Тема 4. Язык SQL-Structured Query Language

Часть 1_2 (продолжение)

1. Многотабличные запросы
2. Агрегатные функции
3. Операторы UNION, EXCEPT, INTERSECT

Многотабличные запросы

Многотабличные запросы

Операция соединения

Если результирующая таблица запроса должна содержать столбцы из разных исходных таблиц, то целесообразно использовать механизм соединения таблиц.

Многотабличные запросы

Декартово произведение – способ 1

- **Вывести все возможные пары «продавец – заказчик».**

```
SELECT   *
FROM Customers , Salespeople
```

Многотабличные запросы

Декартово произведение – способ 2

- **Вывести все возможные пары продавец – заказчиков.**

```
SELECT   *
FROM Customers CROSS JOIN
          Salespeople
```

Многотабличные запросы

Операция соединения

Соединение трактуется как выборка из декартова произведения.

Следовательно, допустимо

```
SELECT  *
FROM Имя_таблицы1, Имя_таблицы2
WHERE условие_соединения
```

Многотабличные запросы

Операция соединения

Доступ к полям из разных таблиц осуществляется через точку, следующим образом:

Имя_таблицы.*имя_поля*

Многотабличные запросы

Псевдонимы таблиц

- **Вместо имен таблиц можно использовать *псевдонимы*, назначенные им в конструкции FROM.**
- **В этом случае имена таблиц и назначаемые им псевдонимы должны разделяться пробелами.**

Многотабличные запросы

Псевдонимы таблиц

```
SELECT   *
FROM Имя_таблицы1  псевдоним1,
     Имя_таблицы2  псевдоним2
        …
WHERE
```

Многотабличные запросы

Псевдонимы **могут использоваться с целью уточнения имен столбцов во всех тех случаях, когда возможна неоднозначность в отношении того, к какой таблице относится тот или иной столбец.**

Псевдонимы могут применяться в любом месте, где требуется указание имени таблицы (для которой он определен).

Многотабличные запросы

- **Пример 18.** **Вывести всех заказчиков, основные продавцы которых работают в Москве.**

```
SELECT   *
FROM Customers c, Salespeople s
WHERE c.snum = s.snum and
          s.city = ‘Москва’
```

Многотабличные запросы

Простое (внутреннее) соединение:

- **выборка из декартова произведения**
- **INNER JOIN (или JOIN)**

Многотабличные запросы

Пример 18. способ 2

```
SELECT   *
FROM Customers c INNER JOIN
   Salespeople s ON  c.snum = s.snum
WHERE  s.city = ‘Москва’
```

```
SELECT   *
FROM Customers c JOIN Salespeople
   s ON  c.snum = s.snum
WHERE  s.city = ‘Москва’
```

Выполнение соединения

1. Формируется декартово произведение таблиц, указанных в конструкции FROM
2. Применение условий поиска к каждой строке таблицы декартова произведения (при наличии WHERE).
3. Формируется каждая отдельная строка результирующей таблицы.

Выполнение соединения

4. **Если в исходном запросе присутствует конструкция SELECT DISTINCT, из результирующей таблицы удаляются все строки-дубликаты.**
5. **Если запрос содержит конструкцию ORDER BY, осуществляется переупорядочивание строк результирующей таблицы.**

Многотабличные запросы

- **Чаще всего многотабличные запросы выполняются для двух таблиц, соединенных связью типа «1:М» или «М:1»**
- **Однако возможны и другие соединения**

Многотабличные запросы

Пример 19. Подобрать продавцов и заказчиков, живущих в одном городе

```
SELECT *
FROM Salespeople s JOIN Customers c
                   ON  s.city = c.city
```

Многотабличные запросы

Объединение таблицы с самой собой

Пример. Вывести все пары заказчиков, имеющих одинаковый рейтинг

```
SELECT s.cname, c.cname, s.rating
FROM Customers s JOIN Customers c
                 ON s.rating = c.rating
                 and s.cnum < c.cnum
```

Многотабличные запросы

Внешнее соединение

- LEFT OUTER JOIN – левое внешнее соединение (LEFT JOIN)
- RIGHT OUTER JOIN – правое внешнее соединение (RIGHT JOIN)
- FULL OUTER JOIN – полное внешнее соединение (FULL JOIN)

