СУБД

Триггеры Oracle

Лекция 11

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры уровня DML

Триггер уровня DML является объектом реляционной базы данных (специальный вид хранимой процедуры), который активизирует выполнение хранимой (или встроенной) PL/SQL-процедуры при изменении пользователем данных в таблице. Событие, управляющее запуском триггера, описывается в виде логических условий. Например, попытка модифицировать данные в таблице активизирует триггер, соответствующий данной команде манипулирования данными. Число триггеров на таблицу базы данных не ограничено.

Обычно триггеры используют для реализации ограничений ссылочной целостности, для предотвращения несогласованных изменений в базе данных (поддержка целостности базы данных), для выполнения скрытых операций при модификации, а также для снижения сетевого трафика за счет передачи обработки на сервер. Операции завершения транзакции выполняются после обработки триггеров.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры уровня DML

Триггер уровня DML является объектом реляционной базы данных (специальный вид хранимой процедуры), который активизирует выполнение хранимой (или встроенной) PL/SQL-процедуры при изменении пользователем данных в таблице. Событие, управляющее запуском триггера, описывается в виде логических условий. Например, попытка модифицировать данные в таблице активизирует триггер, соответствующий данной команде манипулирования данными. Число триггеров на таблицу базы данных не ограничено.

Oracle поддерживает три вида триггеров: предваряющие (**BEFORE**), замещающие (**INSTEAD OF**) и завершающие (**AFTER**). Как и логично было бы ожидать, предваряющие триггеры вызываются перед обработкой запроса на вставку, обновление или удаление, замещающие - вместо него, а завершающие - после обработки запроса. Всего имеется девять возможных типов триггеров: предваряющий триггер вставки, обновления и удаления, замещающий триггер вставки, обновления и удаления и завершающий триггер вставки, обновления и удаления.

Обычно триггеры используют для реализации ограничений ссылочной целостности, для предотвращения несогласованных изменений в базе данных (поддержка целостности базы данных), для выполнения скрытых операций при модификации, а также для снижения сетевого трафика за счет передачи обработки на сервер. Операции завершения транзакции выполняются после обработки триггеров.

E-mail: helen.matat@infostroy.com.ua

Триггеры уровня DML

При выполнении команды UPDATE с помощью триггера можно проверить, что модифицируемые данные удовлетворяют ограничениям целостности базы данных до выполнения операции (при этом возможен доступ к новым данным!). После выполнения операции с помощью триггера можно выполнить скрытую обработку данных с учетом поступивших изменений (старые данные также могут быть доступны).

При выполнении команды INSERT также можно проверить данные до вставки в таблицу на допустимость ограничениям целостности, а после - выполнить операции над только что вставленными данными.

При выполнении команды DELETE можно проверить данные до их удаления или восстановить данные после удаления.

.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Синтаксис команды CREATE TRIGGER

