Лекция 5

SQL запросы

Язык реляционных баз данных SQL

Язык SQL (Structured Query Language) появился в середине 70-х годов и был разработан в рамках экспериментальной реляционной СУБД System R.

В основу SQL положена комбинация реляционного исчисления кортежей и реляционной алгебры. При этом возможности SQL шире, чем у этих средств.

SQL в том или ином варианте присутствует практически во всех коммерческих СУБД.

Существует несколько стандартов языка, но все они во многом сводятся в основном к аккуратной технической обработке идей SQL, впервые появившегося в системе System R.

Первоначально SQL был ориентирован главным образом на удобную и понятную пользователю формулировку запросов. Позже в него помимо операторов формулирования запросов были включены и другие средства, делающие его полным языком БД,

Формулирование запросов к РБД

Запрос к РБД формулируется оператором выборки данных *SELECT*.
Средствами SQL можно формулировать простые запросы к соединениям нескольких отношений и вложенных подзапросов в предикатах (условиях) выборки.

Результатом выполнения оператора *SELECT* будет некоторое отношение, в общем случае являющееся не множеством, а мультимножеством кортежей (в нем могут присутствовать кортежи-дубликаты).

В результирующем отношении могут выполняться различные группирования данных по полям в соответствии с заданными условиями.

Оператор выборки данных SELECT

В простейшем случае оператор SELECT выглядит следующим образом:

SELECT <имена столбцов> FROM <имена таблиц>

В полной форме оператора могут присутствовать дополнительные разделы:

SELECT <имена столбцов>
FROM <имена таблиц>
WHERE <условие соединения> AND <условие выборки записей>
GROUP BY <имена столбцов >
HAVING <условие выборки групп>
ORDER BY <имена столбцов>

Оператор выборки данных SELECT

Результатом выполнения оператора SELECT в простейшей форме

SELECT <имена столбцов> FROM <имена таблиц>

будет таблица, составленная из заданных столбцов указанных таблиц (или одной таблицы).

Если в выборке участвует несколько таблиц, то для однозначной идентификации их столбцов указывается полное имя столбца:

<имя таблицы>.<имя столбца>

Если вместо списка имен столбцов указать символ *, то результирующая таблица будет состоять из всех столбцов всех указанных таблиц.

Примеры SQL-запросов

Рассмотрим несколько групп примеров SQL-запросов.

В этих примерах мы будем использовать базу данных, состоящую из 3 таблиц: *S* (поставщики), *P* (детали) и *SP* (поставки деталей).

Примеры SQL-запросов

Таблица *S. Поставщики*

Номер_Поставщика	Фамилия	Состояние	Город
S1	Смит	20	Лондон
S2	Джонс	10	Париж
S3	Блейк	30	Париж
S4	Кларк	20	Лондон
S5	Адамс	30	Афины

В таблице *S* каждый поставщик имеет уникальный номер, фамилию, значение рейтинга или состояние и местонахождение (город). Первичный ключ таблицы – номер поставщика.

Примеры SQL-запросов

Таблица *P. Детали*

Номер_Детали	Название	Цвет	Вес	Город
P1	гайка	красный	12	Лондон
P2	болт	зеленый	17	Париж
P3	винт	голубой	17	Рим
P4	винт	красный	14	Лондон
P5	кулачок	голубой	12	Париж
P6	блюм	красный	19	Лондон

В таблице *P* каждый вид детали имеет уникальный номер, название, цвет, вес и местонахождение (город). Первичный ключ этой таблицы – номер детали.

Примеры SQL-запросов

Таблица *SP. Поставщики – детали*

Номер_Поставщика	Номер_Детали	Количество
S1	P1	300
S1	P2	200
S1	P3	400
S1	P4	200
S1	P5	100
S1	P6	100
S2	P1	300
S2	P2	400
S3	P2	200
S4	P2	200
S4	P4	300
S4	P5	400

Примеры SQL-запросов

Таблица *SP* связывает детали из таблицы *P* с поставщиками из таблицы *S*.

