RDB & SQL

Session 5

Advanced Grouping Operations

I've completed the **pre-class** content?

Students choose an option

Pear Deck Interactive Slide
Do not remove this bar

No Multiple Choice Response
You didn't answer this question

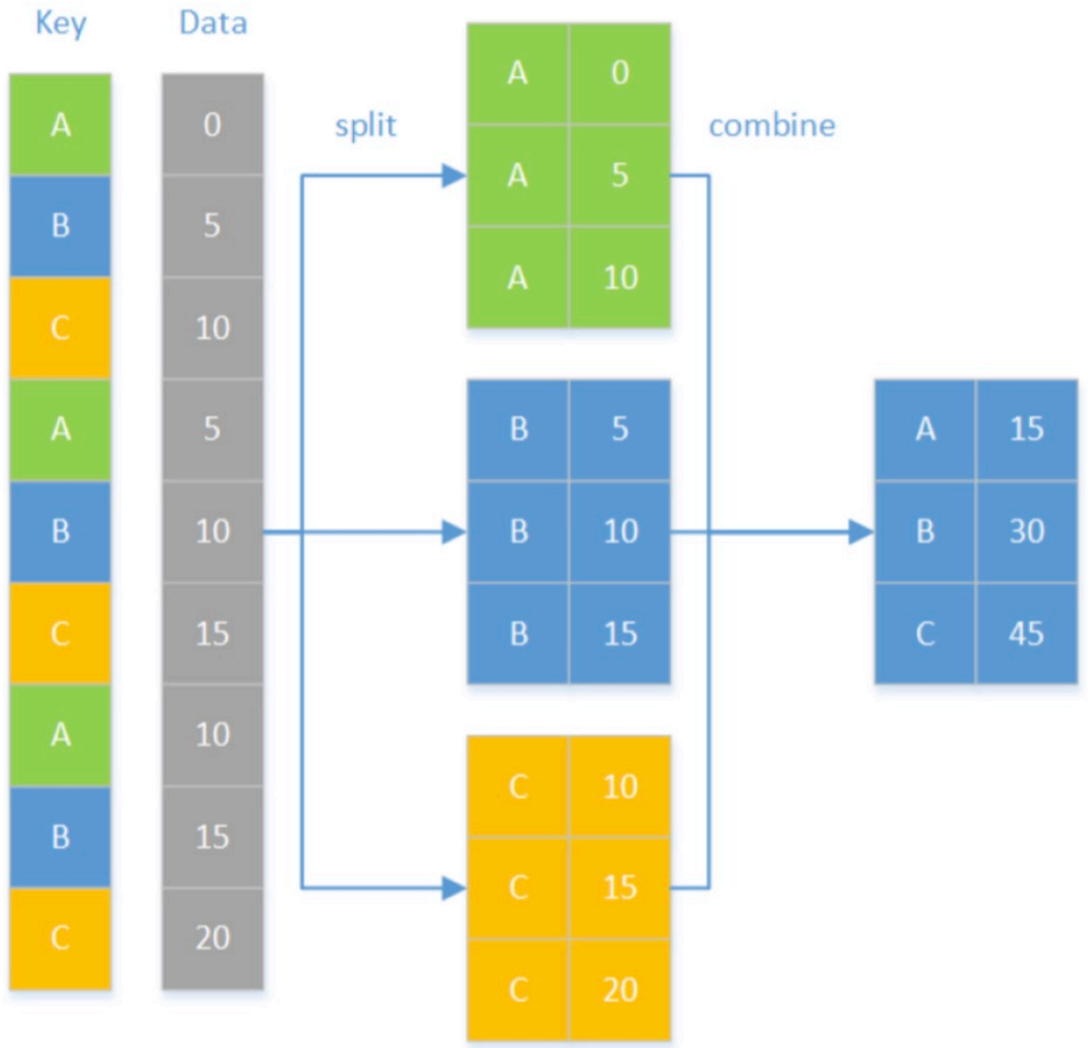
Table of Contents

GROUPING

+

AGGREGATION

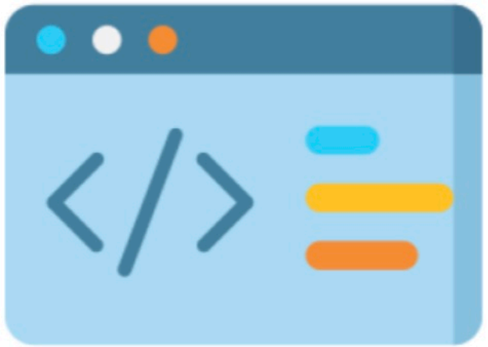
=



Having Clause

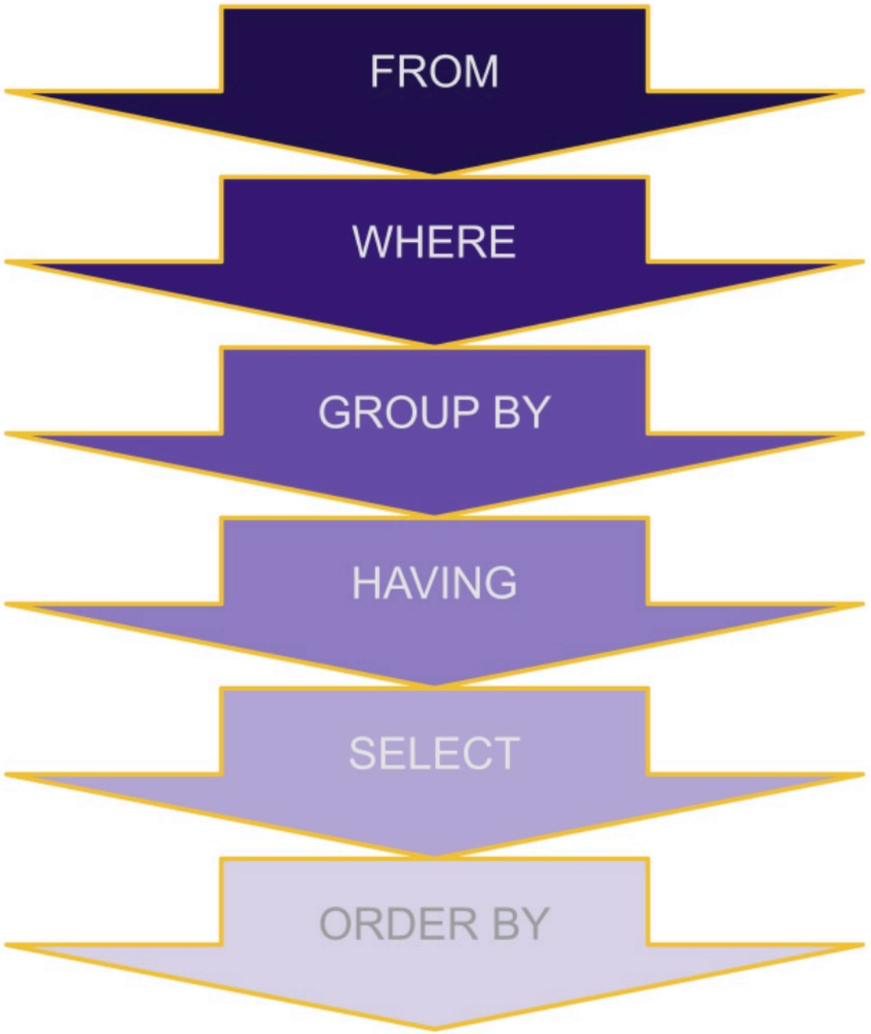
Introduction

- The GROUP BY clause groups rows into summary rows or groups.
- The HAVING clause filters groups on a specified condition.
- You have to use the HAVING clause with the GROUP BY.
- Otherwise, you will get an error.



Syntax

```
1  SELECT column_1, aggregate_function(column_2)
2  FROM table_name
3  GROUP BY column_1
4  HAVING search_condition;
```




```sql
SELECT  brand_name, AVG(list_price) AS avg_list_price
FROM    product.product A, product.brand B
WHERE   A.brand_id = B.brand_id
AND     A.model_year > 2016
GROUP BY
        brand_name
HAVING  AVG(list_price) > 1000
ORDER BY
        AVG(list_price) ASC;
```

Using WHERE, the fields to be grouped over the filtered rows are determined and a new field is created with the aggregate operation.

WHERE is taken into account at an earlier stage of a query execution, filtering the rows read from the tables.

And then HAVING is used if you want to filter the newly created field within the same query.

