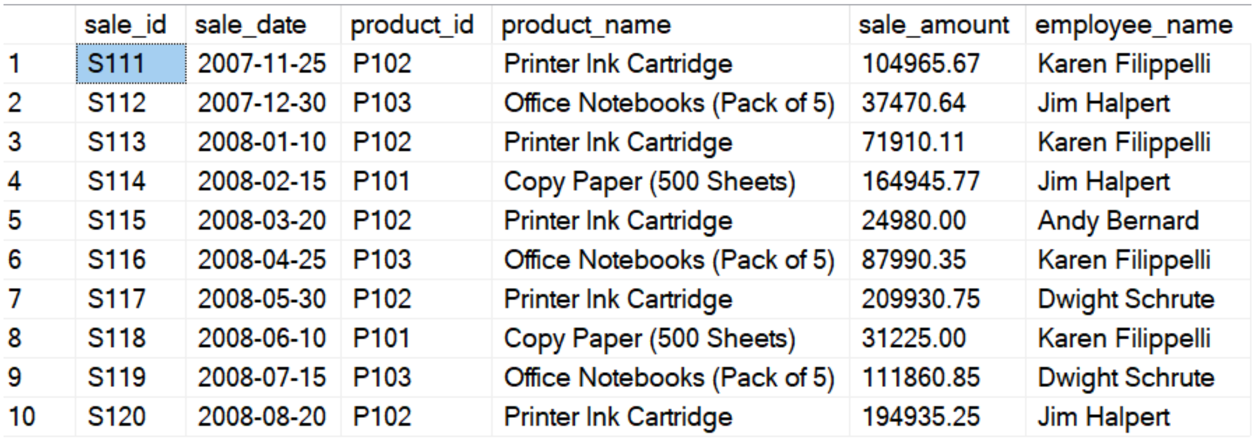
```
SELECT TOP 5 s.sale_id, s.sale_date, s.product_id, p.product_name,

    s.sale_amount, e.employee_name

FROM sales s

JOIN employees e ON e.employee_id = s.employee_id

JOIN products p ON p.product_id = s.product_id

ORDER BY sale_date DESC;
```



	sale_id	sale_date	product_id	product_name	sale_amount	employee_name
1	S125	2008-12-15	P101	Copy Paper (500 Sheets)	179940.60	Jim Halpert
2	S124	2008-12-10	P101	Copy Paper (500 Sheets)	95880.33	Dwight Schrute
3	S123	2008-11-05	P101	Copy Paper (500 Sheets)	37470.66	Jim Halpert
4	S122	2008-10-30	P102	Printer Ink Cartridge	134955.55	Karen Filippelli
5	S121	2008-09-25	P101	Copy Paper (500 Sheets)	79900.71	Stanley Hudson

-- Specific Range of Sales Records from 11th to 20th

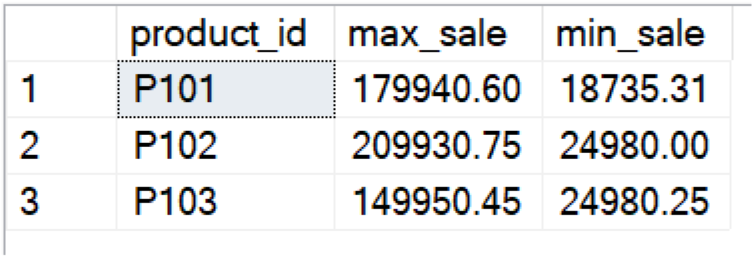
```
SELECT s.sale_id, s.sale_date, s.product_id, p.product_name,
    s.sale_amount, e.employee_name
FROM sales s
JOIN employees e ON e.employee_id = s.employee_id
JOIN products p ON p.product_id = s.product_id
ORDER BY sale_date
OFFSET 10 ROWS
FETCH NEXT 10 ROWS ONLY;
```



