**Лабораторная работа №4**

**Тема: Построение нетривиальных запросов на SQL.**

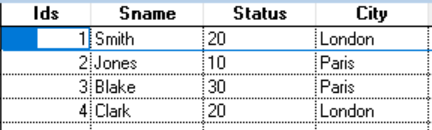
1. Соединение таблицы самой с собой
2. Применение курсора
3. Подзапросы, возвращающие единичное значение
4. Подзапросы, возвращающие множество значений
5. Предикаты на подзапросах, использующие SOME, ANY и ALL
6. Предикаты на подзапросах, использующие операции EXISTS и NOT EXISTS

В стандарте предусмотрены и в VisualFoxPro реализованы две формы языка SQL: интерактивная и встроенная. Продолжаем работать с интерактивной формой языка. Однако запросы на SQL с этого момента будут создаваться и исполняться без использования инструментов QBE в командном окне Command. Все примеры рассматриваются на базе данных Поставщики-Детали.

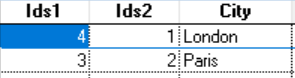
**1. Соединение таблицы самой с собой**

В запросах на выборку таблица может **соединяться сама с собой**. Это классический пример, демонстрирующий необходимость переименования.

Пример1.1. **Определить пары поставщиков из одного города**. Это значит, что из таблицы



требуется получить таблицу



Запрос, реализующий такое требование, должен в подсхеме содержать дважды одну таблицу, т.е. таблица в предложении FROM должна быть представлена двумя своими копиями, каждой копии назначаются псевдонимы. В результирующей таблице также применяются псевдонимы для одноименных атрибутов из двух копий исходной таблицы. Помимо условия соединения в предложении WHERE применяют упорядочение одноименных атрибутов, чтобы избавиться от повторов:

SELECT a.ids as ids1, b.ids as ids2, a.city;

FROM s As a, s As b;

WHERE a.city=b.city AND a.ids>b.ids

**2. Применение курсора**

Когда требование запроса предполагает применение агрегатной функции к агрегатной, что не допускается синтаксисом SQL, а также в других сложных случаях возможно применение курсора. Это значит, что запрос выполняется по этапам, при этом промежуточные результаты сохраняются во временных таблицах – курсоре. Каждый следующий этап может использовать курсор и атрибуты курсора предыдущих этапов наравне с таблицами и атрибутами базы данных.

Пример 2.1. Определить поставщика с максимальной суммарной поставкой.

SELECT S.ids,S.sname, SUM(Sp.qty) as KDet;

FROM s, sp INTO CURSOR tmp;

WHERE S.ids=Sp.ids GROUP BY S.ids,S.sname

SELECT tmp.ids, tmp.sname, KDet From tmp;

WHERE KDet In(Select MAX(Kdet) FROM tmp)



Пример 2.2 Поставщик, поставляющий один тип деталей

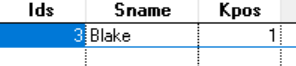
SELECT S.ids,S.sname,COUNT(Sp.qty) as KPos;

FROM s, sp INTO CURSOR tmp;

WHERE S.ids=Sp.ids GROUP BY S.ids, S.sname

SELECT tmp.ids, tmp.sname, KPos From tmp;

WHERE KPos=1



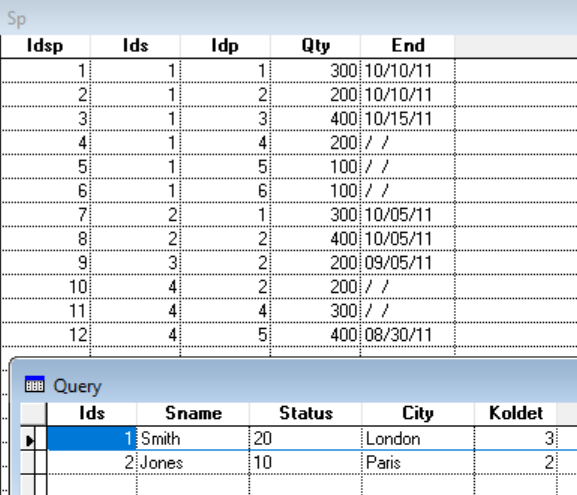
Пример 2.3. Выбрать поставщика, поставившего более одного вида деталей

SELECT sp.ids As NomPos, COUNT(sp.idp)As KolDet FROM sp;

Where NOT Empty(sp.end) Group by sp.ids into cursor t1

SELECT s.\*,t1.KolDet from s,t1 Where s.ids=t1.NomPos AND KolDet>1

*Функция EMPTY(sp.end) возвращает ИСТИНУ, если поле аргумента (sp.end) пусто.*



Пример 2.4. Выбрать клиентов, поставивших детали на максимальную сумму

SELECT s.sname As NomPos,Sum(p.prise\*sp.qty)AS StoimPos;

FROM s,sp,p Where s.ids=sp.ids AND p.idp=sp.idp AND NOT EMPTY(sp.end);

GROUP BY s.sname into cursor t1

SELECT t1.NomPos, t1.StoimPos FROM t1;

