DB-Anwendungsprogrammierung: Trigger

Allgemein:

- Trigger sind ECA-Regeln: Event-Condition-Action

- wenn auslösendes Event auftritt und Condition erfüllt ist, wird Action ausgeführt (ähnlich Prozeduren, die nicht explizit aufgerufen werden müssen

- Trigger nicht in SQL92 enthaelten, für SQL3 vorgesehen

- Trigger in Oracle enthalten, sehr unterschiedlich zum SQL3-Vorschlag

- Trigger heißen auch 'aktive Regeln', DB mit Triggern 'aktive DB'

- Anwendungen von Triggern: Integritätsüberwachungen, Aktualisierung abgeleiteter Daten, Protokollierung von Änderungen, Anstoßen von Geschäftsprozessen

Trigger

Auslösende Ereignisse (Event):

- Einfügen in eine bestimmte Tabelle (nicht in Views)
- Löschen aus einer bestimmten Tabelle (nicht in Views)
- Änderung in einer bestimmten Spalte
- weitere Auslöser denkbar (Datumsabhängigkeit, bei jedem `commit`, vor/nach Ausführung einer Prozedur,...), in Oracle aber nicht vorgesehen

Bedingungen (Condition): in Oracle nur sehr einfache Bedingungen, beziehen sich auf das gerade betroffene Tupel; Aktionsteil kann aber bel. komplex sein

Aktionen (Action): bel. PL/SQL-Code, aber ohne `commit/rollback`; `raise_application_error` aufrufbar -¿ SQL-Anweisung, die zu Fehler geführt hat, kann abgebrochen werden, alle (auch vom Trigger) vorgenommenen Änderungen werden rückgängig gemacht

Trigger

Beispiel:

- zum Warenhaus Relation Bestand mit Attributen Artnr, Anzahl, mindAnzahl; Relation Bestellung mit Attributen Datum und Artnr;
  Bestellungen sollen automatisch erzeugt werden, wenn akt. Bestand unter Mindestbestand sinkt

- ```
  CREATE TRIGGER Nachbestellung
  AFTER UPDATE OF Anzahl ON Bestand
  FOR EACH ROW
  WHEN (new.Anzahl < new.mindAnzahl)
  BEGIN    INSERT INTO Bestellung
  VALUES (SYSDATE, :new.Artnr);
  END;
  ```

- Alternative: Bedingung im Aktiosteil `CREATE TRIGGER`
  ```
  Nachbestellung
  AFTER UPDATE OF Anzahl ON Bestand
  FOR EACH ROW
  BEGIN
  IF :new.Anzahl < :new.mindAnzahl THEN
  INSERT INTO Bestellung
  VALUES (SYSDATE, :new.Artnr);
  END IF;
  END;
  ```
  ```

Trigger

Zeitpunkt des Triggeraufrufs:

- `BEFORE`: Trigger wird vor Ausführung der auslösenden Operation aufgerufen
- `AFTER`: Trigger wird nach Ausführung der auslösenden Operation aufgerufen, kann schon veränderten Zustand sehen

Triggerarten in Oracle:

- Befehlstrigger: wird nur einmal vor/nach Operation aufgerufen; haben keine Möglichkeit, auf die von Änderung betroffenen Tupel zuzugreifen
- Zeilentrigger (`FOR EACH ROW`): wird einmal für jedes geänderte Tupel aufgerufen; in Trigger keine Anfragen erlaubt, die sich auf andere Tupel beziehen; Zugriff auf alten Datensatz: `old`, auf neuen Datensatz: `new`, jeweils implizit definiert vom Typ relation%ROWTYPE (im PL/SQL-Block mit : zugreifen)

Trigger

Sonstiges:

- Action kann selbst DB-Änderungen enthalten -¿ kann weitere Trigger auslösen (Problem: Triggerausführung könnte nicht terminieren)

- in Oracle Trigger formulierbar, die bei verschiedenen Events 'feuern':

  ```
  CREATE TRIGGER Nachbestellung
  AFTER INSERT OR UPDATE ON Bestand
  ...
  ```

  Falls später Unterscheidung nötig: in PL/SQL mit `IF INSERTING THEN...` auf jeweilige Operation beziehen

- in sqlplus: Triggerdefinition mit einem '/' abschließen

- Syntax:

  ```
  CREATE TRIGGER Triggername BEFORE |AFTER Operation ON Relation
  [FOR EACH ROW [WHEN ( Bedingung )] ]
  [DECLARE deklaration;....]
  BEGIN anweisung;.....END;
  ```

  Syntax von `Operation`: `DELETE | INSERT | UPDATE [OF spalte1,...]`
  `[ OR DELETE | INSERT | UPDATE [OF spalte1,...]`
  `[ OR ...]]`

- Löschen von Triggern: `DROP TRIGGER Triggername;`