Многотабличные запросы: Внешнее соединение

- ***Пример 20.*** **Вывести информацию о продавцах с указанием данных по всем заказчикам, с которыми они работают. Если продавец не работает ни с одним заказчиком, то в соответствующих колонках выводить NULL.**

```
SELECT   *
FROM Salespeople s LEFT JOIN Customers c
   ON  c.snum = s.snum
```

Самостоятельно

1. **Напишите запрос, который выдавал бы имена продавца и заказчика для каждого заказа после номера заказа.**
2. **Напишите запрос, который выводил бы всех заказчиков, обслуживаемых продавцом с комиссионными выше 12%. Выведите имя заказчика, имя продавца и ставку комиссионных продавца.**
3. **Напишите запрос, который вычислил бы сумму комиссионных продавца для каждого заказа заказчика с оценкой выше 100.**

Самостоятельно

1. **Напишите запрос, который вывел бы все пары продавцов, живущих в одном и том же городе. Исключите комбинации продавцов с самими собой, а также дубликаты строк, выводимых в обратным порядке.**
2. **Напишите запрос, который выведет все пары заказов по данным заказчикам, имена этих заказчиков и исключит дубликаты из вывода, как в предыдущем вопросе.**
3. **Напишите запрос, который вывел бы имена (cname) и города (city) всех заказчиков.**

Агрегатные функции

Агрегатные функции

- **Агрегатные функции используются для обобщения данных и всегда возвращают одиночное значение.**
- **Аргументами функций являются поля набора данных.**
- ***Замечание*:**
  - **Нельзя одновременно использовать агрегатные функции и поля таблицы в списке SELECT (если отсутствует GROUP BY)**

Агрегатные функции

- **COUNT – количество строк или не-NULL значений полей в запросе**
- **SUM – сумма значений выбранного поля**
- **AVG – среднее значение выбранного поля**
- **MAX – максимальное значение выбранного поля**
- **MIN – минимальное значение выбранного поля**

Агрегатные функции

- **Функции SUM и AVG могут использовать только числовые поля**
- **Функции COUNT, MAX, MIN применяются как к числовым, так и к символьным полям**

Агрегатные функции

- **Функция COUNT**
- **В отличие от других функций, функция COUNT может применяться как к одиночному полю, так и ко всем полям.**

Агрегатные функции

- **Функция COUNT**

Синтаксис 1: COUNT(*)

подсчитывает количество строк, возвращаемых запросом, NULL-значения также учитываются (применяется ко всем полям)

Недопустимо: COUNT(DISTINCT *)

Агрегатные функции

- **Пример 21. Подсчитать количество заказчиков**

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

Результат

EXPR1
7

Агрегатные функции

- **Функция COUNT** ***Синтаксис* 2:**
  COUNT(DISTINCT | ALL поле)

  - **подсчитывает число известных (не NULL) значений поля, используется совместно с DISTINCT | ALL**

COUNT(DISTINCT поле) - подсчет числа неповторяющихся
COUNT(ALL поле) – подсчет всех известных значений (по умолчанию)

Агрегатные функции

- **Пример 22. Подсчитать количество заказчиков, у которых были заказы**

```
SELECT COUNT(DISTINCT cnum)
FROM Orders
```

Результат

EXPR1
7

Агрегатные функции

- **SUM вычисляет арифметическую сумму всех выбранных значений данного поля, игнорируя NULL значения;**
- **AVG вычисляет среднее значение для всех выбранных значений данного поля, игнорируя NULL значения**
  - **если столбец состоит только из пустых значений, то функция AVG также возвратит NULL;**

Агрегатные функции

- **Пример 23. Вывести минимальную и среднюю стоимость заказа заказчика с номером 2006**

```
SELECT MIN(amt), AVG(amt) AS ‘Среднее’
FROM Orders
WHERE cnum = 2006
```

Результат

EXPR1	Среднее
1309.95	3016.475

Агрегатные функции

- **GROUP BY**
- **позволяет группировать данные в подмножества и применять агрегатные функции к полученным подмножествам**
- **Благодаря этому, разрешается в одном запросе выводить значения и полей и агрегатных функций**

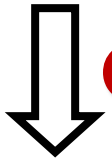
Агрегатные функции

- **Пример 24. Вывести количество заказов продавцов, номера которых меньше 1004**

```
SELECT snum, COUNT(*) AS ‘Кол-во’
FROM Orders
WHERE snum < 1004
GROUP BY snum
```

