Министерство образования Российской Федерации
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

В.М. СТАСЫШИН, Т.Л. СТАСЫШИНА

ПРАКТИКУМ ПО ЯЗЫКУ SQL

Утверждено
Редакционно-издательским советом университета
в качестве учебного пособия

НОВОСИБИРСК
2016

УДК 004.655.3(075.8)
С 779

Рецензенты:
зам. директора ЦИУ *О.Е. Аврунев*,
канд. техн. наук, доцент каф. ТПИ *В.Г. Кобылянский*

Работа подготовлена на кафедре теоретической и прикладной информатики для студентов IV курса ФПМИ дневного отделения (направления 01.03.02, 02.03.03).

Стасышин В.М.
С 779 Практикум по языку SQL: учебное пособие / В.М. Стасышин, Т.Л. Стасышина. – Новосибирск, 2016. – 60 с.

ISBN 978-5-7782-2937-2

Язык SQL (Structured Query Language) является инструментом, предназначенным для выборки и обработки информации из базы данных. На сегодняшний день SQL – единственный стандартный язык баз данных, его поддерживают более ста СУБД, работающих как на мэйнфреймах, так и на серверах и персональных компьютерах.

Предлагаемое учебное пособие по изучению языка SQL содержат набор сгруппированных по темам заданий, выполнение которых позволит студентам получить начальные навыки по работе с базами данных и формированию запросов на языке SQL.

Учебное пособие может быть полезно также специалистам, занимающимся информационными технологиями и самостоятельно осваивающим работу на языке SQL.

УДК 004.655.3(075.8)

Стасышин Владимир Михайлович
Стасышина Татьяна Леонидовна

ПРАКТИКУМ ПО ЯЗЫКУ SQL

Учебное пособие

Редактор *И.Л. Кескевич*
Выпускающий редактор *И.П. Брованова*
Дизайн обложки *А.В. Ладыжская*
Компьютерная верстка *Н.В. Гаврилова*

Налоговая льгота – Общероссийский классификатор продукции
Издание соответствует коду 95 3000 ОК 005-93 (ОКП)

Подписано в печать 31.05.2016. Формат 60 × 84 1/16. Бумага офсетная
Тираж 100 экз. Уч.-изд. л. 3,48. Печ. л. 3,75. Изд. № 63. Заказ № 841
Цена договорная

Отпечатано в типографии
Новосибирского государственного технического университета
630073, г. Новосибирск, пр. К. Маркса, 20

ISBN 978-5-7782-2937-2

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Данное пособие содержит примеры SQL-запросов, аналогичных запросам из вариантов заданий при защите лабораторных работ по курсам «Технологии баз данных» и «Базы данных и экспертные системы».

Схема данных

Как и в лабораторных работах, в пособии используется схема базы данных, содержащая 4 таблицы.

Таблица поставщиков (S)

n_post (номер поставщика)	**name** (Фамилия)	**reiting** (рейтинг)	**town** (город)
S1	Смит	20	Лондон
S2	Джонс	10	Париж
S3	Блейк	30	Париж
S4	Кларк	20	Лондон
S5	Адамс	30	Афины

Таблица деталей (P)

n_det (номер детали)	**name** (название)	**cvet** (цвет)	**ves** (вес)	**town** (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 изделий (J)

n_izd (номер изделия)	**name** (название)	**town** (город)
J1	Жесткий диск	Париж
J2	Перфоратор	Рим
J3	Считыватель	Афины
J4	Принтер	Афины
J5	Флоппи-диск	Лондон
J6	Терминал	Осло
J7	Лента	Лондон

Таблица поставок (SPJ)

n_post (номер поставщика)	**n_det** (номер детали)	**n_izd** (номер изделия)	**kol** (количество)
S1	P1	J1	200
S1	P1	J4	700
S2	P3	J1	400
S2	P3	J2	200
S2	P3	J3	200
S2	P3	J4	500
S2	P3	J5	600
S2	P3	J6	400
S2	P3	J7	800
S2	P5	J2	100
S3	P3	J1	200
S3	P4	J2	500
S4	P6	J3	300
S4	P6	J7	300
S5	P2	J2	200
S5	P2	J4	100
S5	P5	J5	500
S5	P5	J7	100
S5	P6	J2	200
S5	P1	J4	100
S5	P3	J4	200
S5	P4	J4	800
S5	P5	J4	400
S5	P6	J4	500

Результаты запросов, приводимые в примерах, получены при использовании этих данных.

Поскольку основное внимание при подборе заданий уделялось формированию навыков написания запросов, в данных и в формулировках запросов не следует искать глубокого смысла и совпадений с реальностью.

Пояснения к формулировкам запросов

Число поставок…. == число строк таблицы spj ….

Объем поставки == количество поставленных деталей == значение столбца kol таблицы spj.

Количество деталей …. == количество поставленных деталей == значение столбца kol таблицы spj.

Вес поставки == количество поставленных деталей * вес поставленной детали (вес, указанный в таблице p, – это вес одной детали).

Число деталей …. == число разных номеров (наименований, видов, типов) деталей….

Число изделий …. == число разных номеров (наименований, видов, типов) изделий ….

Число поставщиков …. == число разных номеров (наименований, видов, типов) поставщиков ….

Город, из которого сделана поставка, == город, где производится деталь, указанная в поставке.

Город, куда сделана поставка, == город, где собирают изделие, указанное в поставке.

При выполнении заданий прежде всего необходимо проанализировать задачу, т. е. понять, какая именно информация должна быть затребована из таблиц и как ее нужно преобразовать. Хороший подход, особенно на первых порах, это представить, как бы вы работали с таблицами, если бы выполняли запрос вручную. Попытайтесь описать последовательность действий. Это поможет разбить сложный запрос на более простые подзадачи. Примеры данного пособия демонстрируют, как выделять простые подзадачи и собирать из них требуемый запрос.

ПРИМЕР 1 (Лабораторная работа 2 – задание 2)

Найти поставщиков, которые поставляли детали красного цвета для изделий с длиной названия >8. Вывести номера поставщиков без повторений в порядке возрастания номеров.

Анализ задания

Нужно последовательно проверить все поставки:

– по номеру детали определить ее цвет,

– по номеру изделия посмотреть его название и определить, сколько в нем символов.

Если деталь имеет красный цвет и в названии изделия более 8 символов, то номер поставщика из поставки нужно включить в результат. Из полученной выборки показать значения без повторений и упорядочить их.

Подзадача 1

Получить список всех поставок с дополнительной информацией о цвете детали и названии изделия.

Информация о поставках хранится в таблице spj. Информация о цвете деталей хранится в таблице p, столбец cvet. Информация о названии изделий хранится в таблице j, столбец name. Чтобы в каждой строке поставки было видно, какого цвета деталь и как называется изделие, нужно присоединить к каждой строке таблицы spj:

– соответствующую номеру детали строку из таблицы деталей p,

– соответствующую номеру изделия строку из таблицы изделий j.

Используем в запросе две секции join:

```
select spj.*, p.cvet,j.name izd_name
from spj
join p on p.n_det = spj.n_det
join j on j.n_izd = spj.n_izd
```

Результат запроса приведен ниже

n_post	n_det	n_izd	kol	cvet	izd_name
S1	P1	J1	200	Красный	Жесткий диск
S1	P1	J4	700	Красный	Принтер
S2	P3	J1	400	Голубой	Жесткий диск
S2	P3	J2	200	Голубой	Перфоратор
S2	P3	J3	200	Голубой	Считыватель
S2	P3	J4	500	Голубой	Принтер
S2	P3	J5	600	Голубой	Флоппи-диск
S2	P3	J6	400	Голубой	Терминал
S2	P3	J7	800	Голубой	Лента
S2	P5	J2	100	Голубой	Перфоратор
S3	P3	J1	200	Голубой	Жесткий диск
S3	P4	J2	500	Красный	Перфоратор
S4	P6	J3	300	Красный	Считыватель
S4	P6	J7	300	Красный	Лента
S5	P2	J2	200	Зеленый	Перфоратор
S5	P2	J4	100	Зеленый	Принтер
S5	P5	J5	500	Голубой	Флоппи-диск
S5	P5	J7	100	Голубой	Лента
S5	P6	J2	200	Красный	Перфоратор
S5	P1	J4	100	Красный	Принтер
S5	P3	J4	200	Голубой	Принтер
S5	P4	J4	800	Красный	Принтер
S5	P5	J4	400	Голубой	Принтер
S5	P6	J4	500	Красный	Принтер

В целевом списке запись spj.*, p.cvet,j.name izd_name означает, что из таблицы spj выводятся все столбцы, а из таблиц p и j по одному столбцу – цвет детали и название изделия. Для столбца j.name задан псевдоним (синоним) izd_name. Псевдонимы могут быть заданы для любого столбца целевого списка. Как правило, псевдонимы применяют, если в целевом списке есть столбцы с одинаковыми именами или используются функции и выражения.

Подзадача 2

Получить список всех поставок с дополнительной информацией о длине названия изделия.

Добавим еще один столбец – длину названия изделия. Для определения длины названия используем строковую функцию length:

```
select spj.*, p.cvet,
       j.name izd_name,
       length(j.name) len_name
from spj
join p on p.n_det = spj.n_det
join j on j.n_izd = spj.n_izd
```

Результат запроса приведен ниже

n_post	n_det	n_izd	kol	cvet	izd_name	len_name
S1	P1	J1	200	Красный	Жесткий диск	12
S1	P1	J4	700	Красный	Принтер	7
S2	P3	J1	400	Голубой	Жесткий диск	12
S2	P3	J2	200	Голубой	Перфоратор	10
S2	P3	J3	200	Голубой	Считыватель	11
S2	P3	J4	500	Голубой	Принтер	7
S2	P3	J5	600	Голубой	Флоппи-диск	11
S2	P3	J6	400	Голубой	Терминал	8
S2	P3	J7	800	Голубой	Лента	5
S2	P5	J2	100	Голубой	Перфоратор	10
S3	P3	J1	200	Голубой	Жесткий диск	12
S3	P4	J2	500	Красный	Перфоратор	10
S4	P6	J3	300	Красный	Считыватель	11
S4	P6	J7	300	Красный	Лента	5
S5	P2	J2	200	Зеленый	Перфоратор	10
S5	P2	J4	100	Зеленый	Принтер	7
S5	P5	J5	500	Голубой	Флоппи-диск	11
S5	P5	J7	100	Голубой	Лента	5
S5	P6	J2	200	Красный	Перфоратор	10
S5	P1	J4	100	Красный	Принтер	7
S5	P3	J4	200	Голубой	Принтер	7
S5	P4	J4	800	Красный	Принтер	7
S5	P5	J4	400	Голубой	Принтер	7
S5	P6	J4	500	Красный	Принтер	7

Замечание. Функция length относится к типу строковых функций, но не потому, что ее аргументом является символьная строка. Строко-

выми являются функции, вычисляемые на каждой строке (или записи) таблицы и возвращающие некоторое значение для каждой строки. В отличие от строковых агрегатные функции выполняются на группе строк, возвращая одно значение для всей группы.

