# Министерство науки и высшего образования Российской Федерации

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ «ЛЭТИ»  
ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

197376, Санкт-Петербург, ул. проф. Попова, 5.

Факультет компьютерных технологий и информатики

Кафедра вычислительной техники

«ЗАЧТЕНО»

\_\_\_\_\_\_\_\_\_\_ О.А. Жирнова

“\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

ОТЧЁТ  
по дисциплине «Базы данных»

**Лабораторная работа № 2  
««Группировка и агрегирование данных»**

|  |  |
| --- | --- |
| Студент группы 9308 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Н.В. Яловега |

Санкт Петербург 2021

*Цель работы:* знакомство с опциями GROUP BY и HAVING, агрегированием данных.

*Используемая база данных (БД):* AdventureWorks.

**Порядок выполнения**

*Упражнение 1 –* ***и*спользование ключевого слова TOP в команде SELECT**

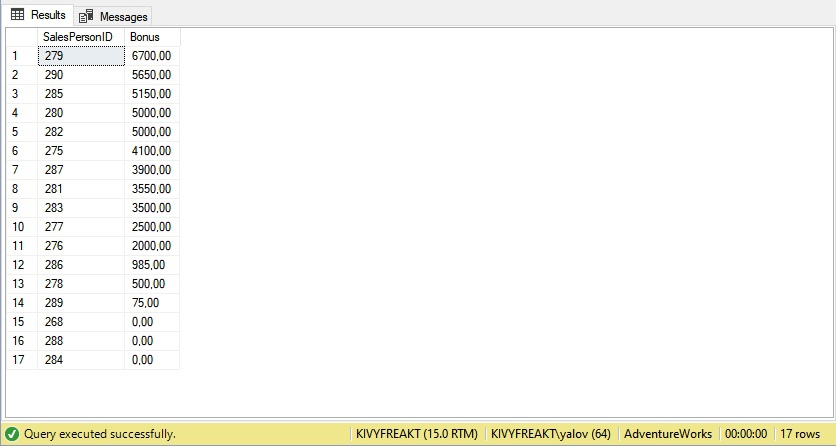
Запрос 1. Из таблицы Sales.SalesPerson выводим значения полей SalesPersonID и Bonus, сортируем по полю Bonus по убыванию.

SELECT SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

Результат выполнения запроса представлен на рисунке 1

Рисунок 1

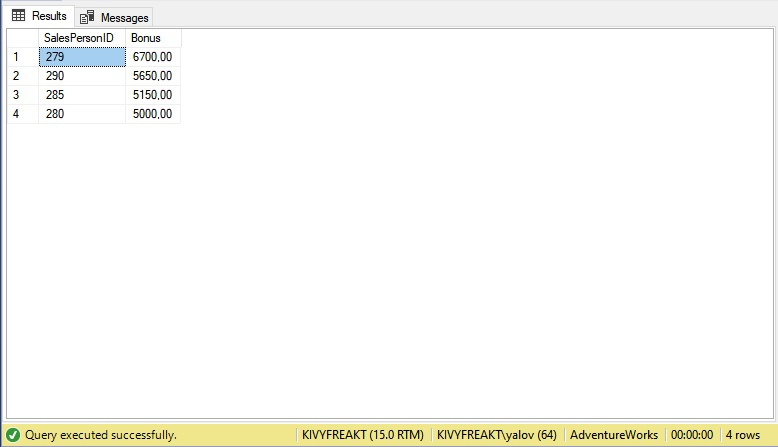
Запрос 2. Ограничим количество возвращаемых записей из предыдущего запроса до 4 самых больших премий для продавцов.

