

БАЗЫ ДАННЫХ

Язык запросов SQL. Использование подзапросов

Понятие подзапроса

Подзапрос используется когда для выполнения одного запроса необходимо значение (значения), вычисляемое с помощью другого запроса. Текст подзапроса должен быть заключен в скобки.

Например.

При использовании условия поиска в предложении WHERE значение, с которым надо сравнивать, заранее не определено и должно быть вычислено в момент выполнения оператора SELECT.

Внутренний подзапрос представляет собой также оператор SELECT, а кодирование его предложений подчиняется тем же правилам, что и основного оператора SELECT.

Внешний оператор SELECT использует результат выполнения внутреннего оператора для определения содержания окончательного результата всей операции.

Внутренние запросы могут быть помещены в предложения WHERE и HAVING внешнего оператора SELECT.

Внутренние операторы SELECT могут применяться в операторах INSERT, UPDATE и DELETE.

Правила и ограничения подзапросов

- фраза ORDER BY не используется, хотя и может присутствовать во внешнем подзапросе;
- список в предложении SELECT состоит из имен отдельных столбцов или составленных из них выражений – за исключением случая, когда в подзапросе присутствует ключевое слово EXISTS;
- по умолчанию имена столбцов в подзапросе относятся к таблице, имя которой указано в предложении FROM. Однако допускается ссылка и на столбцы таблицы, указанной во фразе FROM внешнего запроса, для чего применяются квалифицированные имена столбцов (т.е. с указанием таблицы);
- если подзапрос является одним из двух операндов, участвующих в операции сравнения, то запрос должен указываться в правой части этой операции.

Типы подзапросов

Скалярный подзапрос возвращает единственное значение. В принципе, он может использоваться везде, где требуется указать единственное значение.

Табличный подзапрос возвращает множество значений, т.е. значения одного или нескольких столбцов таблицы, размещенные в более чем одной строке. Он возможен везде, где допускается наличие таблицы.

Подзапросы, возвращающие единичное значение



Пример 1. Определить дату продажи максимальной партии товара.

```
SELECT Дата, Количество FROM Сделка
```

```
WHERE Количество = (SELECT Max(Количество) FROM Сделка)
```

Во вложенном подзапросе определяется максимальное количество товара.

Во внешнем подзапросе – дата, для которой количество товара оказалось равным максимальному.

Необходимо отметить, что нельзя прямо использовать предложение `WHERE Количество = Max(Количество)`, поскольку применять обобщающие функции в предложениях `WHERE` запрещено.

Для достижения желаемого результата следует создать подзапрос, вычисляющий максимальное значение количества, а затем использовать его во внешнем операторе `SELECT`, предназначенном для выборки дат сделок, где количество товара совпало с максимальным значением.

Пример. Возвращение единичного значения



Пример 2. Определить даты сделок, превысивших по количеству товара среднее значение и указать для этих сделок превышение над средним уровнем.

```
SELECT Дата, Количество,  
Количество - (SELECT Avg(Количество) FROM Сделка) AS  
Превышение  
FROM Сделка  
WHERE Количество > (SELECT Avg(Количество) FROM Сделка)
```

В приведенном примере результат подзапроса, представляющий собой среднее значение количества товара по всем сделкам вообще, используется во внешнем операторе SELECT как для вычисления отклонения количества от среднего уровня, так и для отбора сведений о датах.

Пример.
Возвращение
единичного значения



Пример 3. Определить клиентов, совершивших сделки с максимальным количеством товара.

```
SELECT Клиент.Фамилия FROM Клиент INNER JOIN Сделка ON
Клиент.КодКлиента=Сделка.КодКлиента WHERE
Сделка.Количество= (SELECT Max(Сделка.Количество)
FROM Сделка)
```

Здесь показан пример использования подзапроса при выборке данных из разных таблиц.

Пример 4. Определить клиентов, в сделках которых количество товара отличается от максимального не более чем на 10%.

```
SELECT Клиент.Фамилия, Сделка.Количество FROM Клиент
INNER JOIN Сделка ON Клиент.КодКлиента=
Сделка.КодКлиента WHERE Сделка.Количество >= 0.9 * (SELECT
Max(Сделка.Количество) FROM Сделка)
```

Подзапросы в предложении HAVING



Пример 5. Определить даты, когда среднее количество проданного за день товара оказалось больше 20 единиц.

```
SELECT Сделка.Дата, Avg(Сделка.Количество) AS Среднее_за_день
FROM Сделка
GROUP BY Сделка.Дата
HAVING Avg(Сделка.Количество) >20
```

За каждый день определяется среднее количество товара, которое сравнивается с числом 20.