Example:

Instructor Table

	id	name	dept_name	salary
1	10238	Eric	Economics	72000
2	13378	Karl	Music	42000
3	23493	Jason	Philosophy	45000
4	26299	Jane	Computer Science	91000
5	30766	Jack	Economics	68000
6	40284	Mary	Psychology	78000
7	43087	Brian	Physics	93000
8	53695	Richard	Philosophy	54000
9	58248	Joseph	Political Science	58000
10	63172	David	Art History	65000
11	64378	Elvis	Physics	87000
12	96945	John	Computer Science	80000
13	99231	Santosh	Computer Science	74000

Example

Average salary for each department

```
1  SELECT dept_name, AVG(salary) AS avg_salary
2  FROM department
3  GROUP BY dept_name;
4
```

output:

```
1   dept_name    avg_salary
2   -----------  ----------
3   Art History  65000.0
4   Computer Sc  81666.6
5   Economics    70000.0
6   Music        42000.0
7   Philosophy   49500.0
8   Physics      90000.0
9   Political S  58000.0
10  Psychology   78000.0
```

Departments with an average salary greater than 50000

query:

```
1  SELECT dept_name, AVG(salary) AS avg_salary
2  FROM department
3  GROUP BY dept_name
4  HAVING avg_salary > 50000;
5
```

output:

```
1  dept_name    avg_salary
2  -----------  ----------
3  Art History  65000.0
4  Computer Sc  81666.6
5  Economics    70000.0
6  Physics      90000.0
7  Political S  58000.0
8  Psychology   78000.0
```

Query Time

Write a query that checks if any product id is duplicated in product table.

Query Time

Write a query that returns category ids with conditions max list price above 4000 or a min list price below 500.

Expected Output:

Results | Messages

	category_id	max_price	min_price
1	1	16999.95	3.00
2	4	3989.99	1.00
3	5	1799.99	23.99
4	6	359.99	29.99
5	7	2499.00	81.99
6	8	499.95	499.95
7	9	78.99	55.95
8	10	349.95	232.99
9	11	799.98	33.99
10	13	2199.99	49.99
11	14	349.99	39.99
12	15	119.66	75.99
13	16	2399.98	234.99

(local) (15.0 RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 13 rows

Query Time For You

Find the average product prices of the brands. Display brand name and average prices in descending order.

Expected Output:

Results | Messages

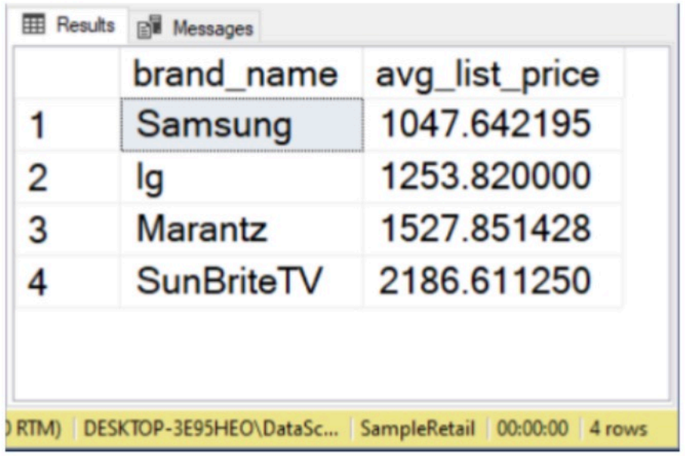
	brand_name	avg_list_price
1	SunBriteTV	2186.611250
2	Marantz	1527.851428
3	lg	1253.820000
4	Samsung	1047.642195
5	svs	933.315000
6	Dell	803.303750
7	BenQ	649.330000
8	Acer	644.450000
9	Sony	583.450000
10	Sennheiser	580.310000
11	Asus	534.462142
12	Apple	527.851875
13	Definitive	493.317500

5.0 RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 40 rows

Query Time For You

Write a query that returns the list of brands whose average product prices are more than 1000

Expected Output:


Results Messages

	brand_name	avg_list_price
1	Samsung	1047.642195
2	lg	1253.820000
3	Marantz	1527.851428
4	SunBriteTV	2186.611250

RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 4 rows

Query Time For You

Write a query that returns the list of each order id and that order's total net amount (please take into consideration of discounts and quantities)

Expected Output:

Results | Messages

	order_id	net_price
1	1	1978.1464
2	2	1370.5907
3	3	806.5405
4	4	246.5820
5	5	529.6224
6	6	2539.2541
7	7	623.9127
8	8	487.7719
9	9	107.9820
10	10	224.9910
11	11	1096.5725
12	12	302.0725

3E95HEO\DataSc... | SampleRetail | 00:00:00 | 1.615 rows

Query Time

Question: Write a query that returns monthly order counts of the States.

