# Группировка. Агрегатные функции

Агрегатные функции выполняют вычисления над значениями в наборе строк. В T-SQL имеются следующие агрегатные функции:

* **AVG**: находит среднее значение
* **SUM**: находит сумму значений
* **MIN**: находит наименьшее значение
* **MAX**: находит наибольшее значение
* **COUNT**: находит количество строк в запросе

В качестве аргумента все агрегатные функции принимают выражение, которое представляет критерий дя определения значений. Зачастую, в качестве выражения выступает название столбца, над значениями которого надо проводить вычисления.

Выражения в функциях **AVG** и **SUM** должно представлять числовое значение. Выражение в функциях **MIN**, **MAX** и **COUNT** может представлять числовое или строковое значение или дату.

Все агрегатные функции за исключением COUNT(\*) игнорируют значения NULL.

### Avg

Функция **Avg** возвращает среднее значение на диапазоне значений столбца таблицы.

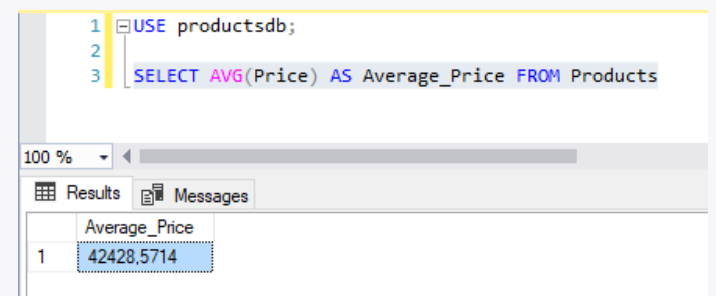
Пусть в базе данных у нас есть таблица товаров Products, которая описывается следующими выражениями (Создать БД Products\_Ivanova):



Найдем среднюю цену товаров из базы данных:



Для поиска среднего значения в качестве выражения в функцию передается столбец Price. Для получаемого значения устанавливается псевдоним Average\_Price, хотя можно его и не устанавливать.



Также мы можем применить фильтрацию. Например, найти среднюю цену для товаров какого-то определенного производителя:



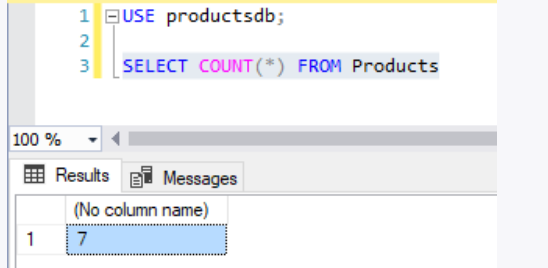
И, кроме того, мы можем находить среднее значение для более сложных выражений. Например, найдем среднюю сумму всех товаров, учитывая их количество:

### 

### Count

Функция **Count** вычисляет количество строк в выборке. Есть две формы этой функции. Первая форма COUNT(\*) подсчитывает число строк в выборке:





Вторая форма функции вычисляет количество строк по определенному столбцу, при этом строки со значениями NULL игнорируются:

### 

### Min и Max

Функции **Min** и **Max** возвращают соответственно минимальное и максимальное значение по столбцу. Например, найдем минимальную цену среди товаров:



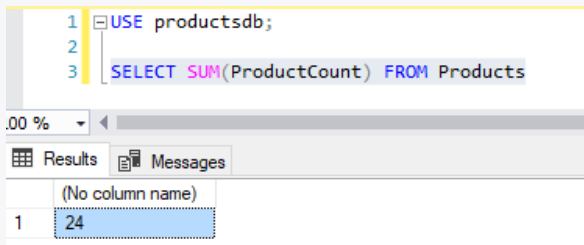
Поиск максимальной цены:



Данные функции также игнорируют значения NULL и не учитывают их при подсчете.