Для каждой поставки имеется номер поставщика, номер детали и количество деталей.

Первичный ключ образуют два атрибута – номер поставщика и номер детали.

Примеры SQL-запросов

Простая выборка

"Выдать номера всех поставляемых деталей".

SELECT Номер_Детали FROM SP

Результат: весть столбец из *SP* с именем *Номер_Детали* (с повторяющимися номерами).

Í î ì åð_Äåòàëè
P1
P2
P3
P4
P5
P6
P1
P2
P2
P2
P4
P5

Запрос

*SELECT * FROM SP*

Результат: вся таблица SP

Оператор выборки данных SELECT

Вместо имени столбца в операторе может быть указано любое выражение, в том числе и константа. В этом случае в качестве значений этого столбца будут выступать результаты вычисления указанного выражения для каждой записи результирующей таблицы.

Использование в запросе выражений позволяет вычислять комбинацию данных из нескольких столбцов, а использование констант позволяет вставлять столбцы с комментариями.

В большинстве случаев имена столбцов в результирующей таблице совпадают с именами столбцов из исходных таблиц, однако они могут быть изменены для сохранения уникальности:

SELECT <имя столбца> AS <новое имя>, ...
FROM <имена таблиц>.

Для столбцов-выражений имена порождаются автоматически.

Примеры SQL-запросов

Выборка вычисляемых значений

"Выдать номера и вес каждой детали в граммах, предполагая, что в таблице P веса деталей даны в фунтах".

*SELECT Номер_Детали, "Вес в граммах=", Вес*454 FROM P*

Результат:

Í î ì åð_Äåòàëè		
P1	Âåñ â ãðàì ì àõ=	5448
P2	Âåñ â ãðàì ì àõ=	7718
-- --	-- -- -- -- -- -- -- --	-- --
P6	Âåñ â ãðàì ì àõ=	8626

Оператор выборки данных SELECT

Для имен таблиц, указанных в запросе, можно задавать синонимы:

SELECT <имена столбцов>
FROM <имя таблицы> <синоним>, ...

В этом случае во всех остальных частях запроса вместо имен этих таблиц следует использовать их синонимы.
Такой прием в частности может быть использован, когда в запросе указано более одного вхождения одной и той же таблицы.

При выполнении запроса в таблице (например, в результате проекции) могут оказаться одинаковые записи. Чтобы исключить дублирующиеся записи, перед именами полей в команде нужно поместить ключевое слово *DISTINCT:*

SELECT DISTINCT <имена столбцов> FROM <имена таблиц>.

Примеры SQL-запросов

Выборка с исключением дубликатов

"Выдать номера всех поставляемых деталей, исключая дубликаты".

SELECT DISTINCT Номер_Детали FROM SP

Результат:

Í î ì åð_Äåòàëè
P1
P2
P3
P4
P5
P6

Оператор выборки данных SELECT

Соединение таблиц в запросе

С точки зрения реляционной алгебры результат выполнения оператора *SELECT* представляет собой проекцию прямого произведения отношений. На практике же обычно требуется получить не прямое произведение отношений, а их соединение.

Для этого в операторе *SELECT* нужно указать *условие соединения*:

SELECT <имена столбцов> FROM <имена таблиц>

WHERE <условие соединения>

В качестве условия соединения может выступать сравнение двух атрибутов таблиц, а также конъюнкция таких условий (с помощью логической связки AND).

Оператор выборки данных SELECT

Задание условий выборки записей

Кроме условия соединения, в запросе может быть указано также и *условие выборки*. В этом случае оно обязательно должно следовать после условия соединения (если последнее задано):

SELECT <имена столбцов> FROM <имена таблиц>
WHERE <условие соединения> AND <условие выборки>

Задание условия выборки позволяет включать в результирующую таблицу не все записи, а только те из них, которые удовлетворяют этому условию.

