Лабораторная работа  
«Вложенные запросы»

## Цель работы:

Изучить операторы MS SQL Server, необходимые для работы с подзапросами. Научиться создавать вложенные запросы. Работа производится с учебной базой данной AdwentureWorks2019.

Язык T-SQL поддерживает написание вложенных запросов. Запрос, который возвращает конечный результат, называют **внешним**. Внутри себя он может использовать результат выполнения других запросов. **Внутренние** запросы вызываются во время выполнения кода и являются аналогами выражений, основанных на переменных или константах. В отличие от выражений их результат может изменяться в зависимости от содержимого таблицы. Использование вложенных запросов избавляет от необходимости хранить промежуточные результаты в отдельных переменных (Бен-Ган, 2015).

## Автономные вложенные запросы

Вложенные запросы всегда вызываются извне. Автономный внутренний запрос не зависит от внешнего, его можно выполнять и отлаживать независимо.

### Скалярные вложенные запросы

Скалярный вложенный запрос всегда возвращает единственное значение.

Пример

Показать продукты с минимальной ценой.

Решение

Решение разобьем на две части. Сначала рассчитаем минимальную цену для всех продуктов и результат запишем в переменную. Затем покажем все продуты с ценой равной значению нашей переменной.

DECLARE @I MONEY;

SET @I = (SELECT MIN(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0);

PRINT @I;

SELECT [Name], ListPrice  
FROM Production.Product  
WHERE ListPrice = @I;

В результате выполнения кода мы получили результат в переменной @I (Рисунок 1) и нашли продукт с рассчитанной минимальной ценой (Рисунок 2).

A picture containing text

Description automatically generated

Минимальное значение цены (Значение переменной @I)

Graphical user interface

Description automatically generated

Результат выполнения запроса для Примера 1

Решение

Решение с подзапросом создается следующим образом: запрос из прошлого решения будет внешним, а расчет минимального значения – внутренним подзапросом, взятым в скобки.

SELECT [Name], ListPrice  
FROM Production.Product  
WHERE ListPrice =   
 (  
 SELECT MIN(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0  
 );

В результате мы получим такой же результат, как и на Рисунке 2.

Пример

Показать фамилию, имена и возраст самого молодого сотрудника.

Фамилия, имя и отчество сотрудников находятся в таблице Person.Person, сведения о дате рождения – в таблице HumanResources.Employee. Решение данной задачи также разделим на две части. Внутренним подзапросом найдем максимальную дату из таблицы HumanResources.Employee. Внешним подзапросом найдем идентификатор сотрудника с максимальной датой рождения. Внутренним соединением с таблицей Person.Person найдем требуемые имена сотрудника и вычислим его возраст.

SELECT  
 p.FirstName  
 ,p.MiddleName  
 ,p.LastName  
 ,DATEDIFF(YEAR,e.BirthDate,GETDATE()) AS Age  
FROM Person.Person AS p  
INNER JOIN HumanResources.Employee AS e  
ON p.BusinessEntityID = e.BusinessEntityID  
WHERE e.BirthDate =  
 (  
 SELECT MAX(BirthDate)  
 FROM HumanResources.Employee  
 );



Результат выполнения запроса для Примера 2

### Вложенные запросы с предикатами

В предыдущих внутренних запросах результатом являлась скалярная величина. Существуют вложенные запросы, в которых в результате выполнения внутреннего запроса возвращается значение в виде столбца.

#### Предикат IN

Если в результате выполнения подзапроса возвращается множество значений (один столбец), то сравнивать его на «равно» нельзя. Для того, чтобы отфильтровать данные во внешнем запросе на «равенство» или «неравенство» полученным значениям во внутреннем запросе, применяется оператор IN (NOT IN). Работа с оператором в подзапросах идентична его работе со множеством имеющихся значений (Васильева, и др., 2020).

Пример

Показать товары, которые продавали максимально часто (первые три места по продажам).

Внутренним подзапросом найдем идентификаторы товаров, продаваемых максимально часто.

SELECT TOP 3 ProductID  
FROM Sales.SalesOrderDetail  
GROUP BY ProductID  
ORDER BY SUM(OrderQty)

В результате выполнения данного запроса получим следующий результат (Рисунок 4).