WHERE StoimPos=(SELECT MAX(StoimPos) FROM t1)



**3. Подзапросы, возвращающие единичное значение**

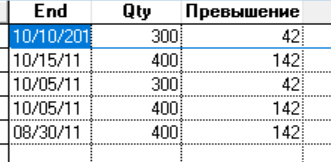
**Вложенные запросы** – это законченные операторы SELECT, внедренные в тело внешнего оператора SELECT. Внутренний оператор SELECT кодируется по тем же правилам, что и внешний оператора SELECT, заключается в круглые скобки. Часто вложенный запрос реализует подзапрос для вычисления условий поиска.  Внешний оператор SELECT использует результат выполнения внутреннего оператора для определения содержания окончательного результата всей операции. Когда есть уверенность, что подзапрос возвращает единственное значение, для сравнения с критерием поиска применяются операторы отношения (=, <, >, <=, >=, <>). В этом случае внутренние запросы помещают непосредственно после оператора сравнения предложения  WHERE (или HAVING) внешнего оператора SELECT.

Пример 3.1. Определить даты поставок, превысивших по количеству деталей среднее значение и указать для этих поставок превышение над средним уровнем

SELECT sp.end, sp.qty,sp.qty-(SELECT Avg(sp.qty);

FROM sp) AS Превышение;

FROM sp WHERE sp.qty>(SELECT Avg(qty)FROM sp)and Not EMPTY(sp.end)

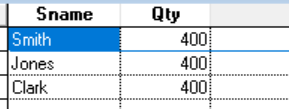


Пример 3.2. Определить поставщиков, поставивших поставки с максимальным количеством деталей

SELECT s.sname, sp.qty;

FROM s,sp Where s.ids=sp.ids and;

sp.qty=(SELECT Max(sp.qty)FROM sp)

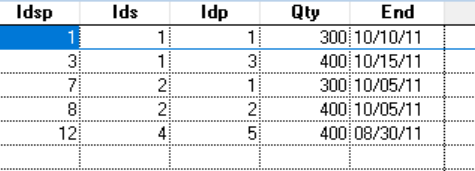


Пример 3.3 Определить поставки, в которых количество поставленных деталей оказалось больше среднего показателя 1-ого поставщика

SELECT sp.\* FROM sp;

WHERE sp.qty>(SELECT avg(sp.qty) FROM sp Where sp.idp=1);

AND NOT EMPTY(sp.end)

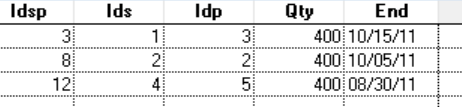


Пример 3.4. Определить поставки, в которых количество поставленных деталей оказалось больше среднего показателя на заданном интервале дат (здесь задан интервал дат от 20 сентября 2011 года до 5 октября 2011 года)

SELECT sp.\* FROM sp;

WHERE sp.qty>(SELECT avg(sp.qty) FROM sp Where sp.end between {^2011/9/20} and {^2011/10/5});

AND NOT EMPTY(sp.end)



*Предложение {^yyyy/mm/dd} или предложение CDOT(‘mm/dd/yy’)применяются для задания конкретной даты.*

**4. Подзапросы, возвращающие множество значений**

В общем случае, подзапрос возвращает множество значений, т.е. значения одного или нескольких столбцов таблицы, размещенные в более чем одной строке. Вложенные подзапросы генерируют непоименованное промежуточное отношение, временную таблицу. Оно может использоваться только в том месте, где появляется в подзапросе. К такому отношению невозможно обратиться по имени из какого-либо другого места запроса. Применяемые к подзапросу операции основаны на тех операциях, которые применяются к множеству, а именно:

• { WHERE | HAVING } выражение [ NOT ] IN ( подзапрос );

• { WHERE | HAVING } выражение оператор\_сравнения { ALL | SOME | ANY } ( подзапрос );

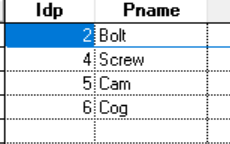
• {WHERE | HAVING } [ NOT ] EXISTS ( подзапрос );

Пример 4.1. Определить название деталей из незавершенных поставок

SELECT p.idp, p.pname FROM p;

WHERE p.idp In (SELECT sp.idp;

FROM sp WHERE EMPTY(sp.end))



Пример 4.2. Определить список деталей, которые поставляют только поставщики со статусом 20

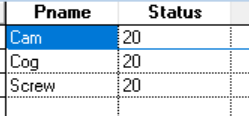
SELECT DISTINCT p.pname, s.status;

FROM p, s, sp WHERE p.idp=sp.idp AND s.ids=sp.ids AND;

s.status='20' and p.pname NOT IN ;

(SELECT p.pname;

FROM p,sp,s WHERE s.ids=sp.ids AND p.idp=sp.idp AND s.status<>’20’)

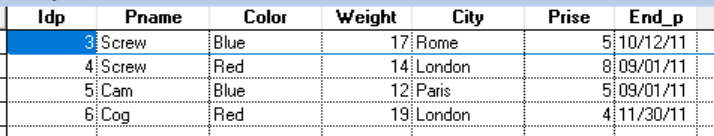


Пример 4.3. Определить список деталей, которые не поставляют поставщики из Парижа.

