Частное учреждение образования
«Колледж бизнеса и права»

УТВЕРЖДАЮ
Заместитель директора
по учебной работе
_____И.В.Малафей
«___»_____2017

Специальность: «Программное обеспечение информационных технологий»	Дисциплина: «Базы данных и системы управления базами данных»

Лабораторная работа № 9

Инструкционно-технологическая карта

Тема: Итоговые запросы на выборку
Цель работы:

– изучить статистические функции, правила обработки значений NULL в статистических функциях;

– познакомиться с правилами выполнения запросов с группировкой CROUP BY. Изучить на примерах особенности выполнения запросов с группировкой;

– познакомиться с правилами выполнения запросов, содержащих условие отбора групп (предложение HAVING). Изучить на примерах особенности выполнения запросов с группировкой, содержащих условие отбора.

Время выполнения: 2 часа

Краткие теоретические сведения

Многие запросы к базе данных не нуждаются в той степени детализации, которую обеспечивают SQL-запросы. Например, во всех перечисленных ниже запросах требуется узнать всего одно или несколько значений, которые подытоживают информацию, содержащуюся в базе данных:

- Какова наименьшая и наибольшая оценка по первому экзамену?
- Какова средняя оценка по первому экзамену?
- Сколько студентов учится в университете?

В SQL запросы такого типа можно создавать с помощью статистических функций и предложений *group by* и *having* инструкции *select.*

Статистические функции

Для подведения итогов по информации, содержащейся в базе данных, в SQL предусмотрены статистические (агрегатные) функции. Статистическая функция принимает в качестве аргумента какой-либо столбец данных целиком, а

возвращает одно значение, которое определенным образом подытоживает этот столбец. Например, функция avg () принимает в качестве аргумента столбец чисел и вычисляет их среднее значение.

Какая средняя оценка по первому и второму экзамену?

```
select avg([оценка 1]) as 'средняя оценка по 1-му экзамену', avg([оценка
2])as 'средняя оценка по 2-му экзамену'
from оценки

средняя оценка по 1-му экзамену средняя оценка по 2-му экзамену
-------------------------------- --------------------------------
3                                3
```

Первая статистическая функция принимает в качестве аргументов все значения, содержащиеся в столбце *оценка 1,* и вычисляет их среднее значение; вторая функция подсчитывает среднее значение столбца *оценка 2.* Результатом запроса является одна строка, представляющая итоги по информации, содержащейся в таблице *оценки.*

Как представлено на рис. 1, в SQL имеется шесть статистических функций, которые позволяют получать различные виды итоговой информации:

- функция ***sum*** () вычисляет сумму всех значений столбца;
- функция ***avg*** () вычисляет среднее всех значений столбца;
- функция ***min*** () находит наименьшее среди всех значений столбца;
- функция *max* () находит наибольшее среди всех значений столбца;
- функция ***count*** () подсчитывает количество значений, содержащихся в столбце;
- функция ***count*** (*) подсчитывает количество строк в таблице результатов запроса.

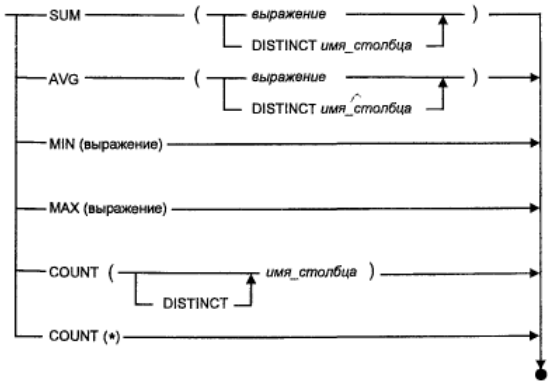

Рис 1. Синтаксическая диаграмма статистических функций

Вычисление суммы значений столбца (функция SUM)

Статистическая функция *sum()* вычисляет сумму всех значений столбца. При этом столбец должен иметь числовой тип данных (содержать целые числа, десятичные числа, числа с плавающей запятой или денежные величины). Результат, возвращаемый этой функцией, имеет тот же тип данных, что и столбец, однако точность результата может быть выше.

