Лабораторная работа 4

Цель работы: использование подзапросов.

При написании запросов может возникнуть необходимость в данных, полученных на предыдущем этапе. Встает вопрос об актуальности данных, рассмотрим следующий пример. Необходимо найти название самого дорогого товара с цветом Red. Пользователь не знакомый с механизмом использования подзапросов может написать следующий набор запросов.

```
SELECT MAX([ListPrice]) FROM [Production].[Product]

WHERE [Color]='Red’
```

Данный запрос вернет таблицу, одна строка и один столбец без имени, со значением 3578,27. Подставив полученный результат в следующий запрос, пользователь получит искомые данные.

```
SELECT [Name]

FROM [Production].[Product]

WHERE [Color]='Red' AND [ListPrice]=3578.27
```

При выполнении данных операций пользователь во-первых – вынужден выполнять промежуточные действия в ручную, во-вторых – актуальность полученных данных может быть поставлена под сомнение.

Использование подзапросов решит обе эти проблемы.

```
SELECT [Name]

FROM [Production].[Product]

WHERE [Color]='Red'AND [ListPrice]=

(SELECT MAX([ListPrice]) FROM [Production].[Product]

WHERE [Color]='Red')
```

В данном случае показан простейший вариант использования подзапроса. Стоит отметить, что использован оператор сравнения на равенство, так как функция MAX гарантированно возвращает одно значение.

Допускается использование логических операторов для сравнения скалярного значения с результатом выполнения подзапроса. Рассмотрим несколько примеров.

Необходимо получить список товаров цвет которых может быть любой, кроме Red, цена которых равна цене любого товара с цветом Red. Для этого можно использовать запрос следующего вида:

```
SELECT [Name]

FROM [Production].[Product]

WHERE [Color]!='Red'AND [ListPrice] = ANY

(SELECT [ListPrice] FROM [Production].[Product]

WHERE [Color]='Red')
```

В оператор сравнения использован вместе с логическим оператором ANY, так как подзапрос возвращает, потенциально, более одного значения. Логический оператор ANY сравнивает скалярное значение с набором значений, состоящим из одного столбца, и условие сравнение должно быть выполнено хотя бы для одного значения из набора.

Необходимо получить список товаров, цена которых больше цены любого из товаров с цветом Red.

```
SELECT [Name]

FROM [Production].[Product]

WHERE [ListPrice] >ALL

(SELECT [ListPrice] FROM [Production].[Product]

WHERE [Color]='Red')
```

Логический оператор ALL сравнивает скалярное значение с набором значений, состоящим из одного столбца, и условие сравнение должно быть выполнено для каждого значения из набора.

Необходимо получить название товаров, чей цвет совпадает с цветом одного из товаров, чья цена больше 3000.

```
SELECT [Name]

FROM [Production].[Product]

WHERE [Color] IN

(SELECT [Color] FROM [Production].[Product]

WHERE [ListPrice]>3000)
```

Логический оператор IN определяет, совпадает ли указанное значение с одним из значений, содержащихся во вложенном запросе или списке.

Подзапросы могут иметь сами содержать подзапросы. Следующий пример находит название категории, где содержится самый дорогой товар.

```
SELECT [Name]

FROM [Production].[ProductCategory]

WHERE [ProductCategoryID] in

(SELECT [ProductCategoryID]

FROM [Production].[ProductSubcategory]

WHERE [ProductSubcategoryID] in

(SELECT [ProductSubcategoryID]

FROM [Production].[Product]

WHERE [ListPrice] =

(SELECT MAX([ListPrice])

FROM [Production].[Product])))
```

Данную задачу можно было бы решить с помощью соединения таблиц, упорядочивания и выражения TOP, но использование подзапросов в данном случае предпочтительней, так как это позволяет избежать ресурсоемкой операции соединения. Это утверждение справедливо только для ситуаций, когда подзапрос простой.

Запрос может использовать более одного подзапроса на одном уровне вложенности. Например, необходимо получить с помощью одного запроса список товаров, у которых цвет совпадает с цветом самого дорогого товара, и стиль совпадает со стилем самого дорого товара.

```
SELECT [Name]
```

```
FROM [Production].[Product]
WHERE [Color] in
(SELECT [Color]
FROM [Production].[Product]
WHERE [ListPrice] =
(SELECT MAX([ListPrice])
FROM [Production].[Product]))
AND
[Style] in
(SELECT [Style]
FROM [Production].[Product]
WHERE [ListPrice] =
(SELECT MAX([ListPrice])
FROM [Production].[Product]))
```

Все рассмотренные ранее подзапросы являлись простыми, они возвращали фиксированный, неизменный набор данных, вне зависимости с какой из строк запроса верхнего уровня в данный момент работает СУБД.

