МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Севастопольский государственный университет»

SQL. АГРЕГАТНЫЕ ФУНКЦИИ

Методические указания
к лабораторной работе №2
по дисциплине
«Управление данными»
для студентов, обучающихся по направлениям
09.03.02 «Информационные системы и технологии» и 09.03.03 «Прикладная информатика» по учебному плану подготовки бакалавров
дневной и заочной форм обучения

Севастополь
2020

УДК 004.92

SQL. Агрегатные функции. Методические указания к лабораторной работе №2 по дисциплине «Управление данными», для студентов, обучающихся по направлениям 09.03.02 «Информационные системы и технологии» и 09.03.03 «Прикладная информатика» по учебному плану подготовки бакалавров дневной и заочной форм обучения /Сост. Ю.В. Доронина, А.В. Волкова - Севастополь: Изд-во СГУ, 2020. – 7 с.

Цель методических указаний: изучить возможности обработки данных с помощью агрегатных функций языка SQL.

1 ОСНОВНЫЕ ПОЛОЖЕНИЯ

1.1 Агрегатные функции

Агрегатные функции – это функции, которые работают не с одним значением, взятым из строки таблицы, а с группой значений.

Общий алгоритм, по которому работают агрегатные функции следующий:

1) производится упорядочивание таблицы по тем полям, по которым осуществляется группировка;

2) от каждой сформированной группы вычисляется требуемое значение, которое и заносится в результат.

Существуют следующие агрегатные функции:

- **COUNT** считает количество строк, которые вернул запрос;
- **SUM** суммирует значение всех полей, по указанному атрибуту;
- **AVG** находит среднее значение поля, по указанному атрибуту;
- **MAX** находит максимальное значение поля, по указанному атрибуту;
- **MIN** находит минимальное значение поля, по указанному атрибуту.

1.2 Тестовая база данных

Рассмотрим работу агрегатных функций на примере базы данных, состоящий из трех таблиц – «Торговый агент», «Покупатель», «Сделка».

Таблица «Торговый агент» *(*Salespeople), представленная на рисунке 1, содержит следующие атрибуты:

- SNUM – номер агента;
- SNAME – имя агента;
- CITY – город, где работает агента;
- COMM – комиссионные торгового агента.

SNUM	SNAME	CITY	COMM
1001	Peel	London	0.12
1002	Serres	San Jose	0.13
1004	Motika	London	0.11
1007	Rifkin	Barcelona	0.15
1003	Axelrod	New York	0.10

Рисунок 1 – Таблица «Торговый агент»

Таблица «Покупатель» (Customers), представленная на рисунке 2, состоит из следующих атрибутов:

- CNUM – номер покупателя;
- CNAME – имя покупателя;
- CITY – город проживания покупателя;
- RATING – некоторый рейтинг, присвоенный покупателю;
- SNUM – номер торгового агента, за которым закреплен покупатель.

CNUM	CNAME	CITY	RATING	SNUM
2001	Hoffman	London	100	1001
2002	Giovanni	Rome	200	1003
2003	Liu	SanJose	200	1002
2004	Grass	Berlin	300	1002
2006	Clemens	London	100	1001
2008	Cisneros	SanJose	300	1007
2007	Pereira	Rome	100	1004

Рисунок 2 - Таблица «Покупатель»

Таблица «Сделка» (Orders), представленная на рисунке 3, содержит следующие атрибуты:
- ONUM – номер сделки;
- AMT – сумма сделки
- ODATE – дата свершения сделки
- CNUM – номер покупателя
- SNUM – номер торгового агента

ONUM	AMT	ODATE	CNUM	SNUM
3001	18.69	10/03/1990	2008	1007
3003	767.19	10/03/1990	2001	1001
3002	1900.10	10/03/1990	2007	1004
3005	5160.45	10/03/1990	2003	1002
3006	1098.16	10/03/1990	2008	1007
3009	1713.23	10/04/1990	2002	1003
3007	75.75	10/04/1990	2004	1002
3008	4723.00	10/05/1990	2006	1001
3010	1309.95	10/06/1990	2004	1002
3011	9891.88	10/06/1990	2006	1001

Рисунок 3 - Таблица «Сделка»

1.3 Действия с базовыми агрегатными функциями

Чтобы найти сумму всех покупок в таблице «Сделки», можно ввести следующий запрос:

```
SELECT SUM(amt) FROM Orders;
```

Результат его выполнения – число 26658.4

Запрос

```
SELECT AVG(amt) FROM Orders;
```

вернет среднее значение сумм сделок -2665.84.

Функция COUNT отличается от других агрегатных функций тем, что она не выполняет математических действий над значением столбца. Она считает число значений в данном столбце, или число строк в таблице. Если необходимо подсчитать количество различных значений некоторого поля в таблице, функцию COUNT надо использовать с DISTINCT. Например, чтобы подсчитать количество продавцов в настоящее время описанных в таблице сделок, можно использовать следующий запрос:

```
SELECT COUNT(DISTINCT snum) FROM Orders;
```

Результат – 5, так как конструкция (**DISTINCT** snum) исключает записи с повторяющимися значениями.

Иногда возникает необходимость решить обратную задачу – подсчитать количество значений поля вместе с повторениями. Для этого существует описатель **ALL** (подразумевается по умолчанию).

