

Dynamic SQL

Interrogazioni parametriche in PL/SQL

```
procedure doIt(  
    X      in VarChar2  
)  
y VarChar2;  
begin  
    select s.nome into y  
    from  Studenti s  
    where Cognome = X;  
end;
```

Parametrizzazioni impossibili

```
procedure doIt(  
    X      in VarChar2  
)  
y VarChar2;  
begin  
    select s.nome into y  
    from X s;  
    select s.X into y  
    from studenti s;  
    for stud in ( select * from studenti order by X )  
    loop ... end loop;  
    for stud in ( select * from studenti X )  
    loop ... end loop;  
end;
```

Dynamic SQL

- PL/SQL offre alcuni costrutti per l'esecuzione di codice PL/SQL preso in input come un valore di tipo VarChar2
- Questi strumenti sono utilizzati per eseguire codice creato a run-time
- Questo aiuta a risolvere i problemi di parametrizzazione
- Meno controlli statici

EXECUTE IMMEDIATE

- *EXECUTE IMMEDIATE* esegue blocchi PL/SQL
 - Begin <commands> End;oppure query SQL che ritornano una sola riga
- La sintassi è della forma:

```
EXECUTE IMMEDIATE dynamic_string  
[INTO {define_variable[, define_variable]... | record}]  
[USING bind_argument[, bind_argument]...]
```

Esempio

```
nome          varchar2(20) := 'Bob';  
cognome       varchar2(20) := 'Elephant';  
sql_stmt      varchar2(250);  
plsql_block   varchar2(250);  
persona       persone%ROWTYPE;  
BEGIN  
  EXECUTE IMMEDIATE  
    'CREATE TABLE auto (targa NUMBER, modello Varchar2(20))';  
  sql_stmt := 'INSERT INTO persone VALUES (:1, :2, :c)';  
  EXECUTE IMMEDIATE sql_stmt USING nome, cognome , 1970;  
  sql_stmt := 'SELECT * FROM persone  
              WHERE nome = "Bob" and cognome = "Elephant" ';  
  EXECUTE IMMEDIATE sql_stmt INTO persona;  
  plsql_block := 'BEGIN user.cercaPersona(:1,:2); END;';  
  EXECUTE IMMEDIATE plsql_block USING nome, cognome;  
END;
```

OPEN FOR

- OPEN ... FOR esegue query multiriga e immette il risultato in un cursore (tipo REF CURSOR, weakly typed)
- La sintassi è della forma:

```
OPEN cursor_variable FOR dynamic_string  
[USING bind_argument[, bind_argument]...];
```

Esempio d'uso di OPEN...FOR

- Cartella *ese7*
- Compilare nell'ordine:
 - *Create.sql*
 - *ModGUI.sql*
 - *Ese7.sql*
- Modificare *host* e *user*
- Test:
*http://oracle1.cli.di.unipi.it/pls/USER.p_ese7.imm
ettiparametri*

Tipo del cursore

```
TYPE EmpCurTyp IS REF CURSOR;  
pers EmpCurTyp; /* weakly typed cursor variable */  
sqlString := ' select * from persone  
    where ((:1 is null) Or (upper(rtrim(Nome)) like upper(rtrim(:2))))  
    And ((:3 is null) Or (upper(rtrim(Cognome)) like upper(rtrim(:4))))  
    And ((:5 is null) Or (AnnoNascita = :6)) ';  
if not(Ordine is null) then sqlString := sqlString || ' order by ' || Ordine;  
end if;  
OPEN pers FOR sqlString  
USING ilNome, ilNome, ilCognome, ilCognome, LAnnoNascita, LAnnoNascita;  
loop  
    FETCH pers INTO laPersona;  
    EXIT WHEN pers%NOTFOUND;  
    ....  
end loop;
```

Generazione di stringa/statement

```
TYPE EmpCurTyp IS REF CURSOR;  
pers EmpCurTyp; /* weakly typed cursor variable  
sqlString := ' select * from persone  
    where ((:1 is null) Or (upper(rtrim(Nome)) like upper(rtrim(:2))))  
    And ((:3 is null) Or (upper(rtrim(Cognome)) like upper(rtrim(:4))))  
    And ((:5 is null) Or (AnnoNascita = :6)) ';  
if not(Ordine is null) then sqlString := sqlString || ' order by ' || Ordine;  
end if;  
OPEN pers FOR sqlString  
USING  
    ilNome, ilNome, ilCognome, ilCognome, LAnnoNascita, LAnnoNascita;  
loop  
    FETCH pers INTO laPersona;  
    EXIT WHEN pers%NOTFOUND;  
    ....  
end loop;
```

Esecuzione di statement

```
TYPE EmpCurTyp IS REF CURSOR;  
pers EmpCurTyp; /* weakly typed cursor variable  
sqlString := ' select * from persone  
    where ((:1 is null) Or (upper(rtrim(Nome)) like upper(rtrim(:2))))  
    And ((:3 is null) Or (upper(rtrim(Cognome)) like upper(rtrim(:4))))  
    And ((:5 is null) Or (AnnoNascita = :6)) ';  
if not(Ordine is null) then sqlString := sqlString || ' order by ' || Ordine;  
end if;  
OPEN pers FOR sqlString  
USING  
    ilNome, ilNome, ilCognome, ilCognome, LAnnoNascita, LAnnoNascita;  
loop  
    FETCH pers INTO laPersona;  
    EXIT WHEN pers%NOTFOUND;  
    ....  
end loop;
```

Esercizio

- Completare l'interfaccia dell'esercizio precedente in modo da fornire un radio button per la scelta:
 - Ordina in modo crescente o ascendente
- Permettere all'utente di scegliere i campi da visualizzare nel risultato tramite un insieme di check box (una per campo)