



Работа с СУБД **SQL**

Агрегатные функции.

• REC Проверить, идет ли запись

**Меня хорошо видно
&& слышно?**



Проверка


- настройка микрофона и аудио
- проверка работы чата





Агрегатные функции

Агрегатные функции



Производят подсчет на основе нескольких показателей (все строки одного столбца) и возвращают атомарный результат (число).

Агрегатные функции

Производят подсчет на основе нескольких показателей (все строки одного столбца) и возвращают атомарный результат (число).

Основные аналитические функции:

- **SUM:** сумма значений столбца

Агрегатные функции

Производят подсчет на основе нескольких показателей (все строки одного столбца) и возвращают атомарный результат (число).

Основные аналитические функции:

- **SUM:** сумма значений столбца
- **COUNT:** количество строк или ненулевых значений в столбце

Агрегатные функции

Производят подсчет на основе нескольких показателей (все строки одного столбца) и возвращают атомарный результат (число).

Основные аналитические функции:

- **SUM:** сумма значений столбца
- **COUNT:** количество строк или ненулевых значений в столбце
- **AVG:** среднее значение столбца

Агрегатные функции

Производят подсчет на основе нескольких показателей (все строки одного столбца) и возвращают атомарный результат (число).

Основные аналитические функции:

- **SUM:** сумма значений столбца
- **COUNT:** количество строк или ненулевых значений в столбце
- **AVG:** среднее значение столбца
- **MAX:** максимальное значение столбца

Агрегатные функции: пример

Подсчитать общую сумму покупок.

```
SELECT Sum(sales_amount) AS  
total_sales  
FROM sales;
```


ID	Date	Sales_amount
1	19/05/2023	10.45
2	20/05/2023	250
3	21/05/2023	67.20

СМОТРИ В БУДУЩЕЕ. ИНВЕСТИРУЙ В
ЗНАНИЯ.



GROUP BY

GROUP BY



Разбивает таблицу по строкам на несколько групп. Группы определяются значениями одного или нескольких столбцов. Аналитические функции считаются внутри каждой группы отдельно.

- Раздел GROUP BY позволяет выполнять группировку строк таблиц по определённым критериям.
- GROUP BY почти всегда используется вместе с функциями агрегирования

Синтаксис:

GROUP BY: пример 1

Подсчитать общую сумму покупок по каждой категории товаров.

```
SELECT product_category,  
       Sum(sales_amount) AS  
total_sales  
FROM   sales  
GROUP BY product_category;
```

ID	Product_category	Sales_amount
1	Food	10.45
2	Beverages	250
3	Food	67.20

GROUP BY: пример 1

Подсчитать общую сумму покупок по каждой категории товаров.

```
SELECT product_category,  
       Sum(sales_amount) AS  
total_sales  
FROM   sales  
GROUP BY product_category;
```

ID	Product_category	Sales_amount
1	Food	10.45
2	Beverages	250
3	Food	67.20

Product_category	total_sales
Food	77.65
Beverages	250

GROUP BY: пример 2



Какой результат выдаст запрос?

```
SELECT client,  
       date,  
       Sum(sales_amount)  
FROM   sales  
GROUP BY client
```

ID	client	sales_amount	date
1	Петр	102	2012-10-23
2	Василий	47	2012-10-27
3	Мария	18	2012-11-05
4	Мария	20	2012-11-14
5	Петр	160	2012-12-03

GROUP BY: пример 3



Как будет выглядеть следующий запрос?

Подсчитать количество покупок по каждому клиенту, который совершил как минимум 2 покупки.

ID	client	sales_amount	date
1	Петр	102	2012-10-23
2	Василий	47	2012-10-27
3	Мария	18	2012-11-05
4	Мария	20	2012-11-14
5	Петр	160	2012-12-03

<https://www.db-fiddle.com/f/kVSgXux6384wV7LUZJwnQn/9>

GROUP BY + HAVING



Как будет выглядеть следующий запрос?