Подзадача 3

Получить список всех поставок с красными деталями и изделиями, имеющими названия длиннее 8.

Добавим условие отбора строк. Нужно оставить только строки, где цвет деталей красный и длина названия изделия больше 8. Добавим секцию where в запрос:

```
select spj.*, p.cvet,
       j.name izd_name,
       length(j.name) len_name
from spj
join p on p.n_det = spj.n_det
join j on j.n_izd = spj.n_izd
where length(j.name)>8
        and
        p.cvet='Красный'
```

Результат запроса приведен ниже

n_post	n_det	n_izd	kol	cvet	izd_name	len_name
S1	P1	J1	200	Красный	Жесткий диск	12
S3	P4	J2	500	Красный	Перфоратор	10
S5	P6	J2	200	Красный	Перфоратор	10
S4	P6	J3	300	Красный	Считыватель	11

Окончательный запрос

Найти поставщиков, которые поставляли детали красного цвета для изделий с длиной названия >8. Вывести номера поставщиков без повторений в порядке возрастания номеров.

Осталось скорректировать целевой список запроса и добавить сортировку результата. Нужно оставить только один столбец (номер поставщика), причем вывести значения столбца без повторов, для чего используем предикат distinct. Чтобы результат был упорядочен, добавим секцию order by:

Запрос:

```
select distinct  spj.n_post
from spj
join p on p.n_det = spj.n_det
join j on j.n_izd = spj.n_izd
where length(trim(j.name))>8
        and
        p.cvet='Красный'
order by 1
```

Результат запроса:

n_post
S1
S3
S5
S4

На первый взгляд, добавление ключевого слова distinct не влияет на полученный результат, если его убрать, запрос вернет те же данные. Но такое поведение имеет место только на конкретных данных, которые сейчас заполняют таблицу spj. Добавьте в таблицу spj новую поставку: поставщик S5, деталь P1, изделие J3, количество 300 – и посмотрите, какой результат вернет запрос с использованием distinct и без. Поскольку запрос следует писать так, чтобы при любых данных в таблицах он возвращал правильный результат, использование предиката distinct обязательно в случаях, когда требуется выдать значения без повторений.

В секции order by указывают столбцы, по которым нужно выполнить сортировку. Эти столбцы можно задавать либо именами столбцов, либо их номерами в целевом списке. В последнем случае следует помнить, что перестановка столбцов в целевом списке отразится на порядке выдачи результата.

Задание для самостоятельной работы

1. *Вывести о каждой поставке информацию:*
- *город, откуда сделана поставка*;
- *город, куда поставлены детали*;
- *количество деталей*;
- *вес детали.*

2. *Вывести без повторений пары цвет детали – город, куда поставлена деталь.*

ПРИМЕР 2 (Лабораторная работа 4 – задание 1)

Выдать число изделий, для которых поставлялись детали из города, где проживает поставщик с максимальным рейтингом.

Анализ задания

Есть один или несколько поставщиков с наибольшим рейтингом.

Города, где эти поставщики проживают, составляют некоторое множество.

Нужно выбрать поставки такие, что поставляемая деталь произведена в городе из указанного множества.

Подсчитать число разных изделий в полученной выборке поставок.

Подзадача 1

Найти наибольший рейтинг поставщика.

Информация о рейтингах поставщиков хранится в таблице s, столбец reiting. Нужно найти наибольшее значение в столбце reiting по всей таблице s. Для этого используем агрегатную функцию max:

Запрос:

```
select max(s.reiting) maxr
from s
```

Результат запроса:

maxr
30

Замечание. Если в целевом списке стоит агрегатная функция, а секции group by в запросе нет, разбиения строк таблицы на группы не происходит. Точнее, вся таблица считается одной группой, и агрегатная функция выполняется на всей таблице целиком, возвращая единственную строку.

Подзадача 2

Найти города, где проживают поставщики с максимальным рейтингом.

Информация о городах, где проживают поставщики, хранится в таблице s, столбец town. Нужно получить значения столбца town из тех строк таблицы s, где рейтинг поставщика наибольший. Выполним сначала такой запрос:

Запрос:

```
select *
from s
where s.reiting=30
```

Результат запроса:

n_post	name	reiting	town
S3	Блейк	30	Париж
S5	Адамс	30	Афины

Для текущих данных этот запрос выводит поставщиков с максимальным рейтингом. Но если в таблицу s добавить нового поставщика с рейтингом 50, запрос перестанет выдавать верный результат. Чтобы запрос правильно работал на любых данных, нужно максимальный

рейтинг получать на лету. Для этого число 30 заменим запросом из подзадачи 1.

Запрос:

```
select s1.town
from s s1
where s1.reiting=(select max(s.reiting)
                  from s)
```

Результат запроса:

Town
Париж
Афины

Поскольку в последнем запросе таблица s присутствует в двух экземплярах, одному из них присвоен псевдоним s1. При использовании псевдонима для таблицы обращение к столбцам таблицы обозначается так:

<псевдоним таблицы>.<имя столбца>

Замечание. В запросе практически везде, где может стоять константа или имя столбца, можно написать взятый в скобки подзапрос. На такой подзапрос накладывается естественное ограничение – он должен возвращать единственное значение (одна строка, один столбец) соответствующего типа. Такого вида подзапрос может стоять даже в целевом списке select-запроса (см. ПРИМЕР 13).

Подзадача 3

Выбрать поставки такие, что деталь поставки произведена в городе из множества, полученного в предыдущем запросе.

Информация о поставках хранится в таблице spj, а информация о городах, где производят детали, хранится в таблице p, столбец town. К каждой строке таблицы поставок spj присоединим данные о детали.

```
select *
from spj
join p on p.n_det=spj.n_det
```

Результат запроса приведен ниже

n_post	n_det	n_izd	kol	n_det	name	cvet	ves	town
S1	P1	J1	200	P1	Гайка	Красный	12	Лондон
S1	P1	J4	700	P1	Гайка	Красный	12	Лондон
S2	P3	J1	400	P3	Винт	Голубой	17	Рим
S2	P3	J2	200	P3	Винт	Голубой	17	Рим
S2	P3	J3	200	P3	Винт	Голубой	17	Рим
…	…	…	…	…	…	…	…	…

Выберем поставки такие, где город изготовления деталей – Париж или Афины.

```
select *
from spj
join p on p.n_det=spj.n_det
where p.town in ('Париж','Афины')
```

Результат запроса приведен ниже

n_post	n_det	n_izd	kol	n_det	name	cvet	ves	town
S2	P5	J2	100	P5	Кулачок	Голубой	12	Париж
S5	P2	J2	200	P2	Болт	Зеленый	17	Париж
S5	P2	J4	100	P2	Болт	Зеленый	17	Париж
S5	P5	J5	500	P5	Кулачок	Голубой	12	Париж
S5	P5	J7	100	P5	Кулачок	Голубой	12	Париж
S5	P5	J4	400	P5	Кулачок	Голубой	12	Париж

Как и в предыдущей подзадаче, этот запрос выводит нужный результат только на текущих данных. Чтобы запрос правильно работал всегда, нужно множество городов получать на лету. Для этого список ('Париж','Афины') заменим запросом из подзадачи 2:

```
select *
from spj
join p on p.n_det=spj.n_det
where p.town in (select s1.town
                 from s s1
                 where s1.reiting=(select max(reiting)
                                   from s))
```

Нетрудно убедиться, что результат остался прежним.

Замечание. Так же как и в случае с константой, любой список в запросе можно заменить на взятый в скобки подзапрос, который должен возвращать один столбец соответствующего типа.

Окончательный запрос

Выдать число изделий, для которых поставлялись детали из города, где проживает поставщик с максимальным рейтингом.

Осталось подсчитать число разных изделий в выборке поставок, полученной в подзадаче 3. Для этого используем агрегатную функцию count, которая подсчитает число разных значений в столбце n_izd.

Запрос:

```
select count(distinct spj.n_izd) kol_izd
from spj
join p on p.n_det=spj.n_det
where p.town in (select s1.town
                 from s s1
                 where s1.reiting=(select max(reiting)
                                   from s))
```

Результат запроса:

kol_izd
4

Агрегатная функция count возвращает количество непустых (отличных от null) значений в столбце, заданном в качестве ее аргумента. Если перед именем столбца поставить предикат distinct, функция count вернет количество разных непустых значений в столбце. Если в качестве аргумента указана звездочка, т. е. count(*), функция вернет количество строк.

Задание для самостоятельной работы

Определить средний вес поставки в город, где производят изделие с самым длинным названием.

ПРИМЕР 3 (Лабораторная работа 2 – задание 1)

Выбрать изделия, для которых поставлялись красные детали, поставлявшиеся для изделия с названием длиной больше 11.

Анализ задания

Есть одно или несколько изделий, длина названия которых превышает 11 символов.

Для этих изделий сделаны поставки деталей. Перечень номеров этих деталей составляет некоторое множество. Каждая деталь из этого множества имеет тот или иной цвет, и можно выделить подмножество красных деталей. Нужно выбрать изделия, для которых поставлялись детали из этого подмножества красных деталей.

Подзадача 1

Найти изделия, длина названия которых превышает 11 символов.

Информация о названиях изделий хранится в таблице j, столбец name. Нужно найти длину значения в столбце name для каждой строки таблицы j и отобрать только те строки, для которых длина названия больше 11. Для определения длины названия используем строковую функцию length.

Запрос:

```
select j.*
from j
where length(j.name)>11
```

Результат запроса:

n_izd	name	town
J1	Жесткий диск	Париж

Подзадача 2

Выбрать детали, поставлявшиеся для изделий с названием длиной больше 11.

Информация о поставках хранится в таблице spj. Нужно к каждой строке таблицы spj присоединить информацию о названии изделия, определить длину названия, отобрать те строки таблицы spj, которым соответствует длина названия больше 11 символов, и вывести без повторений столбец номер детали (n_det).

Запрос:

```
select distinct n_det
from spj
join j on spj.n_izd= j.n_izd
where length(j.name)>11
```

Результат запроса:

n_det
P1
P3

Подзадача 3

Выбрать из деталей, полученных в предыдущем запросе, только красные.