	sale_id	sale_date	product_id	product_name	sale_amount	employee_name
1	S111	2007-11-25	P102	Printer Ink Cartridge	104965.67	Karen Filippelli
2	S112	2007-12-30	P103	Office Notebooks (Pack of 5)	37470.64	Jim Halpert
3	S113	2008-01-10	P102	Printer Ink Cartridge	71910.11	Karen Filippelli
4	S114	2008-02-15	P101	Copy Paper (500 Sheets)	164945.77	Jim Halpert
5	S115	2008-03-20	P102	Printer Ink Cartridge	24980.00	Andy Bernard
6	S116	2008-04-25	P103	Office Notebooks (Pack of 5)	87990.35	Karen Filippelli
7	S117	2008-05-30	P102	Printer Ink Cartridge	209930.75	Dwight Schrute
8	S118	2008-06-10	P101	Copy Paper (500 Sheets)	31225.00	Karen Filippelli
9	S119	2008-07-15	P103	Office Notebooks (Pack of 5)	111860.85	Dwight Schrute
10	S120	2008-08-20	P102	Printer Ink Cartridge	194935.25	Jim Halpert

-- Find the maximum and minimum sale amounts per product.

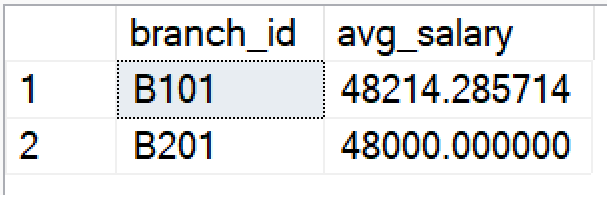
```
SELECT product_id, MAX(sale_amount) AS max_sale, MIN(sale_amount) AS min_sale
FROM sales
GROUP BY product_id;
```



	product_id	max_sale	min_sale
1	P101	179940.60	18735.31
2	P102	209930.75	24980.00
3	P103	149950.45	24980.25

--Branches with average salary over a 40000 amount.

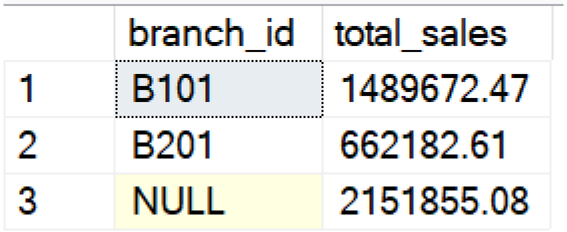
```
SELECT branch_id, AVG(salary) AS avg_salary

FROM employees

GROUP BY branch_id

HAVING AVG(salary) > 40000;
```



	branch_id	avg_salary
1	B101	48214.285714
2	B201	48000.000000

-- Use ROLLUP to get subtotals and grand totals of sales by branch.

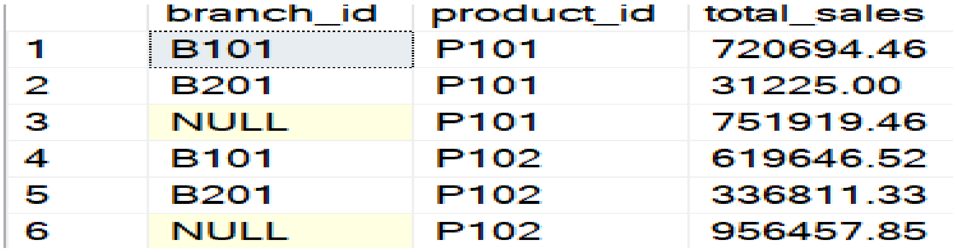
```
SELECT branch_id, SUM(sale_amount) AS total_sales

FROM sales

GROUP BY ROLLUP(branch_id);

-- This query provides a hierarchical summary of sales by branch
```



	branch_id	total_sales
1	B101	1489672.47
2	B201	662182.61
3	NULL	2151855.08

-- Use CUBE to get cross-tabulations for sales by branch and product.

```
SELECT branch_id, product_id, SUM(sale_amount) AS total_sales

FROM sales

GROUP BY CUBE(branch_id, product_id); -- This query provides a multi-dimensional
summary of sales
```

