

Лекция 10. Внешние, перекрестные и естественные соединения

Ранее мы подробно рассмотрели применение внутреннего соединения для связывания нескольких таблиц. Данная лекция посвящена другим способам соединения таблиц, включая внешнее и перекрестное соединения.

Внешние соединения

До сих пор ни в одном из приведенных примеров, включающих запросы к нескольким таблицам, не поднимался вопрос о том, что не все строки таблицы могут соответствовать условиям соединения. Например, при соединении таблицы **account** с таблицей **customer** мы не говорили о возможности отсутствия для значения столбца *cust_id* таблицы **account** соответствующего значения в столбце *cust_id* таблицы **customer**. Если бы такое случилось, некоторые строки одной из таблиц не вошли бы в результирующий набор.

Выведем некоторые столбцы из таблиц **account** и **customer**:

```
SELECT account_id, cust_id  
FROM account;
```

account_id	cust_id
1	1
2	1
3	1
4	2
5	2
6	3
7	3
8	4
9	4
10	4
11	5
12	6
13	6
14	7
15	8
16	8
17	9
18	9
19	9
20	10
21	10
22	11
23	12
24	13

24 rows in set (0.04 sec)

```
SELECT cust_id  
FROM customer;
```

cust_id
1
2
3
4
5
6
7
8
9
10
11
12
13

13 rows in set (0.02 sec)

При соединении этих двух таблиц по столбцу *cust_id* можно ожидать, что в результирующий набор будут включены все 24 строки:

```
SELECT a.account_id, c.cust_id
FROM account a INNER JOIN customer c
ON a.cust_id = c.cust_id;
```

```
+-----+-----+
| account_id | cust_id |
+-----+-----+
|      1 |      1 |
|      2 |      1 |
|      3 |      1 |
|      4 |      2 |
|      5 |      2 |
|      6 |      3 |
|      7 |      3 |
|      8 |      4 |
|      9 |      4 |
|     10 |      4 |
|     11 |      5 |
|     12 |      6 |
|     13 |      6 |
|     14 |      7 |
|     15 |      8 |
|     16 |      8 |
|     17 |      9 |
|     18 |      9 |
|     19 |      9 |
|     20 |     10 |
|     21 |     10 |
|     22 |     11 |
|     23 |     12 |
|     24 |     13 |
+-----+-----+
24 rows in set (0.00 sec)
```

Если же мы соединим таблицу **account** с одной из специализированных таблиц клиентов, например, таблицей **business**, то увидим следующее:

```
SELECT a.account_id, b.cust_id, b.name
FROM account a INNER JOIN business b
ON a.cust_id = b.cust_id;
```

```
+-----+-----+-----+
| account_id | cust_id | name          |
+-----+-----+-----+
|      20 |      10 | Chilton Engineering |
|      21 |      10 | Chilton Engineering |
|      22 |      11 | Northeast Cooling Inc. |
|      23 |      12 | Superior Auto Body   |
|      24 |      13 | AAA Insurance Inc.   |
+-----+-----+-----+
5 rows in set (0.00 sec)
```

Теперь в результирующем наборе только 5 строк вместо 24. Из 13 строк таблицы клиентов только 4 относятся к юридическим лицам. И поскольку у одного из юридических лиц два счета, в общей сложности с юридическими лицами связаны 5 строк таблицы **account**.

Но что делать, если требуется, чтобы запрос возвращал все счета, но при этом включал название фирмы, только если счет связан с юридическим лицом? Это пример, когда необходимо **внешнее соединение (outer join)**.

```
SELECT a.account_id, a.cust_id, b.name  
FROM account a LEFT OUTER JOIN business b  
ON a.cust_id = b.cust_id;
```

account_id	cust_id	name
1	1	NULL
2	1	NULL
3	1	NULL
4	2	NULL
5	2	NULL
6	3	NULL
7	3	NULL
8	4	NULL
9	4	NULL
10	4	NULL
11	5	NULL
12	6	NULL
13	6	NULL
14	7	NULL
15	8	NULL
16	8	NULL
17	9	NULL
18	9	NULL
19	9	NULL
20	10	Chilton Engineering
21	10	Chilton Engineering
22	11	Northeast Cooling Inc.
23	12	Superior Auto Body
24	13	AAA Insurance Inc.

