Лабораторная работа 2

Цель работы: Изучение выражения GROUP BY – группировка данных

Одной из ключевых задач СУБД является выполнение операций связанных с обработкой данных. Это необходимо для анализа информации, а также для формирования сложных запросов. Выражение GROUP BY позволяет разделить результаты выполнения оператора SELECT на группы по признаку, с целью выполнения статистических операций над группами.

Простой запрос:

```
SELECT [Name], [ListPrice], [Color] FROM [Production].[Product]

WHERE [Color] IS NOT NULL
```

В результате выполнения этого запроса пользователь получит название товаров, их стоимость и цвет, таких, что цвет товаров определен. Это полезная и важная информация, однако у пользователя может появиться более сложная задача, например посчитать количество товаров того или иного цвета.

Существует набор функций, которые позволяют выполнить операцию над набором значений в столбце. Такие функции называются агрегирующими:

SUM ([ALL | DISTINCT] expression) – сумма значений в столбце;

MAX([ALL | DISTINCT] expression) – максимальное значение в столбце;

MIN ([ALL | DISTINCT] expression) – минимальное значение в столбце;

AVG ([ALL | DISTINCT] expression) – среднее значение в столбце;

COUNT ({ [[ALL | DISTINCT] expression] | * }) – количество строк в столбце, с учетом NULL значений.

Приведенный общий синтаксис функций рассмотрим на примере:

```
SELECT COUNT(DISTINCT [Color]) FROM [Production].[Product]

WHERE [Color] IS NOT NULL
```

В данном случае будет выполнен SELECT который отберет из таблицы все строки, для которых цвет определен, и будет выполнена агрегирующая функция COUNT, которая посчитает количество различных цветов. Инструкция DISTINCT обеспечивает выбор не повторяющихся значений цвета, а функция COUNT считает количество полученных значений.

Запрос следующего вида фактически возвращает количество строк, для которых определен цвет продукта:

```
SELECT COUNT([Color]) FROM [Production].[Product]

WHERE [Color] IS NOT NULL
```

Он полностью эквивалентен следующему запросу:

```
SELECT COUNT(*) FROM [Production].[Product]

WHERE [Color] IS NOT NULL
```

Рассмотрим работу функции MAX:

```
SELECT MAX([ListPrice]) FROM [Production].[Product]

WHERE [Color]='Red'
```

Этот запрос возвращает максимальную цену продукта, у которого цвет ‘Red’.

Выражение GROUP BY позволяет разделить результат выполнения запроса на группы, и выполнить ту или иную функцию над каждой из групп.

```
SELECT [Color], COUNT(*) as 'Amount'

FROM [Production].[Product]

WHERE [Color] IS NOT NULL

GROUP BY [Color]
```

В данном запросе все строк, для которых выполнено условие – цвет определен, разделены на группы, по цвету, и над каждой группой выполнено функция COUNT. Результатом станет два столбца – цвет, и количество строк с данным цветом в общей выборке.

Необходимо сделать несколько важных дополнений. При группировке запрос возвращает только те данные, которые имеют одно и то же значение для всей группы. Запрос вида:

```
SELECT [Name], [Color], COUNT(*) as 'Amount'

FROM [Production].[Product]

WHERE [Color] IS NOT NULL

GROUP BY [Color]
```

Не будет выполнен, так как в группе может существовать несколько товаров с разными названиями. Даже если фактически в каждой группе с одним и тем же цветом все товары имеют одно и то же название, все равно это будет ошибкой. Так при группировке нельзя использовать псевдонимы столбцов как признак группировки, нельзя группировать по столбцам с типами данных text, ntext, или image, но можно группировать по результату выполнения функции над этими столбцами, например функции приведения типа, или строковой функции. Нельзя использовать в качестве признака группировки подзапросы и данные типа XML. Нельзя использовать в качестве признака группировки столбец индексированного представления.

Если столбец по которому осуществляется группировка содержит значения NULL, то они будут рассмотрены как идентичные, и будут образовывать отдельную группу.

Допускается группировка по результату выполнения простых математических операций, а также группировка по двум и более столбцам:

SELECT Столбец1, Столбец2 FROM Таблица GROUP BY Столбец1, Столбец2;

SELECT Столбец1 + Столбец2 FROM Таблица GROUP BY Столбец1, Столбец2;

SELECT Столбец1 + Столбец2 FROM Таблица GROUP BY Столбец1 + Столбец2;

SELECT Столбец1 + Столбец2 + константное_значение FROM Таблица GROUP BY Столбец1, Столбец2.

Рассмотрим следующий пример:

```
SELECT [Color], [Style], COUNT(*) as 'Amount'

FROM [Production].[Product]

WHERE [Color] IS NOT NULL AND [Style] IS NOT NULL

GROUP BY [Color], [Style]
```

Запрос возвращает, сколько продуктов с общим цветом и стилем упоминается в таблице продукты, с учетом того, что у этих товаров и цвет и стиль определены.

Выражение GROUP BY используется совместно с выражением HAVING, которое определяет условие поиска группы.

```
SELECT [Color], COUNT(*) as 'Amount'

FROM [Production].[Product]

WHERE [Color] IS NOT NULL

GROUP BY [Color]

HAVING COUNT(*)>10
```

Данный запрос выдаст название цветов, количество товаров данного цвета, но только для тех групп, где это количество больше 10. Естественно будет выполнено условие, что цвет товаров определен. Условие поиска группы может содержать так же логические операторы. Вот пример такой возможности.

```
SELECT [Color], COUNT(*) as 'Amount'

FROM [Production].[Product]

WHERE [Color] IS NOT NULL

GROUP BY [Color]

HAVING COUNT(*)>10 AND MAX([ListPrice])>3000
```

Стоит отметить, что операция группировки весьма затратна, с точки зрения процессорной мощности и времени, неправильная конструкция может существенно увеличить время выполнения запроса. Например, стоит задача, рассмотренная в начале работы: посчитать количество товаров каждого цвета, исключив товары, для которых цвет не определен. Выше эта задача была решена, однако формально допустимо следующее решение:

```
SELECT [Color], COUNT(*) as 'Amount'

FROM [Production].[Product]

GROUP BY [Color]

HAVING [Color] IS NOT NULL
```

Такой запрос будет выполняться дольше, так как сначала сформируются все группы, и только потом будет применено условие для выбора группы – отпрошена группа, где цвет не определён. Время на формирование группы с неопределенным цветом фактически будет потрачено впустую.

