Государственный комитет Российской Федерации
по высшему образованию

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ

681
№1350
Я41

Язык структурных запросов SQL

Методические указания
по курсу «Базы данных»
для студентов IV курса ФПМИ
(специальности 010500, 010503) дневного отделения

НОВОСИБИРСК

1996

681.3.01(07)

Методические указания представляют собой практикум по изучению языка SQL. Методические указания могут быть использованы как в рамках исследовательской работы студентов, так и при проведении лабораторных работ. Они могут быть также полезны для специалистов, связанных с информационными технологиями и самостоятельного знакомящихся с основами работы с базами данных средствами SQL.

Составитель: *В.М. Стасышин*, канд. техн. наук, доц.

Рецензент *Н. Л. Долозов*, канд. техн. наук, доц.

Работа подготовлена на кафедре программных систем и баз данных

ВВЕДЕНИЕ

Большинство современных СУБД построено на реляционной модели данных. Для получения информации из отношений (таблиц) базы данных в качестве языка манипулирования данными в теоретическом плане используются три абстрактных языка:

- язык реляционной алгебры;
- язык реляционного исчисления на кортежах;
- язык реляционного исчисления на доменах.

В качестве практического языка работы с данными в середине 70-х годов фирмой IBM разработан язык структурных запросов SQL, ставший впоследствии стандартом de-facto при работе с базами данных. Наметившееся в настоящее время переход к крупным корпоративным СУБД типа Oracle, Informix, Sybase, DB2, Progress, PostgreSQL делает актуальным изучение языка SQL как в практическом плане, так и чисто теоретически, поскольку в основе элементов языка SQL лежат положения теории отношений, теории множеств и логики.

Методические указания по изучению языка SQL

Предлагаемые методические указания по изучению языка SQL содержат набор сгруппированных по темам заданий, выполнение которых позволит слушателям получить начальные навыки по работе с базами данных и формирования запросов на языке SQL. Каждая тема предлагает последовательное изучение конструкций операторов языка SQL по мере их усложнения.

Общая схема выполнения заданий по каждой теме следующая: ставится содержательная задача, предлагается запись запроса на языке SQL, дается результат запроса. Далее необходимо с использованием инструментальных средств работы с SQL подготовить предлагаемый запрос на языке SQL и убедиться в правильности получения решения. При необходимости поясняются конструкции языка SQL и делаются другие замечания методического характера, способствующие усвоению материала.

Будем исходить из того, что в нашем распоряжении имеется база данных (точнее схема базы данных) поставщиков, деталей и поставок, таблицы которой описаны следующим образом:

Таблица поставщиков S:

```
Create table S (n_post char(5) not NULL,
                name char(20),
                reiting smallint,
                town char(15))
```

Таблица деталей P:

```
Create table P (n_det char(6) ,
                name char(20),
                cvet char(7),
                ves smallint,
                town char(15))
```

Таблица поставок SP:

```
Create table SP (n_post char(5) ,
                 n_det char(6),
                 date_post date,
                 kol smallint)
```

Замечания. Структура приведенной базы данных максимально упрощена: в таблицах отсутствуют ограничения, первичные ключи и пр. Сделано это осмысленно, поскольку предметом данного методического пособия является изучение основ языка SQL, а не принципов проектирования реляционных баз данных.

Содержание таблиц базы данных следующее:

Таблица поставщиков (S)

Номер_поставщика	Фамилия	Рейтинг	Город
S1	Смит	20	Лондон
S2	Джонс	10	Париж
S3	Блейк	30	Париж
S4	Кларк	20	Лондон
S5	Адамс	30	Атенс

Таблица деталей (P)

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

Таблица поставок (SP)

Номер поставщика	Номер детали	Дата поставки	Количество
S1	P1	02/01/95	300
S1	P2	04/05/95	200
S1	P3	05/12/95	400
S1	P4	06/15/95	200
S1	P5	07/22/95	100
S1	P6	08/13/95	100
S2	P1	03/03/95	300
S2	P2	06/12/95	400
S3	P2	04/04/95	200
S4	P2	03/23/95	200
S4	P4	06/17/95	300
S4	P5	08/22/95	400

В качестве инструментария для выполнения заданий лабораторных работ в рамках СУБД PostgreSQL можно использовать интерактивную программу phpPgAdmin (или pgAdmin), позволяющей наряду с прочими возможностями подготавливать и выполнять запросы в текстовом редакторе.

Замечание 1. В целях большей наглядности при записи запроса на языке SQL поля таблиц базы данных записаны на русском языке и при переносе текстов запросов необходимо выполнить соответствующие замены с учетом реальных имен полей таблиц.

Замечание 2. К особенностью работы с PostgreSQL относится следующее. Все незащищенные (незаключенные в двойные кавычки) идентификаторы (имена таблиц, столбцов, индексов, представлений и пр.) преобразуются к нижнему регистру. Так любая смешанная комбинация символов разных регистров (stAtEs, STATES) при отсутствии кавычек перед выполнением команды автоматически приводится к виду states. Идентификаторы, заключенные в кавычки, указывают на их буквенную интерпретацию с учетом используемого регистра. Идентификаторы обязательно должны заключаться в кавычки только в двух случаях: если идентификатор совпадает с ключевым словом или в его имени присутствует хотя бы одна прописная буква.