Изображение выглядит как стол

Автоматически созданное описание

Список идентификаторов товаров, продаваемых максимально часто

Отлаженный запрос поместим в предложение WHERE с предикатом IN для фильтрации товаров.

SELECT [Name]  
 ,Color  
 ,Style  
 ,ListPrice  
FROM Production.Product  
WHERE ProductID IN (  
 SELECT TOP 3 ProductID  
 FROM Sales.SalesOrderDetail  
 GROUP BY ProductID  
 ORDER BY SUM(OrderQty));

В результате выполнения запроса получим следующий результат ().

Изображение выглядит как стол

Автоматически созданное описание

Пример

Показать продавцов, получивших премию более $5000.

SELECT FirstName,  
 MiddleName,  
 LastName  
FROM Person.Person  
WHERE BusinessEntityID IN  
 (SELECT BusinessEntityID  
 FROM Sales.SalesPerson  
 WHERE Bonus > 5000);

Внутренний подзапрос вернет из таблицы Sales.SalesPerson идентификаторы продавцов, получивших премию более $5000, а внешний подзапрос из таблицы Person.Person покажет фамилию и имена, соответствующие найденным идентификаторам.

Такой же результат () можно получить с помощью применения оператора внутреннего соединенияINNER JON:

SELECT FirstName,  
 MiddleName,  
 LastName  
FROM Person.Person AS p  
INNER JOIN Sales.SalesPerson AS s  
ON p.BusinessEntityID = s.BusinessEntityID  
WHERE Bonus > 5000;

Изображение выглядит как стол

Автоматически созданное описание

#### Предикат ANY (SOME)

Предикат ANY (SOME) используется, когда необходимо, чтобы результат фильтрации внешнего запроса был меньше (или меньше либо равен)[[1]](#footnote-1) любого значения из множества, возвращаемого внутренним запросом, то есть меньше (меньше или равно) минимального значения из возвращаемого подзапроса.

Результат выполнения запроса

SELECT Field1  
 ,Field2  
 ,...  
FROM Table  
WHERE Field1 <= ANY (1, 2, 3)

эквивалентен выполнению запроса

SELECT Field1  
 ,Field2  
 ,...  
FROM Table  
WHERE Field1 <= 1

Пример 5

Показать товары, цена которых больше минимальной средней цены в каждом цвете.

Определим сначала список средних цен для каждого цвета

SELECT AVG(ListPrice)  
FROM Production.Product  
WHERE ListPrice > 0  
AND Color IS NOT NULL  
GROUP BY Color;

Результат представлен на ().

Изображение выглядит как стол

Автоматически созданное описание

Как видно, минимальная средняя цена равна $9,245.

Теперь запишем общий запрос:

SELECT [Name]  
 ,ListPrice  
FROM Production.Product  
WHERE ListPrice < ANY  
 (SELECT AVG(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color)  
ORDER BY ListPrice;

В результате выполнения данного запроса, получим:

Изображение выглядит как стол

Автоматически созданное описание

Пример 6

Показать товары, цена которых меньше максимальной средней цены в каждом цвете.

SELECT [Name]  
 ,ListPrice  
FROM Production.Product  
WHERE ListPrice < ANY  
 (SELECT AVG(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color)  
ORDER BY ListPrice DESC;

Результат выполнения запроса показан на

Изображение выглядит как стол

Автоматически созданное описание

Из предыдущего примера (Пример 5) видно, что максимальная средняя цена равна $1401,95. Из видно, что все цены в результирующем наборе меньше этого значения.

#### Предикат ALL

Предикат ALL используется, когда в результирующем наборе необходимо провести фильтрацию больше (больше или равно)[[2]](#footnote-2) всех значений из подзапроса, то есть найти значения из внешнего запроса, которые больше или равны максимальному значению из внутреннего запроса.

Результат выполнения данного запроса

SELECT Field1  
 ,Field2  
 ,...  
FROM Table  
WHERE Field1 >= ALL (1, 2, 3)

эквивалентен выполнению следующего запроса:

SELECT Field1

,Field2

,...

