

SQL: операции вставки, удаления, модификации записей

Sumy Educational Center
Software Quality Assurance (QA)

Выражения SQL

<ul style="list-style-type: none">• SELECT	Получение данных
<ul style="list-style-type: none">• INSERT• UPDATE• DELETE	Язык манипулирования данными Data manipulation language (DML)
<ul style="list-style-type: none">• CREATE• ALTER• DROP• RENAME• TRUNCATE	Язык определения данных Data definition language (DDL)
<ul style="list-style-type: none">• COMMIT• ROLLBACK• SAVEPOINT	Управление транзакциями Transaction control (TC)
<ul style="list-style-type: none">• GRANT• REVOKE	Язык управления данными Data control language (DCL)

INSERT

- Для вставки строк в таблицу используется предложение INSERT:

```
INSERT INTO  table [(column [, column...])]  
VALUES      (value [, value...]);
```

- При этом за один раз вставляется только одна строка

Insert into ... values (...)

- Insert into ... values (...) вставляет 1 строку

```
SQL> INSERT INTO dept (deptno, dname)
      VALUES (1, 'New dep')
```

- В качестве значений можно использовать
 - Выражения (в т.ч. с функциями)
 - Null
 - Default

```
SQL> INSERT INTO dept (deptno, dname, loc)
      VALUES (20, 'New dep', DEFAULT)
```

Как проверить, вставлены или нет значения?

```
SQL> INSERT INTO dept (deptno, dname, loc)
        VALUES (20, 'New dep', DEFAULT)
```

```
SQL> SELECT *
2 FROM dept
3 WHERE deptno = 20
4 AND dname = 'New dep';
```

Вставка строк с NULL значениями

- Неявный способ: просто пропустить столбцы с NULL-значениями

```
INSERT INTO dept (deptno, dname)
VALUES (30, 'Purchasing');
```

```
1 rows inserted
```

- Явный способ: вместо значений подставить константу NULL

```
INSERT INTO dept
VALUES (100, 'Finance', NULL);
```

```
1 rows inserted
```

Insert into ...

- Если в новой записи указаны значения всех атрибутов перечень атрибутов можно не указывать

```
SQL> INSERT INTO DEPT (deptno, dname, loc)
      2                VALUES (1, 'New dep', null)
```

=

```
SQL> INSERT INTO DEPT
      2                VALUES (1, 'New dep', null)
```

Вставка специальных значений

- Функция `SYSDATE` содержит текущую дату

```
INSERT INTO empl(empno, ename,  
                hire_date, mng, depno)  
VALUES          (113, 'Louis',  
                SYSDATE, 205, 110);
```

```
1 rows inserted
```


Вставка определенных значений времени

```
INSERT INTO emp1 (empno, ename,  
                 hire_date, mng, depno)  
VALUES          (113, 'Louis',  
               TO_DATE('02 19, 2013', 'MM DD, YYYY'),  
               205, 110);
```

```
1 rows inserted
```

Вставка значений из запроса

- Команда вставки значений из запроса имеет синтаксис:

```
INSERT INTO table [(column [, column...])  
select-query
```

Таким образом можно вставить несколько строк

Копирование строк из другой таблицы

- Пример:

```
INSERT INTO sales_reps(id, name, salary, comm)
  SELECT empno, ename, sal, comm
  FROM emp
  WHERE job LIKE 'SAIL%';
```

4 rows inserted

- Помните:
 - При такой записи не используется предложение `VALUES`
 - Количество и тип столбцов таблицы, в которую вставляет значения `INSERT` должно совпадать с столбцами подзапроса

UPDATE

- Для обновления существующих значений используется предложение UPDATE:

```
UPDATE    table
SET       column = value [, column = value, ...]
[WHERE    condition];
```

- UPDATE может изменять несколько строк за один раз (если это требуется).

UPDATE

- Для того чтобы указать, какие строки нужно обновить используется условие `WHERE`:

```
UPDATE emp
SET     depno = 50
WHERE  empno = 113;
1 rows updated
```

- Если не указать условие `WHERE` – обновятся значения всех строк:

```
UPDATE     emp
SET        depno = 50;
22 rows updated
```

- Указание `SET column_name= NULL` устанавливает значение атрибута в `NULL`.

В чем разница запросов?

```
UPDATE emp
SET     depno = 50
WHERE  empno = 113;
```

```
UPDATE      emp
SET         depno = 50;
```

Update

- Установить всем сотрудникам премию в размере 12% старой зарплаты и добавить к зарплате 15%

```
UPDATE emp
SET sal=sal+sal*0.15, comm = sal*0.12
```

Использование подзапросов при обновлении

- Установить служащему 113 должность и зарплату служащего 205.

```
UPDATE emp
SET job = (SELECT job
           FROM emp
           WHERE empno = 205),
      sal = (SELECT sal
           FROM emp
           WHERE empno = 205)
WHERE empno = 113;
```

```
1 rows updated
```


Обновление строк на основе других таблиц

```
UPDATE copy_emp
SET depno = (SELECT depno
             FROM emp
             WHERE empno = 100)
WHERE job = (SELECT job
             FROM emp
             WHERE empno = 200);
```

```
1 rows updated
```

DELETE

- Предложение `DELETE` удаляет строки из таблицы:

```
DELETE [FROM] table  
[WHERE condition];
```

Удаление строк из таблицы

- Для определения, какие строки удалятся используйте WHERE:

```
DELETE FROM dept
WHERE  dname = 'Finance';
1 rows deleted
```

- Если не указать условия удаления удалятся все строки:

```
DELETE FROM  copy_emp;
22 rows deleted
```

Условие удаления может содержать под-запрос

```
DELETE FROM empl
WHERE deptno IN
           (SELECT deptno
            FROM dept
            WHERE dname LIKE '%Public%');
```

```
1 rows deleted
```

Delete vs Drop

- В чем разница между:

```
SQL> drop table departments;  
SQL> delete from departments;
```

TRUNCATE

- Очищает таблицу
- Относится к data definition language (DDL), а не DML
- Синтаксис:

```
TRUNCATE TABLE table_name;
```

- Пример:

```
TRUNCATE TABLE emp;
```

Q&A

Thank You

