Лабораторная работа №6

Использование объединяющих, вложенных и корректирующих запросов языка SQL

Цель: изучить возможности оператора UNION, вложенных запросов, а также операторов языка SQL, изменяющих данные в БД.

Для получения **теоретических сведений** настоятельно рекомендуется при домашней подготовке изучить материалы по тематике лабораторной работы, представленные в открытых источниках.

Далее следует краткий конспект теоретического материала.

Теоретические сведения

Помимо использования многотабличных запросов, SQL позволяет формировать таблицу результатов на основе данных, содержащихся в нескольких исходных таблицах, с помощью объединения запросов, а также вложенных запросов.

Оператор UNION

Чтобы таблицы результатов запроса можно было объединить с помощью оператора UNION, они должны удовлетворять следующим условиям:

1) таблицы должны содержать одинаковое число столбцов;

2) тип данных каждого столбца первой таблицы должен совпадать с типом данных соответствующего столбца во второй таблице;

3) ни одна из таблиц не должна быть отсортирована с помощью предложения ORDER BY, однако объединенные результаты запроса можно отсортировать.

Имена столбцов в объединяемых таблицах не обязательно должны быть одинаковыми.

Поскольку оператор UNION объединяет строки из двух таблиц результатов, в объединенной таблице могут содержаться повторяющиеся строки. По умолчанию оператор UNION в процессе своего выполнения удаляет повторяющиеся строки.

В случае необходимости сортировки результата выборки, использующей объединение, в предложении ORDER BY следует указывать номера столбцов.

Оператор UNION можно использовать многократно, чтобы объединять результаты трех или более запросов, указывая с помощью круглых скобок порядок выполнения операторов:

```
(SELECT * FROM A) UNION (SELECT * FROM B) UNION (SELECT * FROM C)
```

Вложенные запросы

Механизм вложенных запросов позволяет использовать результат одного запроса в качестве составной части другого запроса.

Вложенным, или подчиненным, запросом (подзапросом), называется запрос, содержащийся в списке возвращаемых столбцов или предложениях WHERE или HAVING другого оператора. Вложенные запросы позволяют естественным образом формулировать запросы, которые используют результаты других запросов.

Например, из учебной базы данных «Торговая компания» можно определить, кто из сотрудников является самым молодым по возрасту:

```
SELECT Фамилия, ДатаРождения FROM Сотрудники
WHERE ДатаРождения = (SELECT MAX(ДатаРождения) FROM Сотрудники)
```

Вложенный запрос, указанный в условии предложения WHERE, вычисляет максимальную дату рождения сотрудника, а главный (внешний) запрос сравнивает дату рождения каждого сотрудника с вычисленной датой и, в зависимости от результата сравнения, либо добавляет сотрудника в таблицу результатов запроса, либо нет.

Вложенный запрос (подзапрос) – это оператор SELECT, заключенный в круглые скобки. Между вложенным запросом и обычным оператором SELECT имеется ряд отличий:

1) таблица результатов вложенного запроса всегда состоит из одного столбца, т.е. список возвращаемых столбцов должен иметь только один элемент;

2) во вложенный запрос не может входить предложение ORDER BY;

3) вложенный запрос не может быть запросом на объединение;

4) имена столбцов, используемые во вложенном запросе, могут являться ссылками на столбцы таблиц главного (внешнего) запроса.

Для иллюстрации 4-го отличия, определим, кто из поставщиков одних товаров одновременно является клиентом, заказывающим другие товары. Таблица результатов формируется оператором:

```
SELECT * FROM Поставщики
WHERE Название = (
   SELECT Название FROM Клиенты
   WHERE Клиенты.Название = Поставщики.Название
)
```

Столбец Поставщики.Название во вложенном запросе является примером внешней ссылки. Внешняя ссылка представляет собой имя столбца, принадлежащего таблице, указанной в предложении FROM главного запроса, и не входящего ни в одну из таблиц, перечисленных в предложении FROM вложенного запроса.

Если во вложенном запросе имеется внешняя ссылка, он называется **связанным подзапросом**. Особенностью связанного подзапроса является то, что он выполняется многократно, по одному разу для каждой строки таблицы, указанной в главном запросе.