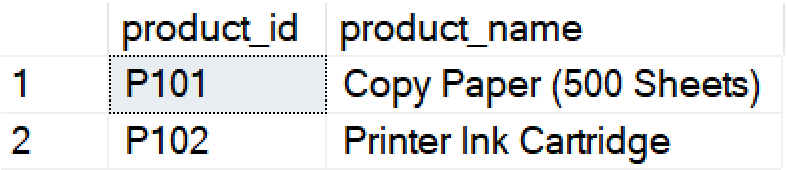


	branch_id	product_id	total_sales
1	B101	P101	720694.46
2	B201	P101	31225.00
3	NULL	P101	751919.46
4	B101	P102	619646.52
5	B201	P102	336811.33
6	NULL	P102	956457.85

--- SUBQUERIES AND COMPLEX JOINS ---

--Retrieve products with sales amounts greater than the average sale amount of all products.

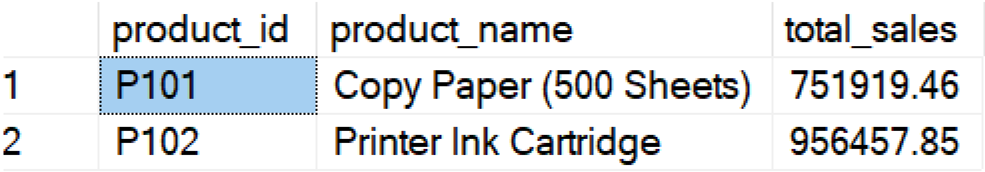
```
SELECT product_id, product_name
FROM products
WHERE product_id IN (
  SELECT product_id
  FROM sales
  GROUP BY product_id
  HAVING SUM(sale_amount) > (
    SELECT AVG(total_sales)
    FROM (
      SELECT product_id, SUM(sale_amount) AS total_sales
      FROM sales
      GROUP BY product_id
    ) AS subquery
  )
);
```



	product_id	product_name
1	P101	Copy Paper (500 Sheets)
2	P102	Printer Ink Cartridge

-- Can you provide the same list of products, but include the sales amount column with the data

```
SELECT p.product_id, p.product_name, s.total_sales
FROM products p
JOIN (
  SELECT product_id, SUM(sale_amount) AS total_sales
  FROM sales
  GROUP BY product_id
  HAVING SUM(sale_amount) > (
    SELECT AVG(total_sales)
    FROM (
      SELECT product_id, SUM(sale_amount) AS total_sales
      FROM sales
      GROUP BY product_id
    ) AS subquery
  )
) s ON p.product_id = s.product_id;
```

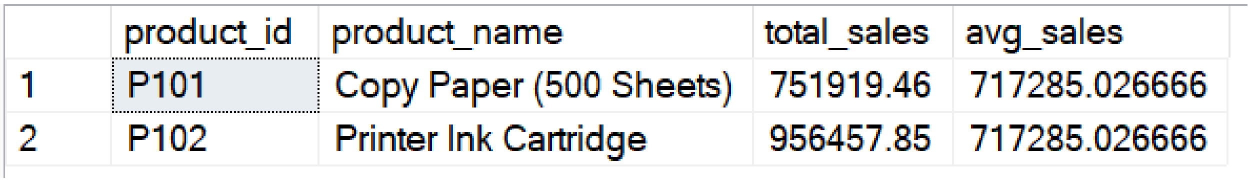


	product_id	product_name	total_sales
1	P101	Copy Paper (500 Sheets)	751919.46
2	P102	Printer Ink Cartridge	956457.85

-- Can you include the average sales data along with the current data for these products?

