Использование подзапросов

Подзапрос — это запрос, содержащийся в выражении ключевого слова WHERE другого запроса с целью дополнительных ограничений на выводимые данные. Подзапросы называют также *вложенными запросами.* Подзапрос в содержащем его запросе используют для наложения условий на выводимые данные. Подзапросы могут использоваться с операторами SELECT, INSERT, UPDATE или DELETE.

В некоторых случаях подзапрос можно использовать вместо связывания таблиц, тем самым связывая данные таблиц неявно. При использовании в запросе подзапроса сначала выполняется подзапрос, а только потом — содержащий его запрос, причем с учетом условий выполнения подзапроса. Результаты выполнения подзапроса используются при обработке условий в выражении ключевого слова WHERE основного запроса Подзапрос можно использовать либо в выражении ключевого слова WHERE, либо в выражении ключевого слова HAVING главного запроса. Логические операции и операции сравнения типа =, >, <, о, IN, NOT IN, AND, OR и т п. можно использовать как в подзапросе, так и для обработки результатов подзапроса в выражениях ключевых слов WHERE и HAVING.

При составлении подзапросов необходимо придерживаться следующих правил.

• Подзапрос необходимо заключить в круглые скобки.

• Подзапрос может ссылаться только на один столбец в выражении своего ключевого слова SELECT, за исключением случаев, когда в главном запросе используется сравнение с несколькими столбцами из подзапроса.

• Ключевое слово ORDER BY использовать в подзапросе нельзя, хотя в главном запросе ORDER BY использоваться может. Вместо ORDER BY в подзапросе можно использовать GROUP BY.

• Подзапрос, возвращающий несколько строк данных, можно использовать только в операторах, допускающих множество значений, например в IN.

• В списке ключевого слова SELECT не допускаются ссылки на значения типа BLOB, ARRAY, CLOB **ИЛИ** NCLOB.

• Подзапрос нельзя непосредственно использовать как аргумент допускающей множество значений функции.

• Операцию BETWEEN по отношению к подзапросу использовать нельзя, но ее можно использовать в самом подзапросе. Базовый синтаксис оператора с подзапросом выглядит следующим образом.

SELECT имя_столбца

FROM таблица

WHERE имя_столбца = (SELECT имя__столбца

FROM таблица

WHERE условия);

Отступы используются исключительно в целях оформления. Практика показывает, что чем аккуратнее выглядят операторы SQL, тем проще они для понимания и тем легче искать и исправлять в них ошибки, если таковые вдруг обнаруживаются.

Рассмотрим примеры правильного использования операции BETWEEN в операторе с подзапросом.

Вот пример правильного использования BETWEEN:

SELECT имя_столбца

FROM таблица

WHERE имя_столбца ОПЕРАЦИЯ (SELECT имя_столбца

FROM таблица

WHERE значение BETWEEN значение);

Подзапросы в операторе SELECT

Чаще всего используются подзапросы с оператором SELECT, хотя, конечно, используются и подзапросы с операторами манипуляций данными. Подзапросы в операторе SELECT извлекают данные для главного запроса.

Базовый синтаксис соответствующего оператора следующий

```
SELECT имя_столбца [, имя_столбца ]
FROM таблица1 [, таблица2 ]
WHERE имя_столбца ОПЕРАЦИЯ
(SELECT имя_столбца [, имя_столбца ]
FROM таблица1 [, таблица2 ]
[ WHERE ]);
```

Например,

```
SELECT E.EMP_ID, E.LAST_NAME, E.FIRST_NAME
EP.PAY__RATE
FROM EMPLOYEE_TBL E, EMPLOYEE_PAY_TBL EP
WHERE E.EMP_ID = EP.EMP_ID
AND EP.PAY_RATE > (SELECT PAY_RATE
FROM EMPLOYEE_PAY_TBL
WHERE E.EMP_ID = '313782439');
```

Подзапросы в операторе INSERT

Подзапросы могут использоваться и с операторами языка манипуляций данными (DML). Первым из таких операторов мы рассмотрим оператор INSERT. Оператор INSERT использует данные, возвращаемые подзапросом, для помещения их в другую таблицу. Выбранные в подзапросе данные можно модифицировать с помощью символьных или числовых функций, а также функций дат и времени.

