**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра САПР**

**ОТЧЕТ**

**по лабораторной работе № 2**

**по дисциплине "Базы данных"**

**Тема: "Группировка и агрегирование данных"**

| Студент гр. 2311 |  | Завьялов Н.С. |
| --- | --- | --- |
| Преподаватель |  | Новакова Н.Е. |

Санкт-Петербург

2024

*Цель работы:* Знакомство с опциями GROUP BY и HAVING, а также агрегированием данных.

*Используемая база данных:* AdventureWorks

**Упражнение 1 – использование ключевого слова TOP в команде SELECT.**  
Запрос 1.:

SELECT TOP(4) SalesPersonID

FROM AdventureWorks.Sales.SalesPerson

ORDER BY Bonus DESC

Результаты выполнения запроса

SalesPersonID

-------------

279

290

285

280

(4 rows affected)

Completion time: 2024-09-25T21:56:51.0286873+03:00

Запрос 2. Использование WITH TIES вместе с TOP:

SELECT TOP(4) WITH TIES SalesPersonID

FROM AdventureWorks.Sales.SalesPerson

ORDER BY Bonus DESC

Результаты выполнения запроса

SalesPersonID

-------------

279

290

285

280

282

(5 rows affected)

Completion time: 2024-09-25T22:08:48.9484768+03:00

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

Запрос 1. Использование COUNT(\*) :

SELECT COUNT(\*)

FROM AdventureWorks.HumanResources.Employee

Результаты выполнения запроса

-----------

290

(1 row affected)

Completion time: 2024-09-25T22:13:01.5089273+03:00

Запрос 2. :

SELECT COUNT(\*)

FROM AdventureWorks.HumanResources.Employee

WHERE ManagerID IS NOT NULL

Результаты выполнения запроса

-----------

289

(1 row affected)

Completion time: 2024-09-25T22:18:58.1839570+03:00

Запрос 3. Использование GROUP BY :

SELECT ProductID,

SUM(OrderQty) AS 'Суммарное кол-во'

FROM AdventureWorks.Sales.SalesOrderDetail

GROUP BY ProductID

Результаты выполнения запроса

ProductID Суммарное кол-во

----------- ----------------

925 625

902 36

...

933 858

984 450

(266 rows affected)

Completion time: 2024-09-26T13:15:08.3147509+03:00

Запрос 4. :

SELECT ProductID,

SUM(OrderQty) AS 'Суммарное кол-во'

FROM AdventureWorks.Sales.SalesOrderDetail

GROUP BY ProductID

ORDER BY 'Суммарное кол-во'

Результаты выполнения запроса

ProductID Суммарное кол-во

----------- ----------------

897 4

942 7

...

870 6815

712 8311

(266 rows affected)

Completion time: 2024-09-26T13:17:24.3136749+03:00

Запрос 5. Использование HAVING :

SELECT ProductID,

SUM(OrderQty) AS 'Суммарное кол-во'

FROM AdventureWorks.Sales.SalesOrderDetail

GROUP BY ProductID

HAVING SUM(OrderQty) >= 2000

ORDER BY 'Суммарное кол-во'

Результаты выполнения запроса

ProductID Суммарное кол-во

----------- ----------------

871 2025

852 2072

...

870 6815

712 8311

(38 rows affected)

Completion time: 2024-09-26T13:21:14.9305634+03:00

Запрос 6. Использование GROUP BY для формирования нескольких групп:

SELECT ProductID,

SpecialOfferID,

SUM(UnitPrice)/COUNT(UnitPrice) AS AvrgUnitPrice,

SUM(LineTotal) AS SumLineTotal

FROM AdventureWorks.Sales.SalesOrderDetail

GROUP BY ProductID, SpecialOfferID

ORDER BY ProductID

Результаты выполнения запроса

ProductID SpecialOfferID AvrgUnitPrice SumLineTotal

----------- -------------- --------------------- ---------------------------------------

707 11 15,7455 2971.175850

707 8 16,8221 2452.662180

...

999 2 527,3902 76871.032436

999 1 428,3185 438795.874000

(484 rows affected)

Completion time: 2024-09-26T13:37:07.8897782+03:00

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

Запрос 1. Использование ROLLUP :

SELECT SalesQuota,

SUM(SalesYTD) AS TotalSalesYTD,

GROUPING(SalesQuota) AS GROUPING\_SalesQuota

FROM AdventureWorks.Sales.SalesPerson

GROUP BY ROLLUP(SalesQuota)

Результаты выполнения запроса

SalesQuota TotalSalesYTD GROUPING\_SalesQuota

--------------------- --------------------- -------------------

NULL 1533087,5999 0

250000,00 33461260,59 0

300000,00 9299677,9445 0

NULL 44294026,1344 1

(4 rows affected)

Completion time: 2024-09-26T14:02:36.3517854+03:00

Запрос 2. Использование CUBE :

SELECT ProductID,

SUM(LineTotal) AS Total

FROM AdventureWorks.Sales.SalesOrderDetail

WHERE UnitPrice < 5

GROUP BY CUBE(ProductID, OrderQty)

ORDER BY ProductID

Результаты выполнения запроса

ProductID Total

----------- ---------------------------------------

NULL 61159.530000

NULL 833.124000

NULL 1466.154000

...

NULL 136.800000

NULL 86579.210714

709 136.800000

709 111.150000

...

923 7425.120000

923 7425.120000

(119 rows affected)

Completion time: 2024-09-26T14:11:21.8260306+03:00

**Вывод**

В ходе выполнения лабораторной работы было проведено знакомство с опциями GROUP BY и HAVING, а именно:

**HAVING** выполняет почти ту же функцию, что и WHERE – накладывает ограничения на выборку. Однако если WHERE применяется до группировки, то HAVING выполняется после группировки.

**GROUP BY** используется для группировки данных по каким-либо полям. Далее полученные группы можно анализировать, а также применять к ним агрегатные функции.

Также у GROUP BY существует дополнительный синтаксис: GROUP BY ROLLUP() и GROUP BY CUBE().

При использовании оператора **ROLLUP()** в последней строке появляется сводное значение по всем значениям указанного столбца

При использовании оператора **CUBE()** к результату добавляются результаты всех перекрестных вычислений.

Для удобства пользования операторами ROLLUP() и CUBE() существует функция grouping(). При добавлении её в поле вывода операции SELECT, она будет выдавать 1, если в данной строке производится группировка по выбранному параметру и 0, если не производится.

Также в данной лабораторной работе были рассмотрены ключевые слова TOP и WITH TIES.

SELECT **TOP (n)** – будут выбраны только первые n результатов

SELECT TOP (n) **WITH TIES –** будут выбраны первые n результатов, а также все результаты, у которых совпадает первое совпавшее поле списка запрашиваемых полей.

Также в данной работе использовались следующие агрегатные функции:

**sum()** – сумма по столбцу;

**avg()** – среднее значение по столбцу;

**count()** – количество строк;