

TOP 20 SQL QUERY PATTERNS

Every Data Analyst Should Know.



BASIC SELECT PATTERN

```
SELECT column1, column2  
FROM table_name;
```



FILTERING WITH WHERE

```
SELECT *  
FROM orders  
WHERE order_date >=  
'2025-01-01';
```

MULTIPLE CONDITIONS (AND / OR)

```
SELECT *  
FROM customers  
WHERE country =  
'India'  
AND city = 'Delhi';
```



AGGREGATION PATTERN

```
SELECT SUM(sales),  
COUNT(*), AVG(price)  
FROM orders;
```

GROUP BY PATTERN

```
SELECT category, SUM(sales)  
FROM products  
GROUP BY category;
```

HAVING PATTERN

```
SELECT category, SUM(sales)  
AS total_sales  
FROM products  
GROUP BY category  
HAVING total_sales > 10000;
```

INNER JOIN PATTERN

```
SELECT o.order_id,  
       c.customer_name  
FROM orders o  
INNER JOIN customers c  
ON o.customer_id =  
   c.customer_id;
```

LEFT JOIN PATTERN

```
SELECT o.order_id,  
       c.customer_name  
FROM orders o  
INNER JOIN customers c  
ON o.customer_id =  
   c.customer_id;
```

SUBQUERY PATTERN

```
SELECT *  
FROM employees  
WHERE salary > (  
    SELECT AVG(salary)  
    FROM employees  
);
```

CORRELATED SUBQUERY

```
SELECT e1.name, e1.salary  
FROM employees e1  
WHERE salary > (  
    SELECT AVG(salary)  
    FROM employees e2  
    WHERE e1.department =  
    e2.department  
);
```



WINDOW FUNCTION PATTERN

```
SELECT name,  
       salary,  
       RANK() OVER (ORDER BY  
salary DESC) AS  
rank_salary  
FROM employees;
```



PARTITION BY PATTERN

```
SELECT department,  
       name,  
       ROW_NUMBER() OVER  
       (PARTITION BY department  
        ORDER BY salary DESC)  
FROM employees;
```



RUNNING TOTAL PATTERN

```
SELECT order_date,  
       SUM(sales) OVER  
       (ORDER BY order_date) AS  
       running_total  
FROM orders;
```



TOP N RECORDS PATTERN

```
SELECT *  
FROM employees  
ORDER BY salary DESC  
LIMIT 5;
```



NTH HIGHEST SALARY PATTERN

```
SELECT DISTINCT salary  
FROM employees  
ORDER BY salary DESC  
LIMIT 1 OFFSET 2;
```



CASE WHEN PATTERN

```
SELECT name,  
       CASE  
         WHEN salary > 100000  
         THEN 'High'  
         ELSE 'Normal'  
       END AS salary_level  
FROM employees;
```

DATE EXTRACTION PATTERN

```
SELECT EXTRACT(YEAR FROM  
order_date) AS order_year,  
COUNT(*)  
FROM orders  
GROUP BY order_year;
```



SELF JOIN PATTERN

```
SELECT e1.name AS  
employee,  
       e2.name AS manager  
FROM employees e1  
JOIN employees e2  
ON e1.manager_id =  
e2.employee_id;
```



DUPLICATE DETECTION PATTERN

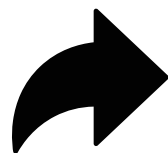
```
SELECT email, COUNT(*)  
FROM users  
GROUP BY email  
HAVING COUNT(*) > 1;
```

CTE (WITH CLAUSE) PATTERN

```
WITH sales_summary AS (  
    SELECT customer_id,  
    SUM(sales) AS total_sales  
    FROM orders  
    GROUP BY customer_id  
)  
SELECT *  
FROM sales_summary  
WHERE total_sales > 5000;
```

IF YOU FOUND IT HELPFUL.

Follow **Aditya Kalyan** for more Data
Analytics Content.



Data Detective Club