ORACLE

Corso SQL APEX ITALIA MARITTIMA

Lezione 6 - 5 Febbraio 2025

Ing. Roberto Capancioni

Chi Sono

Ing. Roberto Capancioni

– Trainer Oracle Academy

capancioni.com

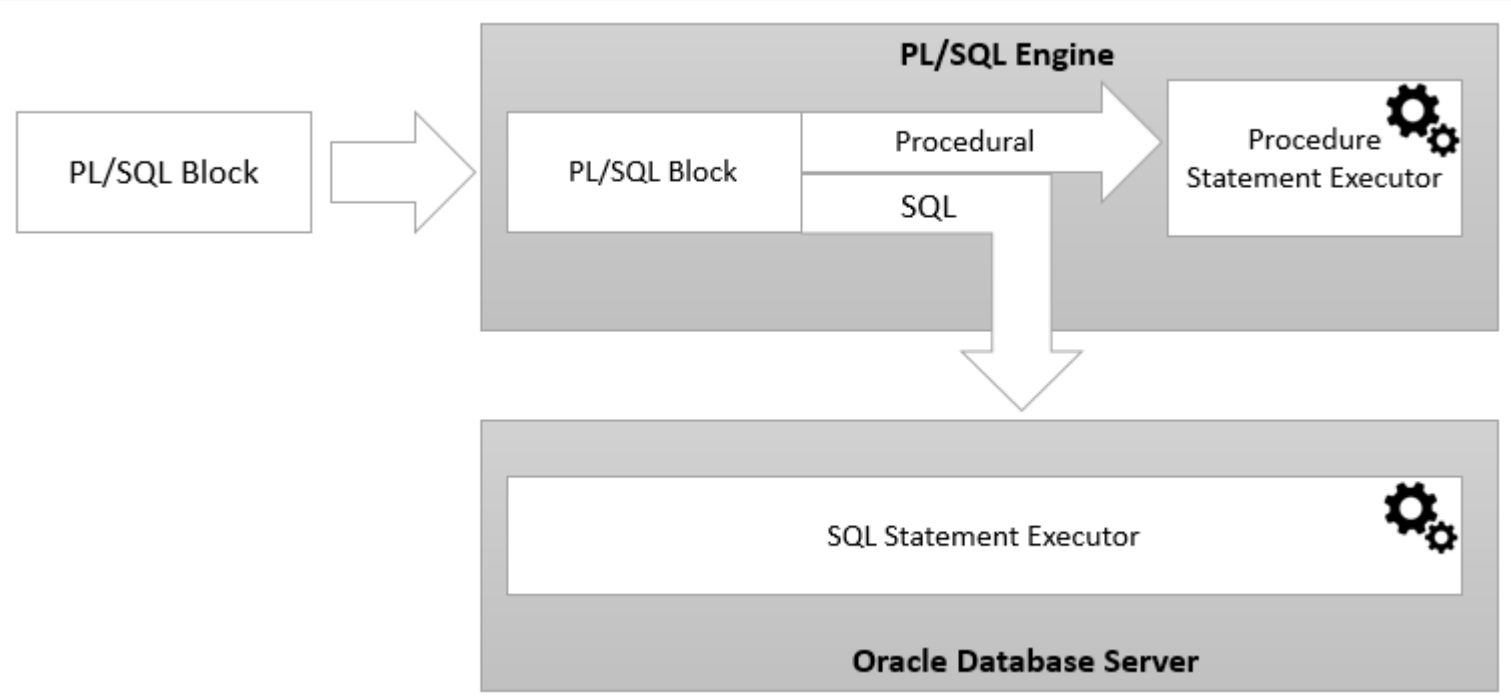
Email: sviluppo@capancioni.com
Linkedin: https://www.linkedin.com/in/robertocapancioni

PL/SQL

PL/SQL

- Aggiunge costrutti procedurali al linguaggio SQL
- E’ un linguaggio altamente strutturato
- E’ integrato all’interno del database Oracle

Procedural
Language extensions to the
Structured
Query
Language



PL/SQL

```
BEGIN
    DBMS_OUTPUT.put_line ('Hello World!');
END;
/
/;
```

PL/SQL

```
DECLARE
   l_message VARCHAR2( 255 ) := 'Hello World!';
BEGIN
   DBMS_OUTPUT.PUT_LINE( l_message );
END;
/
```

