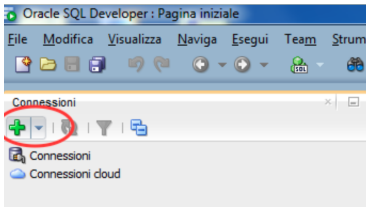
Basi di Dati
Oracle SQL PLUS - Trigger

La finalità di questa esercitazione è di scrivere trigger in SQL da eseguire su un database Oracle.

Connessione al database

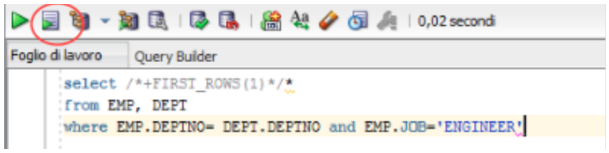
1. Aprire il programma Oracle SQL Developer.
2. Cliccare su *Crea Nuova Connessione*



Materiale disponibile

Sul sito del corso sono disponibili alcuni script contenenti istruzioni SQL per creare le basi dati necessarie per l'esercitazione.

Gli script possono essere caricati aprendo File->Apri e selezionando il file .sql e successivamente cliccando il pulsante *Esegui Script*.



Comandi utili

Definition 0.1 — *Cancellazione di un trigger:*

```
DROP TRIGGER nome_trigger;
DROP TRIGGER "nome_trigger";
```

Definition 0.2 — *Sostituzione (aggiornamento) di un trigger esistente (anziché cancellarlo e ricrearlo):*

```
CREATE OR REPLACE TRIGGER nomeTrigger
```

Definition 0.3 — *Visualizzazione dei trigger generati:*

```
SELECT  trigger_name, triggering_event, table_name,
        status, description, action_type, trigger_body
FROM    user_triggers;
```

Definition 0.4 — *Disabilitazione di un trigger esistente:*

```
ALTER TRIGGER triggerName DISABLE
```

Definition 0.5 — *Visualizzazione degli errori dei trigger:*

```
SELECT * FROM USER_ERRORS;
```

Consigli

Per creare le tabelle degli esercizi, sono disponibili gli script $script\_db\_es1.sql$, $script\_db\_es2.sql$.
Per la creazione dei trigger, prestare attenzione alla sintassi e ai seguenti dettagli:

- assegnare un nome opportuno alle variabili, evitando parole chiave come MIN, MAX, ...
- dichiarare variabili diverse su righe diverse e non sulla stessa riga separate da punto e virgola

  ```
  MyVarUno NUMBER;
  MyVarDue NUMBER;
  MyVarTre VARCHAR2(16);
  ```

- terminare il trigger con il carattere / come nel seguente esempio:

  ```
  ...
  END;
  /
  ```

  N.B. La creazione del trigger può andare a buon fine anche in assenza del carattere di terminazione ***;***

- terminare le istruzioni con il carattere *;* assegnare nuovi valori alle variabili con :=, es.

```
UPDATE tablename
SET varname=newvalue
WHERE column=:NEW.attribute;
IF A<3 OR A=3 THEN
MyVar:='Tre';
ELSE
IF A>3 AND A<5 THEN
MyVar:='Quattro';
ELSE
MyVar:='Altro';
END IF;
END IF;
```

Prima di iniziare l'esercitazione si consiglia di eliminare gli eventuali trigger già presenti sull'utente scelto, che potrebbero alterare i risultati degli esercizi proposti.

N.B. Copiando e incollando porzioni di codice direttamente dal testo dell'esercitazione in formato pdf al Browser Web prestare attenzione all'eventuale conversione o codifica errata dei caratteri, che potrebbe generare il seguente errore: *ora-00911: invalid character*

Esercizio 1

Si consideri il seguente database

IMP(EMPNO, DEPTNO, ENAME, JOB, SAL)
DIP(DEPTNO, DNAME, LOC, MINSAL, MAXSAL)

Nel caso un dipartimento passi dal ruolo (attributo DNAME) di 'ACCOUNTING' al ruolo di 'SALES', il salario per tutti gli impiegati del dipartimento viene incrementato di 100.
Definire un trigger che implementi tale funzionalità. Lo svolgimento dell'esercizio è strutturato nei seguenti passi:

1. Creare la base dati utilizzando lo script $create\_db\_es1.sql$
2. Creare il trigger, eventualmente mediante uno script.
3. Verificare il contenuto delle tabelle IMP e DIP
4. Modificare il nome del dipartimento 'ACCOUNTING':

```
UPDATE DIP SET DNAME = 'SALES' WHERE DNAME='ACCOUNTING';
```

5. Verificare il contenuto delle tabelle IMP e DIP

Durante l'esercitazione dovranno essere eseguiti i seguenti passi:

- scrivere il trigger;
- verificare l'output generato in cui si osserva il risultato ottenuto.

