

Pivot Table in SQL



how can i pivot a table in sql?

Pivoting a table in SQL typically involves transforming row values into column headers. There are two main ways to achieve this:

1. Using `PIVOT` (SQL Server, Oracle)

If you are using **SQL Server** or **Oracle**, you can use the `PIVOT` operator.

Example:

Imagine you have a sales table:

Salesperson	Year	Sales
Alice	2023	1000
Alice	2024	1500
Bob	2023	1200
Bob	2024	1800

You want to pivot this so that years become columns:

```
sql

SELECT *
FROM (
    SELECT Salesperson, Year, Sales
    FROM SalesData
) AS SourceTable
PIVOT (
    SUM(Sales)
    FOR Year IN ([2023], [2024])
) AS PivotTable;
```

2. Using `CASE WHEN` (Works in MySQL, PostgreSQL, and others)

If your database does not support `PIVOT`, you can use `CASE WHEN` with `SUM()`:

```
sql

SELECT
    Salesperson,
    SUM(CASE WHEN Year = 2023 THEN Sales ELSE 0 END) AS Sales_2023,
    SUM(CASE WHEN Year = 2024 THEN Sales ELSE 0 END) AS Sales_2024
FROM SalesData
GROUP BY Salesperson;
```

3. Using `GROUP_CONCAT()` or `STRING_AGG()` for Dynamic Pivoting

If you have an unknown number of columns, you may need to generate a dynamic SQL query using `GROUP_CONCAT()` (MySQL) or `STRING_AGG()` (SQL Server/PostgreSQL).

Would you like an example of that as well?



what if i have one column with the key and another with the value? and i want to create separate columns for some keys

If you have a table with a **key-value** structure and want to create separate columns for certain keys, you can use either `PIVOT` (if your database supports it) or `CASE WHEN`.

Example Table (`KeyValueTable`)

ID	Key	Value
1	Name	Alice
1	Age	25
1	City	Paris
2	Name	Bob
2	Age	30
2	City	London

Solution 1: Using `CASE WHEN` (Works in MySQL, PostgreSQL, SQL Server, etc.)

```
sql
SELECT
  ID,
  MAX(CASE WHEN Key = 'Name' THEN Value END) AS Name,
  MAX(CASE WHEN Key = 'Age' THEN Value END) AS Age,
  MAX(CASE WHEN Key = 'City' THEN Value END) AS City
FROM KeyValueTable
GROUP BY ID;
```

Result

ID	Name	Age	City
1	Alice	25	Paris
2	Bob	30	London

♦ Why `MAX()`?

Since `CASE WHEN` returns `NULL` when the condition is false, `MAX()` helps pick the non-null value.

Solution 2: Using `PIVOT` (SQL Server, Oracle)



sql

```

SELECT *
FROM (
    SELECT ID, Key, Value FROM KeyValueTable
) AS SourceTable
PIVOT (
    MAX(Value)
    FOR Key IN ([Name], [Age], [City])
) AS PivotTable;

```

When to Use Each Approach?

-  `CASE WHEN` → Works in all databases.
-  `PIVOT` → More concise, but only in **SQL Server** and **Oracle**.

Would you like a **dynamic pivot** solution if the keys are unknown? 🚀



how can i do this in spark?

In **Apache Spark (PySpark)**, you can pivot a key-value table using the `.groupBy().pivot().agg()` functions.

Example Data (`KeyValueTable`)

ID	Key	Value
1	Name	Alice
1	Age	25
1	City	Paris
2	Name	Bob
2	Age	30
2	City	London

Solution: Using `.pivot()` in PySpark

python

```

from pyspark.sql import SparkSession
from pyspark.sql.functions import col

# Initialize Spark session
spark = SparkSession.builder.appName("PivotExample").getOrCreate()

# Sample Data
data = [
    (1, "Name", "Alice"),
    (1, "Age", "25"),
    (1, "City", "Paris"),
    (2, "Name", "Bob"),
    (2, "Age", "30"),
    (2, "City", "London")
]

```

```

]

# Create DataFrame
df = spark.createDataFrame(data, ["ID", "Key", "Value"])

# Pivot the DataFrame
pivot_df = df.groupBy("ID").pivot("Key").agg(first("Value"))

# Show result
pivot_df.show()

```

Output

ID	Name	Age	City
1	Alice	25	Paris
2	Bob	30	London

Explanation

- `.groupBy("ID")` → Groups the data by `ID`.`
- `.pivot("Key")` → Converts unique values in `Key` column into separate columns.`
- `.agg(first("Value"))` → Picks the first value for each pivoted column.`

✅ This approach is scalable and efficient in Spark! 🚀

Would you like a **dynamic pivot** where columns are unknown in advance? ↻



what if i only want to do it for certain keys?

If you only want to pivot for **specific keys** in **PySpark**, you can pass a **list of keys** to `.pivot()`.

Example: Pivot Only "Name" and "Age"

```

python

from pyspark.sql import SparkSession
from pyspark.sql.functions import first

# Initialize Spark session
spark = SparkSession.builder.appName("PivotExample").getOrCreate()

# Sample Data
data = [
    (1, "Name", "Alice"),
    (1, "Age", "25"),
    (1, "City", "Paris"),
    (2, "Name", "Bob"),
    (2, "Age", "30"),
    (2, "City", "London")
]

# Create DataFrame
df = spark.createDataFrame(data, ["ID", "Key", "Value"])

# Pivot only for "Name" and "Age"
pivot_df = df.filter(df.Key.isin(["Name", "Age"])) \

```

```
.groupBy("ID") \
.pivot("Key", ["Name", "Age"]) \
.agg(first("Value"))

# Show result
pivot_df.show()
```

Output

ID	Name	Age
1	Alice	25
2	Bob	30

Explanation

- `.filter(df.Key.isin(["Name", "Age"]))`` → Keeps only rows with `Key` **"Name"** or **"Age"**.
- `.pivot("Key", ["Name", "Age"])`` → Only creates columns for "Name" and "Age".
- `.agg(first("Value"))`` → Gets the first value for each pivoted column.

♦ The `"city"` key is **ignored** because we filtered it out.

Would you like a solution that dynamically selects keys based on a condition? 🚀

ChatGPT kan begå misstag. Kontrollera viktig information.