Условие выборки может быть простым или содержать подзапрос, а также может состоять из нескольких частей, соединенных логическими связками *AND*, *OR* и *NOT*.

Примеры SQL-запросов

Ограниченная выборка

"Выдать номера поставщиков, которые находятся в Париже и имеют состояние больше 20".

```
SELECT Номер_Поставщика  FROM S
    WHERE Город="Париж" AND Состояние>20
```

Результат:

Íîì åð_Ïîñòàâùèêà
S3

Оператор выборки данных SELECT

К простым условиям выборки относятся:

- сравнение значения атрибута со значением другого атрибута или любого выражения:

  <атрибут> <сравнение> <атрибут>

  <атрибут> <сравнение> <выражение>,

где в качестве операции сравнения могут использоваться символы равенства (=), неравенства (< >, !=, #) и меньше/больше (<, <=, >, >=);

- проверка на принадлежность (не принадлежность) значения атрибута заданному интервалу:

  <атрибут> BETWEEN <начало> AND <конец>

  <атрибут> NOT BETWEEN <начало> AND <конец>;

Оператор выборки данных SELECT

- проверка на принадлежность (не принадлежность) значения атрибута заданному множеству значений:

  <атрибут> IN (<набор значений>)

  <атрибут> NOT IN (<набор значений>);

- проверка на соответствие (не соответствие) значения символьного атрибута заданному образцу:

  <атрибут> LIKE <образец>

  <атрибут> NOT LIKE <образец>,

Примеры SQL-запросов

Выборка с использованием BETWEEN

"Выдать сведения о деталях, вес которых находится в диапазоне от 16 до 19 включительно".

SELECT Номер_Детали, Название, Цвет, Вес, Город FROM P
WHERE Вес BETWEEN 16 AND 19

Результат: очевиден.

Выборка с использованием IN

"Выдать сведения о деталях зеленого и красного цвета".

SELECT Номер_Детали, Название, Цвет, Вес, Город FROM P

WHERE Цвет IN ("красный", "зеленый")

Результат: очевиден.

Примеры SQL-запросов

Выборка с использованием предиката LIKE

"Выдать название и номера деталей, у которых название заканчивается на букву '*т*' ".

SELECT Номер_Детали, Название FROM P
*WHERE Название LIKE "*т"*

Результат:

Í î ì åð_Äåòàëè	Í àçâàí èå
P2 P3 P4	áîëò âèíò âèíò

Оператор выборки данных SELECT

Упорядочение записей

Для упорядочения записей в результирующей таблице запроса необходимо в разделе ORDER BY указать имена одного или нескольких столбцов, по которым последовательно будет производиться упорядочение записей:

SELECT <имена столбцов> FROM <имена таблиц>
ORDER BY <имена столбцов>

Сначала записи упорядочиваются по первому столбцу, затем для записей с одинаковым значением в этом столбце – по второму столбцу и т.д. Если после имени столбца стоит признак *ASC*, то для этого столбца упорядочение производится по возрастанию значений, если *DESC*, то по их убыванию. По умолчанию происходит упорядочение по возрастанию (*ASC*).

Вместо имен столбцов можно также указывать их порядковые номера в результирующей таблице.

Примеры SQL-запросов

"Выдать номера и вес каждой детали в граммах, предполагая, что в таблице P веса деталей даны в фунтах. Результат упорядочить по возрастанию номера детали в рамках возрастания веса в граммах”

```
SELECT Номер_Детали, "Вес в граммах=", Вес*454  FROM P
        ORDER BY  3, Номер_Детали
```

Предупреждение.

Поле в разделе *ORDER BY* должно включать столбцы результирующей таблицы, иначе будет выдана ошибка.