Подсчитать количество покупок по каждому клиенту, который совершил как минимум 2 покупки.

```
SELECT client,  
       Count(*) AS order_count  
FROM   sales  
GROUP BY client  
HAVING order_count >= 2;
```

ID	client	sales_amount	date
1	Петр	102	2012-10-23
2	Василий	47	2012-10-27
3	Мария	18	2012-11-05
4	Мария	20	2012-11-14
5	Петр	160	2012-12-03

Функции агрегирования

```
агрегатная_функция ( [ ALL | DISTINCT ] выражение [ , ... ]  
    [ ORDER BY предложение_order_by ] )  
    [ FILTER ( WHERE условие_фильтра ) ]
```

```
агрегатная_функция ( * )  
    [ FILTER ( WHERE условие_фильтра ) ]
```

```
агрегатная_функция ( [ выражение [ , ... ] ] ) WITHIN GROUP ( ORDER BY предложение_order_by )  
    [ FILTER ( WHERE условие_фильтра ) ]
```

Важно!

1. Функции агрегирования не работают со значениями NULL.
2. Раздел WHERE не допускает использования функций агрегирования.

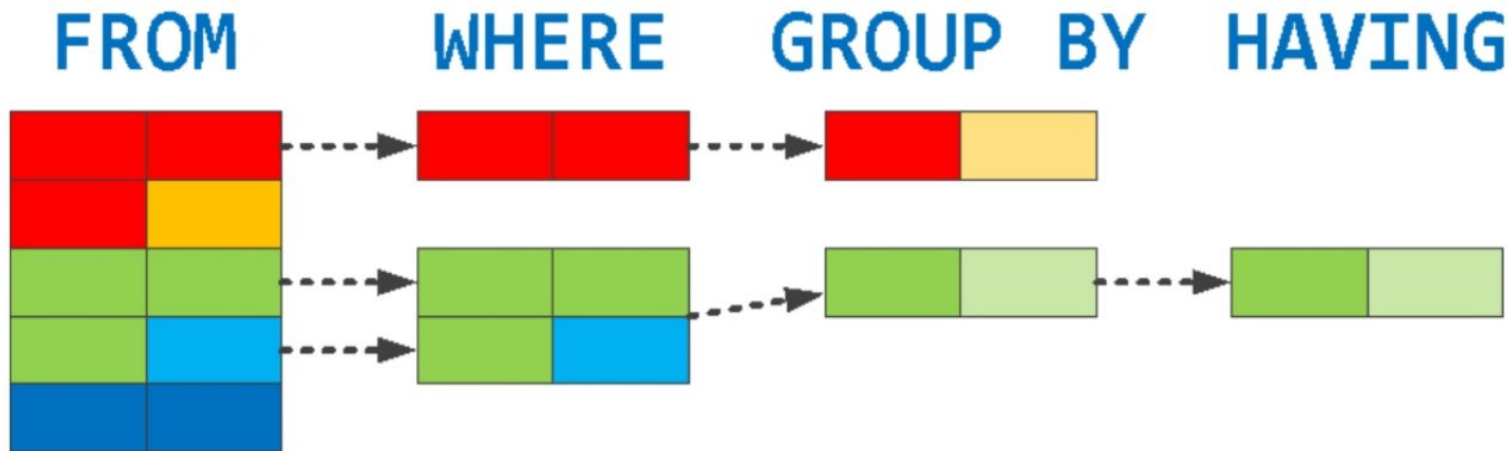
AVG(<поле>)	Среднее значение для указанного столбца или выражения
COUNT(<поле>)	Количество строк, исключая NULL-строки в указанном столбце
COUNT(*)	Общее количество строк, включая NULL-строки
MAX(<поле>)	Максимальное значение в указанном столбце
MIN(<поле>)	Минимальное значение в указанном столбце
SUM(<поле>)	Сумма всех значений в указанном столбце
STDEV(<поле>)	Статистическое стандартное отклонение для значений столбца
VAR(<поле>)	Несмещенная оценка дисперсии величин указанного столбца