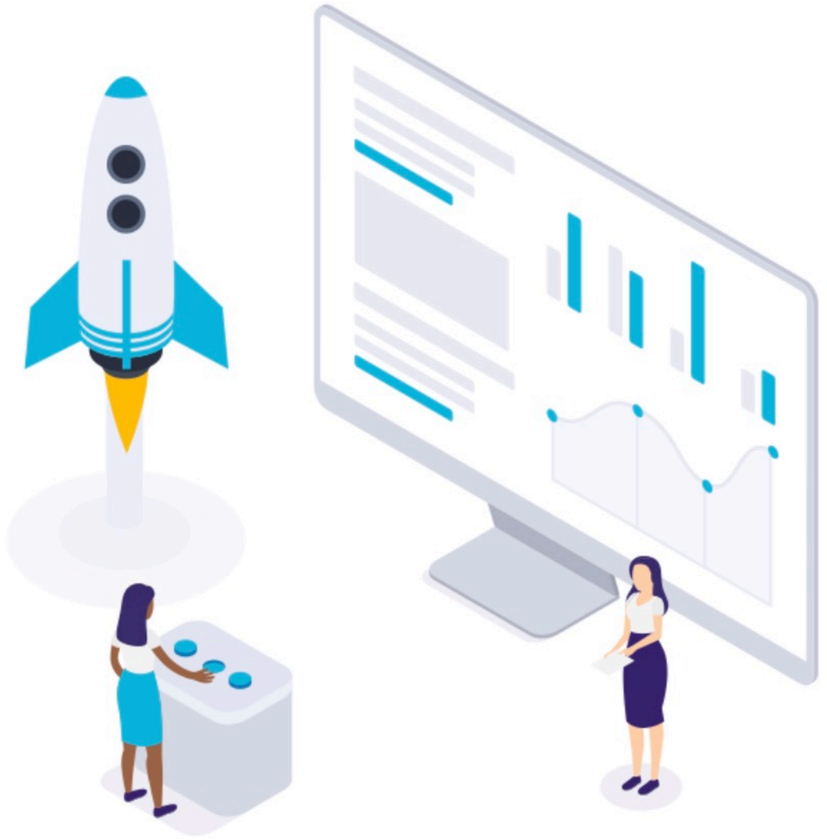
Expected Output:

Results | Messages

	state	YEARS	months	NUM_OF_ORDERS
1	AK	2018	1	3
2	AK	2018	2	2
3	AK	2018	6	1
4	AK	2018	7	1
5	AK	2018	8	1
6	AK	2018	9	2
7	AK	2018	10	1
8	AK	2018	11	1
9	AK	2018	12	1

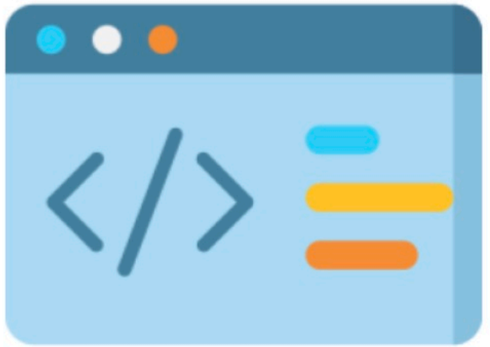
Query exec... | (local) (15.0 RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 576 rows

2 Grouping Sets



Introduction

- The GROUP BY operator groups all specified fields together for the aggregate operation.
- However, using GROUPING SETS, individual grouping can be done for each of the fields in the parentheses.



Syntax

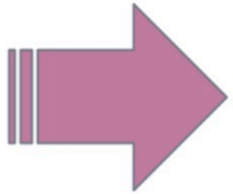
```
SELECT
    column1,
    column2,
    aggregate_function (column3)
FROM
    table_name
GROUP BY
    GROUPING SETS (
        (column1, column2),
        (column1),
        (column2),
        ()
);
```

In order to reach meaningful summary statistics you can use GROUPING SETS;

- When you cannot decide which field to use,
- When you want to see the results of different grouping combinations