Условия поиска с вложенным запросом

Вложенный запрос обычно является частью условия поиска в предложении WHERE или HAVING. Возможны следующие условия поиска с вложенным запросом:

1) **сравнение** с результатом вложенного запроса. Сравнивает значение выражения с одним значением, возвращенным вложенным запросом;

2) **проверка на принадлежность** результатам вложенного запроса. Проверяет значение выражения на равенство с одним из значений множества, возвращенного вложенным запросом.

3) **проверка на существование** - проверяет наличие строк в таблице результатов вложенного запроса. При этом условие поиска EXISTS не использует результаты вложенного запроса. Проверяется только наличие результатов. По этой причине в SQL смягчается правило о единственности возвращаемого столбца во вложенном запросе: при проверке EXISTS допускается использование формы SELECT * …;

4) **многократное сравнение** - сравнивает значение выражения с каждым из значений множества, возвращаемого вложенным запросом:

- при проверке ANY проверяемое значение поочередно сравнивается с каждым значением, содержащимся в столбце, который сформулирован вложенным запросом. Если любое из этих сравнений дает результат TRUE, то проверка ANY возвращает значение TRUE;
- при проверке ALL проверяемое значение поочередно сравнивается с каждым значением, содержащимся в столбце. Если какое-либо из сравнений не дает результат TRUE, то проверка ALL возвращает значение FALSE.

Например, с использованием многократного сравнения можно определить сотрудников, которые оформили наибольшее число заказов:

```
SELECT Фамилия, Имя, "оформил", COUNT(КодЗаказа), "заказов"
FROM Сотрудники, Заказы
WHERE Сотрудники.КодСотрудника = Заказы.КодСотрудника
GROUP BY Фамилия, Имя
HAVING COUNT(КодЗаказа) >= ALL (
  SELECT COUNT(КодЗаказа)
  FROM Сотрудники, Заказы
  WHERE Сотрудники.КодСотрудника = Заказы.КодСотрудника
  GROUP BY Фамилия, Имя
)
```

Добавление данных в таблицу

Наименьшей единицей информации, которую можно добавить в реляционную БД, является одна строка.

Существует два способа добавления новых строк в БД:

1) однострочный оператор INSERT позволяет добавить в таблицу новую строку;

2) многострочный оператор INSERT обеспечивает извлечение строк из одной части БД и добавление их в другую таблицу.

В предложении INTO **однострочного оператора INSERT** указывается целевая таблица, в которую добавляется новая строка, а в предложении VALUES содержатся значения данных для новой строки. Список столбцов определяет, какие значения, и в какой столбец заносятся.

Например, можно задать запросы для внесения в БД сведений о новых компаниях, осуществляющих доставку следующими способами:

а) если список столбцов не указан, последовательность значений данных должна точно соответствовать порядку столбцов в таблице:

```
INSERT INTO Доставка
VALUES (1234, 'ООО ВычТехника', '(495) 123-4567')
```

б) в списке столбцов их имена могут указываться в любой удобной последовательности, определяющей порядок значений данных в предложении VALUES:

```
INSERT INTO Доставка (Название, Телефон, КодДоставки)
VALUES ('ООО ВычТехника', '(495) 123-4567', 1234)
```

в) можно не указывать часть столбцов и их значений, воспользовавшись тем, что при добавлении в таблицу новой строки всем столбцам, имена которых отсутствуют в списке столбцов оператора INSERT, СУБД автоматически присваивает значение NULL либо значение, заданное по умолчанию при описании структуры таблицы:

```
INSERT INTO Доставка (Телефон, Название)
VALUES ('(495) 123-4567', 'ООО ВычТехника')
```

Многострочный оператор INSERT добавляет в целевую таблицу одну или несколько строк. Источником новых строк служит запрос на чтение, содержащийся в операторе INSERT.

Например, можно добавить в таблицу ДОСТАВКА сведения о новой компании, у которой название и телефон будут такими же, как у клиента с кодом ANTON:

```
INSERT INTO Доставка (Телефон, Название)
  SELECT Телефон, Название FROM Клиенты
  WHERE КодКлиента = 'ANTON'
```

Обновление существующих данных

Наименьшей единицей информации, которую можно обновить в реляционной БД является значение одного столбца в одной строке.