Esercizio 2

Si consideri il seguente database

IMP(<u>EMPNO</u>, ENAME, JOB, SAL)
SUMMARY(<u>JOB</u>, NUM)

Verificare che siano state create correttamente le tabelle necessarie effettuando le seguenti query:

```
SELECT * FROM IMP;
SELECT * FROM SUMMARY;
```

Nelle tabelle SUMMARY, il campo NUM indica il numero di impiegati in IMP che svolgono uno stesso lavoro. Si scrivano i trigger per mantenere la consistenza tra la tabella IMP e SUMMARY in caso di:

- inserimento di un record in IMP
- aggiornamento del campo JOB in IMP

Creare la base dati utilizzando lo script $create\_db\_es1.sql$

Soluzioni

Esercizio 1

Database prima dell'esecuzione del trigger

Tabella IMP

EMPNO	DEPTNO	ENAME	JOB	SAL
7000	10	SMITH	CLERK	850
7010	10	SCOTT	ANALYST	1000
7020	10	BLAKE	SALESMAN	1400
7030	10	SMITH	MANAGER	2500
7040	20	SMITH	CLERK	800
7050	20	SCOTT	ANALYST	1600
7060	30	ADAMS	CLERK	000
7070	30	JAMES	CLERK	1000
7080	40	ALLEN	CLERK	850

Tabella DIP

DEPTNO	DNAME	LOC	MINSAL	MAXSAL
10	ACCOUNTING	NEW YORK	100	2500
20	RESEARCH	DALLAS	150	3000
30	SALES	CHICAGO	120	2500
40	OPERATIONS	BOSTON	200	2100

Codice

Trigger per la gestione dell'aggiornamento del campo DNAME in DIP

```
CREATE OR REPLACE TRIGGER UP_SAL
AFTER UPDATE OF DNAME ON DIP
FOR EACH ROW
WHEN (OLD.DNAME='ACCOUNTING' AND NEW.DNAME='SALES')
BEGIN
-- aggiorna salario impiegati dipartimento cambiato
UPDATE IMP
SET SAL=SAL+100
WHERE DEPTNO =:OLD.DEPTNO;
END;
/
```

Istruzione di aggiornamento

```
UPDATE DIP SET DNAME = 'SALES' WHERE DNAME='ACCOUNTING';
```

Database dopo l'esecuzione del trigger

Tabella IMP

EMPNO	DEPTNO	ENAME	JOB	SAL
7000	10	SMITH	CLERK	950
7010	10	SCOTT	ANALYST	1700
7020	10	BLAKE	SALESMAN	1500
7030	10	SMITH	MANAGER	2600
7040	20	SMITH	CLERK	800
7050	20	SCOTT	ANALYST	1600
7060	30	ADAMS	CLERK	900
7070	30	JAMES	CLERK	1000
7080	40	ALLEN	CLERK	850

Tabella DIP

DEPTNO	DNAME	LOC	MINSAL	MAXSAL
10	SALES	NEW YORK	100	2500
20	RESEARCH	DALLAS	150	3000
30	SALES	CHICAGO	120	2500
40	OPERATIONS	BOSTON	200	2100

Commento

Il trigger *UP_SAL* implementa la funzionalità richiesta ed è attivato in seguito ad un'operazione di update sull'attributo *DNAME* della tabella *DIP* (trigger di tipo after).

Il trigger ha inoltre granularità di tupla in quanto occorre poter accedere all'attributo *DNAME* del record aggiornato (e di quello originale) per verificare che il tipo di aggiornamento sia quello richiesto ('ACCOUNTING' -> 'SALES').