Нельзя, например, написать:

```
SELECT Номер_Поставщика FROM S
                ORDER BY  Город
```


Примеры SQL-запросов

Запросы к нескольким таблицам

Простое эквисоединение

"Выдать сведения о таких поставщиках и деталях, которые размещены в одном и том же городе".

```
SELECT  S.*,  P.*
    FROM  S, P
    WHERE  S.Город = P.Город
```

Результат: таблица, полученная путем соединения таблиц *S* и *P* по значению атрибута *Город.*

Примеры SQL-запросов

Соединение двух таблиц с дополнительным условием

"Выдать сведения о таких поставщиках и деталях, которые размещены в одном и том же городе и их состояние больше 20".

```
SELECT  S.*,  P.*
    FROM  S, P
    WHERE  S.Город = P.Город  AND  S.Состояние > 20
```

Результат: ограничение результирующей таблицы из предыдущего примера.

Примеры SQL-запросов

Соединение трех таблиц

"Выдать информацию о поставщиках и деталях, размещенных в одном городе, и количество деталей больше 100".

```
SELECT S.Номер_Поставщика,P.Номер_Детали, SP.Количество
FROM  S, P, SP
WHERE  S.Город = P.Город  AND
        P.Номер_Детали = SP.Номер_Детали  AND
       SP.Количество > 100
```

Результат:

Íîì åð_Ïîñòàâùèêà	Íîì åð_Äåòàëè	Êîëè÷åñòâî
S1	P1	300
S1	P4	200
S2	P2	400
S3	P2	200
S4	P4	300

Примеры SQL-запросов

Соединение таблицы с ней самой

"Выдать все пары поставщиков, находящихся в одном городе".

```
SELECT  ПЕРВАЯ.Номер_Поставщика
                ВТОРАЯ.Номер_Поставщика
FROM S ПЕРВАЯ, S ВТОРАЯ
WHERE  ПЕРВАЯ.Город = ВТОРАЯ.Город  AND
                ПЕРВАЯ.Номер_Поставщика <ВТОРАЯ.Номер_Поставщика
```

Результат:

Í î ì åð_Ï î ñòàâù èêà	Í î ì åð_Ï î ñòàâù èêà
S1 S2	S4 S3

Оператор выборки данных SELECT

Условия с подзапросом содержат внутри себя вложенный запрос к тем же или другим таблицам. Этот подзапрос должен формировать таблицу, состоящую из одного столбца, который интерпретируется как множество значений для последующей проверки истинности условия.

К таким условиям относятся:

- проверка на непустоту результата подзапроса:
  EXISTS (<подзапрос>)

т.е. существует ли хотя бы одна запись во множестве, образованном результатом подзапроса;

- сравнение значения атрибута со всеми значениями результата подзапроса:
  <атрибут> <сравнение> ALL (<подзапрос>)

т.е. сравнимо ли значение атрибута со всеми значениями из множества, образованного результатом подзапроса;

Оператор выборки данных SELECT

- сравнение значения атрибута с хотя бы одним значением результата подзапроса:
  <атрибут> <сравнение> SOME (<подзапрос>)

т.е. сравнимо ли значение атрибута хотя бы с одним значением из множества, образованного результатом подзапроса;

- проверка на принадлежность (не принадлежность) значения атрибута множеству, образованному результатом подзапроса:
  <атрибут> IN (<подзапрос>)
  <атрибут> NOT IN (<подзапрос>)

т.е. принадлежит ли (не принадлежит ли) значение атрибута множеству, образованному результатом подзапроса.

Использование подзапросов

Использование подзапросов

Простой подзапрос

"Выдать фамилии поставщиков, которые поставляют деталь P2".

```
SELECT Фамилия
FROM S
WHERE Номер_Поставщика IN
        (SELECT Номер_Поставщика
        FROM SP
        WHERE Номер_Детали = "P2")
```

Результат:

Ôàì èëèÿ
Ñì èò Äæîí ñ Áëåéê Êëàðê

Замечание. Этот же результат можно получить путем соединения таблиц.

