



Основные типы SQL запросов

Типы запросов

Есть четыре основных типа запросов данных в SQL, которые относятся к **языку манипулирования данными** (Data Manipulation Language или DML):

- 1) **SELECT** – выбрать строки из таблиц;
- 2) **INSERT** – добавить строки в таблицу;
- 3) **UPDATE** – изменить строки в таблице;
- 4) **DELETE** – удалить строки в таблице;

Каждый из этих запросов имеет различные операторы и функции, которые используются для того, чтобы произвести какие-то действия с данными.



SELECT

SELECT *select_list* [**INTO** *new_table*]
[**FROM** *table_source*] [**WHERE** *search_condition*]
[**GROUP BY** *group_by_expression*]
[**ORDER BY** *order_expression* [**ASC** | **DESC**]]

SELECT

- `SELECT * FROM table_name;`
- `SELECT column1, column2 FROM table_name;`
- `SELECT * FROM table_name WHERE column1=3;`
- `SELECT * FROM table_name WHERE column1 = 'abc';`

INSERT

- INSERT INTO *table_name*
VALUES (*value1,value2,value3,...*);
- INSERT
INTO *table_name* (*column1,column2,column3,...*)
VALUES (*value1,value2,value3,...*);

UPDATE

- *UPDATE table_name
SET column1=value1,column2=value2,...
WHERE some_column=some_value;*

DELETE

- DELETE FROM *table_name*
WHERE *some_column=some_value*;



Пакеты

Что такое пакет?

- Пакет по своей сути представляет собой именованный раздел объявлений
- В него могут входить, различные объявления:
 1. **процедуры**
 2. **функции**
 3. **типы**
 4. **курсоры**

Из чего состоит?

- Каждый пакет состоит из двух частей:
 1. описание (заголовок)
 2. тело

Описание (заголовков) пакета

CREATE OR REPLACE PACKAGE имя_модуля {IS AS}

описание_процедуры |

описание_функции |

объявление_переменной |

определение_типа |

объявление_исключительной_ситуации |

объявление_курсора |

END [имя_модуля];

Тело пакета

```
CREATE OR REPLACE PACKAGE BODY имя_модуля {IS AS}  
    код_инициализации_процедуры |  
    код_инициализации_функции |  
END [имя_модуля];
```

Замечания

- Тело пакета не является обязательной частью. Если, к примеру, заголовок модуля содержит описание только нескольких типов и курсоров, то создавать тело модуля нет необходимости. Такой способ целесообразен при объявлении глобальных переменных
- При использовании пакетов, производительность системы увеличивается



Триггеры

Определения

Триггер – это процедура, которая автоматически запускается при возникновении определенных событий.

Событие триггера – событие, управляющее запуском триггера; описывается в виде логических условий.

В Oracle различают следующие типы триггеров:

- **простые триггеры** – они привязаны к определённой таблице, срабатывают при поступлении команд DML и выполняются в рамках этой команды;
- **триггеры `INSTEAD OF`** – они привязаны к определённой таблице, выполняются вместо события триггера, которое является командой DML;
- **триггеры для событий уровня схемы** – они срабатывают при выполнении команд DDL и при наступлении таких событий, как подключение и отключение от базы данных, а также возникновение серверной ошибки.

Назначение триггеров

- Проверка ограничений целостности.
- Автоматизация обработки данных.
- Логирование действий пользователей.
- Установка начальных значений при добавлении данных в таблицы.
- Проверка прав доступа.

Синтаксис создания обычного триггера

```
CREATE [OR REPLACE] TRIGGER <имя триггера>
  { BEFORE | AFTER }
  { INSERT | DELETE | UPDATE [ OF column_commalist ] }
  ON <имя таблицы>
  [ REFERENCING old_or_new_values_alias_list ]
  [ FOR EACH { ROW | STATEMENT } ]
  [ WHEN <условие> ]
[ DECLARE
  -- описание переменных, констант и др. элементов программы
]
BEGIN
  -- программа на процедурном языке (PL/SQL)
  [ EXCEPTION
    -- обработка исключительных ситуаций
  ]
END;
/
```

Основные параметры простого триггера

- INSERT | DELETE | UPDATE [of column] – событие триггера.
Событием триггера может быть одна команда или любая комбинация указанных команд.
- BEFORE | AFTER – время срабатывания триггера: перед выполнением события триггера или после него.
Ограничения целостности проверяются во время выполнения события триггера.
- ON <имя таблицы> – таблица, к которой привязан триггер.
- FOR EACH { ROW | STATEMENT } – область действия триггера (для каждой строки или для команды).
- WHEN <условие> – условие срабатывания триггера.
Если оно не выполняется, триггер не будет запущен.

