Добавление, изменение и удаление информации в таблицах

Пример

INSERT INTO addresses (name, phone) VALUES ("NIck Petrov", "785-91-91")

Другой, более сложный пример

INSERT INTO addresses (name, phone, addresses) VALUES (select name, phone, addresses from prod_addresses)

В этом примере таблица prod_addresses содержит входную информацию. Информация, которая вводится в таблицу addresses, перемещена из трех столбцов таблицы prod_addresses, которая может содержать и другие столбцы. В результате выполнения этого кода для каждой строки таблицы prod_addresses будет создана новая строка.

Обновление строк

Для изменения существующих значений в столбцах таблицы можно использовать инструкцию **UPDATE**

Синтаксис

UPDATE имя_таблицы

SET имя_столбца = значение

WHERE имя_столбца оператор_сравнения значение

Пример

UPDATE employees

SET department = "Sales", badge = 1232

WHERE name = "Petrov"

UPDATE employees

SET department = NULL

WHERE name = "Petrov"

UPDATE pays

SET rate = rate + 2

Удаление строк

Синтаксис

DELETE[FROM] имя_таблицы

WHERE имя_столбца = значение

Пример

DELETE from employees

WHERE department = "SALES"

Если в инструкции **DELETE FROM** не указывать предложение **WHERE**, то будут удалены все строки таблицы:

delete from employee

Добавление столбцов с помощью инструкции ALTER TABLE

Синтаксис

ALTER TABLE имя_таблицы

ADD имя_столбца тип данных

Пример

ALTER TABLE employees

ADD wayclass char(2) null

Удаление столбца из таблицы

Определяется новая таблица, которая содержит только два из трех столбцов существующей таблицы. Инструкция **INSERT** используется вместе с инструкцией **SELECT**

Пример

```
CREATE TABLE employees5 (

name char(20),

badge int

)

INSERT INTO employees5

SELECT name, badge FROM employees5
```

Выполнение реляционных объединений

Пример

```
CREATE TABLE employees (

name char(20),

badge int,

department char(20)

)

CREATE TABLE pays (

hours_worked int,

rate int,

badge int

)
```

Можно объединять и отображать строки разных таблиц и манипулировать ими с помощью тех же инструкций, которые используются при работе с одной таблицей. Строки нескольких таблиц можно объединять по разному. Первый способ, который мы рассмотрим, называется естественным объединением. Естественные объединения выполняются путем подбора строк с одинаковыми значениями в общих для нескольких таблиц столбцах.

Необходимо определить одну из этих таблиц так, чтобы одни из столбцов первой таблицы дублировались во второй, тогда этот столбец будет общим для обеих таблиц. Если в этом столбце хранятся значения, которые делают строки первой таблицы уникальными, то такой столбец может быть первичным ключом для первой таблицы. Столбец, который продублирован во второй таблице, называется внешним ключом.

Пример

SELECT name, department, hours_worked FROM employees, pays WHERE employees.badge = pays.badge

Столбцы name и department из одной таблицы, а также столбец hours_worked из другой таблицы отображается после того, как с помощью столбца badge объединяются строки с одинаковыми номерами из этих двух таблиц

Если реализовать этот запрос без предложения where, то sql сервер формирует результат объединений строк в обеих таблицах. Если при объединении не использовать парные столбцы, то каждая строка второй таблицы просто добавляется к каждой строке первой таблицы.

Пример

SELECT * FROM employees, pays

Комбинации всех строк второй таблицы со всеми строками первой таблицы дает в результате их перекрестное произведение(декартово). Если таблица employees и таблица pays содержат по 10 строк каждая, то в результате передается произведение 100 строк

Пример

SELECT e.badge, p.id FROM employees e, pays p WHERE e.badge = p.uid

Объединение один ко многим

В предыдущем примере для каждой строки таблицы employees в таблице pays имелась только одна парня строка с одинаковым для обеих таблиц значением столбца badge. Бывают таблицы, в которых не одно, а несколько значений столбца одной таблицы будут совпадать со значением столбца другой таблицы. Допустим в таблицу employees были добавлены 3 работника, каждый с одной и той же фамилией и одним табельным номером. В этом случае столбец badge уже не определен в качестве первичного ключа, поэтому допустимы повторяющиеся табельные номера.

Тогда:

SELECT name, pays.badge, hourse_worked, rate FROM employees, pays WHERE employees.badge = pays.badge

Объединение многие к одному - несколько одинаковых значений столбца badge таблицы employees совпадают с одним значением одноименного столбца таблицы pays.

Если изменить порядок задания объединяемых таблиц, то также объединение превратится в отношение один ко многим.

Пример

SELECT name, pays.badge, hourse_worked, rate FROM pays, employees WHERE employees.badge = pays.badge

Объединение многие ко многим

Иногда нужно объединить таблицы, в каждой из которых имеется несколько строк с одинаковыми значениями. Такое объединение называется многие ко многим. Если в нашем примере добавить к таблице pays несколько строк, в которых в столбце badge одно и то же значение, то получится как раз такое объединение.

Пример:

SELECT name, pays.badge, hourse_worked, rate FROM pays, employees WHERE employees.badge = pays.badge

Внешние объединения

В предыдущих примерах делались объединения без учета строк, которые не имеют соответствующих или парных строк в каждой из двух таблиц. Внешние объединения используются для получения строк таблиц, имеющих как одинаковые, так и не одинаковые значения.

Предложение **WHERE** инструкции **SELECT** знает, что возвращаемый результат должен содержать строки, содержащие одинаковые значения в общих столбцах обеих таблиц, а также остальные строки таблицы employees, заданной слева от комбинации символов *=.

Пример

SELECT * FROM employees, pays WHERE employees.badge = pays.badge