Добавим в запрос подзапрос.

Пример 6. Определить даты, когда среднее количество проданного за день товара оказалось больше среднего показателя по всем сделкам вообще.

```
SELECT Сделка.Дата, Avg(Сделка.Количество) AS Среднее_за_день
FROM Сделка
GROUP BY Сделка.Дата
HAVING Avg(Сделка.Количество) >
(SELECT Avg(Сделка.Количество) FROM Сделка)
```

Внутренний подзапрос определяет средний по всем сделкам показатель, с которым во внешнем запросе сравнивается среднее за каждый день количество товара.

Подзапросы, возвращающие множество значений

Во многих случаях значение, подлежащее сравнению в предложениях WHERE или HAVING, представляет собой не одно, а несколько значений.

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, в свою очередь, 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);

Использование операций IN и NOT IN

Оператор IN используется для сравнения некоторого значения со списком значений, при этом проверяется, входит ли значение в предоставленный список или сравниваемое значение не является элементом представленного списка.

Пример.
Возвращение
множества значений



Пример 7. Определить список товаров, которые имеются на складе.

```
SELECT Название FROM Товар
WHERE КодТовара IN (SELECT КодТовара FROM Склад)
```

Пример 8. Определить список отсутствующих на складе товаров.

```
SELECT Название FROM Товар
WHERE КодТовара NOT IN (SELECT КодТовара FROM Склад)
```

Пример 9. Определить товары, которые покупают клиенты из Одессы.

```
SELECT DISTINCT Товар.Название, Клиент.ГородКлиента
FROM Товар INNER JOIN
(Клиент INNER JOIN Сделка ON Клиент.КодКлиента=Сделка.КодКлиента)
ON Товар.КодТовара=Сделка.КодТовара
WHERE Клиент.ГородКлиента = 'Одесса'
```

В результат включаются товары, приобретенные клиентами из Одессы, однако не исключено, что покупателями таких товаров были и клиенты из других городов.

Пример.
Возвращение
множества значений



Введение в запрос фразы "только" требует использования операции NOT IN.

Пример 10. Определить товары, покупку которых осуществляют только клиенты из Одессы, и никто другой.

```
SELECT DISTINCT Товар.Название, Клиент.ГородКлиента
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)
ON Товар.КодТовара = Сделка.КодТовара
WHERE Товар.Название NOT IN
(SELECT Товар.Название FROM Товар INNER JOIN
(Клиент INNER JOIN Сделка ON Клиент.КодКлиента =
Сделка.КодКлиента) ON Товар.КодТовара = Сделка.КодТовара
WHERE Клиент.ГородКлиента <> 'Одесса')
```

Пример.
Возвращение
множества значений



Пример 11. Какие товары ни разу не купили клиенты из Одессы?

```
SELECT DISTINCT Товар.Название, Клиент.ГородКлиента
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)
ON Товар.КодТовара=Сделка.КодТовара
WHERE Товар.Название NOT IN
(SELECT Товар.Название
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента=Сделка.КодКлиента)
ON Товар.КодТовара=Сделка.КодТовара
WHERE Клиент.ГородКлиента='Одесса')
```

Во вложенном запросе определяется список товаров, приобретаемых клиентами из Одессы. Во внешнем запросе выбираются только те товары, которые не входят в этот список.

Пример.
Возвращение
множества значений



Пример 12. Определить фирмы, покупающие товары местного производства.

```
SELECT DISTINCT Клиент.Фирма, Клиент.ГородКлиента,  
Товар.ГородТовара  
FROM Товар INNER JOIN (Клиент  
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)  
ON Товар.КодТовара = Сделка.КодТовара  
WHERE Клиент.ГородКлиента = Товар.ГородТовара
```

В результате выполнения запроса перечисляются сделки, когда клиенту был продан товар, изготовленный в его городе, что совсем не исключает наличие сделок этих же клиентов, связанных с приобретением товара из другого города.