Example

```
...
GROUP BY
GROUPING SETS
(
(Product),
(Product, Size)
)
```



Product	Size	Sum_Sales
Jean	Null	81
Jean	Small	13
Jean	Medium	55
Jean	Large	13
T-Shirt	Null	66
T-Shirt	Small	30
T-Shirt	Medium	20
T-Shirt	Large	16
Jacket	Null	28
Jacket	Small	11
Jacket	Medium	13
Jacket	Large	4

Query Time

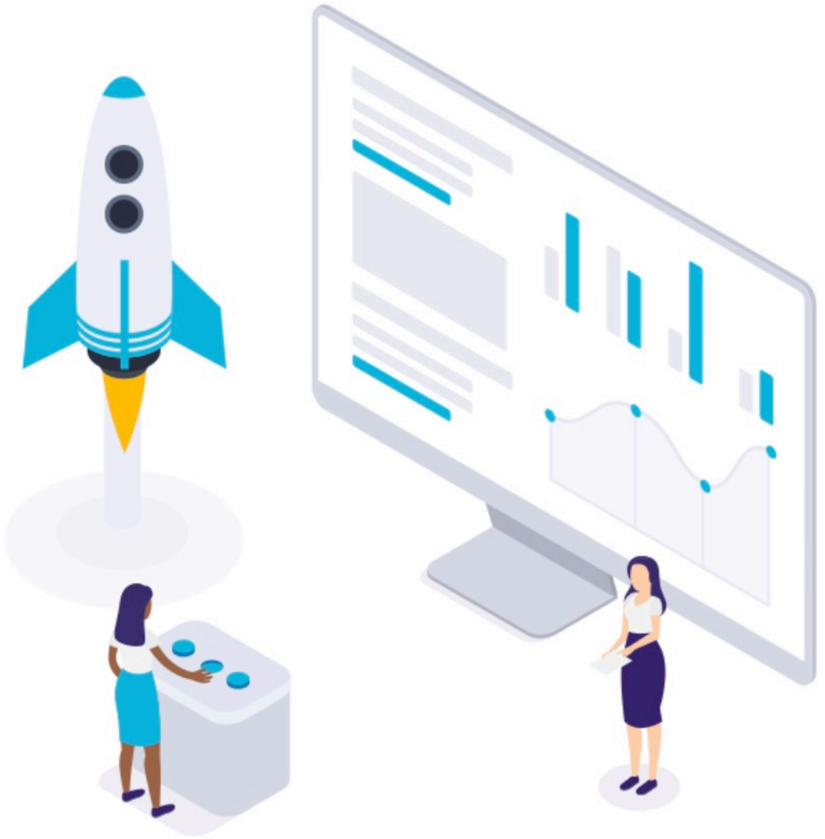
Summary Table:

Results | Messages

	Brand	Category	Model_Year	total_sales_price
1	Acer	Computer Accessories	2018	546.0000
2	Acer	Computer Accessories	2019	1396.0000
3	Acer	Computer Accessories	2021	24381.0000
4	Alpine	Car Electronics	2018	2255.0000
5	Alpine	Speakers	2018	2643.0000
6	Alpine	Speakers	2019	128.0000
7	Alpine	Speakers	2021	9782.0000
8	Apple	Computer Accessories	2018	6150.0000
9	Apple	Computer Accessories	2019	9075.0000
10	Apple	Computer Accessories	2021	39423.0000
11	Apple	mp4 player	2018	24646.0000
12	Asus	Computer Accessories	2018	1737.0000
13	Asus	Computer Accessories	2019	3274.0000
14	Asus	Computer Accessories	2021	5490.0000

Query executed successfully. | (local) (15.0 RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 139 rows

4 Rollup

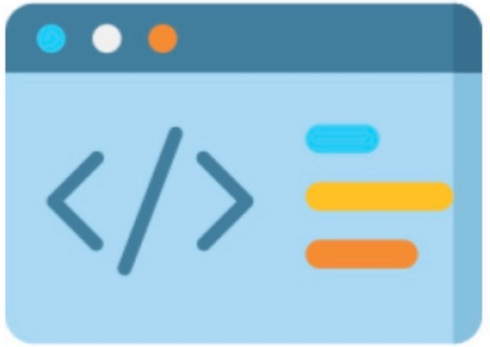


Introduction

ROLLUP operator creates a group for each combination of column expressions.

It makes grouping combinations by subtracting one at a time from the column names written in parentheses, in the order from right to left.

Therefore, the order in which the columns are written is important.