Например, необходимо определить количество торговых агентов snum с учетом повторений.

```
SELECT COUNT (ALL snum) FROM Customers;
```

Результат – 7.

Чтобы подсчитать общее число строк в таблице, используется функция COUNT со звездочкой вместо имени поля, как, например, в следующем примере.

```
SELECT COUNT(*) FROM Customers;
```

Результат – 7.

Внимание! Только **COUNT**(*) может подсчитывать значения NULL. Все остальные функции игнорируют неопределенные значения.

1.4 Простые вычисления

Столбцы, значение которых может быть получено с помощью простых арифметических действий через другие столбцы в БД, как правило, не хранятся. SQL предоставляет простой способ производить подобные вычисления. Также можно помещать в некоторый столбец константу.

Например, чтобы представить комиссионные торгового агента в процентном отношении, а не в виде десятичного числа можно записать такой запрос:

```
SELECT snum, sname, city, comm*100 FROM Salespeople;
```

Результат выполнения запроса:

SNUM	SNAME	CITY	MULTIPLY
1001	Peel	London	12.00
1002	Serres	San Jose	13.00
1004	Motika	London	11.00
1007	Rifkin	Barcelona	15.00
1003	Axelrod	New York	10.00

Можно дополнить указанный запрос знаком процента, выводимым после поля comm*100:

```
SELECT snum, sname, city, comm*100 AS amount, ' % ' FROM Salespeople;
```

Результат выполнения запроса:

SNUM	SNAME	CITY	AMOUNT	CONSTANT
1001	Peel	London	12.00	%
1002	Serres	San Jose	13.00	%
1004	Motika	London	11.00	%
1007	Rifkin	Barcelona	15.00	%
1003	Axelrod	New York	10.00	%

Знак процента в данном случае – константное выражение. При выполнении вычислений в запросе допустимы арифметические действия – сложение, вычитание, умножение, деление.

1.5 Вычисления в агрегатных функциях

Допустим, необходимо вывести 10% от стоимости самой дорогой сделки. Тогда можно записать такой запрос:

```
SELECT MAX(amt*0.1) FROM Orders;
```

Результат выполнения запроса:

MAX
989.188

Таким образом, можно выполнять вычисления с использованием других полей и арифметических действий.

Внимание! Вложение агрегатов не допускается.

1.6 Использование GROUP BY

В случае если таблица рассматривается целиком, в списке вывода запроса присутствуют только агрегатные функции.

В случае если производится группировка таблицы по какому-то полю, в списке вывода могут присутствовать те поля, относительно которых таблица группируется и агрегатные функции. При этом все поля, не охваченные агрегатными функциями, должны быть перечислены в предложении GROUP BY.

Предположим, что необходимо найти сделку с максимальной стоимостью для каждого торгового агента. Можно сделать персональный запрос для каждого из них, выбрав MAX(amt) из таблицы сделок. Можно записать следующий запрос:

```
SELECT snum, MAX(amt) FROM Orders GROUP BY snum;
```

Результат выполнения запроса:

snum	MAX
1001	9891.88
1002	5160.45
1003	1713.23
1004	1900.10
1007	1098.16

В данном случае таблица группируется относительно номера торгового агента, поэтому поле snum присутствует в списке вывода запроса и в предложении GROUP BY.

1.7 Использование HAVING

HAVING подобен WHERE – он задает условия отбора групп строк так же, как это делает WHERE для каждой строки.

Внимание! В предложении HAVING нельзя проверять имена атрибутов на какое-либо условие – для этого существует WHERE. То, что проверяется в HAVING должно иметь только одно значение для группы.

Например, если необходимо узнать, какие торговые агенты имеют заработок от одной сделки более чем 3000, и в какой день, можно использовать запрос вида:

```
SELECT snum, odate, MAX(amt) FROM Orders
GROUP BY snum, odate HAVING MAX (amt) > 3000.00;
```

Результат работы запроса:

Snum	Odate	MAX
1001	10/05/1990	4723.00
1001	10/06/1990	9891.88
1002	10/03/1990	5160.45

2 ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

1. Ознакомиться с принципами работы агрегатных функций COUNT, SUM, AVG, MAX, MIN.
2. Продемонстрировать использование COUNT(*).
3. Продемонстрировать выполнение простых вычислений в запросе.
4. Использовать простое вычисление как параметр агрегатной функции.
5. Ознакомиться с использованием предложения GROUP BY, продемонстрировать его работу.
6. Ознакомиться с использованием предложения HAVING, продемонстрировать его работу.
7. Составить и защитить отчет.

3 ЗАДАНИЕ НА РАБОТУ

Вариант задания, соответствует варианту, полученному в Лабораторной работе №1.

4 СОДЕРЖАНИЕ ОТЧЕТА

1. Отчет состоит из титульного листа, цели работы, описания процесса выполнения работы и вывода.

2. Отчет должен содержать таблицу исходных данных, тексты запросов и результаты их выполнения.

5 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение агрегатных функций?
2. Возможности предложения GROUP BY?
3. Условия отбора групп. Предложения HAVING и WHERE назначение и отличия в использовании?
4. Простые вычисления над данными?
5. Требования к списку выводимых столбцов фразы SELECT при задании группировки таблицы по какому-либо полю?