Информация о цвете деталей хранится в столбце cvet таблицы p. К каждой строке таблицы поставок spj присоединим соответствующую номеру детали строку таблицы p и выберем только строки с красным цветом детали.

Запрос:

```
select distinct spj.n_det
from spj
join j on spj.n_izd= j.n_izd
join p on p.n_det=spj.n_det
where length(j.name)>11
      and p.cvet ='Красный'
```

Результат запроса:

n_det
P1

Окончательный запрос

Выбрать изделия, для которых поставлялись красные детали, поставлявшиеся для изделия с названием длиной больше 11.

Осталось вывести без повторений номера изделий, для которых поставлялись детали из множества, полученного запросом из подзадачи 3.

Для этого используем еще один экземпляр таблицы поставок spj. Присвоим ему псевдоним t, во избежание путаницы с экземпляром spj из подзадачи 3.

Запрос:

```
select distinct t.n_izd
from spj t
where t.n_det in (select spj.n_det
                  from spj
                  join p on p.n_det= spj.n_det
                  join j on j.n_izd= spj.n_izd
                  where length(j.name)>11
                        and p.cvet ='Красный')
```

Результат запроса:

n_izd
J1
J4

Задание для самостоятельной работы

Выбрать поставщиков, которые поставляли детали с весом > 17, поставлявшиеся для изделий из Афин.

ПРИМЕР 4 (Лабораторная работа 4 – задание 1)

Выдать число деталей, поставлявшихся поставщиками, проживающими в городе, где собирают изделие с буквой «ф» в названии.

Анализ задания

Есть одно или несколько изделий, в названии которых встречается буква «ф».

Такие изделия собирают в некотором множестве городов.

Нужно найти поставщиков, которые проживают в каком-либо городе из этого множества городов.

Выбрав поставки найденных поставщиков, следует подсчитать число разных деталей в полученной выборке поставок.

Подзадача 1

Найти изделия, в названии которых встречается буква «ф».

Информация о названиях изделий хранится в таблице j, столбец name. Нужно выбрать те строки таблицы j, где в столбце name встречается буква «ф». Поскольку в названии может встретиться и большая, и

маленькая буква «ф», приведем все символы названия к нижнему регистру, для этого используем строковую функцию lower.

Запрос:

```
select j.*
from j
where lower(j.name) like '%ф%'
```

Результат запроса:

n_izd	name	town
J2	Перфоратор	Рим
J5	Флоппи-диск	Лондон

Замечание. Для приведения к верхнему регистру используется функция upper.

Подзадача 2

Найти множество городов, в которых собирают полученные в подзадаче 1 изделия.

Информация о городе, где собирают изделие, хранится в столбце town, значит, нужно вывести только значения этого столбца.

Запрос:

```
select distinct j.town
from j
where lower(j.name) like '%ф%'
```

Результат запроса:

town
Рим
Лондон

Подзадача 3

Найти поставщиков, которые проживают в каком-либо городе из множества городов, полученного в подзадаче 2.

Информация о поставщиках хранится в таблице s. Нужно выбрать только те строки таблицы s, где город – один из городов, полученных в подзадаче 2.

Запрос:

```
select s.n_post
from s
where s.town in (select j.town
                 from j
                 where lower(j.name) like '%ф%')
```

Результат запроса:

n_post
S4
S1

Подзадача 4

Выбрать поставки найденных поставщиков.

Информация о поставках хранится в таблице spj. Нужно выбрать только те строки таблицы spj, где поставщик принадлежит множеству, полученному в подзадаче 3.

Запрос:

```
select *
from spj
where spj.n_post in (select s.n_post
                     from s
                     where s.town in (select j.town
                                      from j
                                      where lower(j.name) like '%ф%'))
```

Результат запроса:

n_post	n_det	n_izd	kol
S1	P1	J4	700
S1	P1	J1	200
S4	P6	J7	300
S4	P6	J3	300

Окончательный запрос

Выдать число деталей, поставлявшихся поставщиками, проживающими в городе, где собирают изделие с буквой «ф» в названии.

Осталось подсчитать число разных деталей в полученной выборке поставок. Для этого используем агрегатную функцию count.

Запрос:

```
select count(distinct spj.n_det) kol_det
from spj
where spj.n_post in (select s.n_post
                     from s
                     where s.town in (select j.town
                                      from j
                                      where lower(j.name) like '%ф%'))
```

или

```
select count(distinct spj.n_det) kol_det
from spj
join s on s.n_post = spj.n_post
where s.town in (select j.town
                 from j
                 where lower(j.name) like '%ф%')
```

Результат запроса:

kol_det
2

Функция count подсчитывает количество непустых (отличных от null) значений в столбце, переданном ей в качестве аргумента. Если перед названием столбца поставить предикат distinct, будет подсчитываться количество разных значений в столбце.

Задание для самостоятельной работы

Выдать число городов, в которые выполняли поставки поставщики, поставлявшие какую-либо зеленую деталь.

ПРИМЕР 5 (Лабораторная работа 2 – задание 5)

Построить таблицу со списком городов таких, что в городе вместе с поставщиком размещаются либо только детали, либо только изделия, но не то и другое вместе.

Анализ задачи

Есть первое множество городов – города, в каждом из которых размещается один или несколько поставщиков.

Есть второе множество городов – города, в каждом из которых производят одну или несколько деталей.

Есть третье множество городов – города, в каждом из которых собирают одно или несколько изделий.

Нужно из первого множества выбрать только те города, которые есть также либо только во втором множестве, либо только в третьем.

Подзадача 1

Найти первое множество городов – города поставщиков.

Запрос:

```
select distinct town
from s
```

Результат запроса:

town
Афины
Лондон
Париж

Запросы для получения второго и третьего множеств аналогичны.

Подзадача 2

Найти множество городов, в которых размещается либо изделие, либо деталь, либо и то и другое.

Это множество представляет собой объединение второго и третьего множеств городов. Получить объединение двух множеств позволяет операция union.

Запрос:

```
select town
      from j
union
select town
      from p
```

Результат запроса:

town
Афины
Лондон
Осло
Париж
Рим

Предикат distinct можно опустить, поскольку по умолчанию union подразумевает удаление из результата строк-дубликатов.

Подзадача 3

Найти множество городов, в каждом из которых размещается либо изделие, либо деталь, но не деталь и изделие вместе.

Пересечение второго и третьего множеств будет включать в себя города, в которых размещаются и деталь, и изделие. Поэтому, чтобы получить искомое множество городов, следует из множества, полученного запросом из подзадачи 2, вычесть пересечение второго и третьего множеств (т. е. города, где есть и деталь, и изделие). Получить пересечение двух множеств позволяет операция intersect, а разность двух множеств находим при помощи операции except.

Запрос:

```
select town from j
union
select town from p
except
( select town from j
 intersect
 select town from p
)
```

Результат запроса:

town
Афины
Осло

Если в операциях union, except или intersect участвует подзапрос, уже содержащий одну из этих операций, этот подзапрос следует взять в скобки.

Замечание. Операции union, except и intersect могут применяться только к select-запросам. Применить их к таблицам (т. е. написать s union p) нельзя. Кроме того, на запросы, участвующие в этих операциях, накладывается ограничение: у запросов должно быть одинаковое число столбцов, и типы столбцов должны совпадать. Если в целевом списке первого запроса два столбца и первый столбец – дата, а второй –

число, то во втором запросе тоже должно быть два столбца и первый столбец тоже должен быть датой, а второй – числом.

Подзадача 4

Найти список городов таких, что в городе вместе с поставщиком размещаются либо только детали, либо только изделия, но не то и другое вместе.

Для этого нужно получить пересечение первого множества городов и множества, полученного подзадачей 3.

Запрос:

```
select distinct town
    from s
intersect
( select town from j
  union
  select town from p
  except
  ( select town from j
    intersect
    select town from p
  )
)
```

Результат запроса:

town
Афины

Окончательный запрос

Построить таблицу со списком городов таких, что в городе вместе с поставщиком размещаются либо только детали, либо только изделия, но не то и другое вместе.

Осталось упорядочить результат подзадачи 4 и сохранить как таблицу. Это можно сделать разными способами.

Способ 1:

- создать таблицу командой create table;
- вставить данные в таблицу командой insert.

Запрос:

```
create table town (town character(6));
Insert into town ( select distinct town
                   from s
                   intersect
                   ( select town from j
                     union
                     select town from p
```

Сообщение phpPgAdmin:
1 запис(ь/и/ей) обработано.
Время выполнения: 14.577 мсек
SQL-запрос выполнен.

```
                    except
                    ( select town from j
                      intersect
                      select town from p
                    )
                )
            )
```

Способ 2. Сохранить результат запроса как таблицу командой create table в таком варианте.

Запрос:

```
Create table town as
    (select distinct town
    from s
    intersect
    ( select town from j
      union
      select town from p
      except
      ( select town from j
        intersect
        select town from p
      )
    )
)
```

Сообщение phpPgAdmin:
Данных не найдено.
Время выполнения: 77.925 мсек
SQL-запрос выполнен.

Способ 3. Сохранить результат запроса как таблицу командой select в таком варианте.

Запрос:

```
select distinct town into town from s
intersect
( select town from j
  union
  select town from p
  except
  ( select town from j
    intersect
    select town from p
  )
)
```

Сообщение phpPgAdmin:
Данных не найдено.
Время выполнения: 61.267 мсек
SQL-запрос выполнен.

Убедимся, что таблица создана, выполнив запрос.

Запрос:
Select * from town

Результат запроса:

town
Афины

Задание для самостоятельной работы

Построить таблицу с упорядоченным списком городов таких, что в городе размещается более одного поставщика и производится какая-либо деталь, но не собирается ни одно изделие

Измените данные таким образом, чтобы запрос давал другой результат.

ПРИМЕР 6 (Лабораторная работа 2 – задание 3)

Получить список деталей, которые поставляли ТОЛЬКО поставщики, выполнившие поставки в Рим.

Анализ задания

Есть несколько изделий, собираемых в Риме.

Находим множество поставщиков, у которых есть поставки для этих изделий.

В результирующий набор нужно включить детали, которые поставлялись только поставщиками из указанного множества. Если у детали хотя бы одна поставка выполнена поставщиком не из указанного множества, деталь из результата исключается.

Общий подход при написании таких запросов – найти два множества деталей:

– детали, которые поставлялись хотя бы одним поставщиком из указанного множества;

– детали, которые поставлялись поставщиком, не принадлежащим указанному множеству.

Разность первого и второго множеств дает искомое множество деталей.

Подзадача 1

Найти множество поставщиков, сделавших поставки для изделий из Рима.