SELECT TOP 4 SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

Результат выполнения запроса представлен на рисунке 2

Рисунок 2

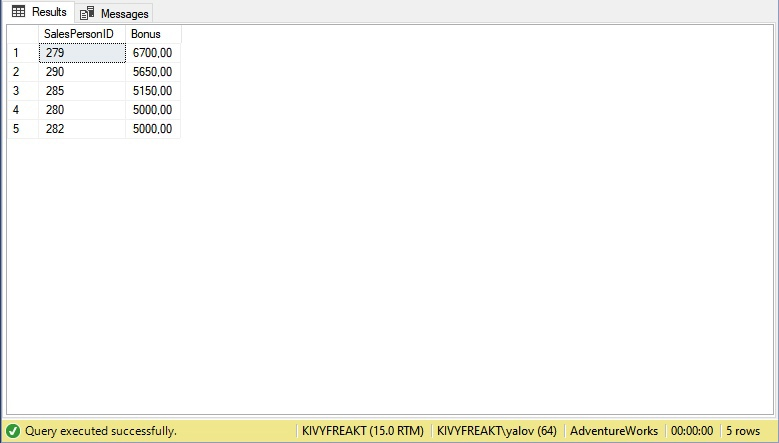
Запрос 3. Теперь сделаем так, чтобы запрос возвращал строки не только со значениями первых четырех самых больших премий для продавцов, но и данные по тем продавцам, чьи премии имеют то же значение, что и последнее значение, полученное в предыдущем задании.

SELECT TOP 4 WITH TIES SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

Результат выполнения запроса приведен на рисунке 3.

Рисунок 3

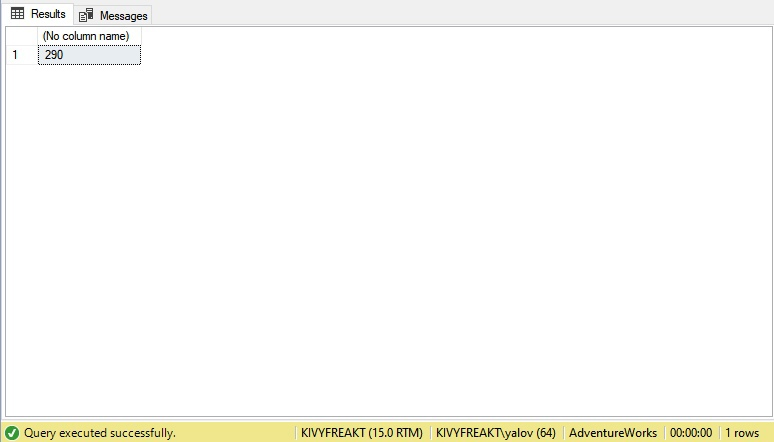
*Упражнение 2* – **использование агрегатных функций и конструкций GROUP BY И HAVING**

Запрос 1.1. Подсчет общего количества строк в таблице Employee схемы HumanResources.

SELECT COUNT(\*)

FROM HumanResources.Employee

Результат выполнения запроса приведен на рисунке 4.

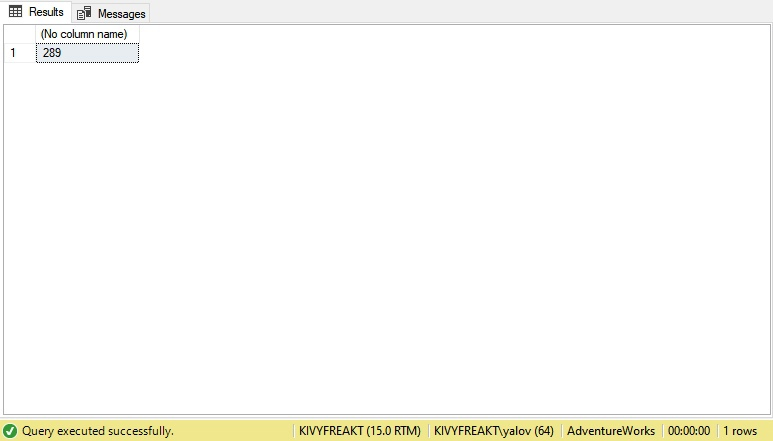
Рисунок 4

Запрос 1.2. Подсчет общего количества сотрудников, имеющих менеджеров (поле ManagerID).