Для создания триггера предусмотрена специальная команда SQL CREATE TPIGGER. Эта команда создает триггер на таблице, которой владеет пользователь.

```
CREATE [OR REPLACE] TRIGGER [имя схемы.]имя триггера
{BEFORE|AFTER}
{INSERT|DELETE|UPDATE [OF имя колонки [, имя колонки s]]}
[OR {INSERT|DELETE|UPDATE [OF имя колонки [, имя колонки s]]}]
ON [имя схемы.]имя таблицы
[FOR EACH ROW]
[WHEN условие]
BEGIN
…
END
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Синтаксис команды CREATE TRIGGER

Ключевое слово OR REPLACE указывает на безусловное замещение старого теста триггера. Если оно не указывается, а триггер определен в базе данных, то замещения старого триггера не происходит, и возвращается сообщение об ошибке.

Определение триггера состоит из нескольких частей:

* задание имени триггера;
* указание команды SQL, к которой относится триггер;
* указание таблицы или представления, для которой определяется триггер;
* задание ограничений триггера;
* задание действия в теле триггера.

Если имя схемы опущено, то триггер создается в схеме текущего пользователя.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Синтаксис команды CREATE TRIGGER

{BEFORE|AFTER} - время действия триггера- до или после выполнения команды манипулирования данными. Нельзя определить два триггера на одну и ту же операцию с одинаковым временем действия.

При создании триггера необходимо указывать, к какой команде манипулирования данными он относится - INSERT, DELETE или UPDATE. Для последней можно указывать конкретные колонки, указав фразу OF имя_колонки [, имя_колонки ...] в предложении UPDATE.

Ключевое слово ON задает имя таблицы или представления, для которого создается триггер.

Необязательное ключевое слово ON EACH ROW определяет триггер как строчный, т.е. запускаемый для каждой строки результирующего множества команды SQL. Если оно опущено, то триггер запускается только один раз в начале обработки команды. Таким образом, условие "для каждой строки" активизируется, только когда есть строки (например, предложение WHERE дает истинное значение условий поиска), в то время как для условия "для каждой команды" триггер сработает и в этом случае.

Дополнительные условия, сужающие область действия триггера, могут быть заданы в предложении WHEN. Условия, задаваемые в этом предложении, являются стандартными для SQL условиями, должны содержать корреляционные имена и не могут содержать запрос. Это предложение может быть указано только для строчного триггера.

Участие в транзакциях

По умолчанию триггеры DML принимают участие в транзакциях, из которых они запускаются, т.е.:

- Если триггер инициирует исключение, то соответствующая часть транзакции будет отменена.
- Если триггер сам выполняет какие-то операторы DML (например, вставляет запись в журнальную таблицу), то такие операторы DML становятся частью главной транзакции.
- Внутри триггера DML нельзя использовать операторы COMMIT и ROLLBACK.

Если использовать триггер DML как автономную транзакцию, то любые команды DML, исполняемые внутри триггера, будут сохранены или отменены посредством явно использованного оператора COMMIT или ROLLBACK, при этом главная транзакция затрагиваться не будет

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Псевдозаписи NEW и OLD

При запуске триггера уровня строки исполняющее ядро PL/SQL создаёт и заполняет две структуры данных, которые работают подобно записям. Это ***псевдозаписи NEW и OLD*** (приставка «псевдо» объясняется тем, что они обладают не всеми свойствами настоящих записей PL/SQL).

OLD хранит начальные значения записи, обрабатываемой триггером, а

NEW — новые значения. Эти записи имеют такую же структуру, как запись, объявленная при помощи атрибута %ROWTYPE на основе таблицы, к которой относится триггер.

Правила при работе с псевдозаписями NEW и OLD :

- Для триггеров, относящихся к командам INSERT, структура OLD не содержит никаких данных, «старого» набора значений не существует.
- Для триггеров, относящихся к командам DELETE, структура NEW не содержит никаких данных, т.к. речь идёт об удалении записи.
- Изменение значений полей структуры OLD запрещено, попытка такого изменения приведёт к возникновению ошибки ORA-04085. Изменение значений полей структуры NEW допустимо.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Псевдозаписи NEW и OLD

- Структура NEW Или OLD не может передаваться как параметр типа запись в процедуру или функцию, вызываемую внутри триггера. Можно передавать только отдельные поля псевдозаписей. gentrigrec.sp - программа, формирующая код, который передаёт значения NEW и OLD записям, которые уже могут передаваться как параметры.
- При ссылке на структуру NEW или OLD внутри анонимного блока триггера необходимо предварять двоеточием соответствующие ключевые слова, например:

```
IF  :NEW.salary > 10000 THEN ...
```

- Выполнение операций уровня записи для структур NEW и OLD не поддерживается. Например, подобный код вызовет ошибку при компиляции триггера:

```
BEGIN  :NEW  :=  NULL; END;
```

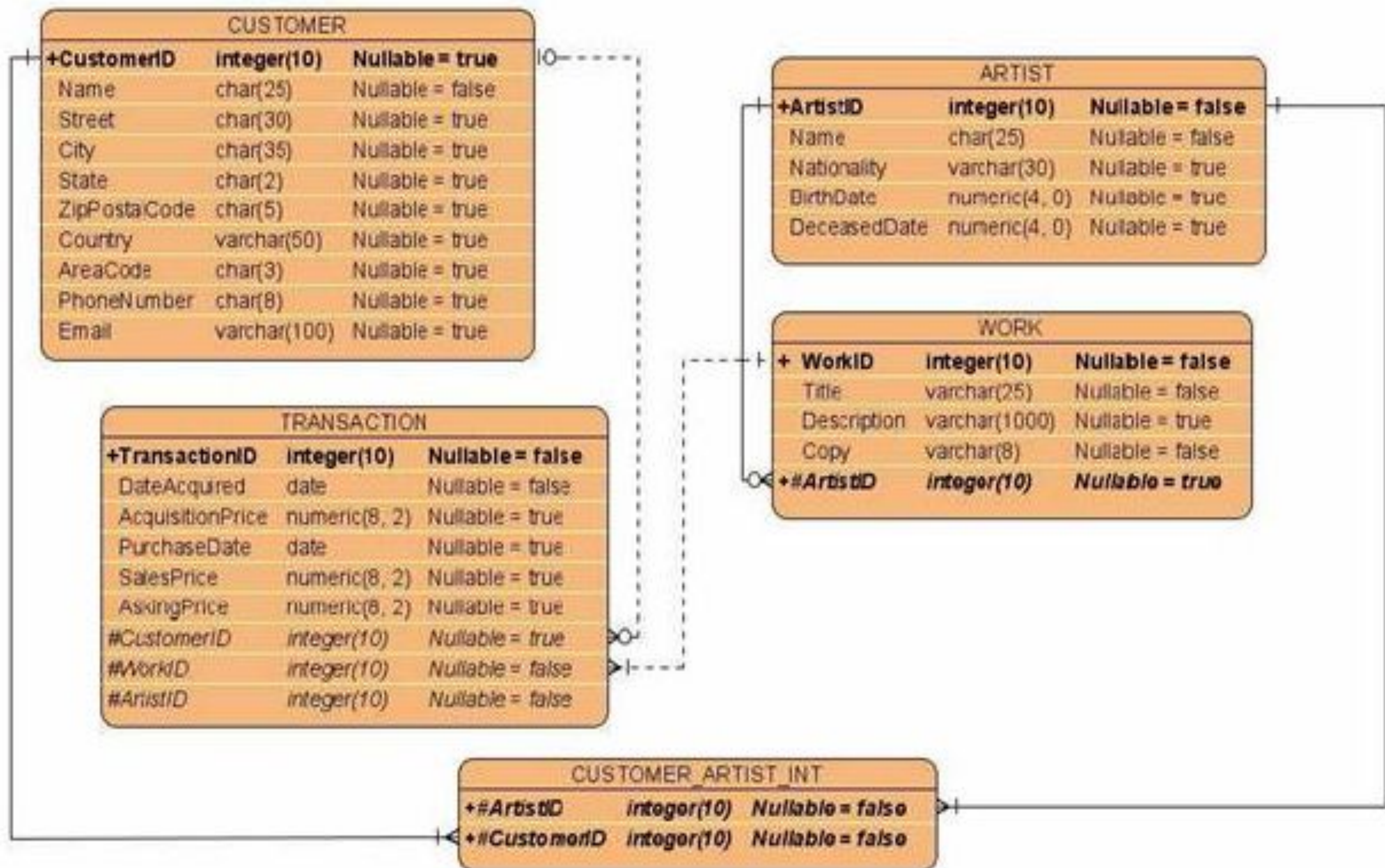
Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Псевдозаписи NEW и OLD

Активизирующий оператор	:OLD	:NEW
INSERT	Не определена, во всех полях содержится NULL значения	Значения, которые будут введены после выполнения оператора.
UPDATE	Исходные значения, содержащиеся в строке перед обновлением данных	Новые значения, которые будут введены после выполнения оператора
DELETE	Исходные значения, содержащиеся в строке перед ее удалением	Не определена, во всех полях содержится NULL значения

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример создания триггера



Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 1 создания триггера

Использование триггеров для проверки допустимости вводимых данных

Предположим, у галереи (см. описание практического примера в гл. 1) есть правило, что ни одна работа не может быть продана менее, чем за 90% от запрошенной цены. Чтобы обеспечить выполнение этого правила, можно написать триггер обновления для таблицы TRANSACTION, сравнивающий значения AskingPrice и SalesPrice. Если правило нарушается, в столбец AskingPrice ставится исходное значение.

Можно использовать две стратегии. Одна заключается в том, чтобы написать предваряющий триггер, который проверяет и переустанавливает, если необходимо, значение столбца SalesPrice до выполнения обновления. Вторая стратегия — написать завершающий триггер, проверяющий и переписывающий строку таблицы TRANSACTION после обновления.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 1 создания триггера

```
CREATE OR REPLACE TRIGGER TRANS_SalesPriceCheck
BEFORE UPDATE ON TRANSACTION
FOR EACH ROW
BEGIN
IF :new.SalesPrice < 0.9 * :old.AskingPrice THEN
UPDATE TRANSACTION
SET
SalesPrice = :old.AskingPrice,
AskingPrice = :old.AskingPrice;
END IF;
END;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 1 создания триггера

Логика работы триггера :

Если новая продажная цена составляет менее 90% от запрашиваемой цены, продажная цена устанавливается равной запрашиваемой цене.

