

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра Систем автоматизированного проектирования

ОТЧЕТ
по лабораторной работе №2
по дисциплине «Базы данных»
Тема: Группировка и агрегирование данных

Студенты гр. 2308

Рымарь М.И.

Мелихов М.А.

Придчин В.Е.

Преподаватель

Горяинов С.В.

Санкт-Петербург

2024

Цель работы

Знакомство с опциями GROUP BY и HAVING, а также агрегированием данных. В лабораторной работе используется база данных AdventureWorks.

Выполнение работы

Упражнение 1 – использование ключевого слова TOP в команде SELECT

Запрос 1: Вывод значений полей SalesPersonID и Bonus из таблицы Sales.SalesPerson (запрос был немного модифицирован, так как предлагаемого поля SalesPersonID в таблице не оказалось, столбцы таблицы представлены на рисунке 1).

```
SELECT BusinessEntityID, TerritoryID, Bonus FROM Sales.SalesPerson ORDER BY Bonus DESC
```

Результат выполнения запроса показан на рисунке 2.

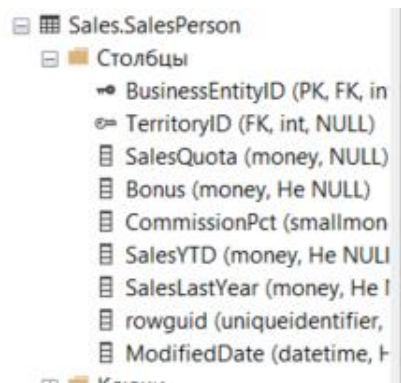


Рисунок 1 – Столбцы таблицы Sales.SalesPerson

BusinessEnt...	TerritoryID	Bonus
279	5	6700,0000
286	9	5650,0000
289	10	5150,0000
280	1	5000,0000
282	6	5000,0000
275	2	4100,0000
284	1	3900,0000
281	4	3550,0000
283	1	3500,0000

(затронута строк: 17)

Время выполнения: 2024-10-19T00:11:38.5407044+03:00

Рисунок 2 – Результат выполнения запроса

Запрос 2: Вывод первых 4 значений полей из таблицы Sales.SalesPerson. Результат выполнения запроса показан на рисунке 3.

```
SELECT TOP (4) BusinessEntityID, TerritoryID, Bonus
FROM Sales.SalesPerson ORDER BY Bonus DESC
```

BusinessEnt...	TerritoryID	Bonus
279	5	6700,0000
286	9	5650,0000
289	10	5150,0000
280	1	5000,0000

(затронута строк: 4)

Время выполнения: 2024-10-19T00:12:29.3364906+03:00

Рисунок 3 – Результат выполнения запроса

Запрос 3: Вывод первых 4 значений полей из таблицы Sales.SalesPerson, включая дублирующие последние значения с использованием команды WITH TIES. Результат выполнения запроса показан на рисунке 4.

```
SELECT TOP (4) WITH TIES BusinessEntityID,
TerritoryID, Bonus FROM Sales.SalesPerson
ORDER BY Bonus DESC
```

BusinessEnt...	TerritoryID	Bonus
279	5	6700,0000
286	9	5650,0000
289	10	5150,0000
280	1	5000,0000
282	6	5000,0000

(затронута строк: 5)

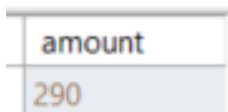
Время выполнения: 2024-10-19T00:13:49.7777633+03:00

Рисунок 4 – Результат выполнения запроса с WITH TIES

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

Запрос 1: Посчитать количество строк в таблице Employee схемы HumanResources. Результат выполнения запроса показан на рисунке 5.

```
SELECT COUNT(*) AS amount FROM HumanResources.Employee
```



amount
290

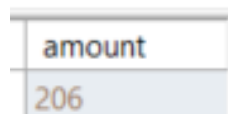
(затронута одна строка)