PL/SQL

Exception	Oracle Error
ACCESS_INTO_NULL	ORA-06530
CASE_NOT_FOUND	ORA-06592
COLLECTION_IS_NULL	ORA-06531
CURSOR_ALREADY_OPEN	ORA-06511
DUP_VAL_ON_INDEX	ORA-00001
INVALID_CURSOR	ORA-01001
INVALID_NUMBER	ORA-01722
LOGIN_DENIED	ORA-01017
NO_DATA_FOUND	ORA-01403
NOT_LOGGED_ON	ORA-01012
PROGRAM_ERROR	ORA-06501
ROWTYPE_MISMATCH	ORA-06504
SELF_IS_NULL	ORA-30625
STORAGE_ERROR	ORA-06500
SUBSCRIPT_BEYOND_COUNT	ORA-06533
SUBSCRIPT_OUTSIDE_LIMIT	ORA-06532
SYS_INVALID_ROWID	ORA-01410
TIMEOUT_ON_RESOURCE	ORA-00051
TOO_MANY_ROWS	ORA-01422
VALUE_ERROR	ORA-06502
ZERO_DIVIDE	ORA-01476

```
DECLARE
      V_RESULT NUMBER;
BEGIN
   V_RESULT := 1 / 0;
   EXCEPTION
     WHEN ZERO_DIVIDE THEN
         DBMS_OUTPUT.PUT_LINE('Non puoi dividere per zero');
     WHEN OTHERS THEN
         DBMS_OUTPUT.PUT_LINE( SQLERRM );
END;
/
```

PL/SQL

```
DECLARE
   l_data date;
BEGIN

   SELECT sysdate
     INTO l_data
     FROM dual;

   DBMS_OUTPUT.PUT_LINE(to_char(l_data,'DD/MM/YYYY HH24:MM:SS'));
END;
/
```

PL/SQL

```
DECLARE
   l_data date;
BEGIN

   SELECT sysdate
     INTO l_data
     FROM dual;

   DBMS_OUTPUT.PUT_LINE(to_char(l_data,'DD/MM/YYYY HH24:MM:SS'));
END;
/
```

PL/SQL

```
DECLARE
  l_venduto NUMBER := 100000;
BEGIN
    IF l_venduto > 100000 THEN
       DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k' );
    ELSIF l_venduto > 50000 THEN
       DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k' );
    ELSE
       DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k' );
    END IF;
END;
/
```

PL/SQL

```
DECLARE
   l_voto CHAR( 1 );
   l_giudizio  VARCHAR2( 20 );
BEGIN
   l_voto := 'B';
   CASE l_voto
   WHEN 'A' THEN
     l_giudizio := 'Eccellente' ;
   WHEN 'B' THEN
     l_giudizio := 'Ottimo' ;
   WHEN 'C' THEN
     l_giudizio := 'Buono' ;
   WHEN 'D' THEN
     l_giudizio := 'Sufficiente' ;
   ELSE
     l_giudizio := 'Insufficiente' ;
   END CASE;
   DBMS_OUTPUT.PUT_LINE( l_giudizio );
END;
/
```

PL/SQL

```
DECLARE
  l_venduto     NUMBER;
  l_commissione NUMBER;
BEGIN
  l_venduto := 150000;
  CASE
  WHEN l_venduto    > 200000 THEN
    l_commissione := 0.2;
  WHEN l_venduto   >= 100000 AND l_venduto < 200000 THEN
    l_commissione := 0.15;
  WHEN l_venduto   >= 50000 AND l_venduto < 100000 THEN
    l_commissione := 0.1;
  WHEN l_venduto    > 30000 THEN
    l_commissione := 0.05;
  ELSE
    l_commissione := 0;
  END CASE;

  DBMS_OUTPUT.PUT_LINE( 'Commissioni:  ' || l_commissione * 100 || '%' );
END;
/
```

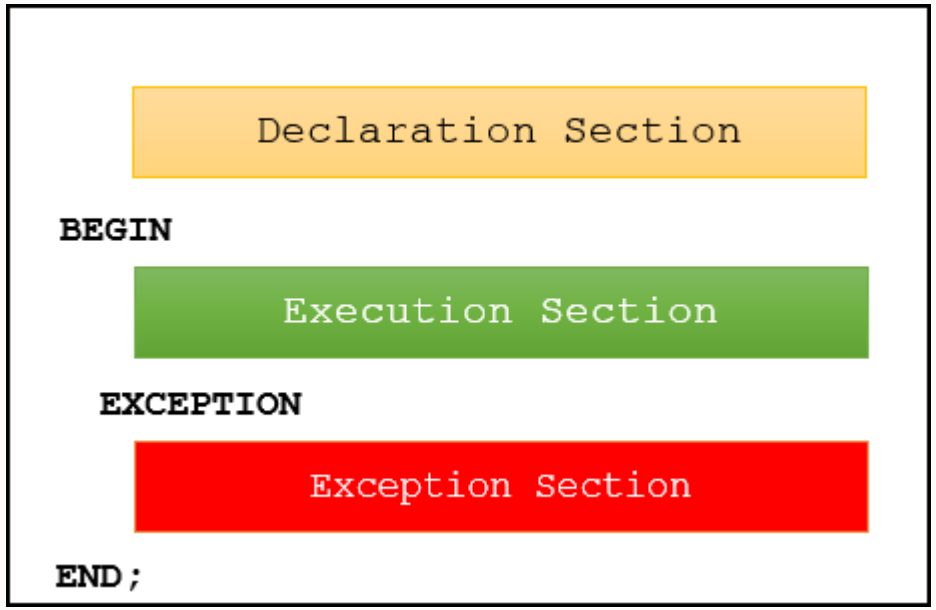
PL/SQL

```
DECLARE

BEGIN
  for c in (select level numero from dual connect by level <=10)
  loop
     DBMS_OUTPUT.PUT_LINE( 'Numero-->  ' || c.numero  );
  end loop;

END;
```