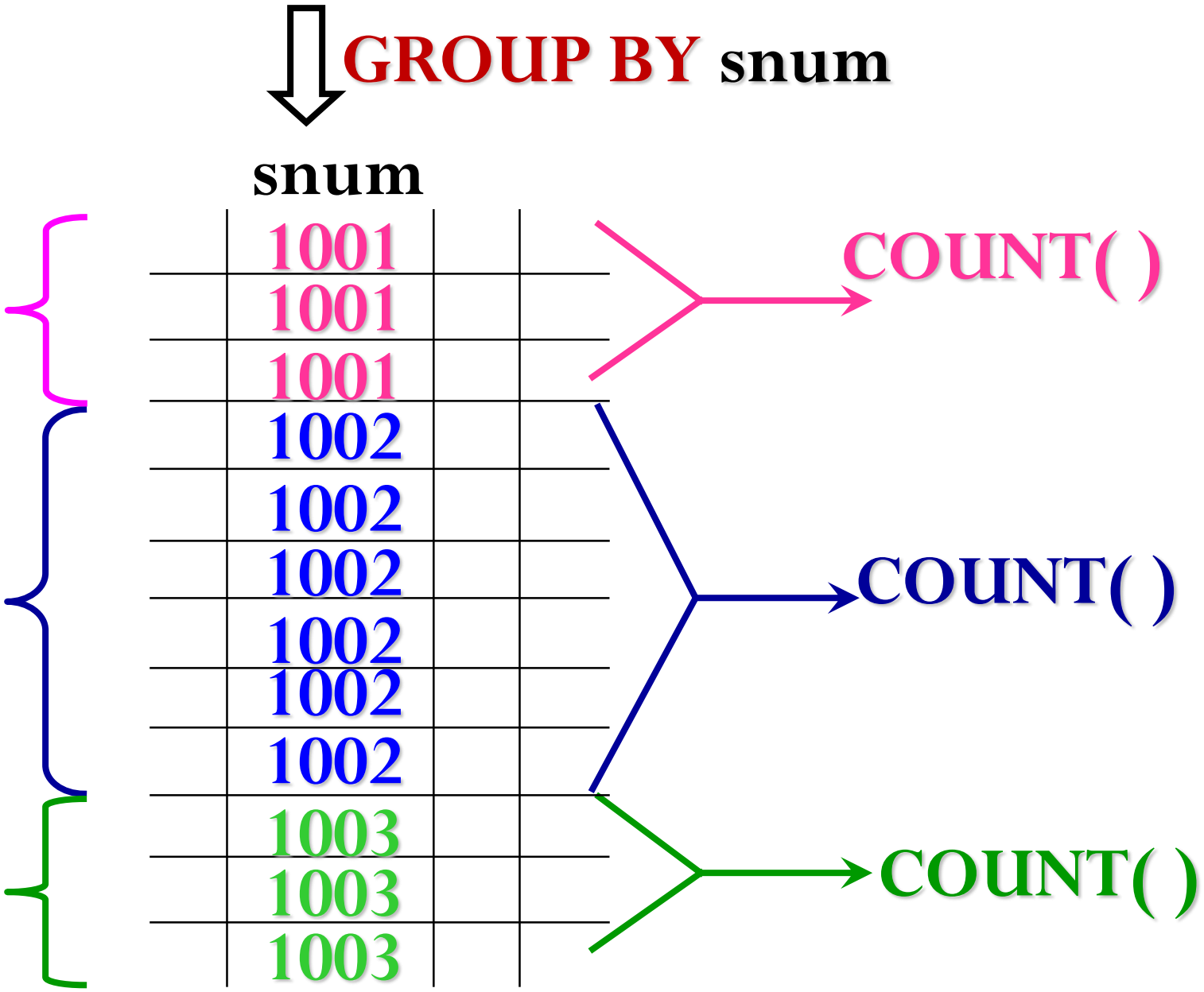
Результат

snum	Кол-во
1001	3
1002	6
1003	3



GROUP BY snum

snum		
1002		
1001		
1003		
1002		
1001		
1002		
1003		
1002		
1002		
1001		
1002		
1003		


GROUP BY snum
snum
1001
1001
1001
1002
1002
1002
1002
1002
1002
1003
1003
1003
COUNT( )
COUNT( )
COUNT( )