К каждой строке таблицы поставок spj присоединяем соответствующую номеру изделия строку из таблицы j. Оставляем только те поставки, где город изделия (столбец j.town) – Рим. Из полученной выборки выводим только столбец n_post (номер поставщика), причем без повторов.

Запрос:

```
select distinct spj.n_post
from spj
join j on j.n_izd=spj.n_izd
where j.town='Рим'
```

Результат запроса:

n_post
S2
S3
S5

Подзадача 2

Найти детали, которые поставлялись хотя бы одним поставщиком из множества, полученного запросом из подзадачи 1.

Выбираем из таблицы spj только те строки, где поставщик (столбец n_post) принадлежит найденному в подзадаче 1 множеству. Из полученной выборки выводим без повторов значения столбца n_det (номер детали).

Запрос:

```
select distinct t.n_det
from spj t
where t.n_post in (select distinct spj.n_post
                   from spj
                   join j on j.n_izd=spj.n_izd
                   where j.town='Рим')
```

Результат запроса:

n_det
P1
P2
P3
P4
P5
P6

Поскольку в запросе теперь два экземпляра таблицы spj, одному из них присваиваем псевдоним t.

Подзадача 3

Найти детали, которые поставлялись поставщиком, не принадлежащим указанному множеству.

Для этого достаточно в условие отбора добавить отрицание.

Запрос:

```
select distinct t.n_det
from spj t
where not t.n_post in (select distinct spj.n_post
                       from spj
                       join j on j.n_izd=spj.n_izd
                       where j.town='Рим')
```

Результат запроса:

n_det
P1
P6

Окончательный запрос

Получить список деталей, которые поставляли ТОЛЬКО поставщики, выполнившие поставки в РИМ.

Осталось найти разность двух множеств. Для этого используем операцию except.

Запрос:

```
select t.n_det
from spj t
where t.n_post in (select distinct spj.n_post
                   from spj
                   join j on j.n_izd=spj.n_izd
                   where j.town='Рим')
except
select t.n_det
from spj t
where not t.n_post in (select distinct spj.n_post
                       from spj
                       join j on j.n_izd=spj.n_izd
                       where j.town='Рим')
```

Результат запроса:

n_det
P2
P3
P4
P5

Поскольку по умолчанию операция except выполняет удаление повторов из результирующего набора, предикат distinct в подзапросах можно не писать.

Задание для самостоятельной работы

Получить список изделий, для которых выполнили поставки ТОЛЬКО поставщики, поставлявшие детали из Парижа.

ПРИМЕР 7 (Лабораторная работа 4 – задание 5)

Выдать полную информацию о деталях, которые поставлялись ТОЛЬКО для изделий с числом поставок не менее 5.

Анализ задания

Для каждого изделия выполнено некоторое количество поставок.

Нужно отобрать только те изделия, для которых сделано 5 или более поставок.

В результирующий набор нужно включить детали, которые поставлялись только для изделий из указанного множества. Если деталь поставлялась хотя бы для одного изделия не из указанного множества, деталь из результата исключается.

Нужно найти два множества деталей:

– детали, которые поставлялись хотя бы для одного изделия из указанного множества;

– детали, которые поставлялись хотя бы для одного изделия, не принадлежащего указанному множеству.

Разность первого и второго множеств дает искомое множество деталей.

Подзадача 1

Найти для каждого изделия число поставок.

Информация о поставках хранится в таблице spj, каждая строка – одна поставка для изделия, номер которого указан в столбце n_izd. Значит, строки таблицы spj нужно разбить на группы (отдельная группа для каждого изделия) и подсчитать число строк в каждой группе. Чтобы разбить таблицу на группы, в запросе следует использовать секцию group by, а для подсчета количества строк требуется функция count.

Запрос:

```
select tj.n_izd, count(*) kp
from spj tj
group by tj.n_izd
```

Результат запроса:

n_izd	kp
J1	3
J2	5
J3	2
J4	8
J5	2
J6	1
J7	3

Подзадача 2

Получить список изделий, имеющих 5 и более поставок.

Запрос из подзадачи 1 нужно изменить таким образом, чтобы в результирующем множестве остались только те группы, для которых значение агрегатной функции больше или равно 5. Для этого добавим условие отбора групп, используем секцию having.

Запрос:

```
select tj.n_izd
from spj tj
group by tj.n_izd
having count(*)>=5
```

Результат запроса:

n_izd
J2
J4

Назначение секции having аналогично where, но where отрабатывает до выполнения группировки и вычисления агрегатных функций, а

having – после, что позволяет использовать в условиях секции having эти функции.

Подзадача 3

Найти детали, которые поставлялись хотя бы для одного изделия из множества, полученного в подзадаче 2.

Выбираем из таблицы spj только те строки, где изделие принадлежит найденному в подзадаче 2 множеству. Из полученной выборки выводим без повторов значения столбца n_det (номер детали).

Запрос:

```
select distinct t.n_det
from spj t
where t.n_izd in (select tj.n_izd
                  from spj tj
                  group by tj.n_izd
                  having count(*)>=5)
```

Результат запроса:

n_det
P1
P2
P3
P4
P5
P6

Подзадача 4

Выдать полную информацию о деталях, которые поставлялись хотя бы для одного изделия из множества, полученного в подзадаче 2.

Информация о деталях хранится в таблице p, нужно присоединить к каждой строке таблицы spj соответствующую номеру детали строку из таблицы деталей p и вывести все столбцы этой таблицы

```
select distinct p.*
from spj t
join p on p.n_det=t.n_det
where t.n_izd in (select tj.n_izd
                  from spj tj
                  group by tj.n_izd
                  having count(*)>=5)
```

Получаем такой результат:

n_det	name	cvet	ves	town
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

Подзадача 5

*Выдать полную информацию о деталях, которые поставлялись для изделий, **не** принадлежащих указанному множеству.*

Для этого достаточно в условие отбора добавить отрицание.

Запрос:

```
select distinct p.*
from spj t
join p on p.n_det=t.n_det
where not t.n_izd in (select tj.n_izd
                      from spj tj
                      group by tj.n_izd
                      having count(*)>=5)
```

Результат запроса:

n_det	name	cvet	ves	town
P1	Гайка	Красный	12	Лондон
P3	Винт	Голубой	17	Рим
P5	Кулачок	Голубой	12	Париж
P6	Блюм	Красный	19	Лондон

Окончательный запрос

Выдать полную информацию о деталях, которые поставлялись ТОЛЬКО для изделий с числом поставок не менее 5.

Осталось найти разность двух множеств. Для этого используем операцию except.

Запрос:

```
select distinct p.*
from spj t
join p on p.n_det=t.n_det
where t.n_izd in (select tj.n_izd
                  from spj tj
                  group by tj.n_izd
                  having count(*)>=5)
except
select distinct p.*
from spj t
join p on p.n_det=t.n_det
where not t.n_izd in (select tj.n_izd
                      from spj tj
                      group by tj.n_izd
                      having count(*)>=5)
```

Результат запроса:

n_det	name	cvet	ves	town
P2	Болт	Зеленый	17	Париж
P4	Винт	Красный	14	Лондон

Операции union, except и intersect могут применяться только к select-запросам с одинаковым составом целевого списка.

Задание для самостоятельной работы

Получить список поставщиков, которые поставляли ТОЛЬКО те детали, которые поставлялись для трех и более изделий.

ПРИМЕР 8 (Лабораторная работа 2 – задание 4)

Вывести полный список городов и для каждого города найти общий вес деталей красного цвета, поставленных в этот город. Города в списке должны быть ВСЕ. Список должен быть упорядочен по алфавиту.

Анализ задания

Есть три множества городов: города деталей, города изделий, города поставщиков. Нужно объединить эти три множества в один список городов.

Для каждой поставки определить цвет поставляемой детали, ее вес и город, куда сделана поставка. Отобрать только поставки красных деталей и подсчитать для каждой поставки ее вес. Отобранные поставки разбить на группы – отдельная группа для каждого города и получить общий вес поставок. Получится второй список городов.

Вывести каждый город из первого списка и, если город есть во втором списке, вывести общий вес поставок красных деталей. Если же города нет во втором списке, вывести пустое значение.

Подзадача 1

Найти полный список городов.

Нужно получить три множества: города из таблицы деталей, города из таблицы изделий, города из таблицы поставщиков – и объединить все три множества в один список и упорядочить по алфавиту. Для этого используем операцию union.

Запрос:

```
select town from p
union
select town from s
union
select town from j
order by 1
```

Результат запроса:

town
Афины
Лондон
Осло
Париж
Рим

Подзадача 2

Вывести вес каждой поставки красных деталей и город, куда сделана поставка.

Информация о поставках хранится в таблице spj. Информация о цвете и весе деталей хранится в таблице p, столбцы cvet и ves. Город, куда сделана поставка,– это город изделия. Информация об изделиях хранится в таблице j, нам нужен столбец town. Присоединим к каждой строке таблицы spj соответствующую номеру детали строку из таблицы деталей p и соответствующую номеру изделия строку из таблицы изделий j. Выведем всю информацию о поставке, а также цвет детали, город изделия и вес поставки. Отберем только красные детали и отсортируем результат по городу.

Запрос:

```
select spj.*,j.town,
       spj.kol*p.ves pves
from spj
join p on spj.n_det=p.n_det
join j on spj.n_izd=j.n_izd
where p.cvet='Красный'
order by 5
```

Результат запроса:

n_post	n_det	n_izd	kol	town	pves
S5	P1	J4	100	Афины	1200
S1	P1	J4	700	Афины	8400
S5	P6	J4	500	Афины	9500
S4	P6	J3	300	Афины	5700
S5	P4	J4	800	Афины	11200
S4	P6	J7	300	Лондон	5700
S1	P1	J1	200	Париж	2400
S5	P6	J2	200	Рим	3800
S3	P4	J2	500	Рим	7000

Подзадача 3

Получить суммарный вес поставок для каждого города.

Для этого в запрос следует добавить секцию group by и использовать агрегатную функцию sum.

Запрос:

```
select j.town, sum(spj.kol*p.ves) sves
from spj
join p on spj.n_det=p.n_det
join j on spj.n_izd=j.n_izd
where p.cvet='Красный'
group by j.town
order by 1
```

Результат запроса:

town	sves
Афины	36000
Лондон	5700
Париж	2400
Рим	10800

Если в запросе есть секция group by, в целевом списке могут стоять только:

– имена столбцов, перечисленных в group by;

– агрегатные функции.

В нашем случае оставляем в целевом списке только город изделия j.town и агрегатную функцию sum для подсчета общего веса поставленных в город деталей.

Окончательный запрос

Вывести полный список городов и для каждого города найти общий вес деталей красного цвета, поставленных в этот город. Города в списке должны быть ВСЕ. Список должен быть упорядочен по алфавиту.