Esercizio 2

Database prima dell'esecuzione del trigger

Tabella IMP

EMPNO	ENAME	JOB	SAL
1	VERDI	SEGRETARIA	800
2	ROSSI	BANCHIERE	900
3	BIANCHI	BANCHIERE	1100

Tabella SUMMARY

JOB	NUM
SEGRETARIA	1
BANCHIERE	2

Codice

Trigger per la gestione dell'inserimento di un record in IMP

```
CREATE OR REPLACE TRIGGER INS_IMP
AFTER INSERT ON IMP
FOR EACH ROW
DECLARE
N NUMBER;
M NUMBER;
BEGIN
--- controlla se ci sono impiegati che fanno lo stesso lavoro
SELECT COUNT(*) INTO N
FROM SUMMARY
WHERE JOB=:NEW.JOB;
IF(N=0) THEN
--- e' il primo impiegato che  fa quel lavoro
INSERT INTO SUMMARY (JOB, NUM)
VALUES(:NEW.JOB, 1);
ELSE
--- esiste almeno un impiegato che fa quel lavoro
SELECT NUM INTO M
FROM SUMMARY
WHERE JOB=:NEW.JOB;
UPDATE SUMMARY
SET NUM=M+1
WHERE JOB=:NEW.JOB;
END IF;
END;
/
```

Istruzione di aggiornamento

```
INSERT INTO IMP(EMPNO, ENAME, JOB, SAL) VALUES(4, 'NERI', 'CORRIERE', 750);
```

Database dopo l'esecuzione del trigger

Tabella IMP

EMPNO	ENAME	JOB	SAL
1	VERDI	SEGRETARA	800
2	ROSSI	BANCHIERE	000
3	BIANCHI	BANCHIERE	1100
4	NERI	CORRIERE	750

Tabella DIP

JOB	NUM
SEGRETARIA	1
BANCHIERE	2
CORRIERE	1

Codice

Trigger per la gestione dell'aggiornamento del campo JOB in IMP

```
CREATE OR REPLACE TRIGGER UPD_IMP
AFTER UPDATE OF JOB ON IMP
FOR EACH ROW
DECLARE
N NUMBER;
M NUMBER;
BEGIN
--- conta quanti impiegati hanno il nuovo lavoro
SELECT COUNT(*) INTO N
FROM SUMMARY
WHERE JOB=:NEW.JOB;
--- incrementa il numero di impiegati per il nuovo lavoro
IF (N=0) THEN
--- l'impiegato inserito e' il primo impiegato per il nuovo lavoro
INSERT INTO SUMMARY(JOB, NUM)
VALUES(:NEW.JOB, 1);
ELSE
--- e' presente almeno un altro impiegato per il nuovo lavoro
UPDATE SUMMARY
SET NUM=NUM+1
WHERE JOB =:NEW.JOB;
END IF;
SELECT NUM INTO M
FROM SUMMARY
WHERE JOB=:OLD.JOB;
IF (M=1) THEN
--- e' presente un solo impiegato per il vecchio lavoro.
--- cancella il record dalla tabella SUMMARY
DELETE FROM SUMMARY
WHERE JOB =:OLD.JOB;
ELSE
--- decrementa NUM nel corrispondente record nella tabella SUMMARY
UPDATE SUMMARY
SET NUM=NUM-1
WHERE JOB=:OLD.JOB;
END IF;
END;
/
```

Istruzione di aggiornamento

```
    UPDATE IMP SET JOB='CORRIERE' WHERE EMPNO=2;
```

Database dopo l'esecuzion del trigger

Tabella IMP

EMPNO	ENAME	JOB	SAL
1	VERDI	SEGRETARIA	800
2	ROSSI	CORRIERE	900
3	BIANCHI	BANCHIERE	1100
4	NERI	CORRIERE	750

Tabella SUMMARY

JOB	NUM
SEGRETARIA	1
BANCHIERE	1
CORRIERE	2

Commento

Commenti Entrambi i trigger sono a granularità di tupla in quanto occorre avere visibilità sul campo JOB della tupla inserita o aggiornata in IMP. In questo modo è possibile verificare se, dopo la modifica su IMP, esiste almeno una tupla (e se si quante) con lo stesso valore di JOB nella tabella SOMMARIO.

N.B. Questa verifica non può essere effettuata eseguendo direttamente un conteggio delle tuple in IMP, in quanto in un trigger a granularità di tupla non è possibile accedere (in lettura/scrittura) alla tabella mutante.