Blocco Anonimo



PL/SQL

BLOCCO ANONIMO

```
DECLARE
  l_venduto NUMBER := 100000;
BEGIN
   IF l_venduto > 100000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k ' );
   ELSIF l_venduto > 50000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k ' );
   ELSE
      DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k ' );
   END IF;
END;
/
```

PROCEDURA

```
CREATE OR REPLACE PROCEDURE D06_VENDUTO
                    (
                     p_venduto IN number default 0
                    )
IS
BEGIN
   IF p_venduto > 100000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k ' );
   ELSIF p_venduto > 50000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k ' );
   ELSE
      DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k ' );
   END IF;
END;
/

BEGIN
  D06_VENDUTO(p_venduto => 100000);
END;
/
```

PL/SQL

BLOCCO ANONIMO

```
DECLARE
  l_venduto NUMBER := 100000;
BEGIN
   IF l_venduto > 100000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k ' );
   ELSIF l_venduto > 50000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k ' );
   ELSE
      DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k ' );
   END IF;
END;
/
```

FUNZIONE

```
CREATE OR REPLACE FUNCTION D06_GET_VENDUTO
                  (
                   p_venduto IN number default 0
                   )
RETURN VARCHAR2
IS
l_return VARCHAR2(100);
BEGIN
   IF p_venduto > 100000 THEN
      l_return := 'Venduto > 100k ';
   ELSIF p_venduto > 50000 THEN
      l_return := 'Venduto > 50k ';
   ELSE
      l_return := 'Venduto <= 50k ';
   END IF;
   RETURN l_return;
END;
/

DECLARE
l_venduto VARCHAR2(100) := null;
BEGIN
  l_venduto := D06_GET_VENDUTO( p_venduto => 100000 );
  DBMS_OUTPUT.PUT_LINE( l_venduto );
END;
/
```

PL/SQL

BLOCCO ANONIMO

```
DECLARE
  l_venduto NUMBER := 100000;
BEGIN
   IF l_venduto > 100000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k ' );
   ELSIF l_venduto > 50000 THEN
      DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k ' );
   ELSE
      DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k ' );
   END IF;
END;
/
```

PROCEDURA

```
CREATE OR REPLACE PROCEDURE D06_GET_VENDUTO
                  (
                   p_venduto IN number default 0 ,
                   p_venduto_txt out varchar2
                   )
IS
BEGIN
   IF p_venduto > 100000 THEN
      l_venduto_txt := 'Venduto > 100k ';
   ELSIF p_venduto > 50000 THEN
      l_venduto_txt := 'Venduto > 50k ';
   ELSE
      l_venduto_txt := 'Venduto <= 50k ';
   END IF;
END;
/

DECLARE
l_venduto_txt VARCHAR2(100) := null;
BEGIN
  D06_GET_VENDUTO( p_venduto => 100000 ,
                   p_venduto_txt =>  l_venduto_txt );
  DBMS_OUTPUT.PUT_LINE( l_venduto_txt );
END;
/
```

PL/SQL

PROCEDURE E FUNZIONI

```
CREATE OR REPLACE PROCEDURE D06_VENDUTO ...
...

CREATE OR REPLACE FUNCTION D06_GET_VENDUTO ...
...
```

PACKAGE

```
CREATE OR REPLACE PACKAGE D06_PKG
IS
PROCEDURE VENDUTO (
                    p_venduto IN number default 0
                   );

FUNCTION GET_VENDUTO (
                       p_venduto IN number default 0
                      )
RETURN VARCHAR2;

END D06_PKG;
/
```

PL/SQL

Specifiche del Package

```
CREATE OR REPLACE PACKAGE D06_PKG
IS
PROCEDURE VENDUTO (
                    p_venduto IN number default 0
                   );

FUNCTION GET_VENDUTO (
                      p_venduto IN number default 0
                     )
RETURN VARCHAR2;

END D06_PKG;
/
```