Новая продажная цена сравнивается со старой запрашиваемой ценой; в противном случае можно было бы, изменив обе цены, успешно совершить обновление, нарушающее данное ограничение. На тот случай, если именно так и произошло, столбец AskingPrice в операторе UPDATE устанавливается равным :old. AskingPrice.

Этот триггер будет вызываться рекурсивно.

Оператор UPDATE в триггере вызовет обновление таблицы TRANSACTION, что, в свою очередь, приведет к повторному вызову триг гера. На этот раз, однако, столбец SalesPrice будет равен :old.AskingPrice, поэтому новых обновлений произведено не будет и рекурсия остановится.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 2 создания триггера

Использование триггеров для присвоения значений по умолчанию

Столбцам таблицы могут присваиваться значения по умолчанию с по мощью квалификатора DEFAULT. В качестве таких значений можно за давать константы или результаты вычисления простых выражений. Если же задание значения по умолчанию требует более сложной логики, необходимо использовать триггер.

Предположим, что у галереи имеется правило, согласно которому запрашиваемая цена произведения устанавливается равной удвоенной стоимости его приобретения или сумме общей стоимости приобретения и чистой выручки от продажи этого произведения в прошлом. Это правило реализуется с помощью завершающего триггера.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 2 создания триггера

Представление, которое используется в рассматриваемом триггере, имеет следующий вид:

```
CREATE VIEW ArtistWorkNet AS
SELECT W.WorkID, Name, Title, Copy, AcquisitionPrice,
SalesPrice, (SalesPrice - AcqisitionPrice) AS NetPrice
FROM TRANSACTION T
JOIN WORK W
ON T.WorkID = W.WorkID
JOIN ARTIST A
ON W.ArtistID = A.ArtistID;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 2 создания триггера

```
CREATE OR REPLACE TRIGGER SetAskingPrice BEFORE INSERT ON TRANSACTION
FOR EACH ROW
DECLARE
avgNetPrice numeric(8,2); newPrice numeric(8,2); rowcount integer; BEGIN
SELECT Count(*) INTO rowcount
FROM TRANSACTION
WHERE WorkID = :new.WorkID;
IF rowcount = 0 THEN
:new.AskingPrice := 2*(:new.AcquisitionPrice);
ELSE
SELECT AVG(NetPrice) INTO avgNetPrice FROM ArtistWorkNet AW WHERE AW.WorkID =
:new.WorkID GROUP BY AW.WorkID;
newPrice := avgNetPrice + :new.AcquisitionPrice; IF newPrice > 2*(:new.AcquisitionPrice)
THEN :new.AskingPrice := newPrice;
ELSE
:new.AskingPrice := 2*(:new.AcquisitionPrice); END IF;
END IF;
END;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 2 создания триггера

Триггер сначала подсчитывает количество строк в таблице TRANSACTION, в которых значение WorkID равно :new.WorkID. Поскольку это предваряющий триггер, произведение еще не добавлено в базу данных, и количество будет равным нулю, если это произведение не появлялось в галерее ранее. В этом случае :new.AskingPrice устанавливается равным удвоенному значению AcquisitionPrice.

Если произведение появлялось в галерее в прошлом, рассчитывается средняя чистая прибыль от его продажи с помощью представления ArtistWorkNet. После этого вычисляется переменная newPrice как сумма средней чистой прибыли и стоимости приобретения. Наконец, :new.AskingPrice присваивается большее из двух значений — newPrice или удвоенное значение AcquisitionPrice. Так как триггер предваряющий, для усреднения можно использовать встроенную функцию AVG: новая строка еще не добавлена в таблицу WORK, поэтому она не будет учтена при расчете среднего значения.

Если в какой-либо из строк представления ArtistWorkNet столбец SalesPrice или AcquisitionPrice является пустым, это может вызвать проблемы при вычислениях в триггере.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 3 создания триггера

Триггер, обновляющий представление

Обновление представлений в ряде случаев может оказаться затрудни тельным. Одна из таких проблем касается представлений, созданных при помощи операции соединения.

