**Подзапросы**

Предположим, необходимо получить список книг, написанных Александром Сергеевичем Пушкиным, из базы данных «Книги», в которой информация об авторах и о книгах хранится в отдельных таблицах. Это можно реализовать при помощи многотабличного запроса:

|  |
| --- |
| **SELECT b.name**  **FROM Authors a, Books b**  **WHERE a.id = b.id\_author AND a.name LIKE “Пушкин А.С.”** |

Логика работы приведенного выше примера реализуется по принципу декартова произведения. А это означает, что СУБД придется перебирать все возможные комбинации записей из таблиц «Авторы» и «Книги».

В данной ситуации более логично было бы сначала найти идентификатор записи в таблице «Авторы», соответствующей автору «Пушкин А.С.», а уже затем искать нужные записи в таблице «Книги». При этом СУБД не придется перебирать все возможные комбинации записей, просто надо будет пройтись по одному разу обе таблицы.

Для реализации этой, а также других весьма полезных и практически востребованных возможностей, используются т.н. **подзапросы** (также используется термин **вложенные запросы**).

**Суть подзапросов заключается в том, что один подзапрос может управлять поведением другого подзапроса.**

Вот как приведенный выше пример может быть реализован при помощи подзапросов:

|  |
| --- |
| **SELECT name**  **FROM Books**  **WHERE id\_author = (**  **SELECT id**  **FROM Authors**  **WHERE name LIKE “Пушкин А.С.”)** |

Сначала выполняется подзапрос, и только после этого, когда получен нужный идентификатор записи из таблицы «Авторы», он подставляется во внешний запрос, после чего этот внешний запрос выполняется.

Подзапросы не обладают свойством коммутативности: их нельзя поставить в левую часть логического выражения. Т.е. вот такой запрос в соответствии со стандартам языка SQL работать не будет:

|  |
| --- |
| **SELECT name**  **FROM Books**  **WHERE (**  **SELECT id**  **FROM Authors**  **WHERE name LIKE 'Пушкин А.С.') = id\_author** |

Впрочем, Access допускает такую форму записи, но в силу того, что по возможности следует ориентироваться на стандарт языка SQL, использовать такую форму записи не рекомендуется.

Очевидно, что в такой ситуации подзапрос не может вернуть больше одной записи. Иначе логическая операция сравнения не сможет быть правильно вычисленной. В самом деле, если в таблице «Авторы» есть несколько Пушкиных А.С., то идентификатору какого из них должен быть равен внешний ключ? СУБД трактует такую ситуацию как неоднозначную и генерирует ошибку.

Если конкретная ситуация позволяет, то для того, чтобы избежать возвращения нескольких записей из подзапроса, можно использовать директиву **DISTINCT**, но, конечно же, она не является универсальным решением данной проблемы.

Решением также может быть использование в подзапросе агрегатных функций, которые по определению возвращают ровно одну запись.

В то же время, если подзапрос вообще не найдет ни одной записи, то никакой ошибки не будет. Это происходит потому, что отсутствие записей в выборке рассматривается как значение **NULL**, сравнение любого значения со значением **NULL** дает опять же **NULL**, что в логических выражениях трактуется как **FALSE**. Таким образом, логическое выражение в секции **WHERE** просто будет ложным, но вполне корректным выражением.

**Конструкция IN** **и подзапросы**

Бывают ситуации, когда возвращение подзапросом нескольких записей будет вполне соответствовать решаемой задаче. Например, если нужно показать все книги всех авторов, родившихся в некотором заданном году. Очевидно, что таких авторов может быть несколько. В этом случае при использовании операции сравнения мы будем получать ошибку.

Для решения этой проблемы может быть использована конструкция **IN**. Эта конструкция будет рассматривать несколько возвращаемых значений как множество и возвращать истину в случае, если искомое значение совпадает хотя бы с одним из значений этого множества.

Таким образом, показать все книги всех авторов, родившихся в некотором заданном году, можно следующим образом:

|  |
| --- |
| **SELECT name**  **FROM Books**  **WHERE id\_author IN (**  **SELECT id**  **FROM Authors**  **WHERE birth\_year = 1950)** |

Аналогичным образом использовать конструкции **IS NULL**/**IS NOT NULL**, **LIKE**, **BETWEEN** вместе с подзапросами нельзя.

**Связанные подзапросы**

Во всех приведенных выше примерах, подзапрос вычисляется **один раз** до выполнения внешнего подзапроса и значение, которое он возвращает, фактически становится постоянным на момент выполнения этого внешнего запроса. Отмечаем, что получается достаточно простая зависимость внешнего запроса от подзапроса при полном отсутствии обратной зависимости. Однако бывают ситуации, в которых то значение, которое должен возвращать подзапрос, в свою очередь должно зависеть от внешнего запроса. Т.е. то значение, которое возвращает подзапрос, должно быть не одним общим для всего внешнего запроса, а отдельно соответствовать каждой записи внешнего запроса.

Такие запросы называются **связанными подзапросами**. Такие подзапросы зависят от каких-либо значений полей внешнего запроса и вычисляются не один раз до выполнения внешнего запроса, а отдельно для каждой записи внешнего запроса.

Например, показать все издательства, среднее количество страниц в книгах которого больше 300:

|  |
| --- |
| **SELECT \***  **FROM Press p**  **WHERE 300 < (**  **SELECT AVG(pages)**  **FROM Books b**  **WHERE p.id = b.id\_press)** |

Для каждой записи из таблицы **Press** отдельно вызывается подзапрос и в него подставляется значение поля **id** этой записи. Таким образом, значение поля **id** выступает в качестве подставляемого параметра подзапроса.

Пример запроса «Показать всех студентов, которые в данный момент читают книгу Бьярна Страуструпа "Язык программирования C++"» будет выглядеть следующим образом:

|  |
| --- |
| **SELECT \***  **FROM Students**  **WHERE students.id IN (**  **SELECT id\_student**  **FROM StudentCards**  **WHERE id\_book = (**  **SELECT id**  **FROM Books**  **WHERE name = 'Язык программирования C++' AND id\_author = (SELECT id**  **FROM Authors**  **WHERE name = 'Бьярн Страуструп')))** |

**Вопросы для самоконтроля**

1. Что такое подзапросы?
2. Что выполняется раньше – внешний запрос или подзапрос?
3. Каковы ограничения при использовании логических операций сравнения и подзапросов?
4. Для чего нужна конструкция **IN** при работе с подзапросами?
5. Что такое связанные подзапросы?
6. Что выполняется раньше – внешний запрос или связанный подзапрос? Корректен ли такой вопрос?