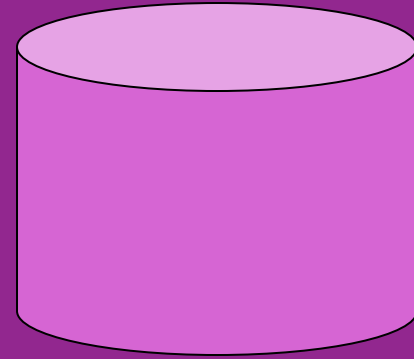




LEEDS
BECKETT
UNIVERSITY



School of Built Environment, Engineering and Computing

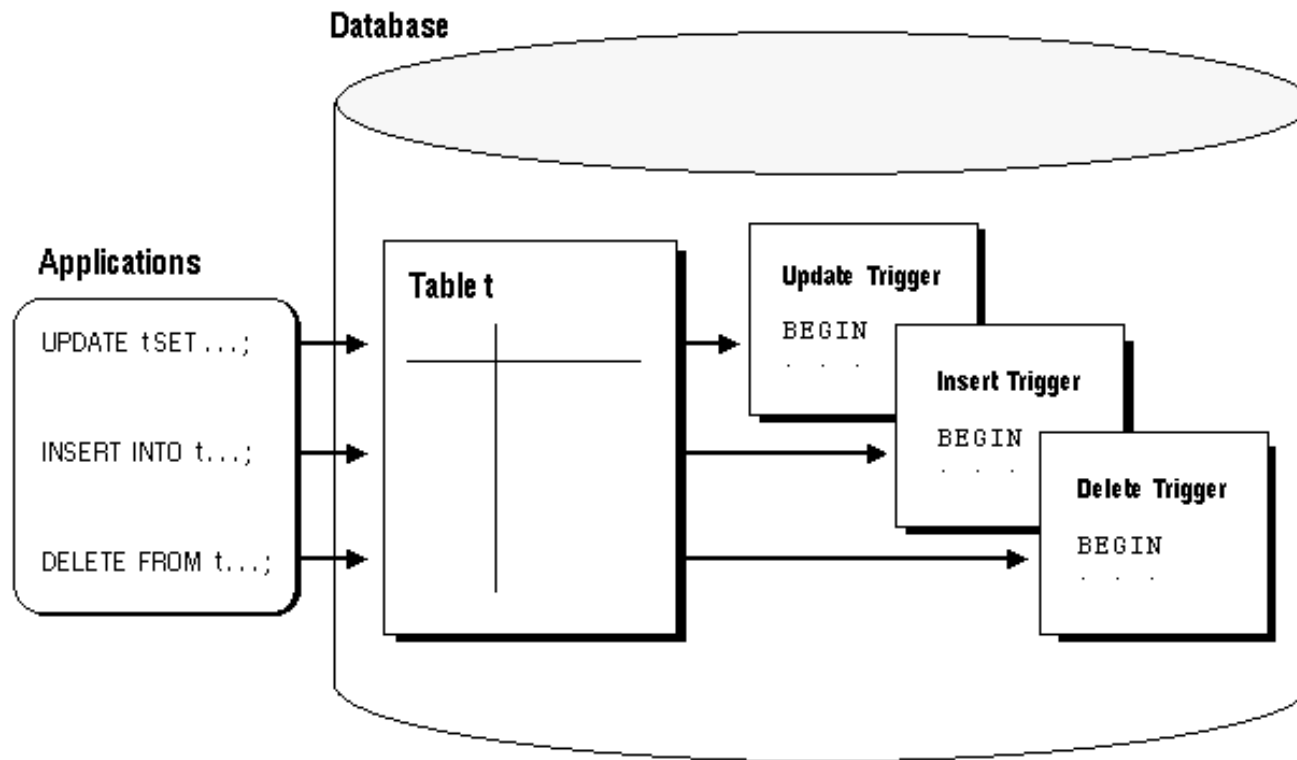
PL/SQL: Triggers

Sanela Lazarevski (Module Leader)

What is a Trigger?

- A PL/SQL procedure that is automatically fired (executed) when an INSERT, UPDATE or DELETE statement is issued against a table (not a view).
- Triggers stored in database (Server side)
- Can also be called from within APEX Form
- Similar to stored procedures but implicitly fired
- Procedure has to be specifically invoked

Database Triggers



<http://docs.oracle.com>

Adding a Trigger using SQL Workshop

The screenshot displays the Oracle Application Express (APEX) SQL Workshop interface. The browser address bar shows the URL: `aet-oracle.aet.leedsbeckett.ac.uk:7777/pls/apex/f?p=4500:1001:6122530713317:::`. The top navigation bar includes tabs for "Campbell, Jackie - Outloo", "Object Browser", and "tmpsalguide oracle table". The main navigation area has "Application Builder", "SQL Workshop" (highlighted with a red circle), "Team Development", and "Packaged Apps". The "Object Browser" section on the left shows a list of objects, with "Triggers" (highlighted with a red circle) selected. The right pane displays the "TMP_SALARY_CHECK" trigger, with the "Code" tab (highlighted with a red circle) active. The code editor shows the following SQL code:

```
1 create or replace trigger tmp_salary_check
2 before insert or update of sal, job
3 on tmpemp
4 for each row
5 when (new.job <> 'PRESIDENT')
6 DECLARE
7     minsall      number;
8     maxsall      number;
9 begin
10     select minsall, maxsall
11     into minsall, maxsall
12     from tmpsalguide
13     where job = :new.job;
14     if (:new.sal < minsall or :new.sal > maxsall)
15     then raise_application_error(-20601,
16         'Salary ' || :new.sal || 'out of range for job ' || :new.job ||
17         'for employee ' || :new.ename);
18     end if;
19 end;
```

The bottom status bar indicates the user is logged in as "cambe10", the schema is "cambe10", and the application version is "Application Express 5.0.0.00.31". Copyright © 1999, 2015, Oracle. All rights reserved.



