# Triggers

# Triggers

- A trigger is a procedure that is invoked by the DBMS as a response to a specified change
- A DB that has a set of associated triggers is referred to as an **active database**
- Triggers are available in most current commercial DB products
  - And are part of the SQL 1999 standard
- Triggers carry out **actions** when their triggering conditions are met
  - Generally SQL constraints only reject transactions

# Why Use Triggers?

- Triggers can implement business rules
  - e.g. creating a new loan when a customer's account is overdrawn
- Triggers may also be used to maintain data in related database tables
  - e.g. Updating derived attributes when underlying data is changed, or maintaining summary data

# Trigger Components

- Event (activates the trigger)
  - A specified modification to the DB
    - May be an insert, deletion, or change
    - The trigger may **fire** before or after the transaction
- Condition
  - A Boolean expression or a query
    - If the query answer set is non-empty it evaluates to true and the trigger action occurs, otherwise false
- Action
  - A trigger's action can be very far-ranging, e.g.
    - Execute queries, Make modifications to the DB, Create new tables, Call host-language procedures

# Triggers

- Synchronization of the Trigger with the activating statement (DB modification)
  - Before
  - After
- Number of Activations of the Trigger
  - Once per modified tuple (FOR EACH ROW)
  - Once per activating statement (default).

# Two kinds of Triggers

- **Statement-level trigger**: executed once for all the tuples that are changed in one SQL statement.

```
REFERENCING  NEW TABLE AS  newtuples,   // Set of new tuples
             OLD TABLE AS  oldtuples     // Set of old tuples
```

- **Row-level trigger**: executed once for each modified tuple.

```
REFERENCING  OLD AS   oldtuple,
             NEW AS   newtuple
```

*newtuples*, *oldtuple*, *newtuple* can be used in the CONDITION and ACTION clauses

# Triggers

- Options for the REFERENCING clause:
  - NEW TABLE: the set of tuples newly inserted (INSERT).
  - OLD TABLE: the set of deleted or old versions of tuples (DELETE / UPDATE).
  - OLD ROW: the old version of the tuple (FOR EACH ROW UPDATE).
  - NEW ROW: the new version of the tuple (FOR EACH ROW UPDATE).
- The action of a trigger can consist of multiple SQL statements, surrounded by BEGIN . . . END.

# Triggers

```
CREATE TRIGGER youngSailorUpdate
   AFTER INSERT ON SAILORS     /* Event */
 REFERENCING NEW TABLE NewSailors
   FOR EACH STATEMENT
      INSERT                                  /* Action */
          INTO YoungSailors(sid, name, age, rating)
          SELECT sid, name, age, rating
          FROM NewSailors N
          WHERE N.age <= 18;
```

- This trigger inserts young sailors into a separate table.
- It has no (i.e., an empty, always true) condition.

# Triggers

```
CREATE TRIGGER notTooManyReservations
   AFTER INSERT ON Reserves                                  /* Event */
   REFERENCING NEW ROW NewReservation
   FOR EACH ROW
   WHEN (10 <= (SELECT COUNT(*)
                FROM Reserves
                WHERE sid =NewReservation.sid))              /* Condition */
        DELETE FROM Reserves R
        WHERE R.sid= NewReservation.sid                      /* Action */
              AND day=
              (SELECT MIN(day) FROM Reserves R2 WHERE R2.sid=R.sid);
```

- This trigger makes sure that a sailor has less than 10 reservations, deleting the oldest reservation of a given sailor, if neccesary.
- It has a non- empty condition (WHEN).

# Triggers in Oracle

## Syntax

CREATE [OR REPLACE] TRIGGER <trigger_name>
{BEFORE|AFTER} {INSERT|DELETE|UPDATE} ON <table_name>
[REFERENCING [NEW AS <new_row_name>]
[OLD AS <old_row_name>]] [FOR EACH ROW
[WHEN (<trigger_condition>)]]
<trigger_body>

# Triggers

## Example

- Create a trigger that checks whether a new tuple inserted into T4
- has the first attribute <= 10.
  - If so, insert the *reverse* tuple into T5.

```
CREATE TABLE T4 (a INTEGER, b CHAR(10));
CREATE TABLE T5 (c CHAR(10), d INTEGER);

CREATE TRIGGER trig1
        AFTER INSERT ON T4
        REFERENCING NEW AS newRow
        FOR EACH ROW
        WHEN (newRow.a <= 10)
        BEGIN
            INSERT INTO T5 VALUES(:newRow.b, :newRow.a);
        END trig1;
```

Examples from http://infolab.stanford.edu/~ullman/fcdb/oracle/or-triggers.html