Оператор выборки данных SELECT

- проверка на непустоту результата подзапроса:
  EXISTS (<подзапрос>)

т.е. существует ли хотя бы одна запись во множестве, образованном результатом подзапроса;

- сравнение значения атрибута со всеми значениями результата подзапроса:
  <атрибут> <сравнение> ALL (<подзапрос>)

т.е. сравнимо ли значение атрибута со всеми значениями из множества, образованного результатом подзапроса;

- сравнение значения атрибута с хотя бы одним значением результата подзапроса:
  <атрибут> <сравнение> SOME (<подзапрос>)

т.е. сравнимо ли значение атрибута хотя бы с одним значением из множества, образованного результатом подзапроса;

Оператор выборки данных SELECT

- проверка на принадлежность (не принадлежность) значения атрибута множеству, образованному результатом подзапроса:

  <атрибут> IN (<подзапрос>)

  <атрибут> NOT IN (<подзапрос>)

т.е. принадлежит ли (не принадлежит ли) значение атрибута множеству, образованному результатом подзапроса.

Использование подзапросов

Подзапрос с несколькими уровнями вложенности

"Выдать фамилии поставщиков, которые поставляют по крайней мере одну красную деталь".

```
SELECT Фамилия
FROM S
WHERE Номер_Поставщика IN
        (SELECT Номер_Поставщика
        FROM SP
        WHERE Номер_Детали IN
            (SELECT Номер_ Детали
            FROM P
            WHERE Цвет = "красный" ) )
```

Результат:

Ôàì èëèÿ
Ñì èò Äæîí ñ Êëàðê

Использование подзапросов

Использование одной и той же таблицы в запросе и подзапросе

"Выдать номера поставщиков, которые поставляют по крайней мере одну деталь, поставляемую поставщиком S2".

```
SELECT Номер_Поставщика
FROM SP
WHERE Номер_Детали IN
    (SELECT Номер_ Детали
    FROM SP
    WHERE Номер_Поставщика ="S2")
```

Результат:

Номер_Поставщика
S1
S2
S3
S4

Использование квантора существования EXISTS

"Выдать фамилии поставщиков детали P1".

```
SELECT Фамилия
FROM S
WHERE EXISTS
        (SELECT *
        FROM SP
        WHERE Номер_Поставщика = S.Номер_Поставщика
        AND Номер_Детали = "P1")
```

Результат:

Фамилия
Смит
Джонс

Использование кванторов EXISTS и ALL

"Выдать фамилию поставщика с максимальным состоянием".

```
SELECT DISTINCT a.Фамилия, a.Состояние
FROM S a
WHERE NOT EXISTS
        (SELECT *
        FROM S b
        WHERE a.Состояние < b.Состояние)
```

либо

```
SELECT DISTINCT a.Фамилия, a.Состояние
FROM S a
WHERE a.Состояние >= ALL
        (SELECT  b.Состояние
        FROM  S b)
```

Результат:

Ôàì èëèÿ
Àäàì ñ

Использование функций в запросе

"Выдать фамилию поставщика с максимальным состоянием".

```
SELECT Фамилия
FROM S
WHERE Состояние = ( SELECT MAX(Состояние)
                    FROM S )
```

Результат:

Ôàì èëèÿ
Àääàì ñ

Использование функций в запросе

"Выдать общее количество поставляемых деталей P2".

```
SELECT SUM(Количество)
FROM SP
WHERE Номер_Детали = "P2"
```

Результат:

1000

Использование функций в запросе

"Выдать общее количество поставщиков“.

SELECT COUNT(*) FROM S

Результат:

5

"Выдать общее количество поставщиков, поставляющих детали в настоящее время".

SELECT COUNT (DISTINCT Номер_Поставщика) FROM SP

Результат:

4

Оператор выборки данных SELECT

Группировка записей и использование функций подсчета

В результирующую таблицу можно помещать не только существующие значения столбцов или результат вычисления выражения для каждой записи, но также и некоторую статистику (количество, сумму, среднее арифметическое и т.п.) по всем значениям столбца. Для этого используются средства группировки записей и набор специальных функций подсчета.