What are they used for?

- automatically generate derived column values
- prevent invalid transactions
- enforce complex security authorizations
- enforce referential integrity across nodes in a distributed database
- enforce complex business rules
- provide transparent event logging
- provide sophisticated auditing
- maintain synchronous table replicates
- gather statistics on table access

What is a Trigger?

- Triggered procedures usually just called triggers.
- A trigger is a piece of code that is automatically run when a specified event occurs.
- We say that the event causes the trigger to fire.
- Two types of Trigger
 - Row level
 - Fired each time table is affected by triggering statement
 - Statement level
 - Fired once by the statement irrespective of number of rows affected.

Server side Triggers

- There are three database events:

INSERT
UPDATE
DELETE

- A trigger may fire

BEFORE

or

AFTER a database event

- Oracle triggers can be

Row-level or Statement-level

This gives us twelve different triggers:

BEFORE INSERT *row*
BEFORE INSERT *statement*
AFTER INSERT *row*
AFTER INSERT *statement*
BEFORE UPDATE *row* ...

Row-level Triggers

- Fire once for each table row affected by the transaction.
- Have access to both the **old** data and the **new** data in the current row.
- Often used to write audit information
E.g. who inserted the row and when?

Statement-level Triggers

- Fire once for each transaction.
- Mainly used to enforce specific security measures on a table.

E.g. Could allow accounts clerks access to the employee table at the financial year end, but not at other times.

Trigger Syntax

```
CREATE OR REPLACE TRIGGER trigger_name  
[BEFORE | AFTER] [DELETE | INSERT | UPDATE]  
ON table_name  
[FOR EACH ROW]  
[WHEN condition]  
[pl/sql block];
```

Specifying the Triggered Action

```
CREATE OR REPLACE TRIGGER student_aft_ins_upd
AFTER INSERT OR UPDATE
ON student

WHEN USER = 'Stephen'
BEGIN
INSERT INTO user_tracking (user_name, table, date)
VALUES (USER, 'student', SYSDATE);
END;
```

The :old and :new Keywords

Row level trigger

```
BEGIN
  IF :old.fee_paid < :new.fee_paid
THEN
  INSERT INTO student_audit (user, date, row, action)
  VALUES (USER, SYSDATE, :old.student_id, 'fee payment');
ELSE
  ...etc.
```

INSERTING, UPDATING & DELETING

AFTER trigger

```
IF INSERTING THEN
```

```
    INSERT INTO student_audit (user, date, row, action)  
    VALUES (USER, SYSDATE, :old.student_id, 'insert');
```

```
ELSE IF DELETING THEN
```

```
    INSERT INTO student_audit (user, date, row, action)  
    VALUES (USER, SYSDATE, :old.student_id, 'delete');
```

```
END IF
```

```
...
```

Etc.

Exclusive arc type Trigger

```
create or replace trigger tmp_excl_arc
Before insert or update of cocustno, indcustno
on tmpord
for each row
begin
    if (:new.cocustno is null) and (:new.indcustno is null)
    then  raise_application_error(-20001,
        'this order must belong to a customer');
    end if;
    if (:new.cocustno is not null) and (:new.indcustno is not null)
    then  raise_application_error(-20002,
        'this order must belong to only one type of customer');
    end if;
end;
```

Statement level trigger – a weekend restriction

```
create or replace trigger tmp_permit_changes
before delete or insert or update
on tmpemp
declare
    dummy integer;
begin
    /* if today is a sat or sun, then return an error */
    if (to_char(sysdate,'DY') = 'SAT' OR to_char(sysdate,'DY') = 'SUN')
    then raise_application_error(-20501,
        'May not change employee table during ' || to_char(sysdate,'DY'));
    end if;
end;
```

PL/SQL – trigger demo

The screenshot displays the Oracle Application Express interface. The browser address bar shows the URL: `aet-oracle.aet.leedsbeckett.ac.uk:7777/pls/apex/f?p=4500:1001:6122530713317:::`. The interface includes a top navigation bar with tabs for 'Object Browser', 'SQL Workshop', 'Team Development', and 'Packaged Apps'. The 'Object Browser' tab is active, showing a tree view of database objects under the 'CAMBE10' schema. The 'Triggers' folder is expanded, and the 'TMP_SALARY_CHECK' trigger is selected. The 'Code' tab is active, displaying the following PL/SQL code:

```
1 create or replace trigger tmp_salary_check
2 before insert or update of sal, job
3 on tmpemp
4 for each row
5 when (new.job <> 'PRESIDENT')
6 DECLARE
7     minsall      number;
8     maxsall      number;
9 begin
10    select minsall, maxsall
11    into minsall, maxsall
12    from tmpsalguide
13    where job = :new.job;
14    if (:new.sal < minsall or :new.sal > maxsall)
15    then raise_application_error(-20601,
16         'Salary ' || :new.sal || ' out of range for job ' || :new.job ||
17         ' for employee ' || :new.ename);
18    end if;
19 end;
```

The interface also shows a 'Save & Compile' button and a 'Download Source' button. The bottom status bar indicates 'Page: 4 of 4', 'Words: 72/693', and the date '17/09/2015'.

PL/SQL

Summary

- Why we need PL/SQL
- Structure
- Syntax and constructs
- Server Side Triggers



LEEDS
BECKETT
UNIVERSITY

Thank you

Any questions?