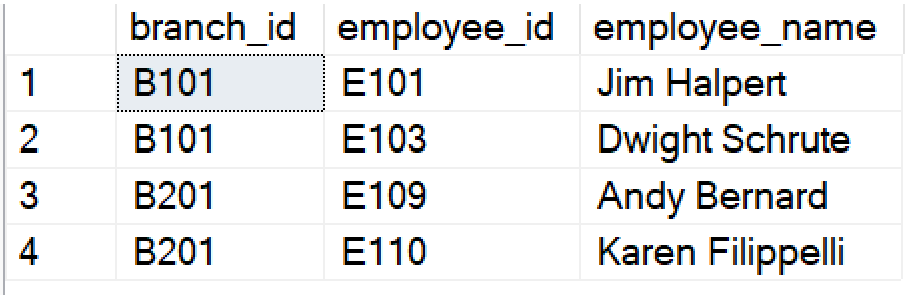
```
WITH AvgSales AS (
  SELECT AVG(total_sales) AS avg_sales
  FROM (
    SELECT product_id, SUM(sale_amount) AS total_sales
    FROM sales
    GROUP BY product_id
  ) AS subquery
)
SELECT p.product_id, p.product_name, s.total_sales, a.avg_sales
FROM products p
JOIN (
  SELECT product_id, SUM(sale_amount) AS total_sales
  FROM sales
  GROUP BY product_id
  HAVING SUM(sale_amount) > (SELECT avg_sales FROM AvgSales)
) s ON p.product_id = s.product_id
CROSS JOIN AvgSales a;
```

-- This query provides a comprehensive comparison of product performance



	product_id	product_name	total_sales	avg_sales
1	P101	Copy Paper (500 Sheets)	751919.46	717285.026666
2	P102	Printer Ink Cartridge	956457.85	717285.026666

```
-- Find employees who have made sales above the average sales amount for their branch

SELECT branch_id, e.employee_id, e.employee_name

FROM employees e

WHERE EXISTS (

  SELECT 1

  FROM sales s

  WHERE e.employee_id = s.employee_id

  AND s.sale_amount > (

    SELECT AVG(sale_amount)

    FROM sales s2

    WHERE s2.branch_id = e.branch_id

  )

);

-- This query identifies top-performing employees within each branch
```

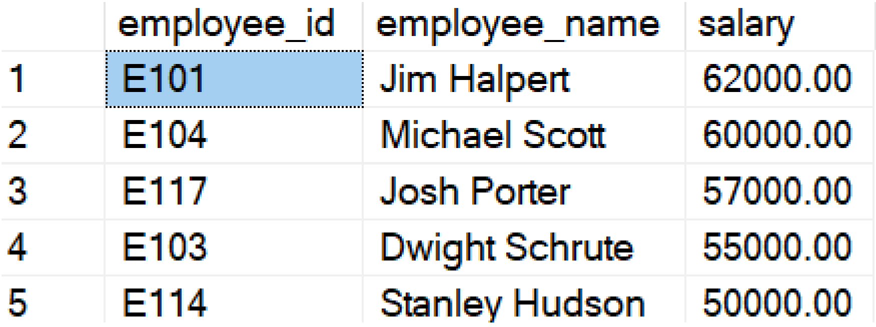


	branch_id	employee_id	employee_name
1	B101	E101	Jim Halpert
2	B101	E103	Dwight Schrute
3	B201	E109	Andy Bernard
4	B201	E110	Karen Filippelli

--- COMMON TABLE EXPRESSIONS ---

--Create a CTE to list employees with the highest salaries

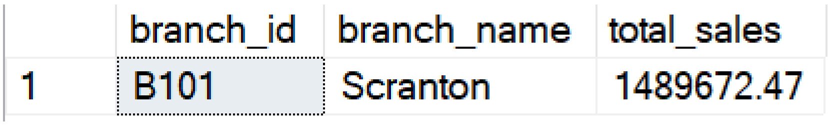
```
WITH HighSalaries AS (
  SELECT TOP 10 employee_id, employee_name, salary
  FROM employees
  ORDER BY salary DESC
)
SELECT * FROM HighSalaries;
```



	employee_id	employee_name	salary
1	E101	Jim Halpert	62000.00
2	E104	Michael Scott	60000.00
3	E117	Josh Porter	57000.00
4	E103	Dwight Schrute	55000.00
5	E114	Stanley Hudson	50000.00