<https://www.db-fiddle.com/f/kVSgXux6384wV7LUZJwnQn/10>

GROUP BY + HAVING

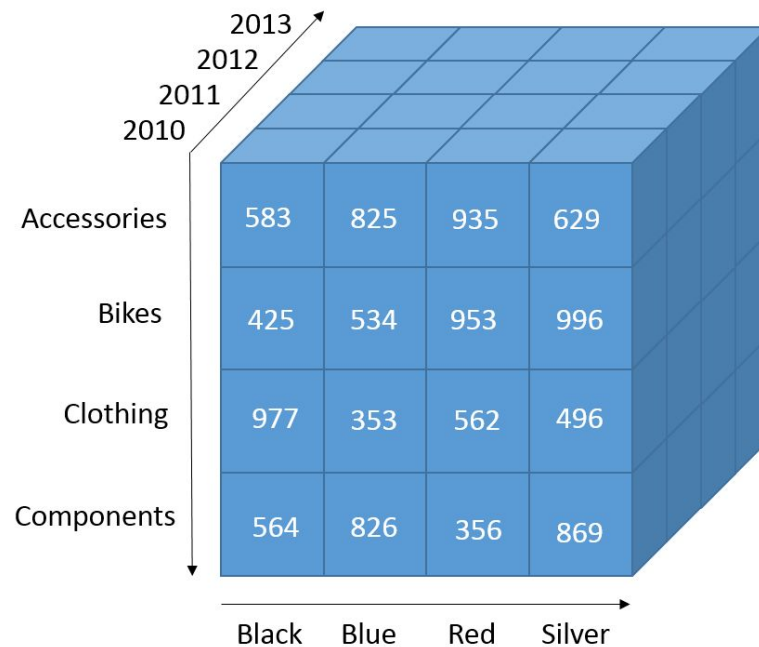
HAVING фильтрует строки внутри каждой группы GROUP BY. Строки фильтруются в соответствии с условием к агрегированному значению (sum, count, avg)

Без HAVING нужно было бы написать несколько запросов.



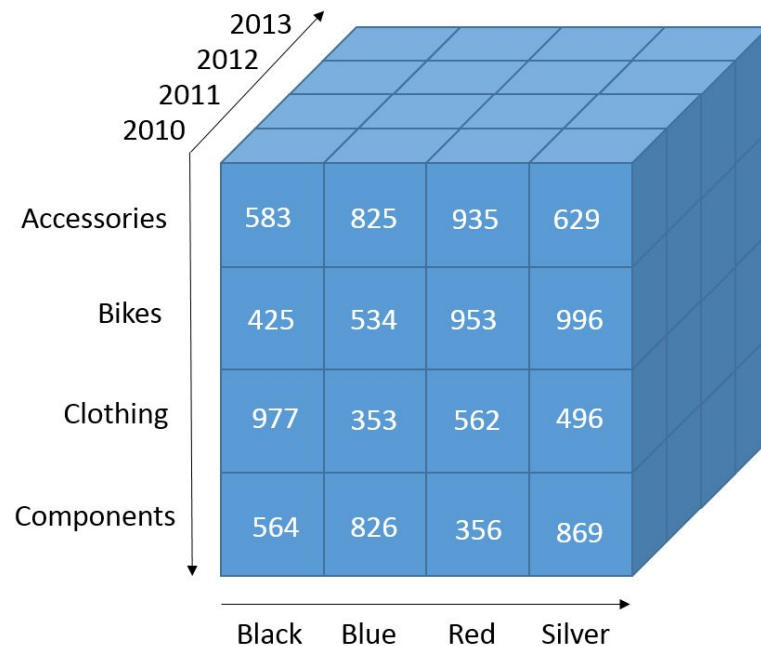
OLAP CUBE

- Метаданные над данными в схеме “звезда” или “снежинка”