Время выполнения: 2024-10-19T00:17:13.7372959+03:00

Рисунок 5 – Результат выполнения запроса

Запрос 2: Посчитать количество сотрудников, имеющих менеджеров, в таблице Employee схемы HumanResources (так как поле ManagerID отсутствует, то я немного модифицирую запрос, подставив другое условие – гендер которых male). Результат выполнения запроса представлен на рисунке 6, столбцы таблицы HumanResources.Employee – на рисунке 7.

```
SELECT COUNT(*) AS amount FROM  
HumanResources.Employee WHERE (Gender = 'M')
```



amount
206

(затронута одна строка)

Время выполнения: 2024-10-19T00:18:17.7690389+03:00

Рисунок 6 – Результат выполнения запроса

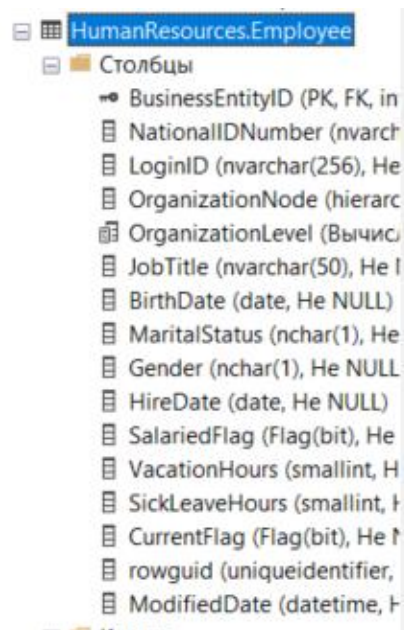


Рисунок 7 – Столбцы таблицы HumanResources.Employee

Запрос 3: Вычислить суммарное количество заказанного товара для каждого продукта. Результат выполнения запроса показан на рисунке 8.

```
SELECT ProductID, SUM(OrderQty) AS amount
FROM Sales.SalesOrderDetail
GROUP BY ProductID
```

ProductID	amount
925	625
902	36
710	90
879	249
733	90
856	1616
756	346

(затронуто строк: 266)

Время выполнения: 2024-10-19T00:20:34.3624172+03:00

Рисунок 8 – Результат выполнения запроса

Запрос 4: Отсортировать результат по суммарному количеству заказанного товара. Результаты запросов показаны на рисунке 9. Минимальное количество – 4, максимальное - 8311.

```
1) . SELECT ProductID, SUM(OrderQty) AS amount
FROM Sales.SalesOrderDetail
GROUP BY ProductID
ORDER BY SUM(OrderQty)
```

```
2) . SELECT ProductID, SUM(OrderQty) AS amount
FROM Sales.SalesOrderDetail
GROUP BY ProductID
ORDER BY SUM(OrderQty) DESC
```

ProductID	amount
897	4
942	7
943	8
911	10
898	15
927	15
744	17
903	25

(затронуто строк: 266)

Время выполнения: 2024-10-19T00:21:58.7494187+03:00

ProductID	amount
712	8311
870	6815
711	6743
715	6592
708	6532
707	6266
864	4247
873	3865
884	3864

(затронуто строк: 266)

Время выполнения: 2024-10-19T00:22:54.9409713+03:00

Рисунок 9 – Результат выполнения запроса

Запрос 5: Модифицировать запрос так, чтобы в выборку попадали те товары, суммарное количество которых не менее 2000. Результат представлен на рисунке 10. Минимальное значение – 2025.

```
SELECT ProductID, SUM(OrderQty) AS amount
FROM Sales.SalesOrderDetail
GROUP BY ProductID HAVING (SUM(OrderQty) >= 2000)
ORDER BY amount
```

ProductID	amount
871	2025
852	2072
784	2111
878	2121
854	2123
858	2188
862	2206

(затронуто строк: 38)

Время выполнения: 2024-10-19T00:38:29.2713759+03:00

Рисунок 10 – Результат выполнения запроса с HAVING

Запрос 6: Использование предложения GROUP BY для формирования нескольких групп. Результат представлен на рисунке 11.

```
SELECT ProductID, SpecialOfferID, AVG(UnitPrice) AS
average, SUM(LineTotal) AS amount
FROM Sales.SalesOrderDetail
GROUP BY ProductID, SpecialOfferID
ORDER BY ProductID
```

ProductID	SpecialOfferID	average	amount
707	11	15,7455	2971,175850
707	8	16,8221	2452,662180
707	3	18,9272	2191,058910
707	1	31,3436	141271,252...
707	2	20,0556	8886,245452
708	8	16,8221	2316,403170
708	11	15,7455	2997,943200
708	3	18,9753	3461,676690
708	2	20,0502	11689,7302...