### Sum

Функция **Sum** вычисляет сумму значений столбца. Например, подсчитаем общее количество товаров:



Также вместо имени столбца может передаваться вычисляемое выражение. Например, найдем общую стоимость всех имеющихся товаров:

### 

### All и Distinct

По умолчанию все вышеперечисленных пять функций учитывают все строки выборки для вычисления результата. Но выборка может содержать повторяющие значения. Если необходимо выполнить вычисления только над уникальными значениями, исключив из набора значений повторяющиеся данные, то для этого применяется оператор **DISTINCT**.



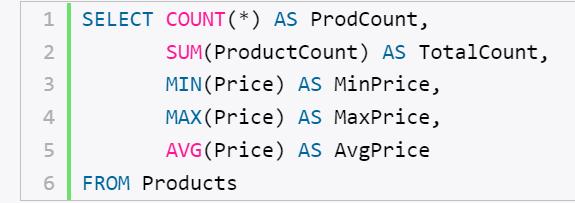
По умолчанию вместо DISTINCT применяется оператор **ALL**, который выбирает все строки:

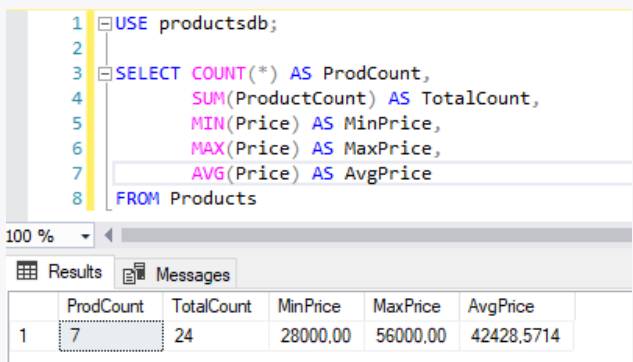


Так как этот оператор неявно подразумевается при отсутствии DISTINCT, то его можно не указывать.

### Комбинирование функций

Объединим применение нескольких функций:





## Операторы GROUP BY и HAVING

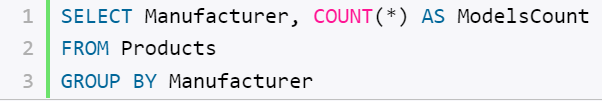
Для группировки данных в T-SQL применяются операторы **GROUP BY** и **HAVING**, для использования которых применяется следующий формальный синтаксис:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | SELECT столбцы  FROM таблица  [WHERE условие\_фильтрации\_строк]  [GROUP BY столбцы\_для\_группировки]  [HAVING условие\_фильтрации\_групп]  [ORDER BY столбцы\_для\_сортировки] |

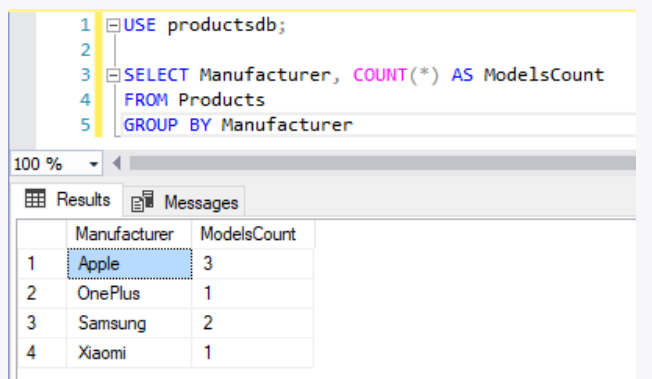
### GROUP BY

Оператор **GROUP BY** определяет, как строки будут группироваться.

Например, сгруппируем товары по производителю

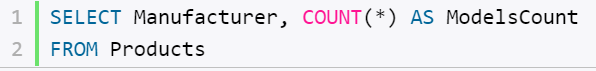


Первый столбец в выражении SELECT - Manufacturer представляет название группы, а второй столбец - ModelsCount представляет результат функции Count, которая вычисляет количество строк в группе.

