Триггеры

Отличие триггеров от процедур

- Синтаксис
- Выполнение
- Цели введения и возможности
- Связь с таблицами, права на создание

Триггеры - синтаксис

```
CREATE [OR ALTER] TRIGGER имя FOR таблица
    {ACTIVE|INACTIVE}
    {BEFORE|AFTER}{DELETE|INSERT|UPDATE}
    [POSITION число]
    AS
    [DECLARE локальные переменные
    . . .]
    BEGIN
    операторы
    END^
```

Триггеры

Совместимость с SQL 2003

```
CREATE TRIGGER имя
    {ACTIVE|INACTIVE}
    {BEFORE|AFTER}
    {DELETE|INSERT|UPDATE}
    [POSITION число]
    ON таблица
AS
. . .
```

- Триггер – элемент схемы БД
- Триггер связан с таблицей, поэтому добавить триггер в схему может только владелец таблицы
- При удалении таблицы, удаляются все ее триггеры

Атрибут состояния триггера

ACTIVE | INACTIVE

Перевод триггера из одного состояния в другое осуществляется командой ALTER TRIGGER.

```
alter trigger TR_DU for TABLE_1 inactive;
```

Операции, активирующие триггер

INSERT | UPDATE | DELETE

Несколько операций

```
create trigger TR_DU for TABLE_1
      before DELETE OR UPDATE
```

логические контекстные переменные для проверки

INSERTING, UPDATING, DELETING

Фазы выполнения

BEFORE | AFTER

Объединение фаз BEFORE и AFTER недопустимо, т.к. триггеры разных фаз имеют разные возможности

Контекстные переменные NEW и OLD

Содержат состояние строки, подвергающейся модификации оператором INSERT, UPDATE или DELETE (там, где это имеет смысл)

OLD - до выполнения операции модификации

NEW - после

В триггерах фазы BEFORE контекстная переменная NEW для операций INSERT и UPDATE может быть подвергнута изменению.

Именно измененное значение будет использовано для завершения выполнения операции.

Индикатор последовательности запуска

POSITION n

Количество триггеров?

Oracle -12 на таблицу

Триггеры строки и триггеры таблицы

Процесс выполнения операции

- Формирование данных и передача серверу (посылка)
- При получении посылки активируются триггеры фазы BEFORE
- Если триггеры завершились успешно, проверка ограничений (CONSTRAINTS)
- Если все ограничения проверены успешно, активируются триггеры фазы AFTER

Процесс выполнения операции

- Если триггеры завершились успешно, сохранение версии данных
- Для фиксации версии данных COMMIT
- Если на любом этапе триггеры или ограничения вызывают исключение транзакция помечается как ROLLED BACK

Использование триггеров

- для проверки и исправления данных, нарушающих ограничение целостности;
- для отмены операций модификации, противоречащих ограничениям целостности в виде требований деловых правил;
- для замены операций модификации данных в таблицах вызовом хранимых процедур, проверяющих и поддерживающих деловые правила;
- для протоколирования событий

Пример – автоматическое исправление

```
create trigger BIU_AGENT for AGENT
   before insert  or update position 0
as
begin
   NEW.NAME_AG = UPPER(NEW.NAME_AG);
end^
```

Пример – запрет ошибочных действий

```
create trigger AU_EMP for EMPLOYEE
    after update position 0
as
begin
    if (NEW.SALARY < OLD.SALARY) then
        exception ERROR_PAY;
end^
```

Отложенные ограничения

- Триггеры можно отключить
- Ограничение срабатывает только при активном триггере

Пример –автоматическая нумерация

```
create trigger BI_OPERATION for OPERATION
  before insert
as
begin
  if (NEW.ID is NULL) then
      NEW.ID=GEN_ID(GEN_OPID,1);
end^
```

Изменение другой таблицы

```
create trigger BI_OPER for OPERATION
    before insert position 0
as
begin
 if (NEW.TYPEOP='R')
  update TOVAR_WH  T
       set     T.KOL         = T.KOL - NEW.KOL
   where     T.ID_WH       = NEW.ID_WH and
                T.ID_TOVAR = NEW.ID_TOVAR;
end^
```

Если такого остатка нет

```
if (NEW.TYPEOP='R')
 begin
  id=null;
  id = (select T.ID from TOVAR_WH  T
          where       T.ID_WH   = NEW.ID_WH and
                      T.ID_TOVAR = NEW.ID_TOVAR);
  if (id IS NOT NULL)
  begin
      update TOVAR_WH  T
      set       T.KOL   = T.KOL - NEW.KOL
      where   T.ID      = :id
  end
  else exception ' нет товара на складе';
end
```

Журнализация событий

```
create trigger AI_TOVAR for TOVAR
ACTIVE AFTER INSERT
POSITION 1
as
declare variable tid integer;
begin
  tid = gen_id(log_table_gen,1);
  insert into log_table (id,
                         operation, date_time, user_name)
  values (:tid, ‘Добавлен новый товар', 'NOW',
                              user);
end^
```

Триггеры базы данных

- Начиная с версии 2.1
- События базы данных
  - CONNECT
  - DISCONNECT
  - TRANSACTION START
  - TRANSACTION COMMIT
  - TRANSACTION ROLLBACK
- Права – только SYSDBA или владелец базы

```
CREATE TRIGGER  имя
[ACTIVE | INACTIVE]
ON событиеБД
[POSITION number]
AS
[<declarations>]
BEGIN
   [<statements>]
END
```

Пример

```
create trigger tr_connect active
on connect
as
begin
    insert into dblog
            values (current_user,
                    current_timestamp,
                    'connect');
end
```

Достоинства и недостатки

- ?