I. Простые запросы на языке SQL

Запрос на языке SQL формируется с использованием оператора Select. Оператор Select используется

- для выборки данных из базы данных;
- для получения новых строк в составе оператора Insert;
- для обновления информации в составе оператора Update.

В общем случае оператор Select содержит следующие восемь спецификаторов, расположенных в операторе в следующем порядке:

- спецификатор Select;
- спецификатор From;
- спецификаторы Join;
- спецификатор Where;
- спецификатор Group by;
- спецификатор Having;
- спецификатор Order by;
- спецификатор Into temp.

Обязательными являются только спецификаторы Select и From. Эти два спецификатора составляют основу каждого запроса к базе данных, поскольку они определяют таблицы, из которых выбираются данные, и столбцы, которые требуется выбрать.

Спецификаторы Join (их в одном операторе Select может быть несколько) используются для перечисления присоединяемых таблиц и указания условия соединения.

Спецификатор Where добавляется для выборки определенных строк или указания условия соединения. Спецификатор Order by добавляется для изменения порядка получаемых данных. Спецификатор Into temp добавляется для сохранения этих результатов в виде таблицы с целью выполнения последующих запросов. Два дополнительных спецификатора оператора Select - Group by (спецификатор группирования) и Having (спецификатор условия выборки группы) - позволяют выполнять более сложные выборки данных.

Упражнения

1. Выбор всех строк и столбцов таблицы.

Пример.

Выдать полную информацию о поставщиках.

```
Select *
from S
```

Символ * после Select означает, что в результат должны быть включены все столбцы таблицы S.

Результат: таблица S в полном объеме.

Подготовьте запрос и проверьте полученный результат.

2. Изменение порядка следования столбцов.

Пример.

Выдать таблицу S в следующем порядке: фамилия, город, рейтинг, номер_поставщика.

```
Select фамилия, город, рейтинг, номер_поставщика
from S
```

Результат: таблица S в требуемом порядке.

Подготовьте запрос и проверьте полученный результат.

3. Выбор заданных столбцов.

Пример.

Выдать номера всех поставляемых деталей.

```
Select номер_детали
from SP
```

Результат: столбец номер_детали таблицы SP

Подготовьте запрос и проверьте полученный результат.

4. Выбор без повторения.

Пример.

Выдать номера всех поставляемых деталей, исключая дублирование.

```
Select distinct номер_детали
from SP
```

Результат: номер_детали

P1
P2
P3
P4
P5
P6

Подготовьте запрос и проверьте полученный результат.

5. Использование в запросах констант и выражений.

Пример.

Выдать вес каждой детали в граммах.

```
Select номер_детали || ' вес в граммах=' || вес*454
 from P
```

|| - оператор конкатинации строк

Результат:
P1 вес в граммах=5448
…
P6 вес в граммах=8226

Подготовьте запрос и проверьте полученный результат.

6.Ограничение в выборке.

Пример.

Выдать номера всех поставщиков, находящихся в Париже и имеющих рейтинг > 20.

```
Select номер_поставщика
from S
where город='Париж' and рейтинг>20
```

Результат:

номер_поставщика
S3

Подготовьте запрос и проверьте полученный результат.

7. Выборка с упорядочиванием.

Пример.

Выдать номера поставщиков, находящихся в Париже в порядке убывания рейтинга.

```
Select номер_поставщика,рейтинг
from S
where город='Париж'
order by рейтинг desc
```

Результат:

номер_поставщика	рейтинг
S3	30
S2	10

Подготовьте запрос и проверьте полученный результат.

8. Упорядочивание по нескольким столбцам.

Пример.

Выдать список поставщиков, упорядоченных по городу, в пределах города - по рейтингу.

```
Select *
from S
order by 4, 3
```

Результат:

Номер_поставщика	Фамилия	Рейтинг	Город
S5	Адамс	30	Атенс
S1	Смит	20	Лондон
S4	Кларк	20	Лондон
S2	Джонс	10	Париж
S3	Блейк	30	Париж

Подготовьте запрос и проверьте полученный результат.

9. Фраза between.

Пример.

Выдать информацию о деталях, вес которых лежит в диапазоне от 16 до 19.

```
Select номер_детали, название, вес
from P
where вес between 16 and 19
```

Результат:

номер_детали	название	вес
P2	Болт	17
P3	Винт	17
P6	Блюм	19

Подготовьте запрос и проверьте полученный результат.

10. Фраза in (not in).

Пример.

Выдать детали, вес которых равен 12, 16 или 17.

```
Select номер_детали, название, вес
from P
where вес in (12, 16, 17)
```

Результат:

номер_детали	Название	вес
P1	Гайка	12
P2	Болт	17
P3	Винт	17
P5	Кулачок	12

Подготовьте запрос и проверьте полученный результат.

11. Выбор по шаблону.

Для запросов с поиском по шаблону, основанных на поиске подстрок в полях типа CHARACTER, по стандарту ANSI используется ключевое слово LIKE.

Включение в выражение ключевого слова NOT порождает условие с обратным смыслом.

СИМВОЛ	ЗНАЧЕНИЕ
LIKE	
%	заменяет последовательность символов
_ (подчерк)	заменяет любой одиночный символ
\	отменяет специальное назначение следующего за ним символа

Примеры.