24 rows in set (0.00 sec)

Внешнее соединение включает все строки одной таблицы и выводит данные второй таблицы только в случае обнаружения соответствующих строк.

Сравнение левосторонних и правосторонних внешних соединений

Ключевое слово **left** свидетельствует о том, что таблица, находящаяся в левой части блока *from*, отвечает за определение числа строк в результирующем наборе, а таблица в правой части предоставляет значения столбцов в случае обнаружения соответствия.

Рассмотрим следующий пример:

```
SELECT c.cust_id, b.name  
FROM customer c LEFT OUTER JOIN business b  
ON c.cust_id = b.cust_id;
```

cust_id	name
1	NULL
2	NULL
3	NULL
4	NULL
5	NULL
6	NULL
7	NULL
8	NULL
9	NULL
10	Chilton Engineering
11	Northeast Cooling Inc.
12	Superior Auto Body
13	AAA Insurance Inc.

13 rows in set (0.00 sec)

Блок *from* определяет *левостороннее внешнее соединение*. Таким образом, в результирующий набор входят все 13 строк таблицы **customer**, а таблица **business** поставляет значения во второй столбец результирующего набора для четырех клиентов-юридических лиц.

Если выполнить предыдущий запрос, но указать *правостороннее внешнее соединение*, будет получен следующий результат:

```
SELECT c.cust_id, b.name  
FROM customer c RIGHT OUTER JOIN business b  
ON c.cust_id = b.cust_id;
```

cust_id	name
10	Chilton Engineering
11	Northeast Cooling Inc.
12	Superior Auto Body
13	AAA Insurance Inc.

4 rows in set (0.00 sec)

Теперь число строк результирующего набора определяется количеством строк таблицы **business**.

Ключевые слова *left* и *right* просто сообщают оптимизатору БД, какая таблица может иметь пробелы в данных. Если нужно провести внешнее соединение таблиц **A** и **B** таким образом, чтобы в результирующий набор входили все строки из **A** и те строки из **B**, для которых есть соответствующие данные, можно задать *A left outer join B* или *B right outer join A*.

Трехсторонние внешние соединения

В некоторых случаях может потребоваться провести внешнее соединение одной таблицы с двумя другими таблицами. Например, нужен список всех счетов с указанием имени и фамилии физического лица, или названия фирмы для юридического лица:

```
SELECT a.account_id, a.product_cd,
       CONCAT(i.fname, ', ', i.lname) person_name,
       b.name business_name
  FROM account a LEFT OUTER JOIN individual i
    ON a.cust_id = i.cust_id
   LEFT OUTER JOIN business b
    ON a.cust_id = b.cust_id;
```

account_id	product_cd	person_name	business_name
1	CHK	James Hadley	NULL
2	SAV	James Hadley	NULL
3	CD	James Hadley	NULL
4	CHK	Susan Tingley	NULL
5	SAV	Susan Tingley	NULL
6	CHK	Frank Tucker	NULL
7	MM	Frank Tucker	NULL
8	CHK	John Hayward	NULL
9	SAV	John Hayward	NULL
10	MM	John Hayward	NULL
11	CHK	Charles Frasier	NULL
12	CHK	John Spencer	NULL
13	CD	John Spencer	NULL
14	CD	Margaret Young	NULL
15	CHK	Louis Blake	NULL
16	SAV	Louis Blake	NULL
17	CHK	Richard Farley	NULL
18	MM	Richard Farley	NULL
19	CD	Richard Farley	NULL
20	CHK	NULL	Chilton Engineering
21	BUS	NULL	Chilton Engineering
22	BUS	NULL	Northeast Cooling Inc.
23	CHK	NULL	Superior Auto Body
24	SBL	NULL	AAA Insurance Inc.

Результаты включают все 24 строки таблицы **account**, а также имена клиентов или названия фирм, поступающие из двух других таблиц в результате внешнего соединения.