Использование GROUP BY допускает использование ORDER BY, TOP.

Существуют специальные конструкции, выполняющие группировку с определёнными правилами. Рассмотрим конструкцию GROUP BY ROLLUP, которая выводить и общие и промежуточные итоги группировки. Частный синтаксис будет такой:

```
SELECT столбец1, столбец2, столбец3, COUNT(*)

FROM таблица

GROUP BY ROLLUP (столбец1, столбец2, столбец3)
```

В данном случае будут выведены итоги как общей группировки по трем столбцам, а также добавлены промежуточные этапы, т.е. строки с группировкой по первым двум столбцам, третий

столбец будет иметь значение NULL, строки с группировкой по первому столбцу, второй и третий будут иметь значение NULL, и строка без группировки, первый, второй и третий столбцы будут иметь значение NULL.

Для понимания ее работы нам понадобиться простой пример группировки.

```
SELECT [Color], [Style], [Class], COUNT(*)

FROM [Production].[Product]

GROUP BY [Color], [Style], [Class]
```

В данном случае пользователь увидит окончательный результат работы группировки, т.е. каждая строка будет содержать информацию о количестве товаров, четвертый столбец, которые попали в группу с одним цветом, одним стилем в этом цвете, и одним классом в этом стиле.

Если выполнить запрос с использованием GROUP BY ROLLUP.

```
SELECT [Color], [Style], [Class], COUNT(*)

FROM [Production].[Product]

GROUP BY ROLLUP ([Color], [Style], [Class])
```

То пользователь получит все те же строки, как и при обычном GROUP BY, а так же дополнительный набор с группировкой по цвету и стилю, дополнительный набор с группировкой только по цвету, и строку без группировки.

Схожим образом работает операция GROUP BY CUBE ( ), только к базовой группировке GROUP BY, добавляются все возможные сочетания параметров группировки. Например, группировка GROUP BY CUBE (столбец1, столбец2), даст все возможные сочетания группировок, базовую группировку (столбец1, столбец2), и возможные сочетания (столбец1,NULL), (NULL,столбец2), (NULL, NULL).

Конструкция GROUP BY GROUPING SETS ( ) позволяет объединить несколько операций GROUP BY.

```
SELECT COUNT(*)

FROM [Production].[Product]

GROUP BY GROUPING SETS (([Color]),([Size]))
```

Данный запрос вернет данный сгруппированные по цвету, и еще набор строк, данные сгруппированные по размеру.

Примеры запросов с ответами.

1. Найти номера первых трех подкатегорий (ProductSubcategoryID) с наибольшим количеством наименований товаров

```
SELECT TOP WITH TIES 3 [ProductSubcategoryID]

FROM [Production].[Product]

WHERE [ProductSubcategoryID] IS NOT NULL

GROUP BY [ProductSubcategoryID]

ORDER BY COUNT(*) DESC
```

2. Разбить продукты по количеству символов в названии, и для каждой группы определить количество продуктов

```
SELECT LEN([Name]), COUNT(*)

FROM [Production].[Product]

GROUP BY LEN([Name])
```

3. Проверить есть ли продукты с одинаковым названием, если есть, то вывести эти названия.

```
SELECT [Name]

FROM [Production].[Product]

GROUP BY [ProductID], [Name]

HAVING COUNT(*)>1
```

Задания для самостоятельного работы:

1. Найти и вывести на экран количество товаров каждого цвета, исключив из поиска товары, цена которых меньше 30;

2. Найти и вывести на экран список, состоящий из цветов товаров, таких что минимальная цена товара данного цвета более 100

3. Найти и вывести на экран номера подкатегорий товаров и количество товаров в каждой категории

4. Найти и вывести на экран номера товаров, и количество фактов продаж данного товара (используется таблица SalesOrderDetail)

5. Найти и вывести на экран список товаров, их номера, которые были куплены более пяти раз

6. Найти и вывести на экран номера покупателей, CustomerID, которые более одного номера чека, SalesOrderID, в один день

7. Найти и вывести на экран все номера чеков, на которые приходится более трех продуктов

8. Найти и вывести на экран все номера продуктов, которые были куплены более трех раз

9. Найти и вывести на экран все номера продуктов, которые были куплены или три или пять раз

10. Найти и вывести на экран все номера подкатегорий, в которым относится более десяти товаров

11. Найти и вывести на экран номера товаров, которые всегда покупались в одном экземпляре за одну покупку

12 Найти и вывести на экран номер чека, SalesOrderID, на который приходится максимальное количество товаров, по на названиям

13 Найти и вывести на экран номер чека, SalesOrderID с наибольшей суммой покупки, исходя из того, что цена товара это UnitPrice, а количество конкретного товара в чеке это OrderQty

14 Определить количество товаров в каждой подкатегории, исключая товары для которых подкатегория не определена, и товары у которых не определен цвет

15 Получить список цветов товаров, в порядке убывания количество товаров данного цвета

16 Вывести на экран товары, их ProductID, такие что всегда покупались в количестве более 1 единицы на один чек, при этом таких покупок было более двух