Для обновления значения одного или нескольких столбцов в выбранных строках одной таблицы предназначен оператор UPDATE.

В операторе указывается целевая таблица, которая должна быть модифицирована. Предложение WHERE отбирает строки таблицы, подлежащие обновлению. В предложении SET указывается, какие столбцы в выбранных строках таблицы должны быть обновлены, и для них задаются новые значения.

Следующий оператор UPDATE изменяет номер телефона компании «Ространс», которая доставляет заказы:

```
UPDATE Доставка SET Телефон='(495) 765-1234'
WHERE Название = 'Ространс'
```

Оператор UPDATE может одновременно обновлять столбцы в нескольких строках, соответствующих условию поиска.

Строки, для которых условие поиска имеет значение TRUE, обновляются, а строки, для которых условие поиска имеет значение FALSE или NULL, не обновляются.

Выражение в операции присваивания может быть любым правильным выражением языка SQL, результирующее значение которого имеет тип данных, соответствующий целевому столбцу.

Значение выражения вычисляется на основе значений строки, которая в данный момент обновляется в целевой таблице.

Выражение не может включать в себя какие-либо агрегатные функции или запросы.

Если выражение содержит ссылку на один из столбцов целевой таблицы, для вычисления выражения используется значение этого столбца в текущей строке, которое было перед обновлением. То же справедливо для ссылок на столбцы в предложении WHERE.

Если предложение WHERE отсутствует в записи оператора UPDATE, обновляются все строки целевой таблицы, например:

```
UPDATE Заказано SET Скидка = Скидка + 0.01
```

Предложение WHERE может содержать вложенный запрос, если необходимо отбирать строки на обновление, опираясь на информацию из других таблиц. Например, можно повысить в должности сотрудников, оформивших более 100 заказов:

```
UPDATE Сотрудники SET Должность = 'Топ-менеджер'
WHERE 100 < (
  SELECT COUNT (*) FROM Заказы
  WHERE Сотрудники.КодСотрудника = Заказы.КодСотрудника
)
```

Удаление существующих данных

Наименьшей единицей информации, которую можно удалить из реляционной БД, является строка.

Для удаления выбранных строк из одной таблицы используется оператор DELETE, в предложении FROM которого указывается таблица, содержащая удаляемые строки, а в предложении WHERE указывается условие, которому должны соответствовать удаляемые строки, например:

```
DELETE FROM Доставка
WHERE КодДоставки=1235
```

Если предложение WHERE отсутствует в записи оператора DELETE, то такой оператор удаляет из таблицы все строки. Например, оператор:

```
DELETE FROM Доставка
```

удалит все строки из таблицы «Доставка».

В предложении WHERE условие поиска может содержать вложенные запросы.

Например, прежде чем удалять информацию из таблицы «Доставка», необходимо из таблицы «Заказы» удалить все строки, связанные с таблицей «Доставка»:

```
DELETE FROM Заказы
WHERE Заказы.Доставка IN (
  SELECT КодДоставки FROM Доставка
)
```

Лабораторное задание

Для БД, разработанной на предыдущих лабораторных работах, **при домашней подготовке** сформулировать и **записать на языке SQL** следующие запросы:

- не менее 3 запросов с объединением;
- не менее 5 вложенных запросов на выборку данных (в том числе с ALL, ANY и EXISTS);
- по одному однострочному запросу на добавление данных для каждой таблицы;
- не менее 3 многострочных запросов на добавление данных;
- не менее 5 запросов на обновление данных;
- не менее 3 запросов на удаление данных, в том числе, с условием, использующим вложенный запрос.

Проверить разработанные запросы, занося результаты в **Отчет.**

Требования к сдаче работы

1. Продемонстрировать разработанные по заданию запросы на языке SQL.
2. Продемонстрировать **отчет** о выполнении лабораторной работы.
3. Ответить на контрольные вопросы и выполнить дополнительные задания, демонстрирующие полученные знания.