--Use a CTE to calculate total sales per branch and then query for branches with sales exceeding a $1MILLION

```
WITH SalesByBranch AS (
  SELECT s.branch_id, b.branch_name, SUM(s.sale_amount) AS total_sales
  FROM sales s
  JOIN branches b ON s.branch_id = b.branch_id
  GROUP BY s.branch_id, b.branch_name
)
SELECT *
FROM SalesByBranch
WHERE total_sales > 1000000;
```

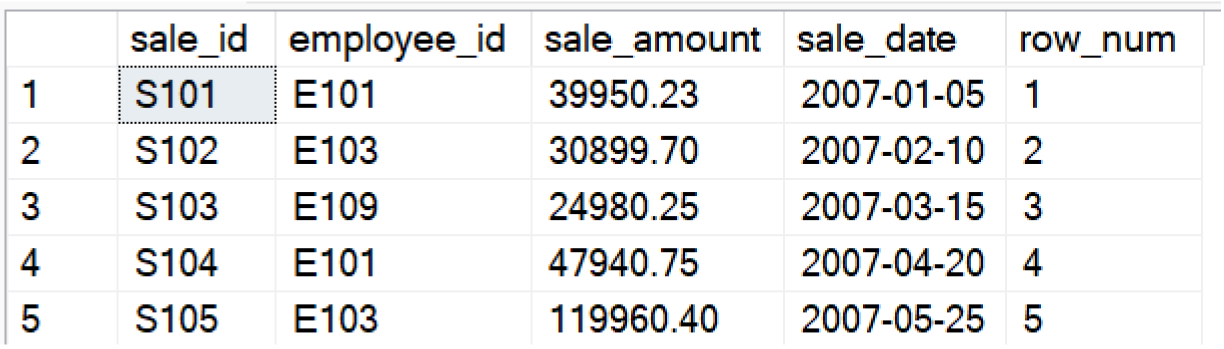


	branch_id	branch_name	total_sales
1	B101	Scranton	1489672.47

--- WINDOW FUNCTIONS ---

--Assign a unique number to each sale record

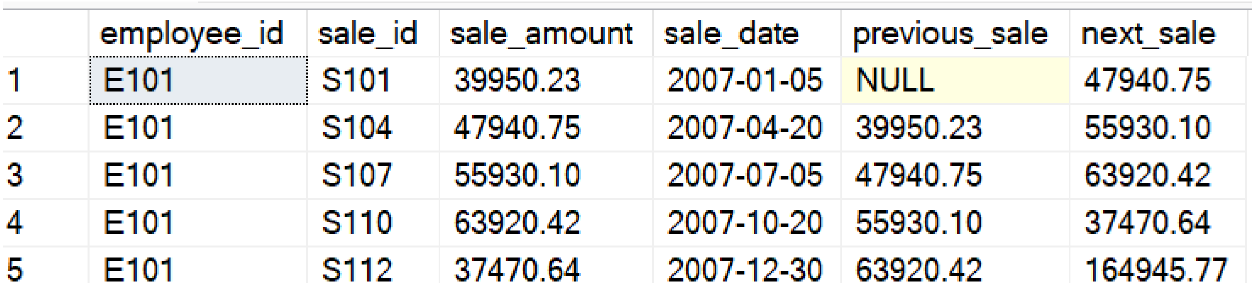
```
SELECT sale_id, employee_id, sale_amount, sale_date,
    ROW_NUMBER() OVER (ORDER BY sale_date) AS row_num
FROM sales;
```



	sale_id	employee_id	sale_amount	sale_date	row_num
1	S101	E101	39950.23	2007-01-05	1
2	S102	E103	30899.70	2007-02-10	2
3	S103	E109	24980.25	2007-03-15	3
4	S104	E101	47940.75	2007-04-20	4
5	S105	E103	119960.40	2007-05-25	5