Осталось соединить запросы из подзадачи 1 и подзадачи 3. Возьмем каждый из запросов в скобки и присвоим им псевдонимы:

– town – запросу из первой подзадачи;

– pt – запросу из подзадачи 3.

Таким образом, мы получили две виртуальные таблицы town и pt. Соединим их, используя секцию join и условие соединения – соответствие города.

Запрос:

```
select town.town, pt.sves
from (select town from p
        union
        select town from s
        union
        select town from j
        ) town
join ( select j.town, sum(spj.kol*p.ves) sves
       from spj
       join p on spj.n_det=p.n_det
       join j on spj.n_izd=j.n_izd
```

Результат запроса:

town	sves
Афины	36000
Лондон	5700
Париж	2400
Рим	10800

```
        where p.cvet='Красный'
        group by j.town
       ) pt on pt.town=town.town
order by 1
```

Замечание. Практически везде, где в запросе пишется имя таблицы, можно поместить подзапрос – виртуальную таблицу. Для этого подзапрос нужно взять в скобки и присвоить ему псевдоним («town» и «pt» в примере выше).

Результат получился не совсем тот, что нужен. Город Осло исчез из списка. Это произошло потому, что соединение join выбрасывает те строки виртуальной таблицы town, для которых нет соответствия в виртуальной таблице pt. В данном случае нам нужно соединение, оставляющее все строки виртуальной таблицы town, – это внешнее соединение left join.

Запрос:

```
select town.town, pt.sves
from ( select town from p
        union
        select town from s
        union
        select town from j
        ) town
left join ( select j.town,
                sum(spj.kol*p.ves) sves
            from spj
            join p on spj.n_det=p.n_det
            join j on spj.n_izd=j.n_izd
            where p.cvet='Красный'
            group by j.town
            ) pt on pt.town=town.town
order by 1
```

Результат запроса:

town	sves
Афины	36000
Лондон	5700
Осло	NULL
Париж	2400
Рим	10800

Внешнее соединение left join добавляет к строке виртуальной таблицы town строку из виртуальной таблицы pt в случае существования подходящей строки. Если подходящей строки нет, вместо строки из pt вставляются пустые значения.

Задание для самостоятельной работы

Вывести полный список поставщиков и для каждого поставщика определить, детали скольких разных цветов были им поставлены для изделия J4. Поставщики в списке должны быть ВСЕ. Список должен быть упорядочен по номеру.

ПРИМЕР 9 (Лабораторная работа 6 – задание 3)

Для каждой детали вывести ее номер и средний объем поставок этой детали поставщиком S2 или 0, если поставок не было. Детали в списке должны быть ВСЕ. Список должен быть упорядочен по номеру детали.

Анализ задания

Есть множество деталей, которые могут поставляться (все детали).

Поставщик S2 поставляет некоторое подмножество деталей.

Для каждой из этих деталей он сделал несколько поставок разного объема, и можно вычислить средний объем поставки детали.

Для всех имеющихся деталей нужно:

– вывести номер детали;

– вывести найденный средний объем для детали или 0, если поставок детали поставщиком S2 не было.

Подзадача 1

Вычислить средний объем поставки для каждой детали, которую поставлял S2.

Информация о том, какой поставщик в каком объеме какие детали поставлял, хранится в таблице spj. Нужно выбрать только поставки поставщика S2, сгруппировать поставки по деталям (секция group by) и для каждой группы определить среднее значение в столбце «Количество» (kol). Для определения среднего используем агрегатную функцию avg.

Запрос:

```
select t.n_det, avg(t.kol) avgkol
from spj t
where t.n_post ='S2'
group by t.n_det
```

Результат запроса:

n_det	avgkol
P3	442.8571428571428571
P5	100.0000000000000000

Чтобы избежать отображения большого количества знаков после запятой, можно использовать строковую функцию round (округлить).

Запрос:

```
select t.n_det, round(avg(t.kol),2) avgkol
from spj t
where t.n_post ='S2'
group by t.n_det
```

Результат запроса:

n_det	avgkol
P3	442.86
P5	100.00

Замечание. Аргументом строковой функции может быть другая строковая или агрегатная функция, если она возвращает значение допустимого типа. Аргументом агрегатной функции может быть строковая функция, но НЕ МОЖЕТ быть другая агрегатная функция. Иными словами, написать max(length(s.name)) допустимо, а max(avg(kol)) – ошибка.

Подзадача 2

Для каждой детали вывести ее номер и средний объем поставок этой детали поставщиком S2.

Информация обо всех имеющихся деталях находится в таблице p. К каждой строке таблицы p нужно присоединить соответствующую строку запроса из подзадачи 1. Для этого используем секцию join, а запрос из подзадачи 1 оформим как виртуальную таблицу a: заключим в скобки и присвоим ему псевдоним a. Поскольку в виртуальной таблице a есть не все детали, используем левое внешнее соединение (left join).

Запрос:

```
select p.n_det, a.avgkol
from p
left join ( select t.n_det, round(avg(t.kol),2) avgkol
            from spj t
            where t.n_post ='S2'
            group by t.n_det
          ) a on p.n_det =a.n_det
```

Результат запроса:

n_det	avgkol
P1	NULL
P2	NULL
P3	442.86
P4	NULL
P5	100.00
P6	NULL

Окончательный запрос

Для каждой детали вывести ее номер и средний объем поставок этой детали поставщиком S2 или 0, если поставок не было. Детали в списке должны быть ВСЕ. Список должен быть упорядочен по номеру детали.

Осталось в столбце avgkol заменить псевдозначение null на 0 и добавить сортировку результата. Несмотря на то что список получился упорядоченным по номеру детали, в общем случае СУБД не гарантирует этого при каждом выполнении запроса. Чтобы результат выводился упорядоченным всегда, нужно добавить секцию order by. Для замены null на 0 воспользуемся конструкцией case, имеющей такой формат:

case when <условие>
then <значение, если условие истинно>
else < значение, если условие ложно>
end

Условие в нашем случае – проверка, что в столбце avgkol не пустое значение. Если условие не выполняется, вместо null будет показан 0.

Запрос:

```
select p.n_det,
       case when not a.avgkol is null
            then a.avgkol
            else 0
       end avgkol
from p
left join ( select t.n_det, round(avg(t.kol),2) avgkol
            from spj t
            where t.n_post ='S2'
            group by t.n_det
          ) a on p.n_det =a.n_det
order by 1
```

Результат запроса:

n_det	avgkol
P1	0
P2	0
P3	442.86
P4	0
P5	100.00
P6	0

Задание для самостоятельной работы

Для каждой детали определить, сколько поставщиков поставляли эту деталь для изделий с длиной названия меньше 7. Детали в списке должны быть ВСЕ. Список должен быть упорядочен по номеру детали.

ПРИМЕР 10 (Лабораторная работа 5 – задание 1)

Получить средний вес поставки для каждого изделия и найти их среднее.

Анализ задания

Для каждого изделия сделано несколько поставок разного объема. Нужно вычислить средний вес поставки для каждого изделия, затем найти среднее этих средних весов.

Подзадача 1

Получить средний вес поставки для каждого изделия.

Информация о поставках хранится в таблице spj. Также потребуется информация о весе деталей, хранящаяся в таблице p, столбец ves. Присоединим к каждой строке таблицы spj соответствующую номеру

детали строку из таблицы деталей p, вычислим вес каждой поставки. Сгруппируем поставки по изделиям и воспользуемся агрегатной функцией avg для определения среднего веса поставки.

Запрос:

```
select t.n_izd,avg(t.kol*p.ves) sves
from spj t
join p on p.n_det=t.n_det
group by t.n_izd
order by 1
```

Результат запроса:

n_izd	sves
J1	4200.0000000000000000
J2	3760.0000000000000000
J3	4550.0000000000000000
J4	6087.5000000000000000
J5	8100.0000000000000000
J6	6800.0000000000000000
J7	6833.3333333333333333

Окончательный запрос

Получить средний вес поставки для каждого изделия и найти их среднее.

Осталось найти среднее значение для столбца sves предыдущего запроса. Оформим предыдущий запрос как виртуальную таблицу b: возьмем запрос в скобки и обозначим псевдонимом b. Чтобы избежать отображения большого количества знаков после запятой, воспользуемся строковой функцией round (округлить).

Запрос:

```
select round(avg(b.sves),2) sr
from ( select avg(t.kol*p.ves) sves
       from spj t
       join p on p.n_det=t.n_det
       group by t.n_izd
     ) b
```

Результат запроса:

sr
5761.55

Задание для самостоятельной работы

Получить число поставок для каждой детали и найти их среднее.

ПРИМЕР 11 (Лабораторная работа 2 – задание 6)

Каждое изделие, в названии которого есть буква «к», перевести в город, в котором проживает поставщик, сделавший для изделия наименьшую по объему поставку. Если таких городов больше одного, перевести в первый по алфавиту из этих городов.

Анализ задания

Есть несколько изделий, в названии которых встречается буква «к».

Для каждого из этих изделий нужно найти поставку с наименьшим количеством деталей (возможно, таких поставок окажется несколько). Получится множество поставок (свое для каждого изделия).

Для каждой поставки из множества нужно определить город поставщика, сделавшего поставку.

Из полученного множества городов взять первый по алфавиту город и именно его записать в качестве города изделия.

Замечание. Ошибки при выполнении операций изменения данных могут повлечь за собой серьезные проблемы. Поэтому при написании сложных запросов на обновление данных рекомендуется сначала написать select-запрос, который вернет:

- список строк, подлежащих обновлению,
- старые значения обновляемых полей,
- новые значения обновляемых полей.

Подзадача 1

Найти множество изделий, в названии которых встречается буква «к».

Информация о названиях изделий хранится в таблице j, столбец name. Нужно выбрать те строки таблицы j, где в столбце name встречается буква «к». Поскольку в названии может встретиться и большая, и маленькая буква «к», приведем все символы названия к нижнему регистру, для этого используем строковую функцию lower.

Запрос:

```
select j.*
from j
where lower(j.name) like '%к%'
```

Результат запроса:

n_izd	name	town
J1	Жесткий диск	Париж
J5	Флоппи-диск	Лондон

Запрос вернул строки, подлежащие обновлению, и текущие значения столбца город (town). Теперь нужно определить для каждой строки новое значение столбца город.

Подзадача 2

Выбрать поставки для изделия J1.

Информация о поставках хранится в таблице spj. Выбираем из таблицы spj только строки, где значение поля n_izd равно J1. Полученную выборку упорядочим по количеству деталей.

