**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра систем автоматизированного проектирования (САПР)**

**отчет**

**по лабораторной работе № 2**

**по дисциплине «Базы данных»**

Тема: **«Группировка и агрегирование данных»**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 3311 | Аршин А. Д  Баймухамедов Р. Р.  Пасечный Л. В. |  |
| Преподаватель | Новакова Н. Е. |  |

Санкт-Петербург

2025

**Цель работы**

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

**Упражнение 1**

Будем использовать ключевые слова TOP в команде SELECT и предложение WITH TIES для возвращения части отсортированных значений из результирующего набора данных

***Задание 1.1***

Из таблицы Sales.SalsePerson выведем значения полей SalesPersonID и Bonus. Отсортируем запрос по полю Bonus по убыванию

*Запрос*

SELECT SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

*Результат*

|  |  |
| --- | --- |
| SalesPersonID | Bonus |
| 279 | 6700,00 |
| 290 | 5650,00 |
| 285 | 5150,00 |
| 280 | 5000,00 |
| 282 | 5000,00 |
| … | … |

(Затронуто строк: 17)

Время выполнения: 2025-09-15T20:30:40.8037693+03:00

***Задание 1.2***

Модифицируем код запроса таким образом, чтобы возвращались только 4 записи о значениях самых больших премий (бонусов) для продавцов

*Запрос*

SELECT TOP 4 SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

*Результат*

|  |  |
| --- | --- |
| SalesPersonID | Bonus |
| 279 | 6700,00 |
| 290 | 5650,00 |
| 285 | 5150,00 |
| 280 | 5000,00 |

(Затронуто строк: 4)

Время выполнения: 2025-09-15T20:38:39.7738473+03:00

***Задание 1.3***

Модифицируем запрос из предыдущего значения так, чтобы он 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 прерыдущем задании. В результате выполнения запроса должно получиться 5 строк

*Запрос*

SELECT TOP 4 WITH TIES SalesPersonID, Bonus

FROM Sales.SalesPerson

ORDER BY Bonus DESC

*Результат*

|  |  |
| --- | --- |
| SalesPersonID | Bonus |
| 279 | 6700,00 |
| 290 | 5650,00 |
| 285 | 5150,00 |
| 280 | 5000,00 |
| 282 | 5000,00 |

(Затронуто строк: 5)

Время выполнения: 2025-09-15T21:18:59.7934505+03:00

**Упражнение 2**

Использование агрегатных функций и конструкций GROUP BY и HAVING

***Задание 2.1.1***

Используя агрегатные функции подсчитаем общее количество строк в таблице Employee схемы HumanResources

*Запрос*

SELECT COUNT(EmployeeID) as 'count of employee'

FROM HumanResources.Employee

*Результат*

|  |
| --- |
| count of employee |
| 290 |

(Затронута 1 строка)

Время выполнения: 2025-09-15T21:48:27.1294134+03:00

***Задание 2.1.2***

Подсчитайте общее количество сотрудников, имеющих менеджеров (поле ManagerID). Запрос выполняется по той же таблице

*Запрос*

SELECT COUNT(EmployeeID) as 'count of employee'

FROM HumanResources.Employee

WHERE ManagerID IS NOT NULL

*Результат*

|  |
| --- |
| count of employee |
| 289 |

(Затронута 1 строка)

Время выполнения: 2025-09-15T21:55:11.9958702+03:00

***Задание 2.2.1***

Используя опции GROUP BY, напишем запрос к таблице Sales.SalesOrderDetail, подсчитывающий суммарное количество заказанного товара (поле OrderQty) для каждого продукта (поле ProductID)

*Запрос*

SELECT ProductID, SUM(OrderQty) as 'Product quantity'

FROM Sales.SalesOrderDetail

GROUP BY ProductID

*Результат*

|  |  |
| --- | --- |
| ProductID | Product quantity |
| 925 | 625 |
| 902 | 36 |
| 710 | 90 |
| 879 | 249 |
| 733 | 90 |
| … | … |

(Затронуто строк: 266)

Время выполнения: 2025-09-15T22:09:14.7087409+03:00

***Задание 2.2.2***

Отсортируем результат запроса по суммарному количеству заказанного товара

*Запрос*

SELECT ProductID, SUM(OrderQty) as 'Product quantity'

FROM Sales.SalesOrderDetail

GROUP BY ProductID

ORDER BY [Product quantity] DESC

*Результат*

|  |  |
| --- | --- |
| ProductID | Product quantity |
| 712 | 8311 |
| 870 | 6815 |
| 711 | 6743 |
| 715 | 6592 |
| 708 | 6532 |
| … | … |

(Затронуто строк: 266)

Время выполнения: 2025-09-15T22:14:15.9769391+03:00

***Задание 2.2.3***

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

*Запрос*

SELECT ProductID, SUM(OrderQty) as 'Product quantity'

FROM Sales.SalesOrderDetail

GROUP BY ProductID

HAVING SUM(OrderQty) >= 2000

ORDER BY [Product quantity] ASC

*Результат*

|  |  |
| --- | --- |
| ProductID | Product quantity |
| 871 | 2025 |
| 852 | 2072 |
| 784 | 2111 |
| 878 | 2121 |
| 854 | 2123 |
| … | … |

(Затронуто строк: 38)

Время выполнения: 2025-09-15T22:20:16.0046907+03:00

***Задание 2.3.1 – 2.3.3***

Используем предложения GROUP BY для формирования нескольких групп. Напишем запрос к таблице Sales.SalesOrderDetail, в списке SELECT которого должны быть представлены поля ProductID, SpecialOfferID, среднее значение цены за единицу товара (поле UnitPrice) и суммарное значение по полю LineTotal. Отсортируем полученный результат по полю ProductID по возрастанию. Дадим псевдонимы тем элементам списка SELECT, которые соответствуют агрегированным значениям. Выполним группировку.