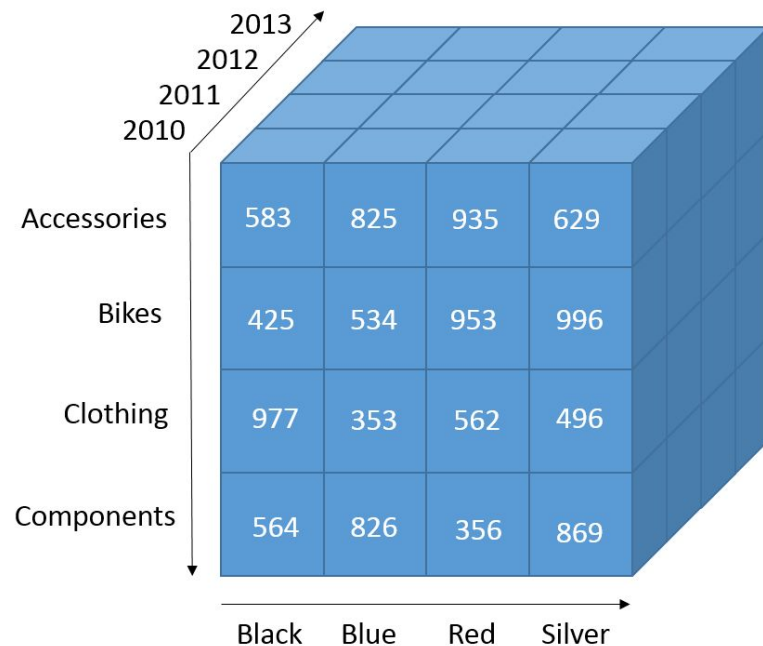
OLAP CUBE

- Метаданные над данными в схеме “звезда” или “снежинка”
- Содержат предвычисления агрегаций и статистик (COUNT, SUM, AVG, MIN, MAX) по разным измерениям



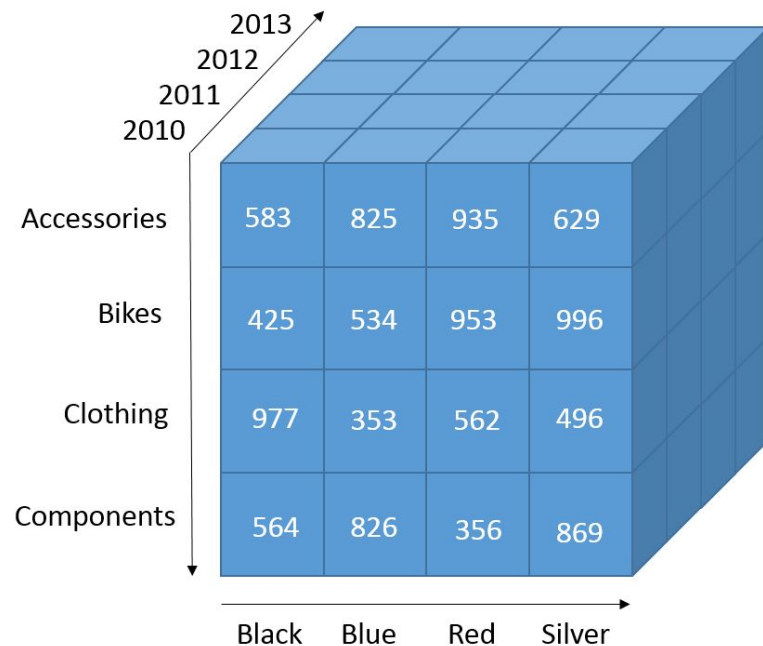
OLAP CUBE

- Метаданные над данными в схеме “звезда” или “снежинка”
- Содержат предвычисления агрегаций и статистик (COUNT, SUM, AVG, MIN, MAX) по разным измерениям
- Ускоряют аналитические запросы