Запрос:

```
select *
from spj
where spj.n_izd='J1'
order by spj.kol
```

Результат запроса:

n_post	n_det	n_izd	kol
S1	P1	J1	200
S3	P3	J1	200
S2	P3	J1	400

Подзадача 3

Присоединить к каждой строке поставки информацию о городе поставщика.

Информация о городах, где размещаются поставщики, хранится в поле town таблицы s. Чтобы для каждой поставки было видно, из какого города поставщик, нужно присоединить к каждой строке таблицы spj соответствующую номеру поставщика строку из таблицы s.

Запрос:

```
select spj.*, s.town
from spj
join s on s.n_post=spj.n_post
where spj.n_izd='J1'
order by spj.kol
```

Результат запроса:

n_post	n_det	n_izd	kol	town
S3	P3	J1	200	Париж
S1	P1	J1	200	Лондон
S2	P3	J1	400	Париж

Подзадача 4

Определить новый город для изделия J1.

Поставок с минимальным количеством деталей две. Они выполнены поставщиками из разных городов, поэтому нужно выбрать тот город, который идет первым по алфавиту. Для этого полученный в подзадаче 3 список нужно дополнительно упорядочить по городу поставщика, взять из получившегося упорядоченного списка только первую строку и вывести из нее столбец town.

Запрос:

```
select s.town
from spj
join s on s.n_post=spj.n_post
where spj.n_izd='J1'
order by spj.kol, s.town
limit 1
```

Результат запроса:

town
Лондон

Подзадача 5

Определить новый город для каждого изделия из списка, полученного в подзадаче 1.

Для этого преобразуем запрос из подзадачи 4: заменим константу 'J1' на текущее изделие j.n_izd, заключим запрос в скобки и присвоим ему псевдоним town_new. Преобразованный запрос поместим в целевой список запроса из подзадачи 1.

Запрос:

```
select j.*,
       (select s.town from spj
       join s on s.n_post=spj.n_post
       where spj.n_izd=j.n_izd
       order by spj.kol, s.town
       limit 1) town_new
from j
where lower(j.name) like '%к%'
```

Результат запроса:

n_izd	name	town	town_new
J1	Жесткий диск	Париж	Лондон
J5	Флоппи-диск	Лондон	Афины

Окончательный запрос

Каждое изделие, в названии которого есть буква «к», перевести в город, в котором проживает поставщик, сделавший для изделия наименьшую по объему поставку. Если таких городов больше одного, перевести в первый по алфавиту из этих городов.

Осталось select-запрос преобразовать в запрос на обновление.

Для этого:

- заменим «select j.*», на «update j set town=»
- уберем псевдоним town_new
- уберем «from j»

Запрос:

```
update j set town= (select s.town
                    from spj
                    join s on s.n_post=spj.n_post
                    where spj.n_izd=j.n_izd
                    order by spj.kol, s.town
                    limit 1 )
where lower(j.name) like '%к%'
```

Результат запроса:

```
2 запис(ь/и/ей) обработано.
Время выполнения: 34.344 мсек
SQL-запрос выполнен.
```

Этот запрос выполняет для каждой из обновляемых строк коррелированный подзапрос, чтобы определить новое значение столбца town.

Убедимся, что данные изменились, выполнив запрос из подзадачи 1:

n_izd	name	town
J1	Жесткий диск	Лондон
J5	Флоппи-диск	Афины

Задание для самостоятельной работы

Каждую красную деталь перевести в город, в котором проживает поставщик, сделавший наибольшую по весу поставку этой детали. Если таких городов больше одного, перевести в первый по алфавиту из этих городов.

ПРИМЕР 12 (Лабораторная работа 3 – задание 2)

Получить номера изделий, для которых поставлялась КАЖДАЯ деталь из списка поставщика S2.

Анализ задания

Есть некоторое множество деталей (детали, поставлявшиеся поставщиком S2).

Нужно найти изделия, для которых поставлялась **каждая** деталь из указанного множества. Если для изделия хотя бы одна деталь не поставлялась, изделие не должно включаться в результирующий список.

Один из подходов при написании таких запросов:

– получить указанное в задании множество деталей;

– подсчитать число деталей в указанном множестве;

– для каждого изделия найти подмножество деталей: детали из указанного множества, поставлявшиеся для изделия;

– подсчитать число деталей в каждом подмножестве;

– выбрать только те подмножества (и соответственно изделия), где число деталей в подмножестве равно числу деталей в исходном множестве.

Подзадача 1

Получить множество деталей, которые поставлял S2.

Информация о том, какой поставщик какие детали поставлял, хранится в таблице spj. Нужно выбрать только поставки поставщика S2 и вывести только значения столбца n_det (номер детали) без повторов.

Запрос:

```
select distinct d.n_det
from spj d
where d.n_post ='S2'
```

Результат запроса:

n_det
P3
P5

Подзадача 2

Подсчитать, сколько деталей в заданном множестве.

Для подсчета числа деталей используем агрегатную функцию count с предикатом distinct в аргументе, чтобы получить число разных значений в столбце n_det (номер детали).

Запрос:

```
select count(distinct d.n_det)
from spj d
where d.n_post ='S2'
```

Результат запроса:

count
2

Подзадача 3

Найти изделия, для которых поставлялась хотя бы одна деталь из заданного множества.

Для этого оставляем только поставки деталей из множества, полученного в подзадаче 1, выводим только столбцы n_izd (номер изделия) и n_det (номер детали) без повторов. Добавим сортировку (секция order by), чтобы было проще визуально проанализировать данные.

Запрос:

```
select distinct t.n_izd, t. n_det
from spj t
where t.n_det in (select d.n_det
                  from spj d
                  where d.n_post ='S2')
order by t.n_izd, t. n_det
```

Результат запроса:

n_izd	n_det
J1	P3
J2	P3
J2	P5
J3	P3
J4	P3

J4	P5
J5	P3
J5	P5
J6	P3
J7	P3
J7	P5

Очевидно, что окончательный запрос (при текущих данных) должен будет вернуть изделия J2, J4, J5, J7.

Подзадача 4

Подсчитаем, сколько деталей из заданного множества поставлялось для каждого изделия.

Для этого множество, полученное подзадачей 3, нужно разбить на группы (отдельная группа для каждого изделия) и подсчитать число разных значений столбца n_det в каждой группе. Чтобы разбить множество на группы, в запрос следует включить секцию group by, а для подсчета числа значений используем функцию count.

Запрос:

```
select t.n_izd, count(distinct t.n_det) count
from spj t
where t.n_det in (select d.n_det
                  from spj d
                  where d.n_post ='S2')
group by t.n_izd
order by t.n_izd
```

Результат запроса:

n_izd	count
J1	1
J2	2
J3	1
J4	2
J5	2
J6	1
J7	2

Окончательный запрос

Получить номера изделий, для которых поставлялась КАЖДАЯ деталь из списка поставщика S2.

Очевидно, что осталось сравнить значения в столбце count запроса из подзадачи 4 со значением, полученным в подзадаче 2, и оставить только те изделия, где эти значения равны.

Для этого добавим условие отбора групп, используем секцию having.

Запрос:

```
select t.n_izd
from spj t
where t.n_det in (select d.n_det
                  from spj d
                  where d.n_post ='S2')
group by t.n_izd
having count(distinct t.n_det)=(select count(distinct d.n_det)
                                from spj d
                                where d.n_post ='S2')
order by t.n_izd
```

Результат запроса:

n_izd
J2
J4
J5
J7

Секция having позволяет оставить в результирующем списке только те группы, для которых значение агрегатной функции удовлетворяет заданному условию.

Измените данные в таблицах и убедитесь, что запрос по-прежнему дает правильный результат.

Например:

- Удалите в таблице spj строку

S5	P5	J4	400

Поскольку теперь деталь P5 не поставляется для изделия J4, это изделие должно исчезнуть из результата запроса.

- Добавьте в таблицу spj строку

S3	P5	J3	100

Поскольку теперь для изделия J3 поставляются обе детали (и P3, и P5), изделие J3 должно присутствовать в результирующем множестве.

Задание для самостоятельной работы

Выдать номера изделий, детали для которых поставляет каждый поставщик, поставляющий какую-либо красную деталь

ПРИМЕР 13 (Лабораторная работа 4 – задание 2)

Поменять местами города, где проживают поставщики с минимальным и максимальным рейтингом, т. е. поставщиков с минимальным рейтингом перевести в город, где проживает поставщик с максимальным рейтингом, и наоборот, поставщиков с максимальным рейтингом перевести в город, где проживает поставщик с минимальным рейтингом. Если городов несколько, брать первый по алфавиту из этих городов.

Анализ задания

Есть поставщики (один или несколько) с минимальным рейтингом. Города, где проживают эти поставщики, образуют некоторое множество. Город 1 – первый по алфавиту из этого множества.

Есть поставщики (один или несколько) с максимальным рейтингом. Города, где проживают эти поставщики, тоже образуют некоторое множество. Город 2 – первый по алфавиту из этих городов.

Нужно всем поставщикам с минимальным рейтингом установить город равным Городу 2, а всем поставщикам с максимальным рейтингом установить город равным Городу 1.

Замечание. Произвести такой обмен можно по-разному. Например, сохранить в каких-либо переменных Город 1 и Город 2, а затем выполнить два оператора update, используя эти переменные. Реализовать этот подход можно либо при помощи клиентской программы, использующей sql, либо применяя функцию на plSQL, либо используя временные таблицы. Однако эту операцию можно выполнить всего одной update-командой. Как и в примере 11 напишем select-запрос, который вернет список строк, подлежащих обновлению, а также старые и новые значения обновляемых полей.

Подзадача 1

Получить город поставщика с минимальным рейтингом (*Город 1*).

Информация о поставщиках хранится в таблице s, столбец town – город, столбец reiting – рейтинг. Нужно выбрать поставщиков с минимальным рейтингом, упорядочить полученную выборку по городу (секция order by) и вывести город только из первой строки (опция limit).

Запрос:

```
select s.town town1
from s
where s.reiting=(select min(reiting)
                 from s s1)
order by s.town
limit 1
```

Результат запроса:

town1
Париж

Тот же результат можно получить более коротким запросом: упорядочим поставщиков по рейтингу, а строки с одинаковым рейтингом – по названию города и возьмем город из первой записи:

```
select s.town town1
from s
order by s.reiting, s.town
limit 1
```

С другой стороны, опция limit присутствует не в каждой СУБД. Запрос без использования limit будет выглядеть так:

```
select min(s.town) town1
from s
where s.reiting=(select min(reiting)
                 from s s1)
```

Агрегатные функции min и max являются перегружаемыми и работают в том числе с символьными данными, возвращая первое и последнее по алфавиту значение соответственно.

