

PL/SQL

Triggers



**DE HOGESCHOOL
MET HET NETWERK**

Elfde-Liniestraat 24, 3500 Hasselt, www.pxl.be



What Are Triggers?

- A trigger is a PL/SQL block that is stored in the database and fired (executed) in response to a specified event.
- The Oracle database automatically executes a trigger when specified conditions occur.

Database triggers

- PL/SQL-code geassocieerd aan een tabel/view
- wordt als object opgeslagen in de DB
- wordt automatisch uitgevoerd (kan niet opgeroepen worden)



bij een event (DML-commando)

INSERT
UPDATE
DELETE

- !!!!! het maakt niet uit vanuit welke omgeving of door wie de gegevens worden gewijzigd!!!!

Syntax voor de creatie van een database trigger

CREATE OR REPLACE TRIGGER *triggernaam*

{BEFORE | AFTER}

→ ***timing***

{DELETE | INSERT | UPDATE [OF *kolom* [,*kolom*]] } [OR ...]

→ ***event***

ON *tabelnaam*

[FOR EACH ROW [WHEN (*voorwaarde*)]]

→ ***frequency***

[DECLARE]

BEGIN

*/*uitvoerbare commando's*/*

[EXCEPTION]

END [*triggernaam*];

Syntax tabel trigger

- **Timing**

- ☐ BEFORE

- als de trigger moet controleren of een actie toegestaan is of niet - voorkomt onnodige rollbacks
 - als je zeker wil zijn dat deze trigger altijd afgaat (deze worden eerst uitgevoerd)

- ☐ AFTER

- als het triggerende commando zeker moet worden uitgevoerd (vooral bij row triggers)
 - wordt pas uitgevoerd nadat alle constraints op de tabel gecontroleerd zijn

- **Event**

- ☐ INSERT - UPDATE - DELETE

- **Frequency** = aantal keer trigger-body uitgevoerd wordt

- ☐ STATEMENT (default)

- 1x per statement / instructie
 - onafhankelijk van het aantal beïnvloede rijen

- ☐ ROW

- 1x per rij die beïnvloed wordt door het event

Statement trigger: voorbeeld

```
CREATE OR REPLACE TRIGGER bds_emp
  BEFORE DELETE
  ON employees
BEGIN
  IF USER != 'JAN' THEN
    RAISE_APPLICATION_ERROR(-20000,
      'u heeft geen rechten voor deze actie');
  END IF;
END;
/
```

```
SQL> delete employees
      2  where employee_id = 206;
delete employees
      *
```

ERROR at line 1:
ORA-20000: u heeft geen rechten voor deze actie
ORA-06512: at "STUDENT.BDS_EMP", line 3
ORA-04088: error during execution of trigger 'STUDENT.BDS_EMP'

Naamgeving triggers

De naamgeving geeft het soort trigger weer

vb: TRIGGER bds_emp

- b = before
de trigger gaat af vóór het DML-statement wordt uitgevoerd
- d = delete
de trigger gaat af bij een delete-statement
- s = statement trigger
deze trigger gaat slechts 1x af per DML-statement, ongeacht hoeveel rijen door het DML-commando worden bewerkt

Statement trigger: voorbeeld uitgebreid

```
CREATE OR REPLACE TRIGGER bdus_emp
  BEFORE DELETE OR UPDATE OF salary
  ON employees
BEGIN
  IF USER != 'JAN' THEN
    IF DELETING THEN
      RAISE_APPLICATION_ERROR(-20000, 'u heeft geen
verwijderrechten');
    ELSE
      RAISE_APPLICATION_ERROR(-20000, 'u heeft geen rechten om het
salary te wijzigen');
    END IF;
  END IF;
END;
```


Functies INSERTING – DELETING - UPDATING

- Te gebruiken indien er meer dan 1 triggerend event is op 1 tabel
- Deze functies geven een boolean terug
- Bij UPDATING kan ook een parameter meegegeven worden
vb. IF updating('salary') THEN