Агрегатные функции GROUP BY

- **В списке полей GROUP BY могут присутствовать любые поля таблиц из списка FROM, включая и те, которых нет среди полей SELECT.**
- **Поля из списка SELECT обязательно указываются в списке GROUP BY**

Агрегатные функции GROUP BY

- **Что будет выведено следующим запросом?**

```
SELECT snum, COUNT(*) AS ‘Кол-во’
FROM Orders
WHERE snum < 1004
GROUP BY snum, odate
```

Агрегатные функции GROUP BY

- **Если полей группировки несколько, то вначале строки группируются по значениям первого столбца, а внутри полученных групп — в подгруппы по значениям второго столбца и т.д.**
- **Т. о., GROUP BY не только устанавливает столбцы, по которым осуществляется группирование, но и указывает порядок разбиения столбцов на группы.**

Агрегатные функции GROUP BY

- **Упорядочивание результатов запроса выполняется последней строкой в запросе.**

```
SELECT snum, COUNT(*) AS ‘Кол-во’
FROM Orders
WHERE snum < 1004
GROUP BY snum, odate
ORDER BY 2
```

Агрегатные функции

Предикат HAVING

- **Позволяет задавать условия на сформированные с помощью GROUP BY наборы данных.**
- **Предикат HAVING определяет критерий, по которому группы следует включать в выходные данные, по аналогии с предложением WHERE, которое осуществляет это для отдельных строк.**

Агрегатные функции - HAVING

```
SELECT snum, COUNT(*)
FROM Orders
WHERE COUNT(*)= 2
GROUP BY snum
```

Неправильно!

Правильно!

```
SELECT snum, COUNT(*)
FROM Orders
GROUP BY snum
HAVING COUNT(*)= 2
```

Агрегатные функции - HAVING

- **В условии, задаваемом предикатом HAVING, указывают только поля или выражения, которые на выходе имеют единственное значение для каждой выводимой группы.**

- ***Следовательно*:**
  в HAVING можно ссылаться только на поля списка GROUP BY и агрегатные функции

Агрегатные функции - HAVING

- **Что выведет следующий запрос?**

```
SELECT snum, COUNT(*)
FROM Orders
GROUP BY snum
HAVING snum IN (1001, 1003)
```

Агрегатные функции - HAVING

```
SELECT snum, COUNT(*)
FROM Orders
GROUP BY snum
HAVING amt < 1000
```

Неправильно!

- **Почему?**

Правильно!

```
SELECT snum, COUNT(*)
FROM Orders
WHERE amt < 1000
GROUP BY snum
```

Агрегатные функции

Пример 25. **Определить даты заказов и продавцов, заключивших в один день больше 3 заказов на сумму 7000**

```
SELECT odate, snum
FROM Orders
GROUP BY odate, snum, amt
HAVING amt = 7000 and COUNT(*)>2
```

Агрегатные функции

Пример 25. **Определить даты заказов и продавцов, заключивших в один день больше 3 заказов на сумму 7000**

```
SELECT odate, snum
FROM Orders
WHERE amt = 7000
GROUP BY odate, snum
HAVING  COUNT(*)>2
```

Агрегатные функции

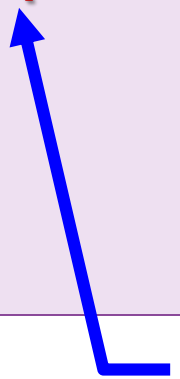
Ограничение

В рамках стандарта ANSI нельзя использовать вложенные агрегатные функции

Агрегатные функции

Пример 26. **Определить номер продавца, общая стоимость заказов которого максимальна**

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



НЕДОПУСТИМО

Самостоятельно

1. **Напишите запрос, который сосчитал бы все суммы продаж на 3 октября.**
2. **Напишите запрос, который сосчитал бы число различных не-NULL-значений поля city в таблице Заказчиков.**
3. **Напишите запрос, который выбрал бы наименьшую сумму для каждого заказчика.**
4. **Напишите запрос, который выбирал бы в алфавитном порядке заказчиков, чьи имена начинаются с буквы G.**
5. **Напишите запрос, который выбрал бы высший рейтинг в каждом городе.**
6. **Напишите запрос, который сосчитал бы число заказчиков, регистрирующих каждый день свои заказы. (Если продавец имел более одного заказа в данный день, он должен учитываться только один раз.)**

Операторы union, except, intersect

Оператор UNION

UNION используется для объединения выходных данных двух или более SQL-запросов в единое множество строк и столбцов

SELECT-запросы должны быть совместимыми по объединению!!!