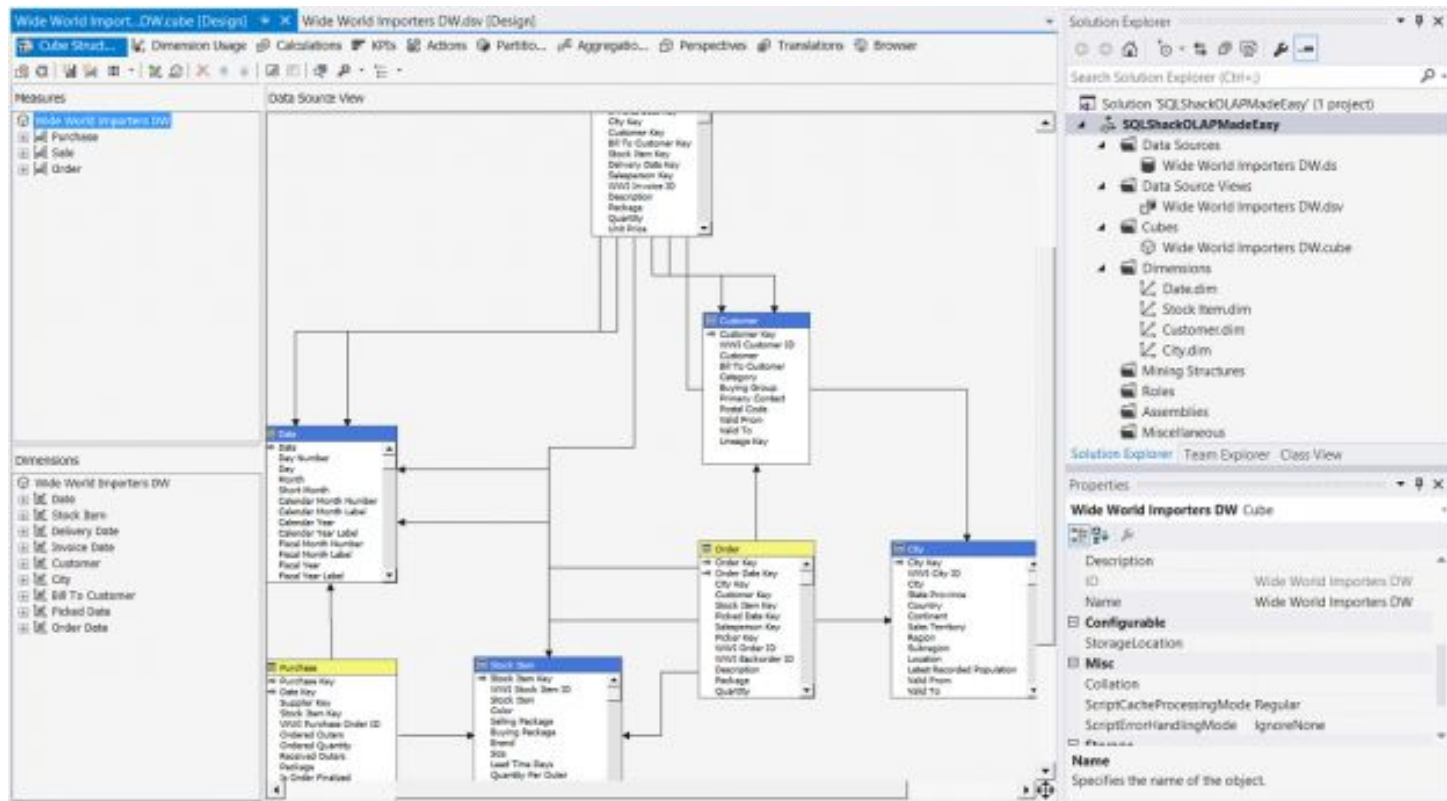
OLAP CUBE

- Метаданные над данными в схеме “звезда” или “снежинка”
- Содержат предвычисления агрегаций и статистик (COUNT, SUM, AVG, MIN, MAX) по разным измерениям
- Ускоряют аналитические запросы
- Позволяют анализировать данные под разным углом (dimensions)



OLAP CUBE: создание

Через графический интерфейс (Microsoft SQL Server):