Syntax

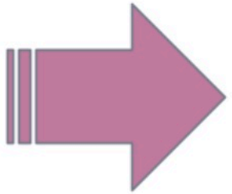
```sql
SELECT
    d1,
    d2,
    d3,
    aggregate_function(c4)
FROM
    table_name
GROUP BY
    ROLLUP (d1, d2, d3);
```

Groups:

- d1, d2, d3
- d1, d2, NULL
- d1, NULL, NULL
- NULL, NULL, NULL

Example

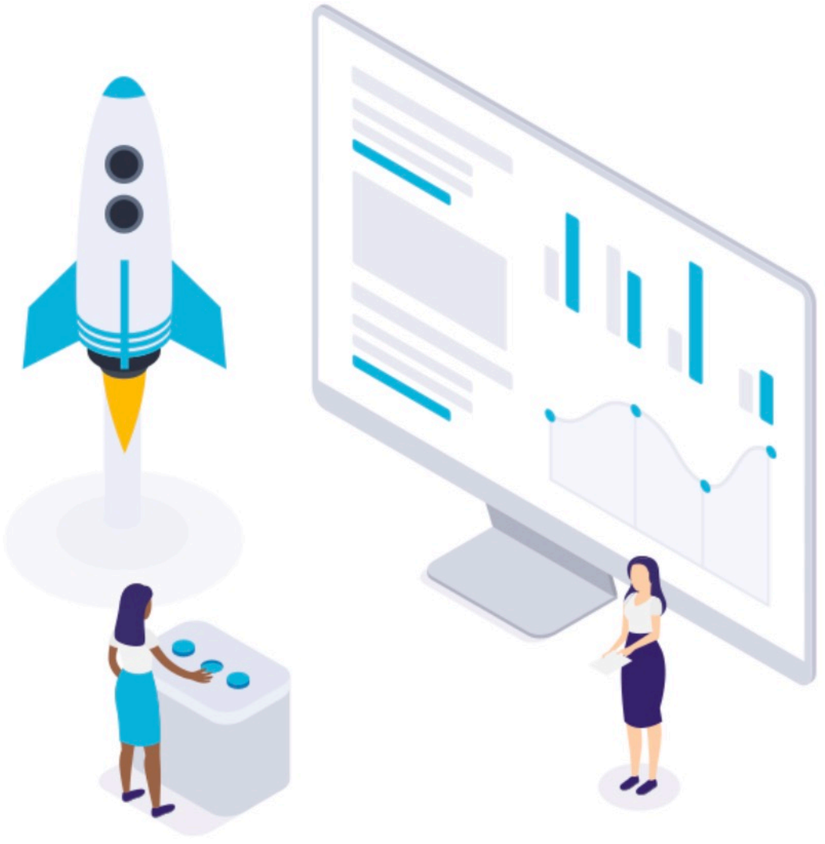
...
WHERE Size = 'Medium'
GROUP BY
ROLLUP (Product, Size, Color)



Product	Size	Color	Sum_Sales
Jean	Medium	Blue	40
Jean	Medium	Black	15
T-Shirt	Medium	White	12
T-Shirt	Medium	Red	8
Jacket	Medium	Black	8
Jacket	Medium	Green	5
Jean	Medium	Null	55
T-Shirt	Medium	Null	20
Jacket	Medium	Null	13
Jean	Null	Null	55
T-Shirt	Null	Null	20
Jacket	Null	Null	13
Null	Null	Null	88

Query Time

5 Cube



Introduction

CUBE operator makes all possible grouping combinations for all fields specified in the select operator.

The order in which the columns are written is not important.

Syntax

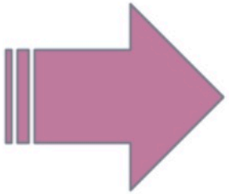
```sql
SELECT
    d1,
    d2,
    d3,
    aggregate_function (c4)
FROM
    table_name
GROUP BY
    CUBE (d1, d2, d3);
```

Groups:

- d1, d2, d3
- d1, d2, NULL
- d1, d3, NULL
- d2, d3, NULL
- d1, NULL, NULL
- d2, NULL, NULL
- d3, NULL, NULL
- NULL, NULL, NULL

Example

...
WHERE Product = 'Jean'
GROUP BY
Cube (Product, Size, Color)