Для того чтобы сократить число соединений в запросе, можно воспользоваться **подзапросами**.

Рекурсивные внешние соединения

В лекции 5 был рассмотрен пример *рекурсивного соединения*, при котором таблица соединялась сама с собой для формирования списка сотрудников и их руководителей. Однако в том запросе в результирующий набор не включались сотрудники, у которых нет начальника. Чтобы это исправить, необходимо просто заменить *внутреннее соединение* на *внешнее*:

```
SELECT e.fname, e.lname,
       e_mgr.fname mgr_fname, e_mgr.lname mgr_lname
  FROM employee e LEFT OUTER JOIN employee e_mgr
    ON e.superior_emp_id = e_mgr.emp_id;
```

fname	lname	mgr_fname	mgr_lname
Michael	Smith	NULL	NULL
Susan	Barker	Michael	Smith
Robert	Tyler	Michael	Smith
Susan	Hawthorne	Robert	Tyler
John	Gooding	Susan	Hawthorne
Helen	Fleming	Susan	Hawthorne
...			

Данный запрос использует *левостороннее внешнее соединение*. Если сделать внешнее соединение *правосторонним*, то будут выбраны все руководители вместе со всеми их подчиненными. При этом некоторые руководители будут выводиться несколько раз, так как у них может быть несколько подчиненных, а также будут выведены сотрудники в столбцах **mgr_fname** и **mgr_lname**, у которых нет подчиненных. Поэтому при использовании внешнего соединения необходимо тщательно продумывать, каким оно должно быть – *левосторонним* или *правосторонним*.

Перекрестные соединения

В лекции 5 также была представлена концепция *декартова произведения*, которое является результатом соединения нескольких таблиц без указания каких-либо условий соединения. Оно используется довольно редко, но если действительно требуется получить декартово произведение двух таблиц, должно быть задано *перекрестное соединение*:

```
SELECT pt.name, p.product_cd, p.name
FROM product p CROSS JOIN product_type pt;
```

name	product_cd	name
Customer Accounts	AUT	auto loan
Customer Accounts	BUS	business line of credit
Customer Accounts	CD	certificate of deposit
Customer Accounts	CHK	checking account
Customer Accounts	MM	money market account
Customer Accounts	MRT	home mortgage
Customer Accounts	SAV	savings account
Customer Accounts	SBL	small business loan
Insurance Offerings	AUT	auto loan
Insurance Offerings	BUS	business line of credit
Insurance Offerings	CD	certificate of deposit
Insurance Offerings	CHK	checking account
Insurance Offerings	MM	money market account
Insurance Offerings	MRT	home mortgage
Insurance Offerings	SAV	savings account
Insurance Offerings	SBL	small business loan
Individual and Business Loans	AUT	auto loan
Individual and Business Loans	BUS	business line of credit
Individual and Business Loans	CD	certificate of deposit
Individual and Business Loans	CHK	checking account
Individual and Business Loans	MM	money market account
Individual and Business Loans	MRT	home mortgage
Individual and Business Loans	SAV	savings account
Individual and Business Loans	SBL	small business loan

24 rows in set (0.00 sec)

Естественные соединения

Если вы не хотите самостоятельно задавать условие соединения, то можно выбрать тип соединения, при котором сервер БД сам определяет необходимые условия соединения указанных вами таблиц. Такой тип соединения называется **естественным соединением** (*natural join*), он делает предположение о необходимых условиях соединения, полагаясь на идентичные имена столбцов в таблицах. Например,

```
SELECT a.account_id, b.cust_id, b.name
FROM account a NATURAL JOIN business b;
```

Поскольку задано естественное соединение, сервер проверил описания таблиц и добавил для этих двух таблиц условие соединения $a.cust_id = b.cust_id$.

Однако если имена соединяемых столбцов не совпадают, то сервер не сможет сформировать правильное соединение, и будут возвращены некорректные результаты. Поэтому следует избегать применения этого типа

соединения и использовать внутренние соединения с явными условиями соединения.

Литература:

1. Алан Бьюли. Изучаем SQL: пер. с англ. – СПб-М.: Символ, O'Reilly, 2007. – 310 с.