а) Выбрать список деталей, начинающихся с буквы "Б"

```
Select номер_детали, название, вес
from P
where название like 'Б%'
```

Результат:

номер_детали	название	вес
P5	Болт	12
P6	Блюм	19

б) Выдать список фамилий поставщиков, третья буква имени которых "а".

```
Select фамилия from S
where name like '__a%'
```

Результат:

фамилия
Адамс
Кларк

Подготовьте запросы и проверьте полученный результат.

II. Использование функций

1. Агрегатные функции.

Среди наиболее часто используемых функций отметим:
Sum - сумма значений по столбцу;
Avg - среднее значение в столбце;
Max - максимальное значение в столбце;
Min - минимальное значение в столбце.

Примеры.

а) Выдать общее количество поставщиков.

```
Select count (*)
from S
```

Результат: 5

Подготовьте запрос и проверьте полученный результат.

б) Выдать общее количество поставщиков, поставляющих в настоящее время детали.

```
Select count (distinct номер_поставщика )
from SP
```

Результат: 4

Подготовьте запрос и проверьте полученный результат.

в) Выдать количество поставок для детали 'P2'.

```
Select count (*)
from SP
where номер_детали='P2'
```

Результат: 4

Подготовьте запрос и проверьте полученный результат.

г) Выдать общее количество поставляемых деталей 'P2'.

```
Select sum (количество)
from SP
where номер_детали='P2'
```

Результат: 1000

Подготовьте запрос и проверьте полученный результат.

д) Выдать средний, минимальный и максимальный объем поставок для поставщика S1 с соответствующим заголовком.

```
Select avg(количество) as average,
       min(количество) as minimum,
       max(количество) as maximum
from SP
where номер_поставщика='S1'
```

Результат:	average	minimum	maximum
	216.6	100	400

Подготовьте запрос и проверьте полученный результат.

2. Строковые функции.

Ниже перечислено несколько функций, относящихся к указанной группе. Общий их перечень достаточно широк.

Substr(s,n,[l]) - функция возвращает подстроку s, начинающуюся с n длиной l;

Lower(s) - функция возвращает строку s, преобразованную к нижнему регистру;

Length(s) - функция возвращает длину строки s.

Пример.

Выдать два первых символа имен поставщиков, преобразованных к нижнему регистру.

```
Select Substr(lower(фамилия), 1, 2)
from s
```

Результат: Первые две буквы фамилии

см
бл
кл
ад
дж

Подготовьте запрос и проверьте полученный результат.

III. Группирование

1. Оператор group by группирует таблицу, представленную фразой from в группы т.о., чтобы в каждой группе все строки имели одно и тоже значение поля, указанного во фразе group by. Далее, к каждой группе перекомпанованной таблицы (а не к каждой строке исходной таблицы) применяется фраза select, в результате чего, каждое выражение во фразе select принимает единственное значение для группы.

Пример.

Выдать для каждой поставляемой детали ее номер и общий объем поставок, за исключением поставок поставщика 'S1'.

```
Select номер_детали, sum( количество)
from SP
where номер_поставщика <>'S1'
group by номер_детали
```

Результат:

Номер_поставщика	(Sum)
P1	300
P2	800
P4	300
P5	400

Подготовьте запрос и проверьте полученный результат.

2. Фраза having.

Фраза having играет ту же роль для групп, что и фраза where для строк и используется для того, чтобы исключать группы, точно так же, как where используется для исключения строк. Выражение во фразе having должно принимать единственное значение для группы.

Пример.

Выдать номера деталей, поставляемых более чем одним поставщиком.

```
Select номер_детали
from SP
group by номер_детали
having count(*) > 1
```

Результат:

Номер_детали
P1
P2
P4
P5

Подготовьте запрос и проверьте полученный результат.

IV. Соединения таблиц.

Классическая реляционная алгебра Кодда включает девять реляционных операций, последовательное применение которых позволяет реализовать выборку любых данных. Три из этих операции так или иначе связаны с соединением таблиц:

- операция взятия декартова произведения;
- операция соединение (соответствующая ей в стандарте ANSI операция носит название операции внутреннего соединение);
- операция эквисоединения.

Операция взятия декартово произведение содержит все возможные комбинации конкатенаций кортежей (строк) из соединяемых таблиц.

Операция соединения представляет собой соединение кортежей соединяемых таблиц по указанному условию соединения. Строки, которые не удовлетворяют условиям соединения, отбрасываются.

Операция эквисоединения является частным случаем операции соединение по условию равенства атрибутов.

Кроме этого существует практически важное расширение операции эквисоединения – естественное (внешнее) соединение.

Внешнее соединение может сохранить строки, для которых не находится соответствия в другой таблице. В этом случае недостающие поля заполняются значениями NULL. Решение о том, войдет ли такая строка в результирующий набор, зависит от того, в какой из соединяемых таблиц отсутствуют данные, и от типа внешнего соединения.

Существуют три разновидности внешних соединений.

- Левое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной слева от ключевого слова JOIN. Отсутствующие поля из правой таблицы заполняются значениями NULL.