Подзапросы так же используются для формирования выборки с использованием конструкции GROUP BY HAVING. Допустим необходимо найти номер подкатегории товаров с наибольшим количеством товаров. Данный запрос можно реализовать несколькими способами, в том числе с использованием подзапроса.

```
SELECT [ProductSubcategoryID]
from [Production].[Product]
GROUP BY [ProductSubcategoryID]
HAVING COUNT(*)=
(SELECT TOP 1 COUNT(*)
from [Production].[Product]
GROUP BY [ProductSubcategoryID]
ORDER BY 1 DESC)
```

Рассмотрим следующую задачу. Необходимо получить список самых дорогих товаров в каждой из подкатегорий. Подобную задачу можно решить с использование сложного, коррелирующего или связанного, подзапроса. Коррелирующим подзапросом называют такой подзапрос, который формирует выборку связанную, зависимую от данных внешнего запроса. Фактически коррелирующий подзапрос выполняется для каждой строки запроса верхнего уровня.

```
SELECT [Name]
FROM [Production].[Product] AS P1
WHERE [ListPrice]=
```

```
(SELECT MAX([ListPrice])
FROM [Production].[Product] AS P2
WHERE P1.ProductSubcategoryID=P2.ProductSubcategoryID)
```

Связанные подзапросы могут использоваться для формирования выводимого столбца. Например, следующий запрос возвращает название продукта и название подкатегории к которой он относится.

```
SELECT [Name],
(SELECT [Name]
FROM [Production].[ProductSubcategory] AS PS
WHERE P1.ProductSubcategoryID=PS.ProductSubcategoryID)
FROM [Production].[Product] AS P1
```

Примеры с решениями:

1 Найти название подкатегории с наибольшим количеством продуктов, без учета продуктов для которых подкатегория не определена (еще одна возможная реализация)

```
SELECT [Name]
FROM [Production].[ProductSubcategory]
WHERE [ProductSubcategoryID] IN
(SELECT [ProductSubcategoryID]
FROM [Production].[Product]
WHERE [ProductSubcategoryID] IS NOT NULL
GROUP BY [ProductSubcategoryID]
HAVING COUNT(*)=
(SELECT TOP 1 COUNT(*)
FROM [Production].[Product]
WHERE [ProductSubcategoryID] IS NOT NULL
GROUP BY [ProductSubcategoryID]
ORDER BY 1 DESC
)
)
```

2 Вывести на экран такого покупателя, который каждый раз покупал только одну номенклатуру товаров, не обязательно в одинаковых количествах, т.е. у него всегда был один и тот же «список покупок».

Вариант 1

```
select [CustomerID], count(*)
from [Sales].[SalesOrderHeader] as SOH1
group by [CustomerID]
```

```
having count(*)>1 and count(*)=all
(select count(*)
from [Sales].[SalesOrderHeader] as SOH inner join
[Sales].[SalesOrderDetail] as SOD
on soh.SalesOrderID=sod.SalesOrderID
group by soh.[CustomerID], sod.ProductID
having soh.CustomerID=soh1.CustomerID)
```

Вариант 2

```
select soh.CustomerID, p.Name
from [Sales].[SalesOrderHeader] as SOH inner join
[Sales].[SalesOrderDetail] as sod
on soh.SalesOrderID=sod.SalesOrderID inner join
[Production].[Product] as p
on sod.ProductID=p.ProductID
where soh.CustomerID in (
select [CustomerID]
from [Sales].[SalesOrderHeader] as SOH1
group by [CustomerID]
having count(*)>1 and count(*)=all
(select count(*)
from [Sales].[SalesOrderHeader] as SOH inner join
[Sales].[SalesOrderDetail] as SOD
on soh.SalesOrderID=sod.SalesOrderID
group by soh.[CustomerID], sod.ProductID
having soh.CustomerID=soh1.CustomerID))
order by 1
```