Вычисление среднего значения столбца (функция AVG)

Статистическая функция *avg*() вычисляет среднее всех значений столбца. Как и в случае с функцией *sum*(), данные, содержащиеся в столбце, должны иметь числовой тип. Поскольку функция *avg*() вначале суммирует все значения, содержащиеся в столбце, а затем делит сумму на число этих значений, возвращаемый ею результат может иметь не такой тип данных, как столбец. Например, если применить функцию *avg*() к столбцу целых чисел, результат будет либо десятичным числом, либо числом с плавающей запятой, в зависимости от используемой СУБД.

Какая средняя оценка по первому и второму экзамену?

```
select avg([оценка 1]) as 'средняя оценка по 1-му экзамену', avg([оценка
2])as 'средняя оценка по 2-му экзамену'
from оценки
```

```
средняя оценка по 1-му экзамену средняя оценка по 2-му экзамену
-------------------------------- --------------------------------
3                                3
```

Вычислить средний бал по первому экзамену среди студентов с кодом 1,3,5

```
select avg([оценка 1]) as 'средняя оценка по 1-му экзамену'
from оценки
where [код студента] in (1,3,5)
```

```
средняя оценка по 1-му экзамену
--------------------------------
4
```

Вычисление экстремумов (функции MIN и MAX)

Статистические функции *min*() и *max*() позволяют найти соответственно наименьшее и наибольшее значения в столбце. При этом столбец может содержать числовые или строковые значения либо значения даты/времени. Результат, возвращаемый этими функциями, имеет точно такой же тип данных, что и сам столбец.

Вывести студентов, у которых сдан первый экзамен на максимальную оценку

```
select max([оценка 2]) as 'максимальная оценка по 2-му экзамену',
min([оценка 2]) as 'минимальная оценка по 2-му экзамену'
from оценки
```

```
максимальная оценка по 2-му экзамену минимальная оценка по 2-му экзамену
------------------------------------ ------------------------------------
5                                    2
```

Кто из студентов родился раньше всех

```
select min([дата рождения]) as 'самая ранняя дата рождения'
from студенты
```

```
самая ранняя дата рождения
--------------------------
1979-02-26 00:00:00.000
```

В случае применения функций *min*() и *max*() к числовым данным числа сравниваются по арифметическим правилам (среди двух отрицательных чисел меньше то, у которого модуль больше; нуль меньше любого положительного числа и больше любого отрицательного). Сравнение дат происходит последовательно (более ранние значения дат считаются меньшими, чем более

поздние). Сравнение интервалов времени выполняется на основании их продолжительности (более короткие интервалы времени считаются меньшими, чем более длинные).

В случае применения функций *min*() и *max*() к строковым данным результат сравнения двух строк зависит от используемой таблицы кодировки. На персональных компьютерах и мини-компьютерах, где используется таблица кодировки ASCII, установлен порядок сортировки, при котором цифры идут перед буквами, а все прописные буквы — перед строчными. Там, где используется таблица кодировки EBCDIC, строчные символы расположены перед прописными, а цифры следуют за буквами.

Отличия в порядке сортировки приводят к тому, что один и тот же запрос, содержащий предложение *order by, в* различных системах может привести к различным результатам.

Хранение в таблицах символов национальных алфавитов (например, кириллицы) может вызвать дополнительные проблемы. В некоторых СУБД для каждого языка используется свой алгоритм сортировки. В других СУБД такие символы сортируются в соответствии с кодом символа. Для решения этой проблемы в стандарт SQL2 включено условие поддержки национальных наборов символов, пользовательских наборов символов и альтернативных последовательностей сортировки. К сожалению, в настоящее время эти возможности реализованы не во всех СУБД. Если в приложении используются символы национальных алфавитов, необходимо проверить, как конкретная СУБД обрабатывает их.

Вычисление количества значений в столбце (функция COUNT)

Статистическая функция *count()* подсчитывает количество значений в столбце. При этом тип данных столбца может быть любым. Функция *count()* всегда возвращает целое число независимо от типа данных столбца.