Базовый синтаксис соответствующего оператора следующий:

INSERT INTO имя_таблицы [(столбец! [, столбец2])]
SELECT [*| столбец1 [, столбец2]]
FROM таблица1 [, таблица2]
[WHERE значение ОПЕРАЦИЯ значение]

Вот пример использования оператора INSERT с подзапросом.

```
INSERT INTO RICH_EMPLOYEES
SELECT E.EMP_ID, E.LAST_NAME, E.FIRST_NAME EP.PAY_RATE
FROM EMPLOYEE_TBL E, EMPLOYEE_PAY_TBL EP
WHERE E.EMP_ID = EP.EMP_ID
AND EP.PAY_RATE >
 (SELECT PAY_RATE
FROM EMPLOYEE_PAY_TBL
WHERE E.EMP_ID = '220984332');
```

Этот оператор INSERT вставляет значения EMP_ID, LAST_NAME, FIRST_NAME и PAY_RATE в таблицу RICH_EMPLOYEES для всех служащих, норма оплаты труда которых превышает норму оплаты труда служащего с табельным номером 220984332.

Подзапросы в операторе UPDATE

Подзапросы можно использовать в операторе UPDATE. С помощью оператора UPDATE с подзапросом можно обновлять данные как одного, так и нескольких столбцов сразу.

Базовый синтаксис оператора следующий.

UPDATE таблица

SET имя_столбца [, имя_столбца] =

(SELECT имя_столбца [,имя_столбца] FROM таблица

[WHERE])

Рассмотрим примеры, разъясняющие использование оператора UPDATE с подзапросом. Сначала рассмотрим запрос, возвращающий табельные номера служащих из Пскова.

Ввод:

```
SELECT EMP_ID
FROM EMPLOYEE_TBL
WHERE CITY = ‘PSKOV';
```

Вот этот оператор UPDATE с подзапросом.

```
UPDATE EMPLOYEE_PAY_TBL
SET PAY_RATE = PAY_RATE * 1.1
WHERE EMP_ID IN (SELECT EMP_ID
FROM EMPLOYEE_TBL
WHERE CITY = ‘PSKOV');
```

Подзапросы в операторе DELETE

Подзапросы можно использовать в операторе DELETE. Базовый синтаксис оператора следующий.

DELETE FROM имя_таблицы

[WHERE ОПЕРАЦИЯ [значение]

(SELECT имя_столбца

FROM имя_таблицы

[WHERE])

В следующем примере из таблицы EMPLOYEE_PAY_TBL удаляется запись с информацией о служащем по имени BRANDON GLASS. Табельный номер этого служащего не известен, но можно создать подзапрос, который найдет этот номер в таблице EMPLOYEEJTBL по значениям столбцов с именами (FIRST_NAME) и фамилиями (LAST_NAME) служащих.

DELETE FROM EMPLOYEE_PAY_TBL WHERE EMP_ID = (SELECT EMP_ID

FROM EMPLOYEE_PAY_TBL

WHERE LAST_NAME = 'GLASS'

AND FIRST_NAME = 'BRAHDON');

Подзапросы внутри подзапросов

Точно так же, как подзапрос можно вложить в главный запрос, подзапрос можно вложить и в подзапрос. В главном запросе подзапрос выполняется до выполнения главного, точно так же и в подзапросе вложенный в него подзапрос будет выполнен первым

По поводу имеющихся ограничений (если они есть вообще) на число вложений одних запросов в другие в рамках одного оператора обратитесь к документации по используемой вами реализации языка, поскольку такие ограничения для разных реализаций могут не совпадать

Базовый синтаксис для операторов, использующих вложенные подзапросы, должен быть следующим

SELECT имя_столбца [, имя_столбца]

FROM таблица1 [, таблица2]

WHERE имя_столбца ОПЕРАЦИЯ (SELECT имя__столбца

FROM таблица

WHERE имя_столбца ОПЕРАЦИЯ (SELECT имя_столбца FROM таблица

[WHEREимя_столбца ОПЕРАЦИЯ значение]))