Вариант 3

```
select soh1.CustomerID
from [Sales].[SalesOrderDetail] as sod1 inner join
[Sales].[SalesOrderHeader] as soh1
on sod1.SalesOrderID=soh1.SalesOrderID
group by soh1.CustomerID
having count(soh1.SalesOrderID)>1 and
count(distinct sod1.ProductID)>1
```

```
 and count(distinct sod1.ProductID)=all
(
select count(*)
from [Sales].[SalesOrderDetail] as sod2 inner join
[Sales].[SalesOrderHeader] as soh2
on sod2.SalesOrderID=soh2.SalesOrderID
group by soh2.CustomerID, sod2.SalesOrderID
having soh2.CustomerID=soh1.CustomerID
)
```

3 Вывести на экран следующую информацию: название товара (первая колонка), количество покупателей, покупавших этот товар (вторая колонка), количество покупателей, совершавших покупки, но не покупавших товар из первой колонки (третья колонка)

```
select
p.[ProductID],
(select count(distinct soh.CustomerID)
from [Sales].[SalesOrderDetail] as sod inner join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where sod.ProductID=p.ProductID),
(select count(distinct soh.CustomerID)
from [Sales].[SalesOrderDetail] as sod inner join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where soh.CustomerID not in
(select distinct soh.CustomerID
from [Sales].[SalesOrderDetail] as sod inner join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID where
sod.ProductID=p.ProductID))
from [Production].[Product] as p
```

4 Найти такие товары, которые были куплены более чем одним покупателем, при этом все покупатели этих товаров покупали товары только из одной подкатегории

```
select name
from [Production].[Product]
where ProductID IN(
```

```
select sod.ProductID
from [Sales].[SalesOrderDetail] as sod join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where soh.CustomerID in(
select soh.CustomerID
from [Sales].[SalesOrderDetail] as sod join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID join
[Production].[Product] as p on
sod.ProductID=p.ProductID
group by soh.CustomerID
having count(distinct p.ProductSubcategoryID)=1)
group by sod.ProductID
having count(distinct soh.CustomerID)>1)
```

5 Найти покупателя, который каждый раз имел разный список товаров в чеке (по номенклатуре)

```
select distinct CustomerID
from [Sales].[SalesOrderHeader]
where CustomerID not in (
select soh.Customerid
from [Sales].[SalesOrderDetail] as sod join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where
exists(select ProductID
from [Sales].[SalesOrderDetail] as sod1 join
[Sales].[SalesOrderHeader] as soh1
on sod.SalesOrderID=soh.SalesOrderID
where soh1.CustomerID=soh.CustomerID and
sod1.ProductID=sod.ProductID and sod.SalesOrderID!=sod1.SalesOrderID
))
```

6 Найти такого покупателя, что все купленные им товары были куплены только им, и никогда не покупались другими покупателями.

Вариант 1

```
select soh1.CustomerID
from[Sales].[SalesOrderDetail] as sod1 inner join
[Sales].[SalesOrderHeader] as soh1
on sod1.SalesOrderID=soh1.SalesOrderID
group by soh1.CustomerID
having count(distinct sod1.productid)=
(select count(distinct sod.ProductID)
from [Sales].[SalesOrderDetail] as sod inner join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where soh.CustomerID=soh1.CustomerID
and sod.ProductID in
(select sod.ProductID
from[Sales].[SalesOrderDetail] as sod inner join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
group by sod.ProductID
having count(distinct soh.CustomerID)=1))
```

Вариант 2

```
select distinct soh.CustomerID
from [Sales].[SalesOrderHeader] as soh
where soh.CustomerID not in(
select distinct soh.CustomerID
from [Sales].[SalesOrderDetail] as sod join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
where ProductID not in(
select sod.ProductID
from [Sales].[SalesOrderDetail] as sod join
[Sales].[SalesOrderHeader] as soh
on sod.SalesOrderID=soh.SalesOrderID
group by sod.ProductID
having count(distinct soh.CustomerID)=1))
```

Задания для самостоятельной работы:

1 Найти название самого продаваемого продукта

2 Найти покупателя, совершившего покупку на самую большую сумм, считая сумму покупки исходя из цены товара без скидки (UnitPrice)

3 Найти такие продукты, которые покупал только один покупатель

4 Вывести список продуктов, цена которых выше средней цены товаров в подкатегории, к которой относится товар

5 Найти такие товары, которые были куплены более чем одним покупателем, при этом все покупатели этих товаров покупали товары только одного цвета и товары не входят в список покупок покупателей, купивших товары только двух цветов

6 Найти такие товары, которые были куплены такими покупателями, у которых они присутствовали в каждой их покупке

7 Найти покупателей, у которых есть товар, присутствующий в каждой покупке/чеке