Какое количество специальностей предлагает ВУЗ

```
select count([код специальности]) as 'количество специальностей'
from специальности
```

```
количество специальностей
-------------------------
5
```

Сколько студентов учится на 5 курсе

```
select count(ФИО) as 'количество студентов на 5 курсе'
from студенты
where курс=5
```

```
количество студентов на 5 курсе
-------------------------------
3
```

Следует заметить, что для функции *count()* важны не конкретные значения ячеек, а важно количество ячеек, удовлетворяющих заданному критерию. Поэтому на самом деле не важно, какой столбец указывать в качестве аргумента. Предыдущий запрос можно переписать и так:

```
select count([код студента]) as 'количество студентов на 5 курсе'
from студенты
```

```
where курс=5
```

В SQL была введена специальная статистическая функция *count (*),* которая подсчитывает строки, а не значения данных. Ниже приведен предыдущий запрос, переписанный с использованием этой функции:

```
select count(*)
from студенты
where курс=5
```

Если применять *count(*) в* качестве функции подсчета строк, то запрос становится более удобочитаемым На практике для подсчета строк всегда используется Функция *count(*)*, а не *count().*

Статистические функции в списке возвращаемых столбцов

Назначение простого запроса, в котором участвуют статистические функции, можно понять достаточно легко. Однако если в список возвращаемых столбцов входит несколько таких функций или если аргументом функции является сложное выражение, понять запрос становится значительно сложнее. В табл. 1 приведены правила выполнения SQL-запросов, расширенные с учетом применения статистических функций. Эти правила являются точным определением того, что означает запрос, а не описанием того, как СУБД на самом деле получает результаты запроса.

Таблица 1. Правила выполнения SQL-запросов

Таблица результатов запроса на выборку генерируется следующим образом:
1. Если запрос представляет собой запрос на объединение (UNION) инструкций SELECT, для каждой из этих инструкций выполнить действия 2—5 и получить отдельную таблицу результатов.
2. Сформировать произведение таблиц, перечисленных в предложении FROM. Если в предложении FROM указана только одна таблица, то произведением будет она сама.
3. Если имеется предложение WHERE, применить заданное в нем условие отбора к каждой строке таблицы произведения и оставить в ней только те строки, для которых это условие выполняется, т.е. имеет значение TRUE; строки, для которых условие отбора имеет значение FALSE или NULL, — отбросить.
4. Для каждой из оставшихся строк вычислить значение каждого элемента в списке возвращаемых столбцов и создать одну строку таблицы результатов запроса. При любой ссылке на столбец берется значение столбца для текущей строки. В качестве аргумента статистической функции используется весь набор строк.
5. Если указан предикат DISTINCT, удалить из таблицы результатов запроса все повторяющиеся строки.
6. Если запрос является запросом на объединение (UNION) инструкций SELECT, объединить результаты выполнения отдельных инструкций в одну таблицу результатов запроса. Удалить из нее повторяющиеся строки, если не указан предикат ALL.
7. Если имеется предложение ORDER BY, отсортировать результаты запроса.

Один из лучших способов понять, как выполняются итоговые запросы со статистическими функциями, — представить запрос разбитым на два этапа. Сначала подумайте, как работал бы запрос без статистических функций, возвращая несколько строк результатов. Затем представьте, как СУБД применяет статистические функции к результатам запроса, возвращая одну итоговую строку.

В списке возвращаемых столбцов вместо имени любого столбца можно указывать статистическую функцию. Например, она может входить в выражение, в котором суммируются или вычитаются значения двух статистических функций. Однако статистическая функция не может быть аргументом для другой такой

функции, поскольку получающееся в подобном случае выражение не имеет смысла. Иногда это правило формулируют таким образом: **"Нельзя вкладывать статистические функции одна в другую"**.

Кроме того, в списке возвращаемых столбцов нельзя одновременно использовать статистические функции и обычные имена столбцов, поскольку в этом также нет смысла. Например, рассмотрим следующий запрос:

```
select ФИО, count([код студента])
from студенты
```

Первый элемент списка возвращаемых столбцов просит СУБД создать таблицу, которая будет состоять из девяти строк и содержать обычные результаты запроса — по одной строке для каждого студента. Второй элемент списка возвращаемых столбцов просит СУБД получить одно результирующее значение, представляющее собой количество столбцов *Студенты*. Между требованиями, одновременно предъявляемыми к двум возвращаемым столбцам, возникает противоречие, что приводит к ошибке. По этой причине либо все ссылки на столбцы в списке возвращаемых столбцов должны являться аргументами статистических функций (и тогда запрос возвращает итоговые результаты), либо в списке возвращаемых столбцов не должно быть ни одной статистической функции (и тогда запрос возвращает обычные результаты). На самом деле это правило несколько сложнее из-за необходимости обработки подчиненных запросов и запросов с группировкой.