(затронуто строк: 484)

Время выполнения: 2024-10-19T00:40:50.5569096+03:00

Рисунок 11 – Результат выполнения запроса

Упражнение 3 – использование операторов ROLLUP и CUBE

Запрос 1: Использование ROLLUP для создания сводных результатов. Результат выполнения запроса представлен на рисунке 12. При выполнении запроса строки с NULL в столбце SalesQuota появляются из-за использования ROLLUP, который добавляет итоговые строки для каждой группы и для общего итога по всем строкам.

```
SELECT SalesQuota, SUM(SalesYTD) AS TotalSalesYTD,  
GROUPING(SalesQuota) AS GroupingFlag  
FROM Sales.SalesPerson  
GROUP BY ROLLUP (SalesQuota)
```

SalesQuota	TotalSalesYTD	GroupingFl...
NULL	1252127,9471	0
250000,0000	27370537,9700	0
300000,0000	7654925,9863	0
NULL	36277591,9034	1

(затронуто строк: 4)

Время выполнения: 2024-10-19T00:42:59.1725541+03:00

Рисунок 12 – Результат выполнения запроса с ROLLUP

Запрос 2: Использование CUBE для создания сводных результатов. Результат выполнения запроса представлен на рисунке 13.

```
SELECT ProductID, OrderQty, SUM(LineTotal) AS  
TotalAmount  
FROM Sales.SalesOrderDetail WHERE (UnitPrice < 5.00)  
GROUP BY CUBE (ProductID, OrderQty)  
ORDER BY ProductID, OrderQty
```


ProductID	OrderQty	TotalAmount
NULL	NULL	86579,210714
NULL	1	61159,530000
NULL	2	833,124000
NULL	3	1466,154000
NULL	4	2059,752000
NULL	5	1873,860000
NULL	6	1941,300000
NULL	7	1860,264000
NULL	8	1460,832000
NULL	9	822,528000

(затронуто строк: 119)

Время выполнения: 2024-10-19T00:45:05.6378920+03:00

Рисунок 13 – Результат выполнения запроса с CUBE

Упражнение 4 – использование предложений COMPUTE и COMPUTE BY в команде SELECT для создания отчётов

Запрос 1: Написать запрос к таблице и изменить его с помощью COMPUTE. Запрос выглядел бы так, но COMPUTE был удалён из SQL Server с 2012 года, вместо него используются ROLLUP или оконные функции. Моя версия 2019 года, к сожалению, не поддерживает эту функцию, нет возможности проверить корректность отработки запроса.

```
SELECT SalesPersonID, CustomerID, OrderDate,
SubTotal, TotalDue
FROM Sales.SalesOrderHeader
ORDER BY SalesPersonID, OrderDate
COMPUTE SUM(SubTotal) AS SumTotal, SUM(TotalDue) AS
TotalDue
```

Выводы

В ходе лабораторной работы было изучено ключевое слово TOP, позволяющее ограничить количество возвращаемых строк в запросе, а также предложение WITH TIES, которое позволяет включать в результат строки, имеющие одинаковые значения с максимальными значениями. Были выполнены

запросы с использованием агрегатных функций, таких как SUM, AVG, MIN, MAX, COUNT, и конструкций GROUP BY и HAVING, что позволило группировать данные и применять фильтрацию к результатам группировки. Кроме того, были изучены и использованы в запросах операторы ROLLUP и CUBE, предоставляющие различные варианты группировки данных. Были также освоены предложения COMPUTE и COMPUTE BY в команде SELECT для создания отчетов, позволяющие вычислять сводные данные по результатам запросов. Полученные знания и навыки позволят эффективно извлекать, группировать и анализировать данные из базы данных, а также создавать сводные отчеты и аналитические документы.