СРСП 7. Запросы с использованием нескольких таблиц

SQL обладает механизмом для одновременной или последовательной обработки данных из нескольких взаимосвязанных таблиц. В нем реализованы возможности "соединять" или "объединять" несколько таблиц и так называемые "вложенные подзапросы".

Операции соединения подразделяются на два вида - внутренние и внешние. Оба вида соединений задаются в предложении WHERE запроса SELECT с помощью специального условия соединения. Внешние соединения поддерживаются стандартом содержат зарезервированное слово "JOIN", в то время как внутренние соединения (или просто соединения) могут задаваться как без использования такого слова, так и с использованием слова "JOIN". Связывание производится, как правило, по первичному ключу одной таблицы и внешнему ключу другой таблицы - для каждой пары таблиц.

Внутреннее соединение возвращает только те строки, для которых условие соединения принимает значение true.

Например, чтобы получить наименование товаров и наименование категории товаров, поставляемых поставщиком с кодом=1 , возможен запрос

```
SELECT   Наименование, Категория товара, Количество, Цена
FROM     Товары, Движение товара
WHERE    Товары.Код товара = Движение товара. Код товара
AND      Код поставщика = 1;
```

Результат выборки

Наименование	Категория товара	Количество	Цена
Товар_2	мебель	100	10
Товар_2	мебель	100	10
Товар_2	мебель	10	12
Товар_2	мебель	100	12

Выборка получена следующим образом: СУБД последовательно формирует строки декартова произведения таблиц, перечисленных во фразе FROM, проверяет, удовлетворяют ли данные сформированной строки условиям фразы WHERE, и если удовлетворяют, то включает в ответ на запрос те ее поля, которые перечислены во фразе SELECT.

Следует подчеркнуть, что в SELECT и WHERE (во избежание двусмысленности) ссылки на все (*) или отдельные столбцы могут (а иногда и должны) уточняться именем соответствующей таблицы, например, Товары.Код товара, Движение товара.Код товара и т.п.

Внешнее соединение возвращает **все** строки из одной таблицы и только те строки из другой таблицы, для которых условие соединения принимает значение true. Строки второй таблицы, не удовлетворяющие условию соединения (т.е. имеющие значение false), получают значение null в результирующем наборе.

Существует следующие виды внешнего соединения: LEFT JOIN, RIGHT JOIN и FULL JOIN.

В левом соединении (LEFT JOIN) запрос возвращает все строки из левой таблицы (т.е. таблицы, стоящей *слева* от зарезервированного словосочетания "LEFT JOIN" и только те из правой таблицы, которые удовлетворяют условию соединения. Если же в правой таблице не найдется строк, удовлетворяющих заданному условию, то в результате они замещаются значениями null.

Для правого соединения - все наоборот.

В полное соединение включаются все строки из обеих таблиц. Для совпадающих строк поля заполняются реальными значениями, для несовпадающих строк поля заполняются в соответствии с правилами левого и правого соединений.

Примеры соединений

Пусть имеются таблицы

Отделения	
Код_отд	Город_отд
ОК3	Гурьев
ОК4	Барнаул
ОК2	Лондон

Объекты	
Код_об	Город_об
О14	Астана
О94	Лондон
О4	Гурьев

Обычное внутреннее соединение этих таблиц выполняется при помощи следующего оператора SQL

```
SELECT    Отделения.*, Объекты.*
FROM      Отделения, Объекты
WHERE     Отделения.Город_отд = Объекты.Город_об
```

Или (что эквивалентно)

```
SELECT    Отделения.*, Объекты.*
FROM      Отделения INNER JOIN Объекты ON Отделения.Город_отд =
Объекты.Город_об
```

Результат выполнения этого запроса имеет вид

Код_отд	Город_отд	Код_об	Город_об
ОК3	Гурьев	О4	Гурьев
ОК2	Лондон	О94	Лондон

Левое внешнее соединение таблиц Отделения и Объекты выглядит так:

```
SELECT    Отделения.*, Объекты.*
FROM      Отделения LEFT JOIN Объекты ON Отделения.Город_отд =
Объекты.Город_об
```

Результат выполнения этого запроса имеет вид

Код_отд	Город_отд	Код_об	Город_об
ОК3	Гурьев	О4	Гурьев
ОК4	Барнаул	NULL	NULL
ОК2	Лондон	О94	Лондон

В результате применения левого внешнего соединения в результирующую таблицу попали не только две строки, в которых имеется соответствие между названиями

городов, но также и та строка первой из соединяемых таблиц (левой), которая не нашла себе соответствия во второй таблице (правой). В этой строке все поля второй таблицы заполнены значениями NULL.