Нетрудно убедиться, что все три варианта дают один и тот же результат.

Подзадача 2

Получить город поставщика с максимальным рейтингом (*Город 2*).

По аналогии с предыдущим запросом возможны три варианта.

Варианты запросов:

```
1. select s.town town2
from s
where s.reiting=(select max(reiting)
                 from s s2)
order by s.town
limit 1
```

Результат запроса:

town2
Афины

```
2. select s.town town2
from s
order by s.reiting desc, s.town
limit 1
3. select min(s.town) town2
from s
where s.reiting=(select max(reiting)
                 from s s2)
```

Подзадача 3

Получить поставщиков с минимальным или максимальным рейтингом (*записи, подлежащие обновлению*) *и города, где они размещаются* (*текущие значения обновляемого столбца*).

Достаточно в секции where усложнить условие.

Запрос:

```
select s.n_post, s.reiting, s.town
from s
where s.reiting=(select min(reiting) from s s1)
      or
      s.reiting=(select max(reiting) from s s2)
```

Результат запроса:

n_post	reiting	town
S2	10	Париж
S3	30	Париж
S5	30	Афины

Подзадача 4

Получить новые значения городов для поставщиков из предыдущей подзадачи.

Выведем в каждой строке запроса из подзадачи 3 Город 1 и Город 2. Для этого поместим непосредственно в целевой список запросы из подзадачи 1 и подзадачи 2. Это допустимо, поскольку оба этих запроса возвращают единственное значение.

Запрос:

```
select s.n_post, s.reiting, s.town,
       ( select s3.town town1
       from s s3
       order by s3.reiting, s3.town
       limit 1) town1,
       ( select s4.town town2
       from s s4
       order by s4.reiting desc, s4.town
       limit 1) town2
from s
```

```
where s.reiting=(select min(reiting)
                 from s s1)
      or
       s.reiting=(select max(reiting)
                 from s s2)
```

Результат запроса:

n_post	reiting	town	town1	town2
S2	10	Париж	Париж	Афины
S3	30	Париж	Париж	Афины
S5	30	Афины	Париж	Афины

Теперь в каждой строке осталось сделать выбор между town1 и town2. Используем для этого конструкцию case, имеющую такой формат

case when <условие>
then <значение, если условие истинно>
else < значение, если условие ложно>
end

В нашем случае <условие> – равенство рейтинга максимальному значению. Если условие выполняется, выбирается значение town1, иначе town2.

Запрос:

```
select s.n_post, s.reiting, s.town,
      case when s.reiting=(select max(reiting) from s s5)
            then ( select s3.town town1
                   from s s3
                   order by s3.reiting, s3.town
                   limit 1)
            else ( select s4.town town2
                   from s s4
                   order by s4.reiting desc, s4.town
                   limit 1)
      end town_new
from s
where s.reiting=( select min(reiting)
                  from s s1)
      or
      s.reiting=( select max(reiting)
                  from s s2)
```

Результат запроса:

n_post	reiting	town	town_new
S2	10	Париж	Афины
S3	30	Париж	Париж
S5	30	Афины	Париж

Окончательный запрос

Поменять местами города, где проживают поставщики с минимальным и максимальным рейтингом.

Осталось select-запрос преобразовать в запрос на обновление. Делаем это так же, как в примере 11:

```
update s set town= (case when s.reiting=(select max(reiting)
                                         from s s5)
                         then ( select s3.town town1
                                from s s3
                                order by s3.reiting, s3.town
                                limit 1)
                         else ( select s4.town town2
                                from s s4
                                order by s4.reiting desc, s4.town
                                limit 1)
                    end)
where s.reiting=(select min(reiting) from s s1)
      or
       s.reiting=(select max(reiting) from s s2)
```

Результат запроса:

```
3 запис(ь/и/ей) обработано.
Время выполнения: 4.964 мсек
SQL-запрос выполнен.
```

Убедимся, что данные изменились, выполнив запрос из подзадачи 3:

n_post	reiting	town
S2	10	Афины
S3	30	Париж
S5	30	Париж

Замечание. Такой запрос не будет работать при попытке поменять местами значения столбцов, являющихся первичным ключом или его

частью. Впрочем, изменять первичный ключ записи – это плохая практика и ее следует избегать в любом случае.

Задание для самостоятельной работы

Поменять местами рейтинги первого и последнего по алфавиту поставщиков.

ПРИМЕР 14 (Лабораторная работа 4 – задание 4)

Выбрать города, в которые не делал поставок ни один поставщик, поставлявший зеленые детали.

Анализ задания

Имеется множество деталей зеленого цвета. Имеется множество поставщиков, поставлявших эти детали.

Нужно получить список городов, куда могут быть сделаны поставки (т. е. города, где собирают изделия), но куда найденные поставщики не делали поставок. Если хотя бы один поставщик делал поставку для изделия из города, город должен быть исключен из результата.

Общий подход при написании таких запросов – найти два множества городов:

– города, куда могут быть сделаны поставки (города изделий);

– города, куда выполнили поставки поставщики зеленых деталей.

Разность первого и второго множеств дает искомое множество городов.

Подзадача 1

Найти множество поставщиков зеленых деталей.

Информация о поставках хранится в таблице spj, информация о цвете деталей хранится в таблице p, столбец cvet. Используем секцию join, чтобы присоединить к каждой строке таблицы spj соответствующую номеру детали строку из таблицы деталей p. Выберем только те поставки, где цвет детали – зеленый, и выведем только значение столбца n_post (номер поставщика) без повторов:

Запрос:

```
select distinct n_post
from spj a
join p on p.n_det=a.n_det
where p.cvet='Зеленый'
```

Результат запроса:

n_post
S5

Подзадача 2

Найти множество городов, куда выполнили поставки поставщики зеленых деталей.

Информация о городах изделий хранится в таблице j, столбец town. Присоединим к каждой строке таблицы spj соответствующую номеру изделия строку из таблицы изделий j. Выберем только строки, где поставщик – один из найденных запросом из подзадачи 1, и выведем только значение столбца j.town (город) без повторов:

Запрос:

```
select distinct j1.town
from spj t
join j j1 on j1.n_izd=t.n_izd
where t.n_post in ( select n_post
                    from spj a
                    join p on p.n_det=a.n_det
                    where p.cvet='Зеленый')
```

Результат запроса:

town
Лондон
Афины
Рим

Подзадача 3

Найти множество городов, куда могут быть сделаны поставки.

Для этого нужно вывести без повторений города (столбец town) из таблицы j.

Запрос:

```
select distinct j.town
from j
```

Результат запроса:

town
Афины
Лондон
Осло
Париж
Рим

Окончательный запрос

Выбрать города, в которые не делал поставок ни один поставщик, поставлявший зеленые детали.

Осталось найти разность запросов подзадачи 3 и подзадачи 2.

Запрос:

```
select town
from j
except
select j1.town
```

Результат запроса:

town
Париж
Осло

```
from spj t
join j j1 on j1.n_izd = t.n_izd
where t.n_post in ( select n_post
                    from spj a
                    join p on p.n_det=a.n_det
                    where p.cvet='Зеленый')
```

Задание для самостоятельной работы

Выбрать детали, не поставлявшиеся ни для одного из изделий, для которых поставлял детали поставщик S3.

ПРИМЕР 15 (Лабораторная работа 4 – задание 3)

Найти поставщиков, имеющих поставки, вес которых составляет менее 20 % от максимального веса поставки красной детали, выполненной этим поставщиком. Вывести номер поставщика, вес поставки, максимальный вес поставки красной детали, сделанной поставщиком.

Анализ задания

Для каждого поставщика нужно отобрать поставки деталей красного цвета.

Найти вес каждой из этих поставок и определить наибольшую среди этих величин (наибольший вес).

Для каждого поставщика этот наибольший вес будет свой.

Затем для каждого поставщика сравнить вес каждой его поставки с найденным наибольшим весом, деленным на 5 (т. е. с 20 % от наибольшего веса). Если вес поставки меньше, включить информацию о поставке в результат, если больше или равен – не включать.

Подзадача 1

Вывести поставки деталей красного цвета и для каждой поставки определить ее вес.

Информация о поставках хранится в таблице spj, информация о деталях хранится в таблице p, нам нужны столбцы cvet и ves. Используем секцию join, чтобы присоединить к каждой строке таблицы spj соответствующую номеру детали строку из таблицы деталей p. Выберем только те поставки, где цвет детали – красный, и добавим в целевой список выражение для вычисления веса поставки.

Запрос:

```
select t.*, t.kol*p.ves pves
from spj t
join p on p.n_det=t.n_det
where p.cvet='Красный'
```

Результат запроса:

n_post	n_det	n_izd	kol	pves
S1	P1	J4	700	8400
S1	P1	J1	200	2400
S3	P4	J2	500	7000
S4	P6	J7	300	5700
S4	P6	J3	300	5700
S5	P1	J4	100	1200
S5	P4	J4	800	11200
S5	P6	J2	200	3800
S5	P6	J4	500	9500

Выражениям в целевом списке рекомендуется всегда присваивать псевдонимы, в данном случае выражение t.kol*p.ves обозначено псевдонимом pves.

Подзадача 2

Найти для каждого поставщика максимальный вес поставки красной детали.

Полученный в предыдущей подзадаче список нужно разделить на группы (отдельная группа для каждого поставщика) и выбрать наибольшее значение в столбце pves в каждой группе. Чтобы разбить список на группы, в запросе следует использовать секцию group by, а для нахождения наибольшего значения – функцию max.

Запрос:

```
select t.n_post, max(t.kol*p.ves) mves
from spj t
join p on p.n_det=t.n_det
where p.cvet='Красный'
group by t.n_post
```

Результат запроса:

n_post	mves
S1	8400
S3	7000
S4	5700
S5	11200

Подзадача 3

Для каждой поставки вывести номер поставщика, вес поставки, максимальный вес поставки красной детали, сделанной поставщиком.

К каждой строке таблицы spj, помимо соответствующей номеру детали строки из таблицы деталей p, присоединим строку из запроса подзадачи 2 по соответствию номера поставщика. Для этого запрос из подзадачи 2 возьмем в скобки, зададим ему псевдоним b.