SELECT COUNT(ManagerID)

FROM HumanResources.Employee

Результат выполнения запроса приведен на рисунке 5.

Рисунок 5

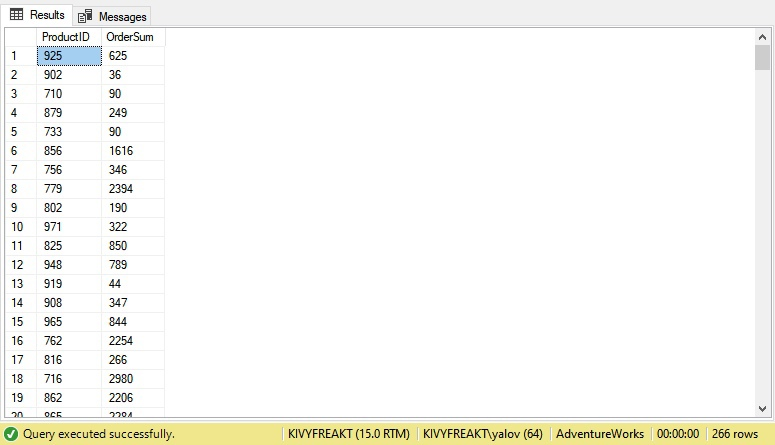
Запрос 2.1. Подсчитаем суммарное количество заказного товара (поле OrderQty) для каждого продукта (поле ProductID).

SELECT ProductID, SUM(OrderQty) AS OrderSum

FROM Sales.SalesOrderDetail

GROUP BY ProductID

Результат выполнения запроса приведен на рисунке 6.

Рисунок 6

Запрос 2.2. Отсортируем прошлый результат запроса по суммарному количеству товара.

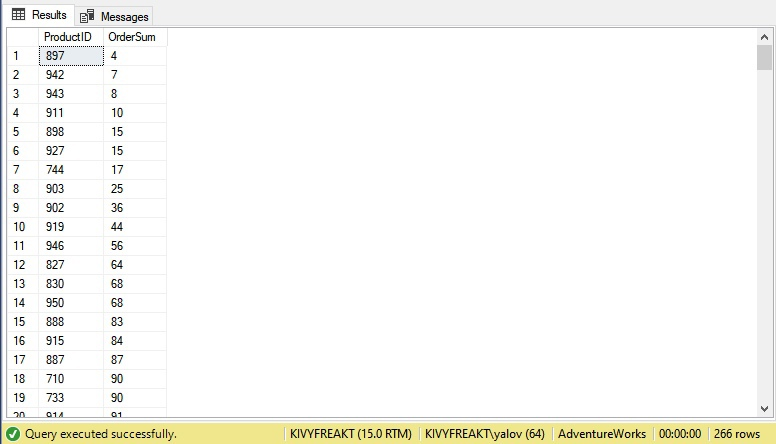
SELECT ProductID, SUM(OrderQty) AS OrderSum

FROM Sales.SalesOrderDetail

GROUP BY ProductID

ORDER BY OrderSum

Результат выполнения запроса приведен на рисунке 7.

Рисунок 7

Запрос 2.3. Модифицируем запрос таким образом, чтобы в результирующий набор попадали только те товары, суммарное значение заказов по которым не менее 2000.