В следующем примере используются два подзапроса, вложенные один в другой . Требуется выяснить, какие покупатели заказали товаров на сумму большую, чем сумма цен всех товаров

Ввод:

```
SELECT CUST_ID, CUST_NAME
FROM CUSTOMER_TBL
WHERE CUST_ID IN (SELECT O.CUST_ID
FROM ORDERS_TBL O, PRODUCTS_TBL P
WHERE O.PROD_ID = P.PROD_ID
AND O.QTY * P.COST > (SELECT SUM(COST)
FROM PRODUCTS_TBL));
```

Вот как последовательно выполнялись запросы в данном операторе.

Ввод:

```
SELECT SUM(COST) FROM PRODUCTS_TBL));
```

Вывод:

SUM(COST)

138.08 1 строка выбрана.

Ввод:

```
SELECT O.CUST_ID
FROM ORDERS_TBL O, PRODUCTS_TBL P
WHERE O.PROD_ID = P.PROD_ID
AND O.QTY * P.COST > 138.08;
```

После подстановки в главный запрос результатов внутреннего подзапроса главный запрос принимает следующий вид

```
SELECT CUST_ID, CUST_NAME
FROM CUSTOMER_TBL
WHERE CUST_ID IN (SELECT O.CUST_ID
FROM ORDERS_TBL O, PRODUCTS_TBL P
WHERE O PROD_ID = P.PROD_ID
AND O.QTY * P.COST > 138.08);
```

При использовании в операторе нескольких подзапросов увеличивается время, необходимое для обработки запроса, и повышается вероятность ошибок из-за усложнения оператора

Связанные подзапросы

Связанные подзапросы допускаются во многих реализациях SQL. Концепция связанного подзапроса определяется стандартом ANSI SQL.

Связанный подзапрос — это подзапрос, зависящий от информации, предоставляемой главным запросом.

В следующем примере в подзапросе определение связи между таблицами CUSTOMER_TBL И ORDERSJTBL использует псевдоним таблицы CUSTOMERJTBL (C), определенный в главном запросе. Этот оператор возвращает имена всех покупателей, заказавших более 10 единиц товара.

Ввод:

```
SELECT C.CUST_NAME
FROM CUSTOMER_TBL C
WHERE 10 < (SELECT SUM(O.QTY)
FROM ORDERS_TBL O
WHERE O.CUST_ID =C.CUST_ID);
```

В случае связанного подзапроса ссылка на таблицу главного запроса должна быть определена до начала выполнения подзапроса.

В следующем операторе этот запрос немного модифицирован, чтобы получить список всех заказчиков с соответствующим количеством заказанных товаров и иметь возможность проверить результаты предыдущего примера.

Ввод:

```
SELECT C.CUST_NAME, SUM(O.QTY)
FROM CUSTOMER_TBL C,
ORDERS_TBL O GROUP BY CUST_NAME;
```

Ключевое слово GROUP BY здесь требуется потому, что по отношению ко второму столбцу используется итоговая функция SUM. Это позволяет подсчитать суммы для каждого из заказчиков В предыдущем примере ключевое слово GROUP BY не требовалось, поскольку там функция зим использовалась для суммирования всех результатов запроса, выполняемого для каждого конкретного заказчика.

Подзапрос представляет собой запрос, выполняемый в рамках другого запроса для задания дополнительных условий на выводимые данные. Подзапрос можно использовать в выражениях ключевых слов WHERE и HAVING. Подзапросы обычно используют в других запросах (операторах DQL — языка запросов к данным), но подзапросы можно использовать и в операторах DML (языка манипуляций данными) таких, как INSERT, UPDATE и DELETE. Все основные правила использования операторов языка манипуляций данными применимы и при использовании в них подзапросов.

Синтаксис подзапросов практически не отличается от синтаксиса обычного запроса, имеются лишь небольшие ограничения. Одним из таких ограничений является запрет на использование в подзапросах ключевого слова ORDER BY, однако, вместо него можно использовать ORDER BY, чем достигается практически тот же эффект. Подзапросы используются для размещения в запросах условий, точные данные для которых не известны, тем самым расширяя возможности и гибкость SQL.