- Правое внешнее соединение. Всегда содержит как минимум один экземпляр каждой строки из таблицы, указанной справа от ключевого слова JOIN. Отсутствующие поля из левой таблицы заполняются значениями NULL.

- Полное внешнее содинение. Всегда содержит как минимум один экземпляр каждой строки каждой из соединяемых таблиц. Отсутствующие поля в записях результирующего набора заполняются значениями NULL.

Для построения соединений стандарт ANSI предусматривает следующую конструкцию спецификаторов from и join:

FROM *источник1*
[Nutural] тип соединения JOIN *источник2* [on *условие* [,...] | Using (*поле1* [,...])]

- *Источник1*. Первый из соединяемых наборов данных (имя таблицы или подзапрос).

- *[Nutural]* Два набора данных соединяются по равным значениям одноименных полей. Конструкции On и Using в этом случае недопустимы.
- *Тип соединения* Возможные виды соединений:
  - [Inner] - внутреннее соединение;
  - Left [Outer] - левое внешнее соединение;
  - Right [Outer] - правое внешнее соединение;
  - Full [Outer] - полное внешнее соединение;
  - Cross – декартово произведение ;
- *Источник2.* Второй из соединяемых наборов данных (имя таблицы или подзапрос).
- *On условие [,...]* Отношение между источниками - критерий, аналогичный тому, который задается в конструкции Where.
- *Using (поле1 [,...])* Одноименные поля источников, по совпадающим значениям которых производится соединение. В отличии от Nutural позволяет ограничиться некоторыми одноименными полями, тогда как Nutural производит соединение по всем одноименным полям.

1. Простое декартово произведение.

Пример.

Выдать информацию обо всех возможных парах поставщик - деталь.

```
Select S.*, P.*
from S
Cross Join  P
```

Результат:

н_пост	фам-я	рейтинг	s.город	н_дет	назв-е	цвет	вес	р.город
S1	Смит	20	Лондон	P1	Гайка	красный	12	Лондон
S1	Смит	20	Лондон	P4	Винт	красный	14	Лондон
S1	Смит	20	Лондон	P6	Блюм	красный	19	Лондон
S2	Джонс	10	Париж	P2	Болт	зеленый	17	Париж
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .

Всего 30 строк.

Подготовьте запрос и проверьте полученный результат.

Замечание: тот же результат может быть получен запросом

```
Select * from S, P
```

В отличие от предыдущего запроса этот запрос написан с отклонением от стандарта ANSI, но он более точно отражает смысл операции взятия декартова произведения.

При написании запросов следует придерживаться стандарта ANSI, позволяющего формировать более читабельные запросы.

2. Простое эквисоединение.

Пример.

Выдать все комбинации информации о поставщиках и деталях, расположенных в одном городе.

```
Select S.номер_поставщика, p.номер_детали, рейтинг
from S
Cross Join  P
where  S.город=P.город
```

Результат:

н_пост	фам-я	рейтинг	s.город	н_дет	назв-е	цвет	вес	p.город
S1	Смит	20	Лондон	P1	Гайка	красный	12	Лондон
S1	Смит	20	Лондон	P4	Винт	красный	14	Лондон
S1	Смит	20	Лондон	P6	Блюм	красный	19	Лондон
S2	Джонс	10	Париж	P2	Болт	зеленый	17	Париж
. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .

Всего 10 строк.

Подготовьте запрос и проверьте полученный результат.

Замечание: тот же результат может быть получен запросом с использованием конструкции операции внутреннего соединения в стандарте ANSI

```
Select S.номер_поставщика, p.номер_детали, рейтинг
from S
Inner Join  P  on  S.город=P.город
```

или запросом, написанным с отклонением от стандарта ANSI

```
Select S.*,P.* from S, P
where S.город=P.город
```

3. Соединение таблиц с дополнительным условием.

Пример.

Выдать все комбинации информации о поставщиках и деталях, расположенных в одном городе, опустив поставщиков с рейтингом = 20.

```
Select S.номер_поставщика, p.номер_детали, рейтинг
from S
Cross Join  P
where  S.город=P.город  and S.рейтинг<>20
```

Результат:

Номер_поставщика	Номер_детали	Рейтинг
S2	P2	10
S2	P5	10
S3	P2	30
S3	P5	30

Подготовьте запрос и проверьте полученный результат.

4. Соединение таблицы с ней самой.

Пример.

Выдать все пары поставщиков из одного города.

```
Select one.номер_поставщика, two.номер_поставщика
from S one
Cross Join  S two
where  one.город = two.город  and one.номер_поставщика <
two.номер_поставщика
```

Результат:

Номер_поставщика	Номер_поставщика
S1	S4
S2	S3

Подготовьте запрос и проверьте полученный результат.

Замечание: замена фразы Cross на Inner в этом и предыдущем примере результата не меняет.

5. Внутреннее соединение

Пример.

Выдать для каждой поставки номер поставщика, его фамилию и количество деталей.

```
Select S.номер_поставщика, S.фамилия, SP.количество
from SP
inner join S on S.номер_поставщика=SP.номер_поставщика
```

или

```
Select S.номер_поставщика, S.фамилия, SP.количество
from SP
natural inner join S
```

Результат:

Номер_поставщика	Фамилия	Количество
S1	Смит	300
S1	Смит	200
S1	Смит	400
S1	Смит	200
S1	Смит	100
S1	Смит	100
S2	Джонс	300
S2	Джонс	400
S3	Блейк	200
S4	Кларк	200
S4	Кларк	300
S4	Кларк	400

Подготовьте запрос и проверьте полученный результат.