Оператор UNION

Синтаксис:

```
SELECT … FROM … WHERE …
UNION [ALL]
SELECT … FROM … WHERE …
UNION …
SELECT … FROM … WHERE …
…
ORDER BY …
```

Сортировка после последнего SELECT

Оператор UNION

- По умолчанию **UNION** автоматически исключает дубликаты строк из вывода.
- Для вывода всех значений используется **UNION ALL**
- В SELECT запросах наравне с полями таблиц можно выводить константы, выражения и т.д.

Оператор UNION

Пример 27: Вывести данные по заказчикам и продавцам, с указанием их статуса

```
SELECT   cname, city, 'Заказчик'
FROM Customers
UNION
SELECT   sname, city, 'Продавец'
FROM Salespeople
ORDER BY 3 , 2, 1
```

Оператор UNION

Часто полезна операция объединения двух запросов, в которой второй запрос выбирает строки, исключенные первым запросом

Оператор UNION

Пример 28: Вывести заказчиков и имена их основных продавцов. Если они проживают в одном городе – выводить слово «MATCHED», иначе «NO MATCH»

Оператор UNION

Пример 28: Решение.

```
SELECT   cname, rating, sname, 'Matched'
FROM Customers c join Salespeople
        on c.snum = s.snum and c.city=s.city
UNION
SELECT   cname, rating, sname, 'No Match'
FROM Customers c join Salespeople
        on c.snum = s.snum and c.city<>s.city
```

Оператор UNION

Объединение более двух SELECT запросов выполняется последовательно сверху вниз.

Для изменения порядка объединения, можно использовать круглые скобки!

Запрос1 **UNION** (Запрос2 **UNION** Запрос3)

Оператор EXCEPT

EXCEPT реализует реляционную операцию – разность.

В результирующий набор помещаются строки первого запроса, отсутствующие в выводе второго запроса.

SELECT-запросы должны быть совместимыми по объединению!!!

Оператор EXCEPT

Синтаксис:

```
SELECT … FROM … WHERE …
EXCEPT
SELECT … FROM … WHERE …
…
ORDER BY …
```

Сортировка после последнего SELECT, по полям первого запроса

Оператор EXCEPT

Пример 29: Вывести список городов, в которых работают заказчики и нет ни одного поставщика

Оператор EXCEPT

Пример 29: Решение.

```
SELECT   city
FROM Customers
EXCEPT
SELECT   city
FROM Salespeople
```

Оператор INTERSECT

INTERSECT реализует реляционную операцию – пересечение.

В результирующий набор помещаются строки, присутствующие в обоих запросах

SELECT-запросы должны быть совместимыми по объединению!!!

Оператор INTERSECT

Синтаксис:

```
SELECT … FROM … WHERE …
INTERSECT
SELECT … FROM … WHERE …
…
ORDER BY …
```

Сортировка после последнего SELECT, по полям первого запроса

Оператор INTERSECT

Пример 30: Вывести список городов, в которых работают и заказчики, и поставщики

Оператор INTERSECT

Пример 30: Решение.

```
SELECT   city
FROM Customers
INTERSECT
SELECT   city
FROM Salespeople
```

Операторы UNION, EXCEPT, INTERSECT

Если оператор EXCEPT или INTERSECT используется в выражении вместе с другими операторами, оно обрабатывается в следующем порядке:

1. **Выражения в скобках.**
2. **Оператор INTERSECT**
3. **Операторы EXCEPT и UNION обрабатываются сверху вниз соответствии с их позицией в выражении.**

Самостоятельно

1. **Создайте объединение из двух запросов которое показало бы имена, города, и оценки всех заказчиков. Те из них которые имеют поле rating=200 и более, должны кроме того иметь слова - " Высокий Рейтинг ", а остальные должны иметь слова " Низкий Рейтинг ".**
2. **Напишите команду которая бы вывела имена и номера каждого продавца и каждого заказчика которые имеют более одного заказа. Результат представьте в алфавитном порядке.**
3. **Сформируйте объединение из трех запросов. Первый выбирает поле snum всех продавцов в Москве; второй, поле cnum всех заказчиков в Самаре; и третий поле onum всех заказов на 3 Октября. Сохраните дубликаты между последними двумя запросами, но устраните любую избыточность вывода между каждым из их и самым первым.**