*Запрос*

SELECT ProductID, SpecialOfferID, AVG(UnitPrice) as 'average unit price', SUM(LineTotal) as 'sum lint total'

FROM Sales.SalesOrderDetail

GROUP BY ProductID, SpecialOfferID

ORDER BY ProductID ASC

*Результат*

|  |  |  |  |
| --- | --- | --- | --- |
| ProductID | SpecialOfferID | average unit price | sum lint total |
| 707 | 11 | 15,7455 | 2971.175850 |
| 707 | 8 | 16,8221 | 2452.662180 |
| 707 | 3 | 18,9272 | 2191.058910 |
| 707 | 1 | 31,3436 | 141271.252000 |
| 707 | 2 | 20,0556 | 8886.245452 |
| … | … | … | … |

(Затронуто строк: 484)

Время выполнения: 2025-09-15T23:13:47.2909946+03:00

**Упражнение 3**

Использование операторов ROLLUP и CUBE

***Задание 3.1.1***

Напишем запрос к таблице Sales.SalesPerson. В списке SELECT укажите полу SalesQuota и суммарное значение по полю SalesYTD. Выполним группировку. Дадим псевдоним TotalSalesYTD для суммы

*Запрос*

SELECT SalesQuota, SUM(SalesYTD) as 'sum of sales'

FROM Sales.SalesPerson

GROUP BY SalesQuota

*Результат*

|  |  |
| --- | --- |
| SalesQuota | sum of sales |
| NULL | 1533087,5999 |
| 250000,00 | 33461260,59 |
| 300000,00 | 9299677,9445 |

(Затронуто строк: 3)

Время выполнения: 2025-09-16T00:10:29.6424470+03:00

***Задание 3.1.2***

Изменим запрос так, чтобы получать сводный результат по полученной выборке. Дополнительно применим функцию GROUPING

*Запрос*

SELECT SalesQuota, SUM(SalesYTD) as 'sum of sales', GROUPING(SalesQuota) AS 'groupung quota'

FROM Sales.SalesPerson

GROUP BY ROLLUP(SalesQuota)

*Результат*

|  |  |  |
| --- | --- | --- |
| SalesQuota | sum of sales | groupung quota |
| NULL | 1533087,5999 | 0 |
| 250000,00 | 33461260,59 | 0 |
| 300000,00 | 9299677,9445 | 0 |
| NULL | 44294026,1344 | 1 |

(Затронуто строк: 4)

Время выполнения: 2025-09-16T10:05:08.8486412+03:00

Первый NULL – значение из данных поля SalesQuota. Второй NULL – результат группировки (итоговая строка, которая добавилась в результате ROLLUP). Функция GROUPING помогает различить эти два случая

***Задание 3.2.1***

Напишем запрос к таблице Sales.SalesOrderDetail. В списке SELECT укажем поле ProductID и сумма по полю LineTotal. Выведем только те значения, для которых UnitPrice < 5.00$. Выполним сортировку и группировку по полю ProductID

*Запрос*

SELECT ProductID, SUM(LineTotal) AS 'Sum of line total'

FROM Sales.SalesOrderDetail

WHERE UnitPrice < 5.00

GROUP BY ProductID

ORDER BY ProductID ASC

*Результат*

|  |  |
| --- | --- |
| ProductID | Sum of line total |
| 709 | 247.950000 |
| 712 | 3448.312275 |
| 870 | 28654.163327 |
| 873 | 8232.597632 |
| 875 | 2458.405400 |
| … | … |

(Затронуто строк: 9)

Время выполнения: 2025-09-16T10:58:15.9541760+03:00

***Задание 3.2.2***

Модифицируем запрос, добавив оператор CUBE, а в группировку добавим поле OrderQty

*Запрос*

SELECT ProductID, SUM(LineTotal) AS 'Sum of line total'

FROM Sales.SalesOrderDetail

WHERE UnitPrice < 5.00

GROUP BY CUBE(ProductID, OrderQty)

ORDER BY ProductID ASC

*Результат*

|  |  |
| --- | --- |
| ProductID | Sum of line total |
| NULL | 113.274000 |
| NULL | 136.800000 |
| … | … |
| 922 | 9480.240000 |
| 923 | 7425.120000 |
| 923 | 7425.120000 |

(Затронуто строк: 119)

Время выполнения: 2025-09-16T11:01:43.1501181+03:00

**Вывод**

В ходе выполнения лабораторной работы были успешно освоены и применены на практике ключевые механизмы агрегирования данных в SQL Server. В результате работы достигнуты следующие цели:

* Отработано использование базовых агрегатных функций (SUM, AVG, COUNT) в сочетании с оператором GROUP BY для получения групповых итогов. Освоено важное правило включения не агрегированных полей в предложение GROUP BY.
* Практически изучено применение оператора HAVING для фильтрации данных после группировки, а также освоены ключевые отличия между HAVING и WHERE.
* Успешно опробованы операторы ROLLUP и CUBE для формирования иерархических итогов и всевозможных комбинаций группировок. Применена функция GROUPING () для идентификации результирующих строк.
* Освоено использование ключевого слова TOP в сочетании с ORDER BY для выборки наиболее значимых записей, а также отработано присвоение понятных псевдонимов агрегированным столбцам с помощью AS.
* Полученные навыки позволяют осуществлять построение комплексных аналитических отчетов и работать с бизнес-данными в реальных проектах. Лабораторная работа выполнена в полном объеме.