Corpo del Package

```
CREATE OR REPLACE PACKAGE BODY D06_PKG
IS
    PROCEDURE VENDUTO (
                                    p_venduto IN number default 0
                                    )
    IS
    BEGIN
        IF p_venduto > 100000 THEN
            DBMS_OUTPUT.PUT_LINE( 'Venduto > 100k ' );
        ELSIF p_venduto > 50000 THEN
            DBMS_OUTPUT.PUT_LINE( 'Venduto > 50k ' );
        ELSE
            DBMS_OUTPUT.PUT_LINE( 'Venduto <= 50k ' );
        END IF;
    END VENDUTO;

    FUNCTION GET_VENDUTO (
                                    p_venduto IN number default 0
                                    )
                                    RETURN VARCHAR2
    IS
    l_return VARCHAR2(100);
    BEGIN
        IF p_venduto > 100000 THEN
            l_return := 'Venduto > 100k ';
        ELSIF p_venduto > 50000 THEN
            l_return := 'Venduto > 50k ';
        ELSE
            l_return := 'Venduto <= 50k ';
        END IF;
        RETURN l_return;
    END GET_VENDUTO;
END D06_PKG;
/
```

PL/SQL

Richiamo Procedura nel Package

```
BEGIN
  D06_PKG.VENDUTO(p_venduto => 100000);
END;
/
```

Richiamo Funzione nel Package

```
DECLARE
  l_venduto VARCHAR2(100) := null;
BEGIN
  l_venduto := D06_PKG.GET_VENDUTO(p_venduto => 100000);
  DBMS_OUTPUT.PUT_LINE( l_venduto );
END;
/
```

Grouping Sets

Grouping Sets

```
select sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
```

```
select cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by cliente
```

```
select zona_cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by zona_cliente
```

```
select zona_cliente,
       cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by zona_cliente,
          cliente
```

Grouping Sets

```
select null zona_cliente,
       null cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 union all
select zona_cliente,
       null cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by zona_cliente
 union all
select null zona_cliente,
       cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by cliente
 union all
select zona_cliente,
       cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by zona_cliente,
          cliente
```

Grouping Sets

```
select zona_cliente,
       cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by
     grouping sets(
       (zona_cliente,cliente),
       (zona_cliente),
       (cliente),
       ()
     )
```

Grouping Sets

```
select zona_cliente,
       cliente,
       grouping(zona_cliente)                    raggr_zona_cliente,
       grouping(cliente)                         raggr_cliente,
       grouping(zona_cliente)*grouping(cliente) raggr_zona_cliente_cliente,
       sum(quantita) quantita,
       sum(importo) importo
  from d05_vendita
 group by
     grouping sets(
       (zona_cliente,cliente),
       (zona_cliente),
       (cliente),
       ()
     )
order by zona_cliente,cliente
```

Grouping Sets

```
select zona_cliente,
       cliente,
       grouping(zona_cliente)                          raggr_zona_cliente,
       grouping(cliente)                       raggr_cliente,
       grouping(zona_cliente)*grouping(cliente)
raggr_zona_cliente_cliente,
       grouping_id(zona_cliente,cliente)
liv_zona_cliente_cliente,
       sum(quantita) quantita,
       sum(importo) importo,
       case
          when sum(quantita)<> 0
          then round(sum(importo)/sum(quantita),2)
          else 0
       end prezzo_medio
  from d05_vendita
 group by
     grouping sets(
       (zona_cliente,cliente),
       (zona_cliente),
       (cliente),
       ()
     )
order by zona_cliente,cliente
```

Grouping Sets con tutte le combinazioni = Cube

```
select zona_cliente,
       cliente,
       grouping(zona_cliente)                    raggr_zona_cliente,
       grouping(cliente)                 raggr_cliente,
       grouping(zona_cliente)*grouping(cliente)
raggr_zona_cliente_cliente,
       grouping_id(zona_cliente,cliente)
liv_zona_cliente_cliente,
       sum(quantita) quantita,
       sum(importo) importo,
       case
          when sum(quantita)<> 0
          then round(sum(importo)/sum(quantita),2)
          else 0
       end prezzo_medio
  from d05_vendita
 group by
     grouping sets(
       (zona_cliente,cliente),
       (zona_cliente),
       (cliente),
       ()
     )
order by zona_cliente,cliente
```

```
cube(zona_cliente,cliente)
```

Cube

```
select zona_cliente,
       cliente,
       grouping(zona_cliente)                      raggr_zona_cliente,
       grouping(cliente)                           raggr_cliente,
       grouping(zona_cliente)*grouping(cliente)    raggr_zona_cliente_cliente,
       grouping_id(zona_cliente,cliente)           liv_zona_cliente_cliente,
       sum(quantita) quantita,
       sum(importo) importo,
       case
          when sum(quantita)<> 0
          then round(sum(importo)/sum(quantita),2)
          else 0
       end prezzo_medio
  from d05_vendita
 group by
      cube(zona_cliente,cliente)
order by zona_cliente,cliente
```