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

« З А Ч Т Е Н О »

_____ О.А. Жирнова
“__” _____ 2021 г.

ОТЧЁТ
по дисциплине «Базы данных»

Лабораторная работа № 2
«Группировка и агрегирование данных»

Студент группы 9308 _____ С. А.Дубенков

Санкт Петербург 2021

Цель работы: знакомство с опциями GROUP BY и HAVING, а также агрегированием данных [лит1].

Используемая база данных (БД): AdventureWorks.

Порядок выполнения

Упражнение 1 – **использование ключевого слова TOP в команде SELECT**

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

```
select SalesPersonID, Bonus from Sales.SalesPerson order by Bonus desc;
```

Результат выполнения запроса представлен на рисунке 1

	SalesPersonID	Bonus
1	279	6700,00
2	290	5650,00
3	285	5150,00
4	280	5000,00
5	282	5000,00
6	275	4100,00
7	287	3900,00
8	281	3550,00
9	283	3500,00
10	277	2500,00
11	276	2000,00
12	286	985,00
13	278	500,00
14	289	75,00
15	268	0,00
16	288	0,00
17	284	0,00

Рисунок 1

2. Код предыдущего запроса был модифицирован так, чтобы возвращались только 4 записи о значениях самых больших премий (бонусов) для продавцов

```
select top(4) SalesPersonID, Bonus from Sales.SalesPerson order by Bonus desc;
```

Результат выполнения запроса представлен на рисунке 2

	SalesPersonID	Bonus
1	279	6700,00
2	290	5650,00
3	285	5150,00
4	280	5000,00

Рисунок 2

3. Код предыдущего запроса был модифицирован так, чтобы он возвращал строки не только со значениями первых четырех самых больших премий для продавцов, но и данные по тем продавцам, чьи пре-

мии имеют тоже 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 представлен на рисунке 3

	SalesPersonID	Bonus
1	279	6700,00
2	290	5650,00
3	285	5150,00
4	280	5000,00
5	282	5000,00

Рисунок 3

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

1. Использование агрегатных функций:

1.1 Подсчет общего количества строк в таблице Employee схемы HumanResources

```
select count(*) as fkc from HumanResources.Employee;
```

Результат выполнения запроса представлен на рисунке 4

	fkc
1	290

Рисунок 4

1.2 Подсчет общего количества сотрудников, имеющих менеджеров

```
select count(ManagerID) as fkc from HumanResources.Employee;
```

Результат выполнения запроса представлен на рисунке 5

	fkc
1	289

Рисунок 5

2. Использование опции GROUP BY

2.1 Запрос к таблице Sales.SalesOrderDetail, подсчитывающий суммарное количество заказанного товара (поле OrderQty) для каждого продукта (поле ProductID) + сортировка

```
select ProductId, sum(OrderQty) as sum from Sales.SalesOrderDetail group by ProductID order by sum desc;
```

Результат выполнения запроса представлен на рисунке 6

	ProductId	sum
1	712	8311
2	870	6815
3	711	6743
4	715	6592
5	708	6532
6	707	6266
7	864	4247
8	873	3865
9	884	3864
10	714	3636
11	859	3464
12	863	3378
13	877	3319
14	867	3296
15	869	3244
16	876	3166
17	921	3095
18	716	2980

Рисунок 6

2.2 Запрос, в котором в результирующий набор попадали только те товары, суммарное значение заказов по которым не менее 2000 (без сортировки)

```
select ProductId, sum(OrderQty) as sum from Sales.SalesOrderDetail group by ProductID having sum(OrderQty) >= 2000;
```

Результат выполнения запроса представлен на рисунке 7

	ProductId	sum
1	779	2394
2	762	2254
3	716	2980
4	862	2206
5	865	2284
6	782	2977
7	922	2376
8	876	3166
9	859	3464
10	770	2270
11	854	2123
12	877	3319
13	871	2025
14	708	6532
15	714	3636
16	883	2848
17	711	6743
18	880	2761

Рисунок 7

3. Использование предложения GROUP BY для формирования нескольких групп:

3.1 Запрос к таблице Sales.SalesOrderDetail, в списке SELECT которого должны быть представлены поля ProductID, SpecialOfferID, среднее значение цены за единицу товара и суммарное значение по полю LineTotal:

```
select ProductID, SpecialOfferID, avg(UnitPrice) as avg, sum(LineTotal) as sum from Sales.SalesOrderDetail group by ProductID, SpecialOfferID;
```

Результат выполнения запроса представлен на рисунке 8

	ProductID	SpecialOfferID	avg	sum
1	815	1	36,447	22013.988000
2	758	1	874,794	621103.740000
3	955	1	1923,6978	869708.736000
4	925	2	144,8782	1561.786996
5	954	14	1030,9491	197210.270400
6	898	1	200,052	3000.780000
7	998	1	439,98	540097.998000
8	754	2	845,6342	20718.037900
9	941	1	48,594	7143.318000
10	854	2	43,4942	12275.803008
11	884	1	43,3937	84893.876000
12	827	1	165,231	10574.784000
13	797	2	1074,87	56093.635360
14	707	11	15,7455	2971.175850
15	958	13	334,0575	55937.928375
16	937	2	46,9742	2301.735800
17	877	4	3,975	89.437500
18	770	1	488,0852	894057.112800

Рисунок 8

3.2 Отсортирован полученный результат по полю ProductID по возрастанию.

```
select ProductID, SpecialOfferID, avg(UnitPrice) as avg,
sum(LineTotal) as sum from Sales.SalesOrderDetail group by
ProductID, SpecialOfferID order by ProductID;
```