Product	Size	Color	Sum_Sales
Jean	Small	Blue	5
Jean	Small	Black	8
Jean	Medium	Blue	40
Jean	Medium	Black	15
Jean	Large	Blue	7
Jean	Large	Black	6
Jean	Small	Null	13
Jean	Medium	Null	55
Jean	Large	Null	13
Jean	Null	Blue	52
Jean	Null	Black	29
Null	Small	Blue	5
Null	Small	Black	8
Null	Medium	Blue	40
Null	Medium	Black	15
Null	Large	Blue	7
Null	Large	Black	6
Jean	Null	Null	81
Null	Small	Null	13
Null	Medium	Null	55
Null	Large	Null	13
Null	Null	Blue	52
Null	Null	Black	29
Null	Null	Null	81

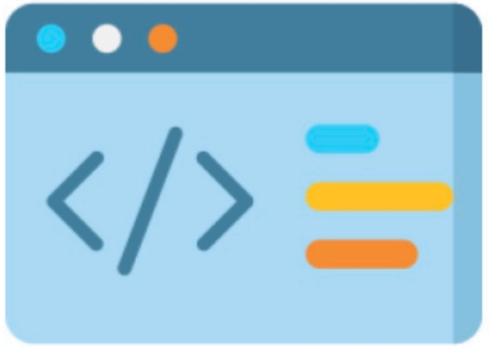
Query Time

3 Pivot

Introduction

The PIVOT operator converts the unique observations seen in the results table into fields and specifies the corresponding aggregate values as rows.

With the PIVOT operator, **GROUP BY is not used.** The logic setup of both operators is completely different.



You follow these steps to make a query a pivot table:

1. First, select a base dataset for pivoting.
2. Second, create a temporary result by using a derived table or common table expression (CTE)
3. Third, apply the PIVOT operator.

Example

Product	Sum_Sales
Rose	53
T-Shirt	81
Cup	15
Teddy Bear	7
Book	28

Rose	T-Shirt	Cup	Teddy Bear	Book
53	81	15	7	28

Syntax

```
SELECT [column_name], [pivot_value1], [pivot_value2], ...[pivot_value_n]
FROM
table_name
PIVOT
(
 aggregate_function(aggregate_column)
 FOR pivot_column
 IN ([pivot_value1], [pivot_value2], ... [pivot_value_n])
) AS pivot_table_name;
```

In this syntax;

[column_name] .....

[pivot value] .....

[aggregate_function] .....

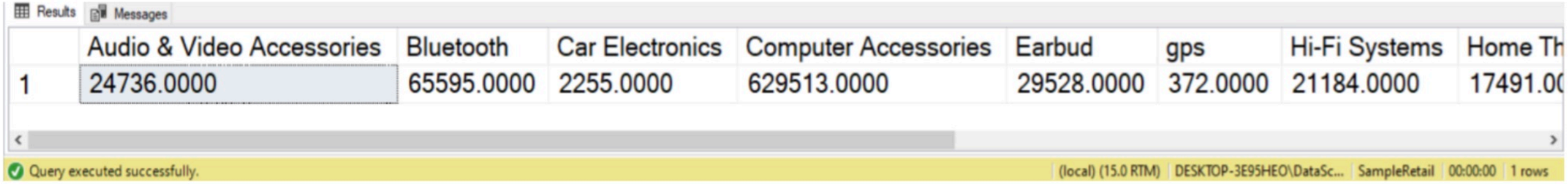
[aggregate column] .....

[pivot column] .....

Query Time For You

Question: Write a query using summary table that returns the total turnover from each category by model year. (in pivot table format)

Expected Output:



Results | Messages

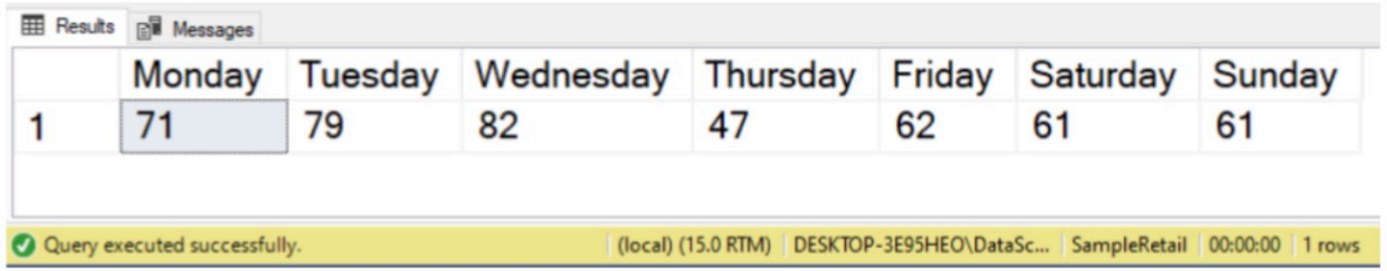
	Audio & Video Accessories	Bluetooth	Car Electronics	Computer Accessories	Earbud	gps	Hi-Fi Systems	Home Th
1	24736.0000	65595.0000	2255.0000	629513.0000	29528.0000	372.0000	21184.0000	17491.0(