Пример. Возвращение множества значений



Введем в запрос фразу "только" – сразу потребуются привлечение операции NOT IN.

Пример 13. Определить фирмы, которые покупают только товары, произведенные в своем городе, и никакие другие.

```
SELECT DISTINCT Клиент.Фирма, Клиент.ГородКлиента, Товар.ГородТовара
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)
ON Товар.КодТовара = Сделка.КодТовара
WHERE Клиент.ГородКлиента NOT IN
(SELECT DISTINCT Клиент.ГородКлиента
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)
ON Товар.КодТовара = Сделка.КодТовара
WHERE Клиент.ГородКлиента <> Товар.ГородТовара)
```

Во вложенном запросе определяется множество фирм, совершивших хотя бы одну покупку товара из чужого города. Затем определяются фирмы, не входящие в это множество.

Ключевые слова 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 запись 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. ANY и ALL



Пример 14. Определить клиентов, совершивших сделки с максимальным количеством товара (эквивалентно запросу 3.)

```
SELECT Клиент.Фамилия, Сделка.Количество  
FROM Клиент INNER JOIN Сделка ON  
Клиент.КодКлиента=Сделка.КодКлиента  
WHERE Сделка.Количество >= ALL(SELECT Количество FROM Сделка)
```

В примере определены клиенты, в сделках которых количество товара больше или равно количеству товара в каждой из всех сделок.

Пример 15. Найти фирму, купившую товаров на сумму, превышающую 1000\$.

```
SELECT Клиент.Фирма, Sum(Товар.Цена*Сделка.Количество) AS  
Общ_стоимость  
FROM Товар INNER JOIN (Клиент  
INNER JOIN Сделка ON Клиент.КодКлиента = Сделка.КодКлиента)  
ON Товар.КодТовара = Сделка.КодТовара  
GROUP BY Клиент.Фирма  
HAVING Sum(Товар.Цена*Сделка.Количество)>1000
```

Пример. ANY и ALL



Пример 16. Найти фирму, которая приобрела товаров на самую большую сумму.

```
SELECT Клиент.Фирма, Sum(Товар.Цена*Сделка.Количество) AS
Общ_стоимость
FROM Товар INNER JOIN (Клиент
INNER JOIN Сделка ON Клиент.КодКлиента=Сделка.КодКлиента)
ON Товар.КодТовара=Сделка.КодТовара
GROUP BY Клиент.Фирма
HAVING Sum(Товар.Цена*Сделка.Количество)>=ALL
(SELECT Sum(Товар.Цена*Сделка.Количество)
FROM Товар INNER JOIN Сделка ON
Товар.КодТовара=Сделка.КодТовара
GROUP BY Сделка.КодКлиента)
```

Вложенный подзапрос подсчитывает общую стоимость покупок каждого клиента. Внешний подзапрос также подсчитывает общую стоимость покупок каждого клиента и определяет тех, для кого эта сумма, по сравнению с другими покупателями, оказалась больше или точно такой же.

Пример. ANY и ALL



Пример 17. Найти фирмы, в сделках которых количество товара превышает такой же показатель хотя бы в одной сделке клиентов из Николаева.

```
SELECT Клиент.Фирма, Сделка.Количество
FROM Клиент INNER JOIN Сделка ON
Клиент.КодКлиента=Сделка.КодКлиента
WHERE Сделка.Количество > ANY
(SELECT Сделка.Количество
FROM Клиент INNER JOIN Сделка
ON Клиент.КодКлиента=Сделка.КодКлиента
WHERE Клиент.ГородКлиента='Николаев')
```

Использование операций **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. EXISTS и NOT EXISTS



Пример 18. Определить список имеющихся на складе товаров (запрос эквивалентен примеру 7).

```
SELECT Название FROM Товар
WHERE EXISTS (SELECT КодТовара FROM Склад
              WHERE Товар.КодТовара = Склад.КодТовара)
```

Пример 19. Определить список отсутствующих на складе товаров (запрос эквивалентен примеру 8).

```
SELECT Название FROM Товар
WHERE NOT EXISTS (SELECT КодТовара FROM Склад
                  WHERE Товар.КодТовара = Склад.КодТовара)
```