6. Соединение трех таблиц.

Пример.

Выдать все пары названий городов таких, что какой-либо поставщик, находящийся в первом из этих городов, поставляет деталь, хранимую в другом городе.

```
Select distinct S.город, P.город
from SP
inner join  S   on  S.номер_поставщика = SP.номер_поставщика
inner join  P   on  P.номер_детали = SP.номер_детали
```

Результат:

s.город	p.город
Лондон	Лондон
Лондон	Париж
Лондон	Рим
Париж	Лондон
Париж	Париж
Париж	Рим

Подготовьте запрос и проверьте полученный результат.

Замечание: еще раз приведем для сравнения запрос, написанный с отклонением от стандарта ANSI и дающим тот же результат

```
Select distinct S.город, P.город
from S, SP, P
where S.номер_поставщика = SP.номер_поставщика
and P.номер_детали = SP.номер_детали
```

и отметим еще раз, что при написании запросов предпочтение следует отдавать конструкциям стандарта ANSI.

7. Простое внешнее соединение двух таблиц.

Пример левого внешнего соединения.

```
Select S.номер_поставщика, S.фамилия, SP.количество
from S
left outer join SP  on (S.номер_поставщика=SP.номер_поставщика)
```

Добавление ключевого слова outer перед именем таблицы SP превращает ее в подчиненную таблицу. Результатом этого внешнего соединения будет получение сведений обо всех поставщиках, независимо делали ли они поставки.

Результат:

Номер_поставщика	Фамилия	Количество
S1	Смит	300
S1	Смит	200
S1	Смит	400
S1	Смит	200
S1	Смит	100
S1	Смит	100
S2	Джонс	300
S2	Джонс	400

S3	Блейк	200
S4	Кларк	200
S4	Кларк	300
S4	Кларк	400
S5	Адамс	

Подготовьте запрос и проверьте полученный результат. Сравните с результатом получения внутреннего соединения.

Полное внешнее соединение даст аналогичный результат, правое - результат, аналогичный внутреннему соединению.

8. Внешнее соединение внутреннего соединения с третьей таблицей.

Для получении внешнего соединения будем использовать представление (View). Представление можно рассматривать как хранимый запрос, на основании которого создается объект базы данных. Этот объект схож с таблицей, но в его содержимом динамически отражаются только те записи, которые были заданы при создании. Создается представление командой Create view

Create view имя_представления as запрос,

а удаляется командой Drop view.

```
create view z1 as
select sp.номер_поствщика, sp.номер_детали, p.название, p.цвет, p.вес
from SP
join  P   on  SP.номер_детали = P.номер_детали
where  цвет in ('Красный', 'Зеленый');

select S.n_post, S.name, z1.n_det, z1.name, z1.cvet, z1.ves
from s
left join z1  on(s.n_post=z1.n_post)
```

Первым оператором выполняет внутреннее соединение таблиц SP и P, а затем оператором Select выполняется внешнее соединение как комбинирование этой информации с данными из главной таблицы S.

Результат:

Номер_пост	Фамилия	Номер_дет	Название	Цвет	Вес
S1	Смит	P1	Гайка	Красный	12
S1	Смит	P2	Болт	Зеленый	17
S1	Смит	P4	Винт	Красный	14
S1	Смит	P6	Блюм	Красный	19
S2	Джонс	P1	Гайка	Красный	12
S2	Джонс	P2	Болт	Зеленый	17
S3	Блейк	P2	Болт	Зеленый	17
S4	Кларк	P2	Болт	Зеленый	17
S4	Кларк	P4	Винт	Красный	14
S5	Адамс				

Подготовьте запрос и проверьте полученный результат.

V. Использование значений oid-столбца

Для нахождения в таблице дубликата значения можно использовать в самосоединении скрытый oid-столбцец. Условие x.oid != y.oid, заданное в следующем примере эквивалентно следующему утверждению: строка x не является той же самой, что и строка y.

Пример.

В таблице описания деталей найти детали с одинаковым весом.

```
Select x.номер_детали, x.название, x.вес
from P x
cross join P y
where x.вес = y.вес   and   x.oid != y.oid
```

Результат:

номер_детали	название	вес
P2	Болт	17
P3	Винт	17
P1	Болт	12
P5	Кулачок	12

Подготовьте запрос и проверьте полученный результат.

VI. Подзапросы

Оператор select, вложенный в спецификатор where другого оператора select (или одного из операторов insert, delete, update), называется подзапросом. В состав каждого подзапроса должны входить спецификаторы select и from. Кроме того, каждый подзапрос должен быть заключен в круглые скобки, чтобы указать серверу баз данных на то, что эту операцию следует выполнить первой.

Подзапросы бывают коррелированными и некоррелированными. Подзапрос является коррелированным, если его значение зависит от значения, производимого внешним оператором select, который содержит этот подзапрос. Любой другой вид запроса называется некоррелированным.