Query executed successfully. | (local) (15.0 RTM) | DESKTOP-3E95HEO\DataSc... | SampleRetail | 00:00:00 | 1 rows

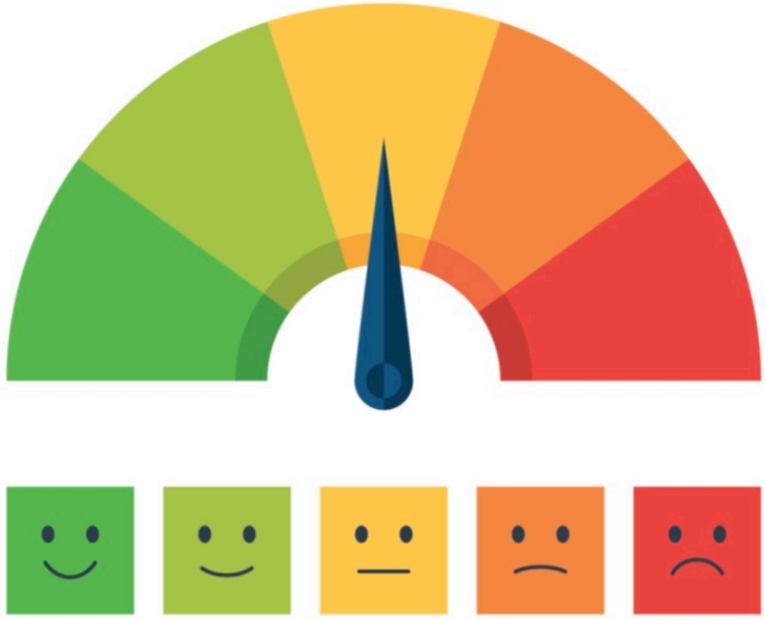
Query Time For You

Question: Write a query that returns count of the orders day by day in a pivot table format that has been shipped two days later.

Expected Output:



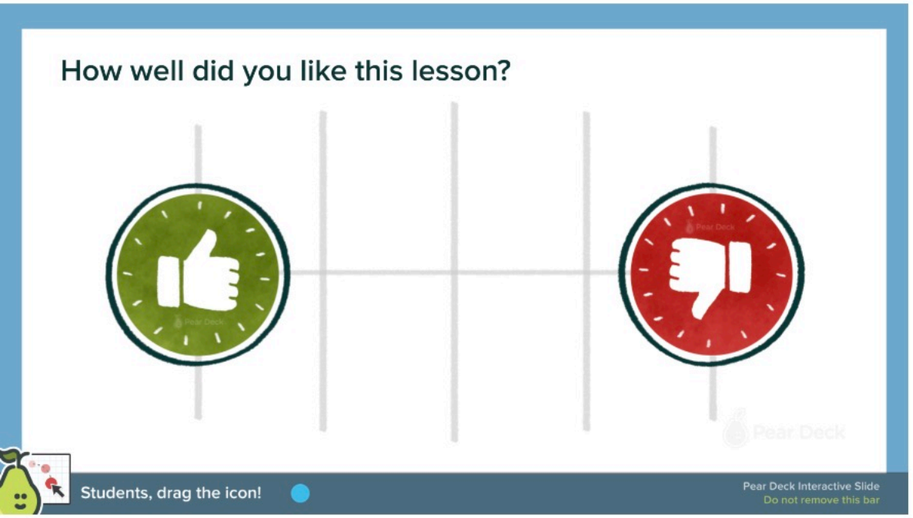
Is everything clear so far?



Students choose an option

Pear Deck Interactive Slide
Do not remove this bar

No Multiple Choice Response
You didn't answer this question



No Draggable™ Response

You didn't answer this question