Статистические функции и значения NULL

Функции *sum(), avg(), min(), max()* и *count()* в качестве аргумента принимают столбец значений и возвращают в качестве результата одно значение. А что происходит, когда в столбце встречается одно или несколько значений *null?* В стандарте ANSI/ISO сказано, что значения *null* статистическими функциями *игнорируются*.

Игнорирование значений *null* не оказывает влияния на результаты, возвращаемые статистическими функциями *min()* и *max()*. Однако оно может привести к проблемам при использовании функций *sum()* и *avg()*.

В стандарте ANSI/ISO определены следующие точные правила обработки значений *null* в статистических функциях:

- если какие-либо из значений, содержащихся в столбце, равны *null*, при вычислении результата функции они исключаются;
- если все значения в столбце равны *null*, то функции *sum(), avg(), min()* и *max()* возвращают значение *null*; функция *count()* возвращает ноль;
- если в столбце нет значений (т.е. столбец пустой), то функции *sum(), avg(), min()* и *max()* возвращают значение *null*; функция *count()* возвращает ноль;
- функция *count(*)* подсчитывает количество строк и не зависит от наличия или отсутствия в столбце значений *null*; если строк в таблице нет, эта функция возвращает ноль.

Удаление повторяющихся строк (предикат DISTINCT)

Предикат *distinct* указывается в начале списка возвращаемых столбцов и служит для удаления повторяющихся строк из таблицы результатов запроса. С помощью этого предиката можно также указать, что перед применением

статистической функции к столбцу из него следует удалить все повторяющиеся значения. Для этого необходимо включить предикат *distinct* перед аргументом статистической функции сразу же после открывающей круглой скобки.

Ниже приведен запрос, который иллюстрируетт удаление повторяющихся значений перед применением статистических функций

В скольких различных городах живут студенты

```
select count(distinct адрес) as 'количество городов'
from студенты
```

```
количество городов
------------------
6
```

При использовании предиката *distinct в* статистической функции ее аргументом должно быть простое имя столбца; аргумент не может быть выражением. Стандарт позволяет использовать предикат *distinct в* функциях *sum()* и *avg()* и не разрешает — в функциях *min ()* и *мах ()*, поскольку в этом нет смысла; тем не менее, в ряде СУБД подобное все-таки допустимо. Кроме того, стандарт *требует* применения предиката *distinct в* функции *count (),* но в ряде СУБД можно использовать функцию *count()* и без него. В функции *count(*)* данный предикат применять нельзя, поскольку она вообще не имеет отношения к столбцам, а просто подсчитывает число строк. В стандарте SQL2 упомянутые ограничения сняты и разрешается использовать предикат *distinct* во всех статистических функциях, а также применять выражения в качестве аргументов дня всех функций.

Кроме того, предикат *distinct в* одном запросе можно употребить только *один* раз. Если он применяется вместе с аргументом одной из статистических функций, его нельзя использовать ни с одним другим аргументом. Если он указан перед списком возвращаемых столбцов, его нельзя употреблять ни в одной статистической функции. Единственным исключением из этого правила является случай, когда предикат *distinct* используется внутри подчиненного запроса, входящего в запрос, в котором предикат уже применяется.

Запросы с группировкой (предложение GROUP BY)

Результаты итоговых запросов, о которых до сих пор шла речь в настоящей главе, напоминают итоговую информацию, находящуюся обычно в конце отчета Эти запросы "сжимают" подробные данные, содержащиеся в отчете, в одну строку итоговых результатов. Но, как известно, в отчетах иногда используются также промежуточные итоги. И точно так же бывает необходимо получать промежуточные итоги результатов запроса. Эту возможность предоставляет предложение *group by* инструкции *select.*

Назначение предложения *group by* проще всего понять на примере.

Какое количество студентов каждого курса обучения

```
select курс, count(фио) as 'количество студентов'
from студенты
group by курс
```

```
курс количество студентов
---- --------------------
1    2
2    2
3    1
4    2
```

```
5     3
```

На логическом уровне запрос выполняется следующим образом:

1. студенты делятся на курсы. На каждом курсе по несколько студентов
2. для каждого курса вычисляется количество столбцов ФИО и генерируется одна итоговая строка результатов. Это строка содержит значение столбца курс и количество студентов на каждом курсе.

Запрос, включающий в себя предложение *group by*, называется запросом с группировкой, поскольку он объединяет строки исходных таблиц в группы и для каждой группы строк генерирует одну строку в таблице результатов запроса. Столбцы, указанные в предложении *group by*, называются **столбцами группировки**, поскольку именно они определяют, по какому признаку строки делятся на группы.