Группировка записей по одному или нескольким атрибутам задается с помощью раздела *GROUP BY*:

SELECT <имена столбцов> FROM <имена таблиц>
GROUP BY <имена столбцов>

Оператор выборки данных SELECT

Группировка записей по заданному атрибуту заключается в том, что все записи с одинаковыми значениями атрибута объединяются в одну группу и в результирующую таблицу попадает только один представитель от каждой группы.

Если задано разбиение по нескольким столбцам, то оно осуществляется последовательно, т.е. сначала все записи разбиваются на группы по первому указанному столбцу, потом внутри каждой группы по второму столбцу и т.д. Вместо имен столбцов можно указывать их порядковые номера в результирующей таблице.

Оператор выборки данных SELECT

Разбиение на группы обычно производится для подсчета статистики по столбцам. Для этого в операторе *SELECT* вместо имени столбца нужно указать одну из стандартных функций от значений этого столбца:

COUNT(<имя столбца>) – количество значений в столбце;

SUM(<имя столбца>) – сумма значений в столбце;

AVG(<имя столбца>) – среднее арифметическое в столбце;

MIN(<имя столбца>) – минимальное значение в столбце;

MAX(<имя столбца>) – максимальное значение в столбце.

Эти функции действуют над всеми значениями столбца внутри каждой группы. В этом случае представитель каждой группы в результирующей таблице будет содержать результат вычисления функции в соответствующем столбце.

Оператор выборки данных SELECT

Если группировка с помощью GROUP BY не задана, то вся исходная таблица считается одной группой и функции подсчета применяются ко всем значениям заданного столбца (или столбцов). Результирующая таблица в этом случае всегда будет состоять лишь из одной записи.

При использовании функций подсчета перед именем столбца можно указать ключевое слово *DISTINCT*, например:

COUNT(DISTINCT <имя столбца>).

В этом случае в подсчете будут участвовать только различные значения в столбце.

В функции *COUNT( )* также вместо имени конкретного столбца можно указать символ *.

Оператор выборки данных SELECT

Вместо имени столбца в вызове функций может быть указано любое выражение. В этом случае будет осуществляться подсчет не значений столбца, а результатов вычисления заданного выражения для всех записей группы.

Таким образом, можно получать статистику не только по данным из столбца, но и по некоторой комбинации данных из одного или нескольких столбцов.

Запросы с группированием данных

"Вычислить общий объем поставок для каждой детали ".

SELECT Номер_Детали, SUM(Количество)

FROM SP

GROUP BY Номер_Детали

Результат:

Í î ì åð_Äåòàëè	
P1	600
P2	1000
P3	400
P4	500
P5	500
P6	100

Оператор выборки данных SELECT

Задание условий выборки групп

Для того чтобы в результирующую таблицу попадали представители не всех групп, а только некоторых из них, удовлетворяющих заданному *условию выборки групп*, необходимо указать это условие в разделе *HAVING*:

SELECT <имена столбцов> FROM <имена таблиц>
GROUP BY <имена столбцов >
HAVING <условие выборки групп>

В этом условии, так же, как и в условии выборки записей, можно использовать операции сравнения, но их аргументами уже могут быть не только значения атрибутов, но и вызовы функций подсчета для значений столбцов. Подзапросы в этом условии использовать не разрешается. Условие может состоять из нескольких частей, соединенных логическими связками *AND, OR* и *NOT*.

Запросы с группированием данных

2. "Выдать номера всех деталей, поставляемых более чем одним поставщиком ".

SELECT Номер_Детали FROM SP

GROUP BY Номер_Детали

HAVING COUNT(*) > 1

Результат:

Í î ì åð_Äåòàëè
P1
P2
P4
P5