Важное свойство коррелированного подзароса состоит в следующим: так как он зависит от значения результата внешнего оператора select, то должен выполняться повторно по одному разу для каждого значения, производимого внешним оператором select. Некоррелированный подзапрос выполняется только один раз.

Подзапрос включается в спецификатор where оператора select с помощью следующих ключевых слов:

ALL
ANY
IN
EXISTS

Некоррелированные подзапросы.

1. Фраза ALL.

Ключевое слово ALL, указываемое перед подзапросом используется для определения того, выполняется ли условие сравнения для каждого возвращаемого подзапросом значения. Если подзапрос не возвращает ни одного значения, то условие поиска считается выполненным.

Пример

Получить перечень поставщиков, рейтинг которых выше рейтинга любого лондонского поставщика.

```
Select x.номер_поставщика, x.фамилия, x.рейтинг
from S x
where x.рейтинг > all
(select y.рейтинг
from S y
where y.город='Лондон')
```

Сначала выполняется независимый внутренний подзапрос, его результатом является выборка (20, 20), затем - внешний запрос, приводящий к результату, записанному ниже.

Результат:	Номер_поставщика	Фамилия	Рейтинг
	S3	Блейк	30
	S5	Адамс	30

Подготовьте запрос и проверьте полученный результат.

2. Фраза ANY.

Ключевое слово ANY, указываемое перед запросом, используется для определения того, выполняется ли сравнение по крайней мере для одного значения, возвращаемого подзапросом. Если подзапрос не возвращает ни одного значения, то условие поиска считается не выполненным.

Пример

Получить перечень поставщиков, рейтинг которых выше рейтинга хотя бы одного парижского поставщика.

```
Select x.номер_поставщика, x.фамилия, x.рейтинг
from S x
where x.рейтинг > any
(select y.рейтинг
from S y
where y.город='Париж')
```

Сначала выполняется независимый внутренний подзапрос, его результатом является выборка (10, 30), затем - внешний запрос, приводящий к результату, записанному ниже.

Результат:	Номер_поставщика	Фамилия	Рейтинг
	S1	Смит	20
	S3	Блейк	30
	S4	Кларк	20
	S5	Адамс	30

Подготовьте запрос и проверьте полученный результат.

3. Фраза IN.

3.1. Простой подзапрос.

Пример

Выдать фамилии поставщиков, поставляющих деталь P2.

```
Select фамилия
from S
where номер_поставщика in
(Select номер_поставщика
from SP
where номер_детали ='P2')
```

Сначала выполняется внутренний подзапрос, его результатом является выборка (S1, S2, S3, S4), затем - внешний запрос, который после подстановки результатов внутреннего подзапроса имеет вид:

```
Select фамилия
        from S
        where номер_поставщика in ('S1', 'S2', 'S3', 'S4')
```

Результат: Фамилия
Смит
Джонс
Блейк
Кларк

Подготовьте запрос и проверьте полученный результат.

3.2. Подзапрос с несколькими уровнями вложенности.

Пример

Выдать фамилии поставщиков, поставляющих 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'))
```

Сначала осуществляется самый внутренний подзапрос, дающий выборку (P1, P4, P6). После подстановки его результатов выполняется второй по вложенности подзапрос, дающий выборку (S1, S2, S4). Подстановка результатов второго выполненного подзапроса во внешний запрос приводит к окончательному результату.

Результат: Фамилия
Смит
Джонс
Кларк

Подготовьте запрос и проверьте полученный результат.

3.3. Использование одной и той же таблицы в подзапросе внешнем запросе.

Пример

Выдать номера поставщиков, поставляющих, по крайней мере, одну деталь, поставляемую поставщиком S2.

```
Select distinct номер_поставщика
from SP spx
where spx.номер_детали in
(Select spy.номер_детали
from SP spy
where spy.номер_поставщика='S2')
```

Сначала выполняется внутренний подзапрос, дающий выборку (P1, P2). Подстановка его результатов во внешний запрос приводит к окончательному результату.

Результат: номер_поставщика
S1
S2
S3
S4

Подготовьте запрос и проверьте полученный результат.

3.4. Подзапрос с оператором сравнения отличным от IN.

Пример

Выдать номера поставщиков, находящихся в том же городе, что и поставщик S1.

```
Select номер_поставщика
from S
where город = (Select город
from S
where номер_поставщика ='S1')
```

Сначала выполняется внутренний подзапрос, дающий единственное значение "Лондон". Подстановка его результатов во внешний запрос приводит к окончательному результату.

Результат: номер_поставщика
S1
S4

Подготовьте запрос и проверьте полученный результат.

Коррелированный подзапросы

3.5. Простой коррелированный подзапрос.

Пример

Выдать фамилии поставщиков, поставляющих деталь P2.