OLAP Cube: SQL-оператор в Oracle

SELECT

department, gender

sum(salary) as Salary_Sum

FROM employee

GROUP BY CUBE (department, gender)

Column name	Type
id	INT
name	VARCHAR
gender	VARCHAR
salary	INT
department	VARCHAR

OLAP Cube: SQL-оператор в Oracle

SELECT

```
department, gender  
  
sum(salary) as Salary_Sum  
  
FROM employee  
  
GROUP BY CUBE (department, gender)
```

Даст сумму зарплат:

1. По департаменту и полу (\approx GROUP BY department, gender)
2. По департаменту (\approx GROUP BY department)
3. По полу (\approx GROUP BY gender)
4. По всей таблице (без GROUP BY)

Column name	Type
id	INT
name	VARCHAR
gender	VARCHAR
salary	INT
department	VARCHAR

OLAP Cube: SQL-оператор в PostgreSQL

SELECT

department, gender

sum(salary) as Salary_Sum

FROM employee

GROUP BY CUBE (department, gender)

<https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-cube/>

Column name	Type
id	INT
name	VARCHAR
gender	VARCHAR
salary	INT
department	VARCHAR

Задача

На какую сумму организации приобрели товары по годам?


```
select OrgName, year(n.Dat) as [Год],  
       sum(Amount*Price) as [Сумма]  
from Org o, Nakl n, SostNakl sn  
where o.Org_ID=n.Org_ID  
      and n.Nakl_ID=sn.Nakl_ID  
      and n.InOut=' - '  
group by OrgName, year(Dat)  
order by OrgName, year(Dat);
```



Задача

На какую сумму организации приобрели товары по годам с суммой покупки более 1000?

```
select OrgName, year(n.Dat) as [Год],  
       sum(Amount*Price) as [Сумма]  
from Org o, Nakl n, SostNakl sn  
where o.Org_ID=n.Org_ID  
       and n.Nakl_ID=sn.Nakl_ID  
       and n.InOut=' - '  
group by OrgName, year(Dat)  
having sum(Amount*Price)>1000  
order by OrgName, year(Dat);
```



Задача

Другое решение:

```
SELECT t.Tovar_ID, t.TovarName, p.Price, p.DateStart  
FROM Tovar t  
left join PriceList p  
      on t.Tovar_ID=p.Tovar_ID  
where t.IsTovar=1  
      and DateStart<=getdate()  
      and (DateEnd is null or DateEnd>=getdate())  
ORDER BY TovarName;
```



Задача

оператор SELECT используется в списке извлекаемых полей:

```
SELECT Tovar.Tovar_ID, Tovar.TovarName,  
      (SELECT Price  
       FROM PriceList  
       WHERE Tovar.Tovar_ID=PriceList.Tovar_ID  
        and DateStart<=getdate()  
        and (DateEnd is null or DateEnd>=getdate()))  
      LIMIT 1) as Price  
FROM Tovar  
where IsTovar=1  
ORDER BY TovarName;
```



Задача

Получить суммы закупок организаций по месяцам 2024г. для организаций, для которых сумма закупок превысила 1000 р. Результатом должна явиться таблица с полями Организация, Месяц, Сумма закупок.

Обратите внимание, что во фразах GROUP BY, HAVING использование псевдонимов полей не допускается.



Задача

Решение:

```
SELECT OrgName, month(Nakl.Dat) AS Mon,  
       SUM(SostNakl.Amount*SostNakl.Price) AS Summa  
FROM Org, Nakl, SostNakl  
WHERE Org.Org_ID=Nakl.Org_ID  
       and Nakl.Nakl_ID=SostNakl.Nakl_ID  
       and Nakl.InOut='-'  
       and year(Nakl.Dat)=2024  
GROUP BY OrgName, month(Nakl.Dat)  
HAVING SUM(SostNakl.Amount*SostNakl.Price)>1000  
ORDER BY OrgName, month(Nakl.Dat);
```





СПАСИБО ЗА ВНИМАНИЕ!

#аис
#учисьваис