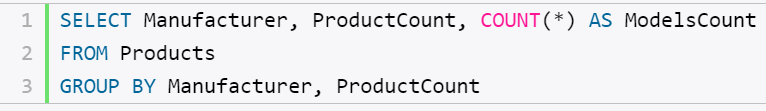


Стоит учитывать, что любой столбец, который используется в выражении SELECT (не считая столбцов, которые хранят результат агрегатных функций), должны быть указаны после оператора GROUP BY. Так, например, в случае выше столбец Manufacturer указан и в выражении SELECT, и в выражении GROUP BY.

И если в выражении SELECT производится выборка по одному или нескольким столбцам и также используются агрегатные функции, то необходимо использовать выражение GROUP BY. Так, следующий пример работать не будет, так как он не содержит выражение группировки:



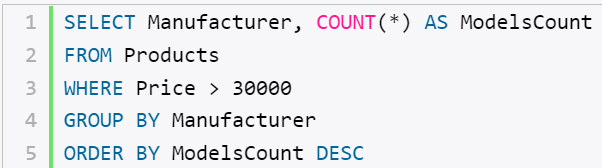
Другой пример, добавим группировку по количеству товаров:

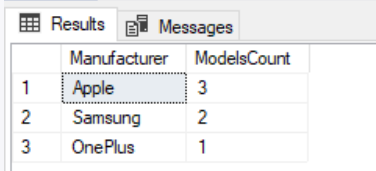


Оператор GROUP BY может выполнять группировку по множеству столбцов.

Если столбец, по которому производится группировка, содержит значение NULL, то строки со значением NULL составят отдельную группу.

Следует учитывать, что выражение GROUP BY должно идти после выражения WHERE, но до выражения ORDER BY:



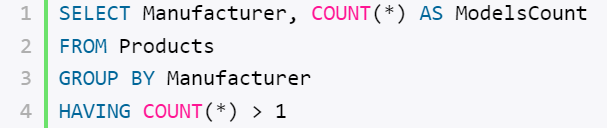


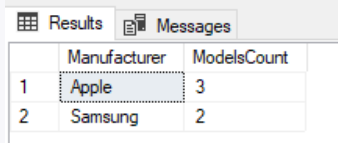
### Фильтрация групп. HAVING

Оператор **HAVING определяет, какие группы будут включены в выходной результат, то есть выполняет фильтрацию групп.**

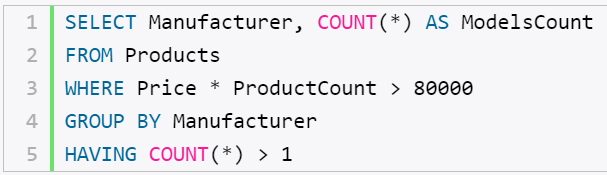
Применение HAVING во многом аналогично применению WHERE. Только есть WHERE применяется к фильтрации строк, то HAVING используется для фильтрации групп.

Например, найдем все группы товаров по производителям, для которых определено более 1 модели:



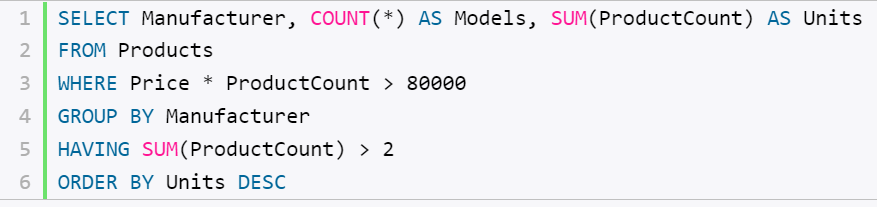


При этом в одной команде мы можем использовать выражения WHERE и HAVING:



То есть в данном случае сначала фильтруются строки: выбираются те товары, общая стоимость которых больше 80000. Затем выбранные товары группируются по производителям. И далее фильтруются сами группы - выбираются те группы, которые содержат больше 1 модели.

Если при этом необходимо провести сортировку, то выражение ORDER BY идет после выражения HAVING:



В данном случае группировка идет по производителям, и также выбирается количество моделей для каждого производителя (Models) и общее количество всех товаров по всем этим моделям (Units). В конце группы сортируются по количеству товаров по убыванию.