Правое внешнее соединение таблиц Отделения и Объекты выглядит так:

```
SELECT    Отделения.*, Объекты.*
FROM      Отделения RIGHT  JOIN Объекты ON Отделения.Город_отд =
Объекты.Город_об
```

Результат выполнения этого запроса имеет вид

Код_отд	Город_отд	Код_об	Город_об
NULL	NULL	О14	Астана
ОК3	Гурьев	О4	Гурьев
ОК2	Лондон	О94	Лондон

В результате применения правого внешнего соединения в результирующую таблицу попали не только две строки, в которых имеется соответствие между названиями городов, но также и та строка второй из соединяемых таблиц (правой), которая не нашла себе соответствия в первой таблице (левой). В этой строке все поля первой таблицы заполнены значениями NULL.

Полное внешнее соединение таблиц Отделения и Объекты выглядит так:

```
SELECT    Отделения.*, Объекты.*
FROM      Отделения FULL  JOIN Объекты ON Отделения.Город_отд =
Объекты.Город_об
```

Результат выполнения этого запроса имеет вид

Код_отд	Город_отд	Код_об	Город_об
NULL	NULL	О14	Астана
ОК3	Гурьев	О4	Гурьев
ОК4	Барнаул	NULL	NULL
ОК2	Лондон	О94	Лондон

В результате применения полного внешнего соединения в результирующую таблицу попали не только две строки, в которых имеется соответствие между названиями городов, но и все строки исходных таблиц, не нашедшие себе соответствия. В этих строках все столбцы той таблицы, в которой не было найдено соответствия, заполняются значениями NULL.

Вложенные подзапросы

Вложенный подзапрос - это подзапрос, заключенный в круглые скобки и вложенный в WHERE (HAVING) фразу предложения SELECT или других предложений, использующих WHERE фразу. Вложенный подзапрос может содержать в своей WHERE (HAVING) фразе другой вложенный подзапрос и т.д. Нетрудно догадаться, что вложенный подзапрос создан для того, чтобы при отборе строк таблицы, сформированной основным запросом, можно было использовать данные из других таблиц.

Существуют простые и коррелированные вложенные подзапросы. Они включаются в WHERE (HAVING) фразу с помощью условий IN, EXISTS или одного из условий сравнения (= | <> | < | <= | > | >=).

Простые вложенные подзапросы обрабатываются системой "снизу вверх". Первым обрабатывается вложенный подзапрос самого нижнего уровня. Множество значений, полученное в результате его выполнения, используется при реализации подзапроса более высокого уровня и т.д.

Запросы с коррелированными вложенными подзапросами обрабатываются системой в обратном порядке. Сначала выбирается первая строка рабочей таблицы, сформированной основным запросом, и из нее выбираются значения тех столбцов, которые используются во вложенном подзапросе (вложенных подзапросах). Если эти значения удовлетворяют условиям вложенного подзапроса, то выбранная строка включается в результат. Затем выбирается вторая строка и т.д., пока в результат не будут включены все строки, удовлетворяющие вложенному подзапросу (последовательности вложенных подзапросов).

Простые вложенные подзапросы

Простые вложенные подзапросы используются для представления множества значений, исследование которых должно осуществляться в каком-либо предикате IN, что иллюстрируется в следующем примере: выдать название поставщиков и банков, в которых они обслуживаются, поставивших товары по цене 20 тенге.

```
SELECT      Наименование, Банк
FROM   Поставщики
WHERE       Код поставщика IN
                    (SELECT Код поставщика
                FROM Движение товара
                 WHERE      Цена = 20 );
```

Результат выборки

Наименование	Банк
Поставщик_5	Народный
Поставщик_4	Альянс
Поставщик_5	Народный

При обработке полного запроса система выполняет прежде всего вложенный подзапрос. Этот подзапрос выдает множество номеров поставщиков, которые поставляют товары по цене 20 тенге, а именно множество (5,4,5). Поэтому первоначальный запрос эквивалентен такому простому запросу:

```
SELECT      Название, банк
FROM        Поставщики
WHERE       Код поставщика IN (4,5);
```