Запрос:

```
select a.n_post, a.kol*pa.ves pves, b.mves
from spj a
join p pa on pa.n_det=a.n_det
join ( select t.n_post, max(t.kol*p.ves) mves
       from spj t
       join p on p.n_det=t.n_det
       where p.cvet='Красный'
       group by t.n_post
       ) b on b.n_post=a.n_post
order by 1,2
```

Результат запроса:

n_post	pves	mves
S1	2400	8400
S1	8400	8400
S3	3400	7000
S3	7000	7000
S4	5700	5700
S4	5700	5700
S5	1200	11200
S5	1200	11200
S5	1700	11200
S5	3400	11200
S5	3400	11200
S5	3800	11200
S5	4800	11200
S5	6000	11200
S5	9500	11200
S5	11200	11200

Окончательный запрос

Найти поставщиков, имеющих поставки, вес которых составляет менее 20 % от максимального веса поставки красной детали, сделанной этим поставщиком. Вывести номер поставщика, вес поставки, максимальный вес поставки красной детали, сделанной поставщиком.

Осталось сравнить значения в столбцах pves и mves и оставить только те строки, где значения столбца pves меньше, чем значение столбца mves, деленное на 5.

Запрос:

```
select a.n_post, a.kol*pa.ves pves, b.mves
from spj a
join p pa on pa.n_det=a.n_det
join ( select t.n_post, max(t.kol*p.ves) mves
       from spj t
       join p on p.n_det=t.n_det
       where p.cvet='Красный'
       group by t.n_post
       ) b on b.n_post=a.n_post
where a.kol*pa.ves< b.mves/5.0
order by 1,2
```

Результат запроса:

n_post	pves	mves
S5	1200	11200
S5	1200	11200
S5	1700	11200

Чтобы исключить из результата поставки с одинаковым весом, можно использовать предикат distinct.

Задание для самостоятельной работы

Найти изделия, имеющие поставки, объем которых в 8 раз меньше объема наибольшей поставки для изделия. Вывести номер изделия, объем поставки, объем наибольшей поставки для изделия.

ПРИМЕР 16 (Лабораторная работа 5 – задание 2)

Для каждого поставщика из Парижа найти число поставок каждой детали, им поставлявшейся. Вывести номер поставщика, город поставщика, номер детали, название детали, число поставок.

Анализ задания

В Париже проживают несколько поставщиков.

Нужно выбрать поставки, сделанные этими поставщиками. Разделить их на группы: каждая группа – поставки одним поставщиком одной из деталей. Определить число поставок в каждой группе. К полученным строкам добавить информацию о городе поставщика и названии детали.

Подзадача 1

Выбрать поставки, сделанные поставщиками из Парижа.

Информация о поставках хранится в таблице spj, информация о поставщиках хранится в таблице s, город в столбце town. Присоединим к каждой строке таблицы spj соответствующую номеру детали строку из таблицы поставщиков s. Выберем только те поставки, где город поставщика – Париж.

Запрос:

```
select spj.*
from spj t
join s s1 on t.n_post = s1.n_post
where s1.town='Париж'
order by t.n_post, t.n_det
```

Результат запроса:

n_post	n_det	n_izd	kol
S2	P3	J1	400
S2	P3	J7	800
S2	P3	J6	400
S2	P3	J5	600
S2	P3	J4	500
S2	P3	J3	200
S2	P3	J2	200
S2	P5	J2	100
S3	P3	J1	200
S3	P4	J2	500

Подзадача 2

Для каждого поставщика из Парижа найти число поставок каждой детали, им поставлявшейся.

Сгруппируем поставки по поставщикам и по деталям, для подсчета числа поставок воспользуемся агрегатной функцией count.

Запрос:

```
select t.n_post, t.n_det, count(*) p_sr
from spj t
join s s1 on t.n_post = s1.n_post
where s1.town='Париж'
group by t.n_post, t.n_det
```

Результат запроса:

n_post	n_det	p_sr
S2	P3	7
S2	P5	1
S3	P3	1
S3	P4	1

Окончательный запрос

Для каждого поставщика из Парижа найти число поставок каждой детали, им поставлявшейся. Вывести номер поставщика, город поставщика, номер детали, название детали, число поставок.

Осталось добавить в каждой строке город поставщика и название детали. Эти столбцы нельзя просто добавить в целевой список, так как в запросе есть группировка. Если в запросе есть секция group by, в целевом списке могут стоять только:

- имена столбцов, перечисленных в group by;
- агрегатные функции.

Возникает соблазн добавить город поставщика и название детали в список группировки, но это решение, во-первых, не является оптимальным; во-вторых, делает запрос менее читабельным.

Поэтому оформим предыдущий запрос как виртуальную таблицу: заключим запрос в скобки и присвоим псевдоним a. Присоединим к виртуальной таблице a соответствующие строки таблиц s и p, выведем нужные столбцы.

Запрос:

```
select a.n_post,s.town, a.n_det,p.name, a.p_sr
from (select t.n_post, t.n_det, count(*) p_sr
      from spj t
      join s s1 on t.n_post = s1.n_post
      where s1.town='Париж'
      group by t.n_post, t.n_det
      ) a
```

```
join p on a.n_det = p.n_det
join s on a.n_post = s.n_post
order by a.n_post, a.n_det
```

Результат запроса:

n_post	town	n_det	name	p_sr
S2	Париж	P3	Винт	7
S2	Париж	P5	Кулачок	1
S3	Париж	P3	Винт	1
S3	Париж	P4	Винт	1

Задание для самостоятельной работы

Для каждой детали из Лондона найти максимальный объем поставки этой детали каждым поставщиком, ее поставлявшим. Вывести номер детали, город детали, номер поставщика, фамилию поставщика, максимальный объем поставки детали поставщиком.

ПРИМЕР 17 (Лабораторная работа 5 – задание 3)

Найти города поставщиков, поставлявших деталь P1, и определить, какой процент составляет суммарный вес поставок поставщиками из каждого города от общего веса поставленных деталей P1. Вывести город, суммарный вес поставок поставщиков из этого города, общий вес поставленных деталей P1, процент.

Анализ задания

Нужно определить общий вес всех поставленных деталей P1. Эта величина – 100 %.

Для каждой поставки детали P1 определить, в каком городе живет поставщик, сделавший поставку. Разделить поставки на группы: каждая группа – поставки поставщиками из одного города. Для каждой группы вычислить суммарный вес деталей P1 и определить, сколько процентов эта величина составляет от 100 %.

Подзадача 1

Определить общий вес всех поставленных деталей P1.

Информация о поставках хранится в таблице spj. Также потребуется информация о весе деталей, хранящаяся в таблице p, столбец ves. Присоединим к каждой строке таблицы spj соответствующую номеру

детали строку из таблицы деталей p, отберем только поставки детали P1 и просуммируем веса поставок.

Запрос:

```
select sum(t.kol*p.ves) oves
from spj t
join p on p.n_det=t.n_det
where t.n_det= 'P1'
```

Результат запроса:

oves
12000

Подзадача 2

Вывести поставки детали P1 с дополнительной информацией о городе поставщика и весе поставки.

Присоединим к каждой строке таблицы spj соответствующую номеру детали строку из таблицы деталей p и соответствующую номеру поставщика строку из таблицы s. Отберем только поставки детали P1. Выведем все столбцы таблицы spj, столбец town (город) из таблицы s и добавим выражение для вычисления веса поставки.

Запрос:

```
select t.*, s.town, t.kol*p.ves pves
from spj t
join s on s.n_post=t.n_post
join p on p.n_det=t.n_det
where t.n_det= 'P1'
```

Результат запроса:

n_post	n_det	n_izd	kol	town	pves
S1	P1	J1	200	Лондон	2400
S1	P1	J4	700	Лондон	8400
S5	P1	J4	100	Афины	1200

Подзадача 3

Вычислить суммарный вес поставок детали P1 поставщиками из каждого города.

Добавим в предыдущий запрос группировку по городу поставщика и просуммируем веса поставок.

Запрос:

```
select s.town, sum(t.kol*p.ves) pves
from spj t
join s on s.n_post=t.n_post
join p on p.n_det=t.n_det
where t.n_det= 'P1'
group by s.town
```

Результат запроса:

town	pves
Афины	1200
Лондон	10800

Подзадача 4

Присоединить к каждой строке предыдущего запроса величину, полученную в первой подзадаче.

Запрос из подзадачи 1 всегда возвращает одну строку, и эту строку нужно присоединить к каждой строке запроса подзадачи 3. Для этого оформим оба запроса как виртуальные таблицы (заключим запросы в скобки и обозначим псевдонимом) и используем соединение cross join (декартово произведение, соединение каждой строки одной таблицы с каждой строкой другой).

Запрос:

```
select a.town, a.pves, b.oves
from ( select s.town, sum(t.kol*p.ves) pves
       from spj t
       join s on s.n_post=t.n_post
       join p on p.n_det=t.n_det
       where t.n_det= 'P1'
       group by s.town
       ) a
cross join ( select sum(t.kol*p.ves) oves
             from spj t
             join p on p.n_det=t.n_det
             where t.n_det= 'P1'
             ) b
```

Результат запроса:

town	pves	oves
Афины	1200	12000
Лондон	10800	12000

Окончательный запрос

Найти города поставщиков, поставлявших деталь P1, и определить, какой процент составляет суммарный вес поставок поставщиками из каждого города от общего веса поставленных деталей P1. Вывести город, суммарный вес поставок поставщиков из этого города, общий вес поставленных деталей P1, процент.

Осталось вычислить, какой процент составляет значение в столбце pves от значения в столбце oves.

Запрос:

```
select a.town, a.pves, b.oves,
       round(a.pves*100.0/b.oves,2) proc
from ( select s.town, sum(t.kol*p.ves) pves
       from spj t
       join s on s.n_post=t.n_post
       join p on p.n_det=t.n_det
       where t.n_det= 'P1'
       group by s.town
    ) a
```

Результат запроса:

town	pves	oves	proc
Афины	1200	12000	10.00
Лондон	10800	12000	90.00

```
cross join ( select sum(t.kol*p.ves) oves
             from spj t
             join p on p.n_det=t.n_det
             where t.n_det= 'P1') b
```

Задание для самостоятельной работы

1. *Найти детали, которые поставлял поставщик S4, и определить, какой процент составляет суммарный вес поставок каждой детали от общего веса деталей, поставленных S4. Вывести номер детали, суммарный вес поставок этой детали, общий вес деталей поставщика S4, процент.*

2. *Найти изделия, для которых поставлял детали поставщик S2, и определить, какой процент составляют поставки для каждого изделия от общего числа поставок S2. Вывести номер изделия, число поставок для этого изделия, общее число поставок поставщика S2, процент.*

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Стасышин В.М.* Язык структурных запросов SQL: учеб. пособие. – Новосибирск: Изд-во НГТУ, 1996. – 33 с.

2. *Грабер М.* Введение в SQL. – М.: ЛОРИ, 1996.

3. *Гроff Д.Р., Вайнберг П.Н.* SQL: полное руководство. – Киев: BMV, «Ирина», 2001.