FROM Table

WHERE Field1 >= 3

Пример

Показать товары, цена которых больше любой средней цены в каждом цвете.

SELECT [Name]  
 ,ListPrice  
FROM Production.Product  
WHERE ListPrice > ALL  
 (SELECT AVG(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color)  
ORDER BY ListPrice;

Итоговый запрос вернет список товаров, цена которых больше или равна максимальной величине из внутреннего запроса, то есть не меньше $1401,95.

Изображение выглядит как стол

Автоматически созданное описание

Как видно из, все товары имеют цены выше, чем максимальная цена из подзапроса ($1401,95).

Пример

Показать товары, цена которых меньше любой средней цены в каждом цвете.

SELECT [Name]  
 ,ListPrice  
FROM Production.Product  
WHERE ListPrice > 0  
 AND ListPrice > ALL  
 (SELECT AVG(ListPrice)  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color)  
ORDER BY ListPrice DESC;

Изображение выглядит как стол

Автоматически созданное описание

Как видно, все цены ниже минимальной средней цены ($9,245).

#### Предикат EXISTS

## Использование подзапросов в предложении FROM

Как известно, использовать агрегатную функцию от агрегатной функции использовать нельзя, но можно написать запрос с подзапросом следующего вида:

SELECT f1(AliasField)  
FROM  
 (  
 SELECT f2(Field1) AS AliasField  
 FROM Table  
 WHERE (Conditions1)– необязательная инструкция  
 GROUP BY Field2– необязательная инструкция  
 HAVING (Conditions2)– необязательная инструкция  
) AS AliasTable

В данном примере f1 и f2 ‑ агрегатные функции.[[3]](#footnote-3)

Запишем запрос с использованием конструкции такого вида для одного из прошлых примеров.

Пример

Показать товары, цена которых меньше любой средней цены в каждом цвете. Решение с помощью двойного вложенного подзапроса.

Напишем сначала запрос, который возвращает максимальное значение средней цены для каждого цвета.

SELECT MAX(AvgListPrice)  
 FROM   
 (  
 SELECT AVG(ListPrice) AS AvgListPrice  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color  
 ) AS T;

В результате выполнения данного запроса, получим следующий результат():

Изображение выглядит как стол

Автоматически созданное описание

Значение максимальной средней цены, полученное с помощью применения подзапроса, совпадает с рассчитанным ранее значением ().

Применим отлаженный запрос с подзапросом в качестве вложенного внутреннего запроса для внешнего запроса, который будет фильтровать товары по цене – отбирать товары с меньшей ценой:

SELECT [Name], ListPrice  
FROM Production.Product  
WHERE ListPrice <  
 (  
 SELECT MAX(AvgListPrice)  
 FROM   
 (  
 SELECT AVG(ListPrice) AS AvgListPrice  
 FROM Production.Product  
 WHERE ListPrice > 0  
 AND Color IS NOT NULL  
 GROUP BY Color  
 ) AS T  
 )  
ORDER BY ListPrice DESC;

Результат выполнения данного запроса совпадает с результатом (), полученным для Примера 8. Аналогичным образом можно записать запросы для Примера 5 ‑ Примера 7 и получить такие же результаты.

## Коррелирующие вложенные запросы

Если результат выполнения внутреннего запроса зависит от внешнего, то такой запрос называется зависимым или коррелирующим. Отладить отдельно внутренний и внешний запросы нельзя. Записываются такие запросы обычно с применением самосоединения таблицы (SELF JOIN).

Пример

Для каждого стиля показать товары с максимальной ценой (стиль определен).

Несмотря на то, что выполнить и отладить внутренний запрос для данного примера отдельно нельзя, запишем его в приближенном к окончательному, виде: найдем максимальную цену товара для каждого стиля.

SELECT MAX(p2.ListPrice)  
FROM Production.Product AS p2  
WHERE p2.Style IS NOT NULL  
GROUP BY p2.Style;

В результате получим следующий набор данных:

Изображение выглядит как стол

Автоматически созданное описание

По условию задачи необходимо показать товар в каждом стиле, цена которого равна полученной максимальной цене.