SELECT ProductID, SUM(OrderQty) AS OrderSum

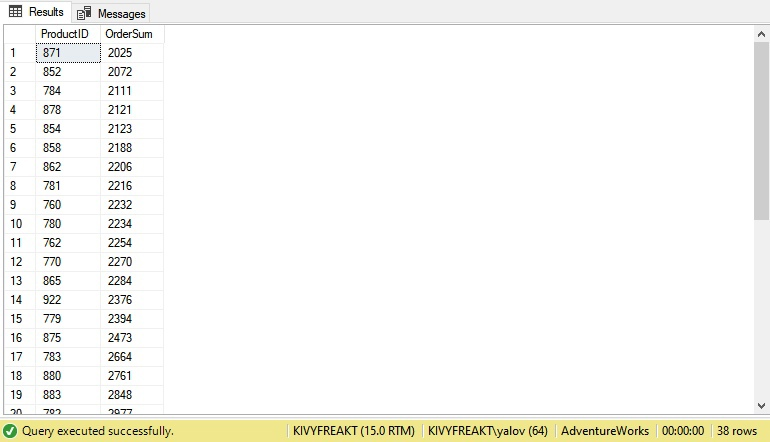
FROM Sales.SalesOrderDetail

GROUP BY ProductID

HAVING SUM(OrderQty)>=2000

ORDER BY OrderSum

Результат выполнения запроса приведен на рисунке 8.

Рисунок 8

Запрос 3.1. Выведем поля ProductID, SpecialOfferID, среднее значение цены за единицу товара (поле UnitPrice) и суммарного значения по полю LineTotal. Выполним группировку.

SELECT ProductID, SpecialOfferID,

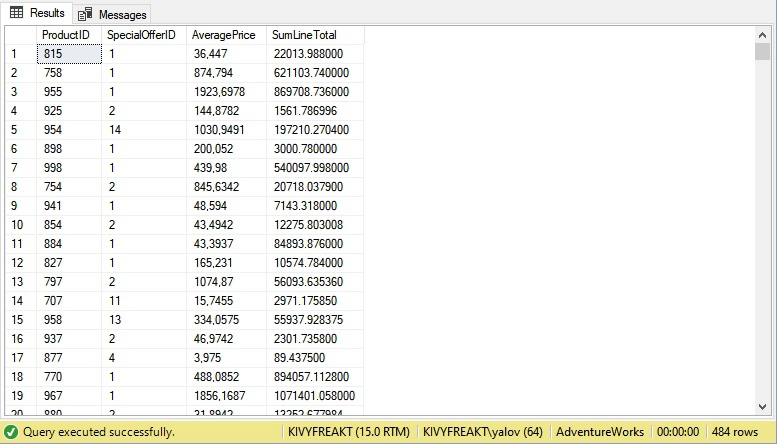
AVG(UnitPrice) AS AveragePrice,

SUM(LineTotal) AS SumLineTotal

FROM Sales.SalesOrderDetail

GROUP BY ProductID, SpecialOfferID

Результат выполнения запроса приведен на рисунке 9.

Рисунок 9

Запрос 3.2. Отсортируем полученные данные по полю ProductID по возрастанию, а также присвоим псевдонимы тем элементам списка SELECT которые соответствуют агрегированным значениям.

SELECT ProductID, SpecialOfferID,

AVG(UnitPrice) AS AveragePrice,

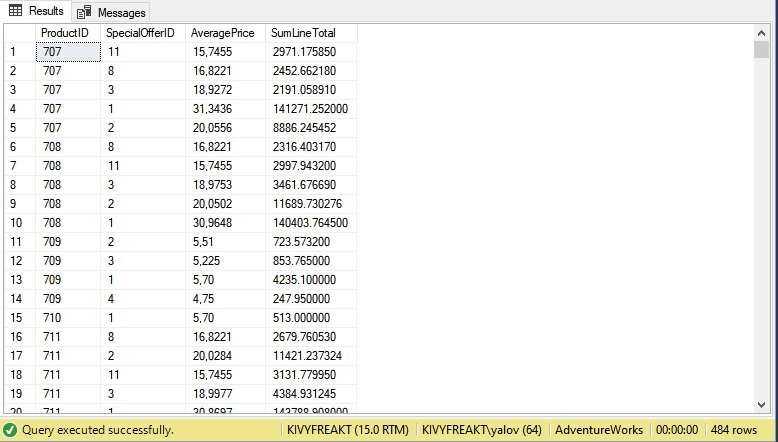
SUM(LineTotal) AS SumLineTotal

FROM Sales.SalesOrderDetail

GROUP BY ProductID, SpecialOfferID

ORDER BY ProductID ASC

Результат выполнения запроса приведен на рисунке 10.