Результат выполнения запроса представлен на рисунке 9

	ProductID	SpecialOfferID	avg	sum
1	707	11	15,7455	2971.175850
2	707	8	16,8221	2452.662180
3	707	3	18,9272	2191.058910
4	707	1	31,3436	141271.252000
5	707	2	20,0556	8886.245452
6	708	8	16,8221	2316.403170
7	708	11	15,7455	2997.943200
8	708	3	18,9753	3461.676690
9	708	2	20,0502	11689.730276
10	708	1	30,9648	140403.764500
11	709	2	5,51	723.573200
12	709	3	5,225	853.765000
13	709	1	5,70	4235.100000
14	709	4	4,75	247.950000
15	710	1	5,70	513.000000
16	711	8	16,8221	2679.760530
17	711	2	20,0284	11421.237324
18	711	11	15,7455	3131.779950

Рисунок 9

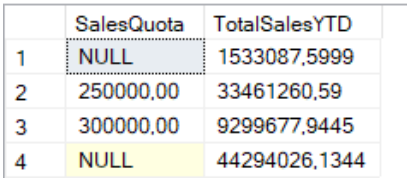
Упражнение 3 – **использование операторов ROLLUP и CUBE**

1. Использование оператора ROLLUP для создания сводных результатов:

1) Запрос к таблице Sales.SalesPerson. В списке SELECT поле SalesQuota и суммарное значение по полю SalesYTD. Выполнена группировку. Дан псевдоним TotalSalesYTD для суммы:

```
select SalesQuota, sum(SalesYTD) as TotalSalesYTD from Sales.SalesPerson group by SalesQuota with rollup;
```

Результат запроса представлен на рисунке 10



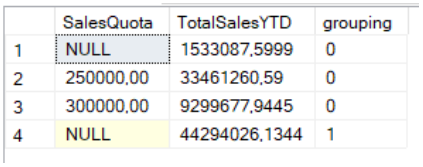
	SalesQuota	TotalSalesYTD
1	NULL	1533087,5999
2	250000,00	33461260,59
3	300000,00	9299677,9445
4	NULL	44294026,1344

Рисунок 10

2) Запрос изменен так, чтобы получать 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 запроса представлен на рисунке 11



	SalesQuota	TotalSalesYTD	grouping
1	NULL	1533087,5999	0
2	250000,00	33461260,59	0
3	300000,00	9299677,9445	0
4	NULL	44294026,1344	1

Рисунок 11

Результат совпал со скриншотом из методических указаний, следовательно, задание выполнено верно. Смысл значений NULL заключается в том, что они обозначают «пустое поле», то есть «поле, не содержащее никакого значения»

2. Использование оператора CUBE для создания сводных результатов

1) Запрос к таблице Sales.SalesOrderDetail. В списке SELECT указаны поле ProductID и сумма по полю LineTotal. Выводить необходимо только те значения, для которых UnitPrice < $5.00. Выполнена сортировка и группировка по полю ProductID:

```
select ProductID, sum(LineTotal) as sumTotal from Sales.SalesOrderDetail where UnitPrice < 5 group by ProductID order by ProductID;
```

Результат запроса представлен на рисунке 12

	ProductID	sumTotal
1	709	247.950000
2	712	3448.312275
3	870	28654.163327
4	873	8232.597632
5	875	2458.405400
6	877	11188.372080
7	921	15444.050000
8	922	9480.240000
9	923	7425.120000

Рисунок 12

2) Предыдущий запрос модифицирован путем добавления оператора CUBE, а в группировку добавлено поле OrderQty:

```
select ProductID, sum(LineTotal) as sumTotal from Sales.SalesOrderDetail where UnitPrice < 5 group by cube(ProductID, OrderQty) order by ProductID ;
```

Результат запроса представлен на рисунке 13

	ProductID	sumTotal
1	NULL	113.274000
2	NULL	136.800000
3	NULL	160.700100
4	NULL	169.533000
5	NULL	186.684750
6	NULL	218.457000
7	NULL	295.928325
8	NULL	307.457905
9	NULL	400.778400
10	NULL	518.771250
11	NULL	751.041500
12	NULL	967.972765
13	NULL	503.943440
14	NULL	669.313736
15	NULL	1070.853300
16	NULL	807.616824
17	NULL	1068.246595
103	877	763.200000
104	877	867.605760
105	877	915.840000
106	877	1001.700000
107	877	186.030000
108	877	1116.180000
109	877	1001.700000
110	877	477.000000
111	877	801.360000
112	877	1087.560000
113	877	11188.372...
114	921	15444.050...
115	921	15444.050...
116	922	9480.240000
117	922	9480.240000
118	923	7425.120000
119	923	7425.120000

Рисунок 13

Вывод: в ходе выполнения данной лабораторной работы была достигнута цель, предполагающая знакомство с опциями GROUP BY и HAVING (например, для формирования нескольких групп), а также с агрегированием данных. Были использованы ключевые слова TOP и предложение WITH TIES для возвращения части отсортированных значений из результирующего набора данных. Для создания сводных результатов были использованы операторы ROLLUP и CUBE и применена функция GROUPING

Список использованных источников

Методические указания к лабораторным работам / Сост.: А. В. Горячев, Н.Е. Новакова. СПб.: Изд-во СПБГЭТУ «ЛЭТИ», 2008. 32 с.