```
Select фамилия
from S
where 'P2' in
(Select номер_детали
from SP
where номер_поставщика= S.номер_поставщика)
```

В коррелированном подзапросе внутренний подзапрос не может быть отработан раз и навсегда, прежде чем будет отработан внешний запрос, поскольку этот внутренний подзапрос зависит от переменной, значение которой изменяется по мере того, как система проверяет различные строки таблицы, участвующие во внешнем запросе. Обработка запроса выполняется по следующей схеме:

- выбирается первая строка из S (номер_поставщика='S1");
- выполняется подзапрос:

(Select номер_детали
from sp
where номер_поставщика='S1')

результатом подзапроса является выборка (P1, P2, P3, P4, P5, P6);

- завершается обработка запроса для первой строки из S, при выполнении которого проверяется условие

'P2' in ('P1', 'P2', 'P3', 'P4', 'P5', 'P6')

- поскольку проверяемое условие - истина, результатом обработки запроса для первой строки из S является фамилия "Смит";
- аналогично повторяется обработка для остальных строк таблицы S.

Результат: Фамилия
Смит
Джонс
Блейк
Кларк

Подготовьте запрос и проверьте полученный результат.

3.6. Коррелированный подзапрос с использованием в коррелированном и внешнем запросе одной и той же таблицы.

Пример

Выдать номера деталей, поставляемых более чем одним поставщиком.

```
Select distinct spx.номер_детали
from SP spx
where spx.номер_детали in
(Select spy.номер_детали
from SP spy
where spy.номер_поставщика<>spx.номер_поставщика)
```

Результат: Номер_детали
P1
P2
P4
P5

Подготовьте запрос и проверьте полученный результат.

4. Фраза EXISTS.

4.1. Квантор существования EXISTS.

В языке SQL предикат с квантором существования представляется выражением вида:

EXISTS (select * from...)

Данное выражение истинно тогда и только тогда, когда результат вычисления подзапроса, представленного с помощью **select * from** является непустым множеством, т.е. когда существует какая-либо запись в таблице, указанной во фразе **from** подзапроса, который удовлетворяет условию **where** этого подзапроса.

Пример

Выдать фамилии поставщиков, поставляющих деталь P2.

```
Select фамилия
from S
where exists
(Select *
from SP
where номер_поставщика = S.номер_поставщика
and номер_детали = 'P2')
```

Последовательность обработки запроса:

- выбирается первая строка из S (номер_поставщика='S1');
- поскольку условие

номер_поставщика = 'S1' и номер_детали = 'P2' - истина, результат обработки запроса для первой строки - фамилия Смит.

Результат:

Фамилия
Смит
Джонс
Блейк
Кларк

Подготовьте запрос и проверьте полученный результат.

4.2. Запрос, реализующий квантор общности.

Квантор общности FORALL в SQL не поддерживается, однако он может быть выражен через квантор существования при помощи тождества

FORALL x(p)=NOT(EXISTS x(NOT(p))).

Пример

Выдать фамилии поставщиков, которые поставляют все детали.

Эквивалентная формулировка задачи может звучать так:
Выдать фамилии поставщиков таких, что для всех деталей существует запись в таблице SP, указывающая, что данный поставщик поставляет эту деталь.

Последнее утверждение, в свою очередь, эквивалентно следующему: выдать фамилии поставщиков таких, что не существует детали такой, что не существует записи в таблице SP, указывающей, что данный поставщик поставляют эту деталь.

```
Select фамилия
from S
where not exists
(Select * from P
where not exists
(Select * from SP
where номер_поставщика=S.номер_поставщика
and номер_детали=P.номер_детали))
```

Результат:

Фамилия
Смит

Подготовьте запрос и проверьте полученный результат.

5. Использование функций в подзапросе.

Пример

Выдать номера поставщиков со значением поля рейтинг меньшим, чем максимальный рейтинг в таблице S.

```
Select номер_поставщика from S
where рейтинг < (Select max(рейтинг) from S)
```

Результат:

номер_поставщика
S1
S2
S4

Подготовьте запрос и проверьте полученный результат.

Пример

Выдать номер_поставщика, рейтинг и город всех поставщиков, у которых рейтинг больше либо равен среднему для их конкретного города (использование функций в коррелированном подзапросе).

```
Select номер_поставщика, рейтинг, город
from S sx
where рейтинг >= (Select avg(рейтинг)
from S sy
where sy.город=sx.город)
```

Результат:

номер_поставщика	Рейтинг	Город
S1	20	Лондон
S3	30	Париж
S4	20	Лондон
S5	30	Атенс

Подготовьте запрос и проверьте полученный результат.

VII. ОБЬЕДИНЕНИЕ

Объединяемые оператором UNION таблицы должны быть совместны по объединению:

- иметь одинаковое число столбцов;
- соответствующие столбцы должны иметь одинаковые типы.

Любое число предложений select может быть соединено оператором union. Избыточные дубликаты исключаются из результата объединения.

Пример

Выдать номера деталей, которые имеют вес более 16 фунтов, либо поставляются поставщиком S2.