Посчитать количество студентов, которые получили по первому экзамену 3,4,5

```
      select [оценка 1] as 'оценка по первому экзамену', count([код студента])
as 'количество студентов'
      from оценки
      where [оценка 1] in(3,4,5)
      group by [оценка 1]
```

```
оценка по первому экзамену количество студентов
-------------------------- --------------------
3                          2
4                          4
5                          3
```

Между статистическими функциями SQL и предложением *group by* существует связь Статистическая функция берет столбец значений и возвращает одно значение. Предложение *group by* указывает, что результаты запроса следует разделить на группы, применить статистическую функцию по отдельности к каждой группе и получить для каждой группы одну строку результатов. В таблице 2 показаны правила выполнения SQL-запроса на выборку, расширенные с учетом операции группировки.

Таблица 2. Правила выполнения SQL-запроса на выборку с учетом операции группировки.

1. Если запрос представляет собой запрос на объединение (UNION) инструкций SELECT, для каждой из этих инструкций выполнить действия 2—6 и получить отдельную таблицу результатов.
2. Сформировать произведение таблиц, перечисленных в предложении FROM. Если в предложении FROM указана только одна таблица, то произведением будет она сама.
3. Если имеется предложение WHERE, применить заданное в нем условие отбора к каждой строке таблицы произведения и оставить в ней только те строки, для которых это условие выполняется, т.е. имеет значение TRUE; строки, для которых условие отбора имеет значение FALSE или NULL, — отбросить
4. Если имеется предложение GROUP BY, разделить строки, оставшиеся в таблице произведения, на группы таким образом, чтобы строки в каждой группе имели одинаковые значения во всех столбцах группировки
5. Для каждой из оставшихся строк (или для каждой группы строк) вычислить значение каждого элемента в списке возвращаемых столбцов и создать одну строку таблицы результатов запроса. При любой ссылке на столбец берется значение столбца для текущей строки (или группы строк). В качестве аргумента статистической функции используются значения столбца из всех строк, входящих в группу, если указано предложение GROUP BY; в противном случае используются значения столбца из всех строк таблицы результатов.
6. Если указан предикат DISTINCT, удалить из таблицы результатов запроса все повторяющиеся строки.
7. Если запрос является запросом на объединение (UNION) инструкций SELECT, объединить результаты выполнения отдельных инструкций в одну таблицу результатов запроса. Удалить из нее повторяющиеся строки, если не указан предикат ALL.
8. Если имеется предложение ORDER BY, отсортировать результаты запроса

SQL позволяет группировать результаты запроса на основании двух или более столбцов. Например, предположим, что вам требуется сгруппировать студентов по оценкам и предметам. Эту задачу выполняет приведенный ниже запрос.

Посчитать сколько студентов сдали первый экзамен на 3,4,5 по соответствующим предметам

```
select [код предмета 1], [оценка 1] as 'оценка по первому экзамену',
count([код студента]) as 'количество студентов'
from оценки
where [оценка 1] in(3,4,5)
group by [оценка 1], [код предмета 1]
```

```
код предмета 1       оценка по первому экзамену количество студентов
-------------------- -------------------------- --------------------
1                    5                          2
2                    3                          1
2                    4                          1
3                    3                          1
3                    5                          1
4                    4                          2
5                    4                          1
```

Обратите внимание на то, что с помощью одного запроса невозможно получить как детальные, так и промежуточные итоговые результаты. Чтобы получить детальные результаты с итогами по группам, необходимо с помощью программного SQL написать приложение и вычислить промежуточные итоговые результаты в программе. В SQL Server это ограничение стандартного SQL устраняется с помощью дополнительного предложения *compute*, которое добавляется в конец инструкции *select*. Предложение *compute* обеспечивает получение промежуточных и общих итогов, как показано в следующем примере:

Вывести среднюю оценку по первому экзамену за каждый предмет

```
select [наименование предмета], [оценка 1] as 'оценка по первому
экзамену'
from оценки, предметы
where предметы.[код предмета] = оценки.[код предмета 1]
order by [код предмета 1], [оценка 1]
compute avg([оценка 1]) by [код предмета 1]
```

```
наименование предмета                              оценка по первому экзамену
-------------------------------------------------- --------------------------
Операционные системы                               5
Операционные системы                               5
avg
-----------
5
наименование предмета                              оценка по первому экзамену
-------------------------------------------------- --------------------------
Офисные пакеты                                     3
Офисные пакеты                                     4
avg
-----------
3
наименование предмета                              оценка по первому экзамену
-------------------------------------------------- --------------------------
Базы данных                                        3
Базы данных                                        5
avg
-----------
4
наименование предмета                              оценка по первому экзамену
-------------------------------------------------- --------------------------
Языки програмирования                              4
```

```
Языки програмирования                                4
avg
-----------
4
наименование предмета                                оценка по первому экзамену
---------------------------------------------------- -------------------------
Проектирование информационных систем                 2
Проектирование информационных систем                 4
avg
-----------
3
```

Приведенный выше запрос генерирует обычные результаты, содержащие одну строку для каждой строки таблицы *Оценки* и отсортированные по столбцам *оценка 1* и *код предмета 1*. Кроме того, он вычисляет средний балл по первому экзамену для каждого из предметов.

Ограничения на запросы с группировкой

На запросы, в которых используется группировка, накладываются дополнительные ограничения. Столбцы с группировкой должны представлять собой реальные столбцы таблиц, перечисленных в предложении *from*. Нельзя группировать строки на основании значения вычисляемого выражения.

Кроме того, существуют ограничения на элементы списка возвращаемых столбцов. Все элементы этого списка должны иметь одно значение для каждой группы строк. Это означает, что возвращаемым столбцом может быть:

- *константа;*
- *статистическая функция,* возвращающая одно значение для всех строк, входящих в группу;
- *столбец группировки,* который по определению имеет одно и то же значение во всех строках группы;
- *выражение,* включающее в себя перечисленные выше элементы.

На практике в список возвращаемых столбцов запроса с группировкой всегда входят столбец группировки и статистическая функция. Если последняя не указана, значит, запрос можно более просто выразить с помощью предиката *distinct* без использования предложения *group by*. И наоборот, если не включить в результаты запроса столбец группировки, вы не сможете определить, к какой группе относится каждая строка результатов!

Еще одно ограничение запросов с группировкой обусловлено тем, что в SQL игнорируется информация о первичных и внешних ключах при анализе правильности запроса с группировкой.

Значения NULL в столбцах группировки

Когда в столбце группировки содержится значение *null*, возникают дополнительные осложнения. Если значение столбца неизвестно, к какой группе его следует отнести? В предложении *where* при сравнении двух значений *null* результат имеет значение *null* (а не *true*), т.е. два значения *null* не считаются одинаковыми. Если такое соглашение применить в предложении *group by*, это приведет к тому, что каждая строка со значением *null* в столбце группировки будет помещена в отдельную группу, состоящую из одной этой строки.

На практике это правило очень неудобно. Поэтому в стандарте ANSI/ISO определено, что два значения *null* в предложении *group by* равны. Если две строки имеют значение *null* в одинаковых столбцах группировки и идентичные значения во всех остальных столбцах группировки, они помещаются в одну группу.

Хотя такой принцип обработки значений *null* определен в стандарте SQL, он реализован не во всех диалектах SQL. Прежде чем рассчитывать на определенные результаты, следует протестировать свою СУБД.

Условия отбора групп (предложение HAVING)

Точно так же, как предложение *where* используется для отбора отдельных строк, участвующих в запросе, предложение *having* можно применить для отбора групп строк. Его формат соответствует формату предложения *where*. Предложение *having* состоит из ключевого слова *having*, за которым следует условие отбора. Таким образом, данное предложение определяет условие отбора для групп строк.

Вывести список студентов очной формы обучения, которые поступили в 2005 году.

В предложении *having* указываются точно такие же условия отбора, что и в предложении *where.*

Таблица 3. Правила выполнения SQL-запросов на выборку с учетом HAVING

Таблица результатов запроса на выборку генерируется следующим образом:

1. Если запрос представляет собой запрос на объединение (UNION) инструкций SELECT, для каждой из этих инструкций выполнить действия 2—7 и получить отдельную таблицу результатов.
2. Сформировать произведение таблиц, перечисленных в предложении FROM. Если в предложении FROM указана только одна таблица, то произведением будет она сама.
3. Если имеется предложение WHERE, применить заданное в нем условие отбора к каждой строке таблицы произведения и оставить в ней только те строки, для которых это условие выполняется, т.е. имеет значение TRUE; строки, для которых условие отбора имеет значение FALSE или NULL, — отбросить.
4. Если имеется предложение GROUP BY, разделить строки, оставшиеся в таблице произведения, на группы таким образом, чтобы строки в каждой группе имели одинаковые значения во всех столбцах группировки.
5. Если имеется предложение HAVING, применить заданное в нем условие отбора к каждой группе строк и оставить в таблице произведения только те группы, для которых это условие выполняется, т.е. имеет значение TRUE; группы, для которых условие отбора имеет значение FALSE или NULL, — отбросить.
6. Для каждой из оставшихся строк (или для каждой группы строк) вычислить значение каждого элемента в списке возвращаемых столбцов и создать одну строку в таблице результатов запроса. При любой ссылке на столбец берется значение столбца для текущей строки (или группы строк). В качестве аргумента статистической функции используются значения столбца из всех строк, входящих в группу, если указано предложение GROUP BY; в противном случае используются значения столбца из всех строк таблицы результатов.
7. Если указан предикат DISTINCT, удалить из таблицы результатов запроса все повторяющиеся строки.
8. Если запрос является запросом на объединение (UNION) инструкций SELECT, объединить результаты выполнения отдельных инструкций в одну таблицу результатов запроса. Удалить из нее повторяющиеся строки, если не указан предикат ALL.
9. Если имеется предложение ORDER BY, отсортировать результаты запроса.

Ограничения на условия отбора групп

Предложение *having* используется для того, чтобы включать и исключать группы строк из результатов запроса, поэтому используемое в нем условие отбора применяется не к отдельным строкам, а к группе в целом. Это значит, что в условие отбора может входить:

- *константа;*

- *статистическая функция*, возвращающая одно значение для всех строк, входящих в группу;
- *столбец группировки,* который по определению имеет одно и то же значение во всех строках группы;
- *выражение,* включающее в себя перечисленные выше элементы.

На практике условие отбора предложения *having* всегда должно включать в себя как минимум одну статистическую функцию. Если это не так, значит, условие отбора можно переместить в предложение *where*. Чтобы определить, где следует указать условие отбора — в предложении *where* или *having*, — необходимо вспомнить, как применяются эти предложения:

- предложение WHERE применяется к отдельным строкам, поэтому выражения, содержащиеся в нем, должны вычисляться для отдельных строк;
- предложение *having* применяется к группам строк, поэтому выражения, содержащиеся в нем, должны вычисляться для групп строк.

Значения NULL и условия отбора групп

Как и условие отбора в предложении *where,* условие отбора в предложении *having* может дать один из следующих результатов:

- если условие отбора имеет значение *true,* группа строк остается и для нее генерируется одна строка результатов запроса;
- если условие отбора имеет значение *false* или *null,* группа строк исключается и строка для нее в результатах запроса не генерируется.

Правила обработки значений *null в* условиях отбора для предложения *having* точно такие же, как и для предложения *where.*

Предложение HAVING без GROUP BY

Предложение *having* почти всегда используется в сочетании с предложением *group by,* однако синтаксис инструкции *select* не требует этого. Если предложение *having* используется без предложения *group by,* СУБД рассматривает полные результаты запроса как одну группу. Другими словами, статистические функции, содержащиеся в предложении *having,* применяются к одной и только одной группе, и эта группа состоит из всех строк. На практике предложение *having* очень редко используется без соответствующего предложения *group by.*

Порядок выполнения работы

1. Изучить теоретический материла.
2. Выполнить все примерами и проверить результаты выполнения запроса на соответствие данным в базе данных.
3. Получить у преподавателя задание для индивидуальной работы.
4. Ответить на контрольные вопросы.

Контрольные вопросы

1. Перечислите и поясните статистические (агрегатные) функции.
2. Опишите правила выполнения SQL- запроса на выборку (со статистическими функциями).
3. Как называются запросы, включающие в себя предложение *group by*.
4. Как называются столбцы, указанные в предложении *group by*.

5. Перечислите ограничения на запросы с группировкой.
6. Опишите правила выполнения SQL-запроса на выборку с учетом предложения having.
7. Перечислите ограничения на условия отбора групп

Преподаватель С.В. Банцевич

Рассмотрено на заседании цикловой
комиссии программного обеспечения
информационных технологий №10
Протокол № от « »_____201_ .
Председатель ЦК Т.Г.Багласова