Row Trigger: voorbeeld

```
CREATE OR REPLACE TRIGGER aur_emp_salary
  AFTER UPDATE OF salary
  ON employees
  FOR EACH ROW
BEGIN
  IF (:NEW.salary - :OLD.salary > 0.1* :OLD.salary) THEN
    RAISE_APPLICATION_ERROR(-20000, 'salary te veel
                                     verhoogd' );
  END IF;
END;
```

Row triggers

- bovenaan de trigger definitie: FOR EACH ROW

- gaat per te bewerken rij af

Vb een delete-commando verwijdt 10 rijen

➤ statement-trigger gaat **1 keer** af

➤ rij-trigger gaat **10 keer** af

Vb een delete-commando verwijdt 0 rijen

➤ statement-trigger gaat **1 keer** af

➤ rij-trigger gaat **niet** af

Row triggers

- mogelijk oude en nieuwe kolomwaarden op te vragen
 - :NEW.kolomnaam
 - :OLD.kolomnaam
- :NEW.salary → bevat de nieuwe waarde voor salary na uitvoering van een INSERT of UPDATE


Let op: bij DELETE is deze variabele leeg
- :OLD.salary → bevat de oude waarde voor salary vóór uitvoering van een UPDATE of DELETE

Let op: bij INSERT is deze variabele leeg
- Enkel bij UPDATE bevatten beiden een waarde

Row Trigger uitbreiding (WHEN): voorbeeld

```
CREATE OR REPLACE TRIGGER aur_emp_sal2
  AFTER UPDATE OF salary
  ON employees
  FOR EACH ROW
  WHEN (OLD.job_id != 'AD_PRES')
BEGIN
  IF (:NEW.salary - :OLD.salary > 0.1* :OLD.salary) THEN
    RAISE_APPLICATION_ERROR(-20000, 'salary te veel
                                verhoogd');
  END IF;
END;
/
```

Enkel mogelijk voor row triggers!



LET OP : geen ':' bij 'OLD'

Volgorde van uitvoering triggers (automatisch)

Indien meerdere triggers op 1 DML-statement

1. Alle BEFORE STATEMENT triggers
2. Voor elke rij uit de ROW triggers
 - a. Alle BEFORE ROW triggers voor die rij
 - b. Triggerende DML-statement + integrity constraints checken voor die rij
 - c. Alle AFTER ROW triggers voor die rij
3. Alle AFTER STATEMENT triggers

Trigger keuze

- Gebruik een **before statement trigger** als de trigger MOET afgaan
- Gebruik eerder een **before statement-trigger** dan een after statement trigger (vooral bij controle of een actie toegestaan is, dit voorkomt rollbacks)
- Gebruik een **row trigger** als de inhoud van de kolommen nodig is
- Gebruik liever een **after row trigger** dan een before row trigger. Oracle controleert dan eerst de constraints.
- Gebruik een **before row trigger** als de inhoud van een kolom in de trigger gewijzigd wordt

Beperkingen van Database Triggers

Commando's COMMIT en ROLLBACK zijn niet toegelaten in triggers

Beheer van triggers

- **Welke triggers bestaan?**

```
SQL> SELECT object_name, created, status  
FROM user_objects  
WHERE object_type = 'TRIGGER';
```

- **Broncode opvragen:**

```
SQL> SELECT line, text  
FROM user_source  
WHERE name = 'AUR_EMP_SALARY';
```

Beheer van triggers – tabel **USER_TRIGGERS**

- SQL> SELECT trigger_type, trigger_body
FROM **user_triggers**
WHERE trigger_name = 'AUR_EMP_SALARY';
- SQL>SELECT trigger_name, trigger_type, triggering_event, table_name, status
FROM **user_triggers**;

Beheer van triggers

- **Verwijderen**
DROP TRIGGER triggernaam
- **Activeren/deactiveren van 1 bepaalde trigger**
ALTER TRIGGER triggernaam ENABLE
ALTER TRIGGER triggernaam DISABLE
- **Activeren/deactiveren van alle triggers van 1 bepaalde tabel**
ALTER TABLE tabelnaam ENABLE ALL TRIGGERS
ALTER TABLE tabelnaam DISABLE ALL TRIGGERS