```
Select номер_детали
        from P
        where вес>16
union
Select номер_детали
        from SP
        where номер_поставщика='S2'
```

Результат:

Номер_детали
P1
P2
P3
P6

Подготовьте запрос и проверьте полученный результат.

VIII. Операторы манипулирования данными. Удаление данных

Общая форма оператора удаления:

delete from таблица [where предикат]

1. Удаление единственной записи.

Пример

Удалить сведения о поставщике S1.

delete from S

where номер_поставщика='S1'

Результат: таблица S с отсутствующей строкой о поставщике S1.

Подготовьте запрос и проверьте полученный результат.

2. Удаление множества записей.

Пример

Удалить сведения обо всех поставщиках из Лондона.

delete from S

where город='Лондон'

Результат: таблица S с отсутствующими строками о поставщиках из Лондона.

Подготовьте запрос и проверьте полученный результат.

3. Удаление с подзапросом.

Пример

Удалить все поставки для поставщиков из Лондона.

delete from SP

where 'Лондон'=

(Select город from S

where S.номер_поставщика=SP.номер_поставщика)

Результат: таблица SP с отсутствующими строками о поставках для поставщиков из Лондона.

Подготовьте запрос и проверьте полученный результат.

4. Удаление всех строк таблицы.

Пример

delete from S

Подготовьте запрос и проверьте полученный результат.

IX. Операторы манипулирования данными. Вставка данных

Общая форма оператора вставки.

Insert into таблица [(поле [,поле]...)]

values (константа [,константа]...) или подзапрос

1. Вставка единственной записи.

Пример

Вставить новую поставку с номером поставщика S2, номером детали P4 и количеством 1000 на дату "30 ноября 1995 г.".

Insert into SP values ('S2', 'P4', '11/30/95', 1000)

Результат: таблица SP с добавленной строкой о поставке поставщиком S2 детали P4.

Подготовьте запрос и проверьте полученный результат.

2. Вставка множества записей.

Пример

Восстановить таблицу S.

Insert into S values ('S1', 'Смит', 20, 'Лондон');
Insert into S values ('S2', 'Джонс', 10, 'Париж');
Insert into S values ('S3', 'Блейк', 30, 'Париж');
Insert into S values ('S4', 'Кларк', 20, 'Лондон');
Insert into S values ('S5', 'Адамс', 30, 'Атенс')

Результат: восстановленная таблица S.

Подготовьте запрос и проверьте полученный результат.

3. Перечисление имен столбцов.

Допускается не задавать значения для каждого столбца, а перечислить имена столбцов после имени таблицы, а потом предоставить значения только для тех столбцов, имена которых указаны.

Пример

Вставить строку о новом поставщике, занеся лишь номер поставщика, фамилию и город.

Insert into S(номер_поставщика, фамилия, город)
values ('S6', 'Боб', 'Нью-Йорк')

Результат: добавленная строка в таблице S.

Подготовьте запрос и проверьте полученный результат.

4. Вставка множества записей как результата подзапроса.

Пример

Для каждой поставляемой детали получить ее номер и общий объем поставки, сохранить результат в базе данных.

```
Create table temp
        (номер_детали char(6),
        объем поставки smallint);
Insert into temp (номер_детали,обьем_поставки)
        Select номер_детали, sum(количество)
                from SP
                group by номер_детали
```

Результат: Сформированная таблица temp, данные в которую занесены как результат указанного оператора.

Подготовьте запрос и проверьте полученный результат.

5. Построение внешнего соединения с использованием оператора Insert.

Пример

Для каждого поставщика получить его номер, фамилию, рейтинг и город вместе с номерами всех поставляемых им деталей. Если поставщик не поставляет никаких деталей, поставить в поле номер_детали значение NN.

```
Create table outside_t
        (номер_поставщика char(5),
        фамилия char(20),
        рейтинг smallint,
        город char(15),
        номер_детали char(6));
Insert into outside_t
Select S.*, SP.номер_детали
from S, SP
where S.номер_поставщика=SP.номер_поставщика;
Insert into outside_t
Select S.*, 'NN'
from S
where not exists
(Select *
from SP
where SP.номер_поставщика = S.номер_поставщика)
```

Результат: Сформированная таблица outside_t с данными, представляющими собой результат внешнего соединения двух таблиц.

Подготовьте запрос и проверьте полученный результат.

X. Операторы манипулирования данными. Обновление данных

Общая форма оператора обновления

Update таблица

set поле=выражение [,поле=выражение]...[where предикат]

1. Обновление единственной записи.

Пример

Изменить цвет детали P2 на желтый, увеличить ее вес на 5 и установить значение города "неопределен".

```
Update P set цвет='желтый',
                    вес=вес+5,
                    город=NULL
                    where номер_детали='P2'
```

Результат: Таблица S с внесенными изменениями.

Подготовьте запрос и проверьте полученный результат.

2. Обновление множества записей.

Пример

Удвоить рейтинг всех поставщиков в Лондоне.

```
Update S set рейтинг=2*рейтинг
                    where город='Лондон'
```

Результат: Таблица S с увеличенным рейтингом для поставщиков из Лондона.

Подготовьте запрос и проверьте полученный результат.

3. Обновление с подзапросом.

Пример

Установить объем поставок, равный нулю для поставщиков из Лондона.

```
update SP set количество=0
where 'Лондон'=
(Select город
from S
where S.номер_поставщика=SP.номер_поставщика)
```

Результат: Таблица SP с внесенными изменениями.

Подготовьте запрос и проверьте полученный результат.

Список литературы

1.Грабер М. Введение в SQL. - М.: ЛОРИ, 1996. - 380 с. .

2. Хансен Г., Хансен Д. Базы данных и управление. - М.: Бином, 1999.

3. Дж.Уорсли, Дж. Дрейк. PostgreSQL для профессионалов. - С-Питербур: Питер, 2003.

4. Конноли Т., Бегг К., Страчан А. Базы данных. Проектирование, реализация и сопровождение. - М.- С./П.- К., 2000.