Напишем внешний запрос для первого стиля – M, зная, что максимальная цена в этом стиле равна $89,99.

SELECT p1.[Name]  
 ,p1.ListPrice  
FROM Production.Product AS p1  
WHERE p1.Style = 'M'  
 AND p1.ListPrice = 89.99

В результате выполнения данного запроса получим:

Изображение выглядит как текст, стол

Автоматически созданное описание

Итоговый запрос, показывающий товары для всех стилей, будет сформирован из двух отдельных запросов с некоторой модификацией:

SELECT p1.[Name]  
 ,p1.ListPrice  
 ,p1.Style – добавим столбец стиль  
FROM Production.Product AS p1  
WHERE  
 -- p1.Style = 'M' AND удалим фильтрацию для стиля  
 p1.Style IS NOT NULL AND – и добавляем фильтрацию для стиля на то, что он заполнен  
 p1.ListPrice = (  
 SELECT MAX(p2.ListPrice)  
 FROM Production.Product AS p2  
 WHERE p1.Style = p2.Style –добавим условие  
 -- p2.Style IS NOT NULL удалим условие  
 --GROUP BY p2.Style – удалим группировку  
 )  
ORDER BY p1.Style; --добавлено для удобства анализа

Итоговый запрос был дополнен строкой в предложении SELECT внешнего запроса, для того чтобы показать стиль, к которому относятся выбранные товары. В предложении WHERE внешнего запроса было убрано ограничение на конкретный вид стиля. Во внутреннем подзапросе были удалены строки на фильтрацию стиля, если тот не определен, потому что добавлено обязательное условие на равенство стиля внутреннего и внешнего запросов, а во внешнем запросе это условие было добавлено. Также во внутреннем запросе удалена строка на группировку по стилю, т. к. максимальная цена рассчитывается для одного конкретного стиля, пришедшего из внешнего запроса.

Запрос будет выполняться следующим образом. Сначала для внешнего запроса фильтрации будут подвержены строки, в которых стиль не определен. Затем для каждой строки из внешнего запроса будет выполнен внутренний запрос. В предложении выбора во внешнем запросе будут показаны товары, с ценой, равной цене, рассчитанной во внутреннем подзапросе.

В результате получим набор, в котором для каждого стиля показаны товары с максимальной ценой.

Изображение выглядит как стол

Автоматически созданное описание

Пример

Показать товары, в Линейке модели которых существует более 4 размеров

Определим сначала Линейки продуктов, в которых имеется более 4 размеров.

SELECT COUNT(DISTINCT p2.Size) AS SizeCount  
 ,p2.ProductLine  
FROM Production.Product AS p2  
WHERE p2. ProductLineIS NOT NULL  
 AND p2.Size IS NOT NULL  
GROUP BY p2.ProductLine  
HAVING COUNT(DISTINCT p2.Size) > 4;

В результате получим следующий набор данных

Изображение выглядит как стол

Автоматически созданное описание

Запишем итоговый запрос, применяя измененный подзапрос.

SELECT p1.[Name]  
 ,p1.ProductLine  
 ,p1.Size  
FROM Production.Product AS p1  
WHERE p1.ProductLine IS NOT NULL  
AND p1.Size IS NOT NULL  
AND  
 ( SELECT COUNT(DISTINCT p2.Size) AS SizeCount  
 FROM Production.Product AS p2  
 WHERE p1.ProductLine = p2.ProductLine  
 AND p2.Size IS NOT NULL  
 ) > 4;

В результате выполнения данного запроса получим список товаров, для которого в представленных линиях продуктов, имеется более четырех размеров.

Изображение выглядит как стол

Автоматически созданное описание

#### Предикат EXISTS

### Возвращение предыдущих или следующих значений

### Использование текущих агрегатов

## Ограничение на подзапросы

1. Также данный предикат можно использовать для отбора больше (или равно), чем любое [↑](#footnote-ref-1)
2. Также данный предикат можно использовать для отбора меньше (или равно), чем все [↑](#footnote-ref-2)
3. Необязательно функции должны быть агрегатные. Конструкцию такого виды можно использовать для любого вида подзапросов. [↑](#footnote-ref-3)