Рассмотрим представление CustomerInterests:

```
CREATE VIEW CustomerInterests AS
SELECT C.Name AS Customer, A.Name AS Artist
FROM CUSTOMER C
JOIN CUSTOMER_ARTIST_INT C1
ON C.CustomerID = C1.CustomerID JOIN ARTIST A
ON C1.ArtistID = A.ArtistID;
```

Оно содержит строки таблиц CUSTOMER и ARTIST, соединенные через таблицу пересечения. Столбцу CUSTOMER.Name дан псевдоним Customer, а столбцу ARTIST.Name — Artist.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример 3 создания триггера

Запрос на изменение имени клиента в представлении CustomerInterests можно интерпретировать как запрос на изменение столбца Name в таблице CUSTOMER. Такой запрос может быть обработан лишь в том случае, если это имя является уникальным в таблице CUSTOMER. В противном случае невозможно будет определить, какую из строк следует обновлять.

Замещающий триггер обновляет имя клиента, если это имя является уникальным в базе данных. Вместо того чтобы подсчитывать количество строк с данным именем клиента и выполнять обновление только в том случае, если такая строка всего одна, триггер обусловливает обновление ключевым словом NOT EXISTS. Такая конструкция триггера позволяет Oracle оптимизировать SQL-оператор и приводит к лучшей производительности.

```
CREATE OR REPLACE TRIGGER CustomerInterestsJJpdate INSTEAD OF UPDATE ON CustomerInterests
FOR EACH ROW
BEGIN
UPDATE CUSTOMER C1 SET C1.Name = :new.Customer WHERE C1.Name = :old.Customer AND NOT EXISTS (SELECT * FROM CUSTOMER C2 WHERE C2.Name = C1.Name
AND C2.CustomerID <> C1.CustomerID);
END;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Определение DML-действия внутри триггера

Oracle предлагает набор функций (называемых также ***операционными директивами***), которые позволяют определить, какое DML-действие вызвало запуск текущего триггера. Каждая такая функция возвращает TRUE или FALSE.

INSERTING Возвращает TRUE, если триггер был запущен в ответ на вставку в таблицу, с которой связан триггер, и FALSE — в противном случае.

UPDATING Возвращает TRUE, если триггер был запущен в ответ на обновление таблицы, с которой связан триггер, и FALSE — в противном случае.

DELETING Возвращает TRUE, если триггер был запущен в ответ на удаление из таблицы, с которой связан триггер, и FALSE — в противном случае.

Используя эти директивы, можно создаь один общий триггер, который будет объединять действия, необходимые для различных видов операций.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры DDL

Oracle позволяет определить триггеры, которые будут запускаться в ответ на исполнение операторов языка ***DDL*** (Data Definition Language — язык определения данных). Оператор DDL — это любой оператор SQL, используемый для создания или изменения объекта базы данных, такого как таблица или индекс.

Каждый из этих операторов приводит к созданию, изменению или удалению объекта базы данных.

Несколько примеров операторов DDL:

CREATE TABLE

ALTER INDEX

DROP TRIGGER

Синтаксис создания таких триггеров практически совпадает с синтаксисом создания триггеров DML, отличие лишь в запускающих их событиях и в том, что триггеры DDL не применяются к отдельным таблицам.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Создание триггера DDL

Для создания (или пересоздания) триггера DDL используйте такую конструкцию:

```
1 CREATE [ OR REPLACE ] TRIGGER имя_триггера
2 { BEFORE | AFTER } {DDL-событие } ON { DATABASE | SCHEMA }
3 [ WHEN ( ... ) }
4 DECLARE
5 Объявления переменных
6 BEGIN
7 ... код ...
8 END;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

События триггера DDL

ALTER Specify ALTER to fire the trigger whenever an ALTER statement modifies a database object in the data dictionary.

Restriction on Triggers on ALTER Operations The trigger will not be fired by an ALTER DATABASE statement.

ANALYZE Specify ANALYZE to fire the trigger whenever the database collects or deletes statistics or validates the structure of a database object.

ANALYZE for information on various ways of collecting statistics

ASSOCIATE STATISTICS Specify ASSOCIATE STATISTICS to fire the trigger whenever the database associates a statistics type with a database object.

AUDIT Specify AUDIT to fire the trigger whenever the database tracks the occurrence of a SQL statement or tracks operations on a schema object.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

События триггера DDL

COMMENT Specify COMMENT to fire the trigger whenever a comment on a database object is added to the data dictionary.