Запуск и выполнение триггеров

Триггер запускается **автоматически** при наступлении события триггера.

Если команда инициирует выполнение более чем одного триггера, то, в зависимости от типов, они выполняются в таком порядке:

1. **Перед началом выполнения команды** (Before-statement trigger)
2. **Перед обработкой записи** (Before-row trigger)
3. **После обработки записи** (After-row trigger)
4. **После окончания выполнения команды** (After-statement trigger)

Триггер INSTEAD OF

- **Особенность выполнения:** триггеры INSTEAD OF выполняются ВМЕСТО тех команд, которые являются событием триггера.
- **Назначение:** обычно триггеры INSTEAD OF применяются для обновления представлений, которые не являются обновляемыми.
- **Ограничения триггеров INSTEAD OF:**
 - нельзя указывать тип BEFORE / AFTER;

Синтаксис триггеров INSTEAD OF

```
CREATE [OR REPLACE] TRIGGER <имя триггера>
  INSTEAD OF { INSERT | DELETE | UPDATE }
  ON { <имя представления> | <имя объектного представления> }
  [ REFERENCEING old_or_new_values_alias_list ]
  [ FOR EACH ROW ]
[DECLARE
-- описание переменных, констант и др. элементов программы
]
BEGIN
  -- программа на процедурном языке (PL/SQL)
  [EXCEPTION
  -- обработка исключительных ситуаций
  ]
END;
/
```

Пример триггера INSTEAD OF. Исходные данные

-- Таблица «Отделы»

```
create table DEPART (  
    did    number(3) primary key,      -- номер отдела  
    dname  varchar2(100) not null);    -- название
```

-- Таблица «Должности»

```
create table POSTS (  
    post  varchar2(50) primary key,    -- название должности  
    sal   number(8,2) default 10000;   -- оклад
```

-- Таблица «Сотрудники»

```
create table EMP (  
    id     number(6) primary key,      -- идентификатор сотрудника  
    name   varchar2(60) not null,      -- ФИО сотрудника  
    did    number(3) references depart, -- номер отдела  
    post   varchar2(50) references posts, -- должность  
    exp    number(4,2) default 1, ...); -- количество ставок (от 0.25 до 1)
```

-- Представление «Должности сотрудников»

```
create or replace view STAFF as  
    select d.did, d.dname, e.id, e.name, e.exp, p.post, p.sal  
    from depart d, emp e, posts p  
    where d.did=e.did and e.post=p.post;
```

Пример триггера INSTEAD OF. Изменение данных

```
CREATE TRIGGER staff_update
INSTEAD OF UPDATE ON staff FOR EACH ROW
BEGIN
    IF :new.id<>:old.id OR :new.did<>:old.did OR :new.name<>:old.name
        OR :new.dname<>:old.dname
        THEN raise_application_error(-20160, 'Нельзя изменять название и
            номер отдела, имя и ID сотрудника через представление STAFF');
    END IF;
    IF :new.post<>:old.post OR :new.EXP<>:old.EXP THEN
        update emp
            SET post=:new.post, EXP=:new.EXP
            where id = :old.id;
    END IF;
    IF :new.sal<>:old.sal THEN
        update posts
            SET sal = :new.sal
            where post = :old.post;
    END IF;
END;
```

Синтаксис триггеров уровня БД

```
CREATE [ OR REPLACE ] TRIGGER <имя триггера>
  { AFTER STARTUP | BEFORE SHUTDOWN |
    AFTER LOGON | BEFORE LOGOFF |
    AFTER SERVERERROR }
  ON DATABASE
  [ WHEN <условие> ]
[ DECLARE
  -- описание переменных, констант и др. элементов программы
]
BEGIN
  -- программа на процедурном языке (PL/SQL)
  [ EXCEPTION
    -- обработка исключительных ситуаций
  ]
END;
/
```


Триггеры для событий уровня схемы и БД (начиная с версии Oracle8i)

<i>Событие уровня БД</i>	<i>Описание триггера</i>
STARTUP	Срабатывает при запуске сервера БД
SHUTDOWN	Срабатывает при запуске сервера БД
SERVERERROR	Срабатывает при возникновении серверной ошибки
LOGON	Срабатывает при успешном подключении к системе клиентского приложения
LOGOFF	Срабатывает перед отключением клиентского приложения
<i>Событие уровня схемы</i>	<i>Описание триггера</i>
CREATE	Срабатывает при добавлении к схеме нового объекта командой CREATE
DROP	Срабатывает перед попыткой удалить объект командой DROP
ALTER	Срабатывает при изменении объекта командой ALTER