Рисунок 10

*Упражнение 3* **– использование операторов ROLLUP И CUBE**

Запрос 1.1. Выведем поля SalesQuota и суммарное значение по полю SalesYTD. Выполним группировку. Также зададим псевдоним TotalSalesYTD для суммы.

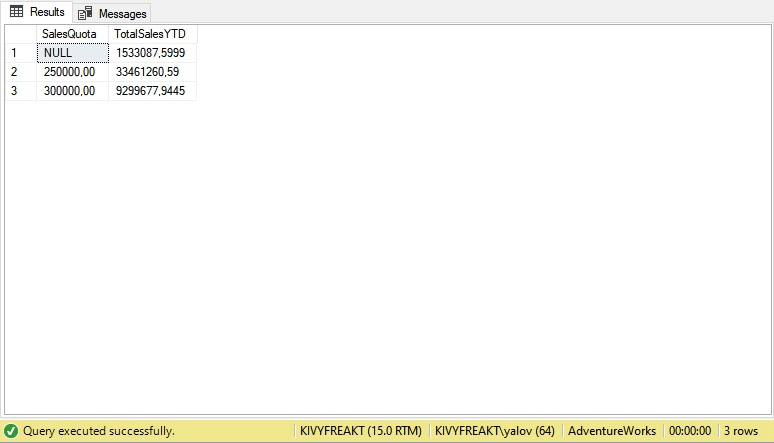
SELECT SalesQuota,

SUM(SalesYTD) AS TotalSalesYTD

FROM Sales.SalesPerson

GROUP BY SalesQuota

Результат выполнения запроса приведен на рисунке 11.

Рисунок 11

Запрос 1.2. Предыдущий запрос был изменен так, чтобы получить сводный результат по полученной выборке, дополнительно применена функция GROUPING.

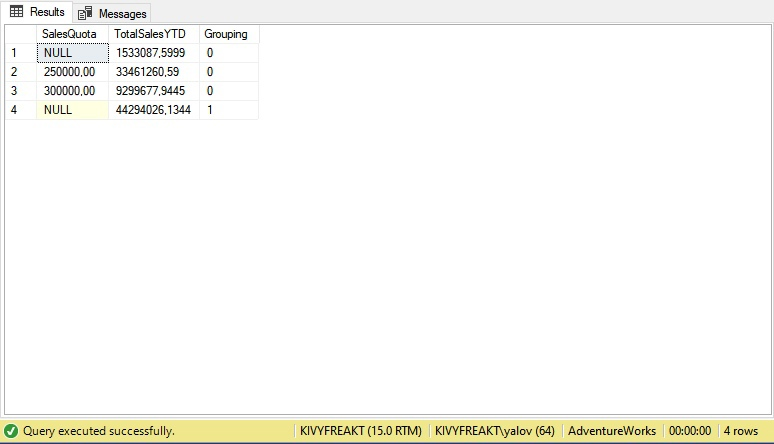
SELECT SalesQuota,

SUM(SalesYTD) AS TotalSalesYTD, GROUPING(SalesQuota) AS 'Grouping'

FROM Sales.SalesPerson

GROUP BY SalesQuota WITH ROLLUP

Результат выполнения запроса приведен на рисунке 12.

Рисунок 12

Запрос 1.3. Результат совпал со скриншотом из методических указаний, следовательно, задание выполнено верно. Смысл значений NULL - «поле, не содержащее никакого значения».

Запрос 2.1. Выведем поля ProductID и сумма по полю LineTotal. Выведены только те значения, для которых UnitPrice < $5.00. Выполнена сортировка и группировка произведена по полю ProductID.