CREATE Specify CREATE to fire the trigger whenever a CREATE statement adds a new database object to the data dictionary.

Restriction on Triggers on CREATE Operations The trigger will not be fired by a CREATE DATABASE or CREATE CONTROLFILE statement.

DISASSOCIATE STATISTICS Specify DISASSOCIATE STATISTICS to fire the trigger whenever the database disassociates a statistics type from a database object.

DROP Specify DROP to fire the trigger whenever a DROP statement removes a database object from the data dictionary.

GRANT Specify GRANT to fire the trigger whenever a user grants system privileges or roles or object privileges to another user or to a role.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

События триггера DDL

NOAUDIT Specify NOAUDIT to fire the trigger whenever a NOAUDIT statement instructs the database to stop tracking a SQL statement or operations on a schema object.

RENAME Specify RENAME to fire the trigger whenever a RENAME statement changes the name of a database object.

REVOKE Specify REVOKE to fire the trigger whenever a REVOKE statement removes system privileges or roles or object privileges from a user or role.

TRUNCATE Specify TRUNCATE to fire the trigger whenever a TRUNCATE statement removes the rows from a table or cluster and resets its storage characteristics.

DDL Specify DDL to fire the trigger whenever any of the preceding DDL statements is issued.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Пример триггера DDL

Пример триггера, выполняющего роль информатора, объявляющего о создании всех объектов:

```
SQL> CREATE OR REPLACE TRIGGER  TRDDL_INF
  2   AFTER CREATE ON SCHEMA
  3   BEGIN
  4      DBMS_OUTPUT.PUT_LINE('I believe you have created something!');
  5   END
  6   /

  TRIGGER created.
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры событий базы данных

Триггеры событий базы данных запускаются при возникновении событий на уровне базы данных.

Существует **пять триггеров событий базы данных**:

Событие	Разрешенное время выполнения	Описание
STARTUP	AFTER	Активизируется после запуска экземпляра
SHUTDOWN	BEFORE	Активизируется при остановке экземпляра. Для заметки, это событие не активизирует триггер, если останов БД аварийный!
SERVERERROR	AFTER	Активизируется при возникновении ошибки ORACLE.
LOGON	AFTER	Активизируется после успешного соединения пользователя с базой данных.
LOGOOFF	BEFORE	Активизируется в начале отключения пользователя.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Создание триггера события базы данных

Синтаксис, используемый для создания такого триггера, очень похож на синтаксис создания триггера DDL:

```
1 CREATE [ OR REPLACE ] TRIGGER имя_триггера
2 { BEFORE | AFTER } { событие_базы_данных } ON { DATABASE | SCHEMA }
3 DECLARE
4 Объявление переменных
5 BEGIN
6 ... код ...
7 END;
```

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры событий базы данных

Эти триггеры представляют собой великолепное средство автоматизации процесса администрирования базы данных и обеспечения детального контроля над базой данных.

Существует ряд ***ограничений,*** накладываемых на использование атрибутов BEFORE и AFTER для определённых событий. Некоторые ситуации представляются просто бессмысленными:

Не бывает триггеров BEFORE STARTUP. Попытка создания такого триггера приводит к появлению сообщения об ошибке:

ORA-30500: database open triggers and server error triggers cannot have BEFORE type

Не бывает триггеров AFTER SHUTDOWN. Попытка создания такого триггера приводит к появлению сообщения об ошибке:

ORA-30501: instance shutdown triggers cannot have AFTER type

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua

Триггеры событий базы данных

Не бывает триггеров BEFORE LOGON. Попытка создания такого триггера приводит к появлению сообщения об ошибке:

ORA-30508: client logon triggers cannot have BEFORE type

Не бывает триггеров AFTER LOGOFF. Попытка создания такого триггера приводит к появлению сообщения об ошибке:

ORA-30509: client logoff triggers cannot have AFTER type

Не бывает триггеров BEFORE SERVERERROR. Попытка создания такого триггера приводит к появлению сообщения об ошибке:

ORA-30500: database open triggers and server error triggers cannot have BEFORE type.

Доцент Кафедри Інформатики ХНУРЕ, начальник Відділу Контроля Якості ПЗ INFOSTROY LTD Матат О.О.
E-mail: helen.matat@infostroy.com.ua