--Use LAG() and LEAD() to compare sales amounts over time for each employee.

```
SELECT employee_id, sale_id, sale_amount, sale_date,
    LAG(sale_amount) OVER (
      PARTITION BY employee_id ORDER BY sale_date
    ) AS previous_sale,
    LEAD(sale_amount) OVER (
      PARTITION BY employee_id ORDER BY sale_date
    ) AS next_sale
FROM sales;
```



	employee_id	sale_id	sale_amount	sale_date	previous_sale	next_sale
1	E101	S101	39950.23	2007-01-05	NULL	47940.75
2	E101	S104	47940.75	2007-04-20	39950.23	55930.10
3	E101	S107	55930.10	2007-07-05	47940.75	63920.42
4	E101	S110	63920.42	2007-10-20	55930.10	37470.64
5	E101	S112	37470.64	2007-12-30	63920.42	164945.77

--- DATA MODIFICATION ---

--Insert new records into employees and sales tables

```
INSERT INTO employees (employee_id, employee_name, branch_id, department, hire_date, salary)

VALUES ('E218', 'New Employee', 'B101', 'Sales', '2024-09-01', 55000);


INSERT INTO sales (sale_id, employee_id, product_id, sale_amount, sale_date, branch_id)

VALUES (1001, 'E218', 'P101', 15000, '2024-09-01', 'B101');
```

--We need to remove an employee and their associated sales records.

```
-- Deleting sales associated with the employee first because of the foreign key constraint

DELETE FROM sales

WHERE employee_id = 'E218';

-- Now deleting it from the employees table

DELETE FROM employees

WHERE employee_id = 'E218';
```

--Update product prices and employee salaries

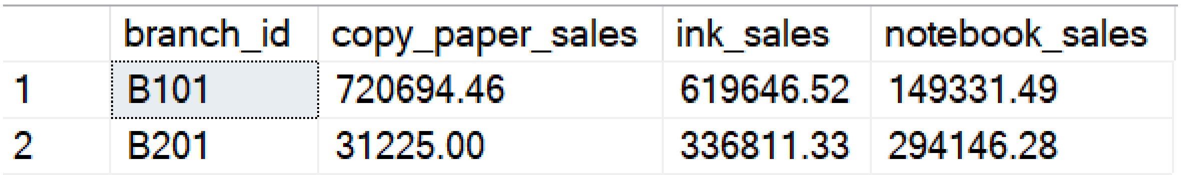
```
UPDATE products

SET price = 17.99

WHERE product_id = 'P103';
```

```
UPDATE employees
SET salary = 62000
WHERE employee_id = 'E101';
```

--Can you summarize the total sales per product by branch?

```
SELECT branch_id,
  SUM(CASE WHEN product_id = 'P101' THEN sale_amount ELSE 0 END) AS copy_paper_sales,
  SUM(CASE WHEN product_id = 'P102' THEN sale_amount ELSE 0 END) AS ink_sales,
  SUM(CASE WHEN product_id = 'P103' THEN sale_amount ELSE 0 END) AS notebook_sales
FROM sales
GROUP BY branch_id;
```

--This query provides a cross-tabulation of sales by product and branch



	branch_id	copy_paper_sales	ink_sales	notebook_sales
1	B101	720694.46	619646.52	149331.49
2	B201	31225.00	336811.33	294146.28

--- SCHEMA MODIFICATION ---

--We need to add a new column to track the email address of each employee.

```
ALTER TABLE employees
ADD email_address VARCHAR(255);
```

--We no longer need the email_address column in the employees table. Can you remove it?

```
ALTER TABLE employees
DROP COLUMN email_address;
```