SELECT ProductID,

SUM(LineTotal) AS SumLineTotal

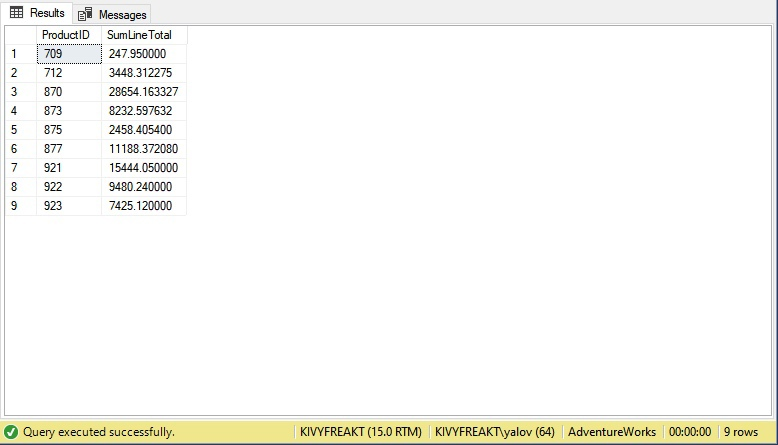
FROM Sales.SalesOrderDetail

WHERE UnitPrice<5.00

GROUP BY ProductID

ORDER BY ProductID ASC

Результат выполнения запроса приведен на рисунке 13.

Рисунок 13

Запрос 2.2. Предыдущий запрос модифицирован путем добавления оператора CUBE, а в группировку добавлено поле OrderQty.

SELECT ProductID,

SUM(LineTotal) AS SumLineTotal

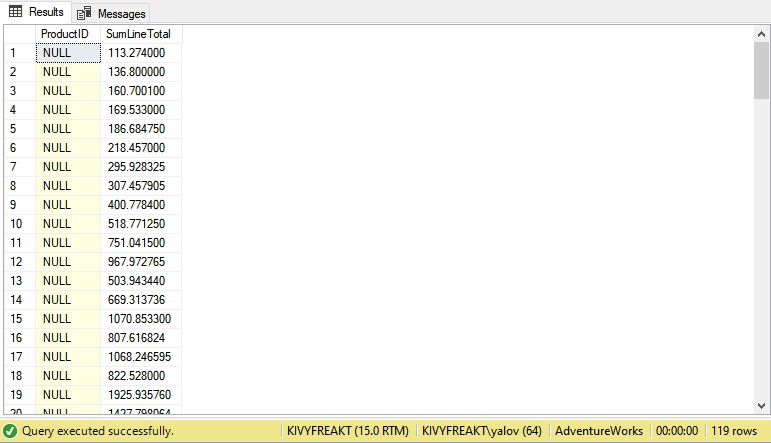
FROM Sales.SalesOrderDetail

WHERE UnitPrice<5.00

GROUP BY CUBE (ProductID, OrderQty)

ORDER BY ProductID ASC

Результат выполнения запроса приведен на рисунке 14.

Рисунок 14

**Выводы**

Ознакомился с опциями GROUP BY и HAVING (например, для формирования нескольких групп), а также с агрегированием данных. Использовал ключевые слова TOP и предложение WITH TIES для возвращения части отсортированных значений из результирующего набора данных. Для создания сводных результатов использовал операторы ROLLUP и CUBE, также применял функцию GROUPING.

**Список использованных источников**

1. Горячев А. В., Новакова Н. Е. Распределенные базы данных. Мет. указания к лаб. работам., СПб. Изд-во СПбГЭТУ «ЛЭТИ», 2008

2. Горячев А.В, Новакова Н.Е. Особенности разработки и администрирования приложений баз данных: учеб. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2016. 68 с.

3. Дейт К. Введение в системы баз данных. : Пер. с англ. – 6-е изд. -К.:Диалектика, 1998.