SELECT DISTINCT p.\*;

FROM p WHERE p.idp NOT IN;

(SELECT sp.idp FROM s,sp WHERE s.ids=sp.ids AND s.city='Paris’)



*Для таблиц, между которыми связь m:n, запрос на исключение реализуется от противного. В рассматриваемом примере во вложенном запросе определяется список деталей, поставляемых поставщиками из Парижа. Во внешнем запросе выбираются те детали, которые не входят в этот список.*

**5. Предикаты на подзапросах, использующие SOME, ANY и ALL**

Ключевые слова ANY и ALL могут использоваться с подзапросами, возвращающими один столбец чисел.

Если подзапросу будет предшествовать ключевое слово ALL, условие сравнения считается выполненным, только когда оно выполняется для всех значений в результирующем столбце подзапроса.

Если записи подзапроса предшествует ключевое слово ANY, то условие сравнения считается выполненным, когда оно выполняется хотя бы для одного из значений в результирующем столбце подзапроса.

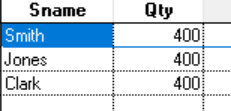
Если в результате выполнения подзапроса получено пустое значение, то для ключевого слова ALL условие сравнения будет считаться выполненным, а для ключевого слова ANY – невыполненным. Ключевое слово SOME является синонимом слова ANY.

Пример 5.1. Определить поставщиков, поставивших максимальное количество деталей (эквивалентно запросу с оператором «=»)

SELECT s.sname, sp.qty FROM s,sp;

WHERE s.ids=sp.ids AND sp.qty>=ALL(SELECT sp.qty FROM sp)

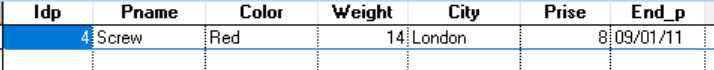
*В примере определены поставщики, в поставках которых количество деталей больше или равно количества деталей в каждой из всех поставок.*



Пример 5.2. Выбрать все детали, цена которых выше цены любой детали из Парижа

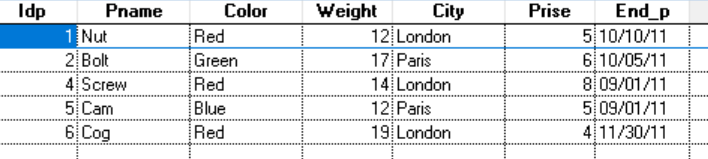
SELECT \* FROM p WHERE p.prise>;

All(SELECT p.prise FROM p WHERE p.city=’Paris’)



Пример 5.3. Выбрать все детали, производимые в городах, где есть поставщики

SELECT \* FROM p WHERE p.city=ANY(SELECT s.city FROM s)



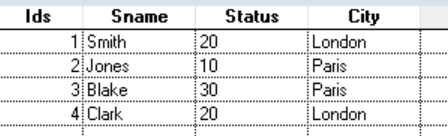
Пример 5.4. Найти поставщиков, в поставках которых количество деталей превышает размер хотя бы одной поставки поставщиков из Парижа.

SELECT s.sname, sp.qty FROM s,sp;

WHERE s.ids=sp.ids AND sp.qty>;

ANY(SELECT sp.qty FROM s,sp ;

WHERE s.ids=sp.ids AND s.city='Paris')



**6.** **Предикаты на подзапросах, использующие операции EXISTS и NOT EXISTS**

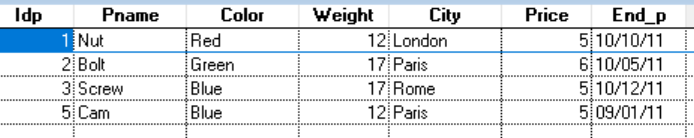
Ключевые слова EXISTS и NOT EXISTS предназначены для использования только совместно с подзапросами. Результат их обработки представляет собой логическое значение TRUE или FALSE. Для ключевого слова EXISTS результат равен TRUE в том и только в том случае, если в возвращаемой подзапросом результирующей таблице присутствует хотя бы одна строка. Если результирующая таблица подзапроса пуста, результатом обработки операции EXISTS будет значение FALSE. Для ключевого слова NOT EXISTS используются правила обработки, обратные по отношению к ключевому слову EXISTS . Поскольку по ключевым словам EXISTS и NOT EXISTS проверяется лишь наличие строк в результирующей таблице подзапроса, то эта таблица может содержать произвольное количество столбцов.

Пример 6.1. Определить список поставленных деталей

SELECT p.\* FROM p;

WHERE EXISTS (SELECT sp.idp FROM sp;

WHERE p.idp=sp.idp AND NOT EMPTY(sp.end))



Пример 6.2. Выбрать список деталей, поставки которых не выполнены.

SELECT p.\* FROM p;

WHERE EXISTS (SELECT sp.idp FROM sp;

WHERE p.idp=sp.idp AND EMPTY(sp.end))