Использование одной и той же таблицы во внешнем и вложенном подзапросе

Выдать номера поставщиков, которые поставляют хотя бы один товар по той же цене, что и поставщик с кодом = 5.

```
SELECT      DISTINCT Код поставщика
FROM        Движение товара
WHERE       Код поставщика IN
                    (SELECT Код поставщика
                FROM Движение товара
                 WHERE      Цена = 20 );
```

Результат выборки:

Код поставщика
5
4

Вложенный подзапрос с оператором сравнения, отличным от IN

Выдать номера поставщиков, обслуживающихся в том же банке, что и поставщик с кодом 5.

```
SELECT      Наименование, Банк
FROM        Поставщики
WHERE       Банк =
            (SELECT Банк
          FROM Поставщики
           WHERE       Код поставщика = 5 );
```

Результат выборки:

Код поставщика
1
3

В подобных запросах можно использовать и другие операторы сравнения (<>, <=, <, >= или >), однако, если вложенный подзапрос возвращает более одного значения и не используется оператор IN, будет возникать ошибка.

Коррелированные вложенные подзапросы

Выдать название поставщиков и банков, в которых они обслуживаются, поставивших товары по цене 20 тенге.

```
SELECT      Наименование, Банк
FROM  Поставщики
WHERE       20 IN
             (SELECT Код поставщика
           FROM     Движение товара
           WHERE Код поставщика = Движение товара.Код поставщика
                        AND   Движение товара.Цена = 20  );
```

Такой подзапрос отличается от рассмотренного тем, что вложенный подзапрос не может быть обработан прежде, чем будет обрабатываться внешний подзапрос. Это связано с тем, что вложенный подзапрос зависит от значения Движение товара.Код поставщика, а оно изменяется по мере того, как система проверяет различные строки таблицы Поставщики. Обработка коррелированного подзапроса должна повторяться для каждого значения извлекаемого из внешнего подзапроса, а не выполняться раз и навсегда.

Запросы, использующие EXISTS

Квантор EXISTS (существует) - понятие, заимствованное из формальной логики. В языке SQL предикат с квантором существования представляется выражением EXISTS (SELECT * FROM ...).

Такое выражение считается истинным только тогда, когда результат вычисления "SELECT * FROM ..." является непустым множеством, т.е. когда существует какая-либо запись в таблице, указанной во фразе FROM подзапроса, которая удовлетворяет условию WHERE подзапроса.

Рассмотрим примеры. Выдать названия поставщиков, поставляющих продукт с номером 11.

```
 SELECT     Название                    Результат:
FROM        Поставщики                  Название
                                        СЫТНЫЙ
```

```
WHERE   EXISTS                                  УРОЖАЙ
   (    SELECT *                                КОРЮШКА
        FROM   Поставки
        WHERE  ПС = Поставщики.ПС               ЛЕТО
        AND    ПР = 11 );
```

Система последовательно выбирает строки таблицы Поставщики, выделяет из них значения столбцов Название и ПС, а затем проверяет, является ли истинным условие существования, т.е. существует ли в таблице Поставки хотя бы одна строка со значением ПР=11 и значением ПС, равным значению ПС, выбранному из таблицы Поставщики. Если условие выполняется, то полученное значение столбца Название включается в результат.

EXISTS представляет собой одну из наиболее важных возможностей SQL. Фактически любой запрос, который выражается через IN, может быть альтернативным образом сформулирован также с помощью EXISTS. Однако обратное высказывание несправедливо.

Выдать название и статус поставщиков, не поставляющих продукт с номером 11.

```
SELECT   Название, Статус                  Название    Статус
FROM     Поставщики                        ПОРТОС      кооператив
WHERE    NOT EXISTS                        ШУШАРЫ      совхоз
   (     SELECT *                          ТУЛЬСКИЙ    универсам
         FROM   Поставки
         WHERE  ПС = Поставщики.ПС         ОГУРЕЧИК    ферма
         AND    ПР = 11 );
```

Задание к СРСП 7:

1. Написать скрипты на создание запросов 1-5 к БД согласно индивидуального варианта задания