

PL-SQL

- Procedural Language extensions to the Structured Query Language
- Block structured and Readable Language
- Embedded language only executed in Oracle database
- It has more procedural constructs to SQL language to overcome some limitations of SQL
- PLSQL is a block structured language

A block has 3 sections

1. Declaration section
2. Execution section (Mandate)
3. Exception handling section

A block without name is Anonymous block and can't be reused because it is not saved in oracle database server - one time use

A named block is saved into Oracle Database Server, and it can be reused later

Syntax:

DECLARE

declaration section

BEGIN

execution section

EXCEPTION

exception section

END;

Comments:

-- (Single Line)

/* ... */ (Multi Line)

Data Types:

2 Types

1. Scalar - Holds single value
 - Number (Integers / Floating points)

- Boolean (True / False / Null)
- Character (Char / Varchar2)
- Datetime

2. Composite - Holds multiple values

Records

Collections

Products Table

PLSQL x SQLAdditions

Limit to 1000 rows

```

1 • create database tasks;
2 • use tasks;
3 • create table Products(
4   Product_Id int primary key,
5   Product_Name varchar(25),
6   Qty int,
7   Price int,
8   Category varchar(15),
9   Product_Purchased_Date date
10 );
11 • insert into Products values
12   (1,'Mobile',7,15000,'Electronics','2021-12-11'),
13   (2,'Mobile Charger',5,1000,'Electronics','2021-10-17'),
14   (3,'Office Chair',10,3000,'Furniture','2022-02-15'),
15   (4,'Office Table',15,6000,'Furniture','2023-07-05'),
16   (5,'Laptop',20,40000,'Electronics','2024-03-03'),
17   (6,'Laptop Charger',16,4000,'Electronics','2024-10-07'),
18   (7,'Pen drive',4,400,'Electronics','2024-11-20');

```

Automatic context help disabled. Use the toolbar manually get help for current caret position or toggle automatic help

Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	21:52:38	create database tasks	1 row(s) affected	0.031 sec
2	21:52:38	use tasks	0 row(s) affected	0.000 sec
3	21:52:38	create table Products(Product_Id int primary key, Product_Name varchar(25), Qty int...	0 row(s) affected	0.297 sec
4	21:52:38	insert into Products values (1,'Mobile',7,15000,'Electronics','2021-12-11'), (2,'Mobile C...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0.000 sec

PLSQL x SQLAdditions

Limit to 1000 rows

```

14   (3,'Office Chair',10,3000,'Furniture','2022-02-15'),
15   (4,'Office Table',15,6000,'Furniture','2023-07-05'),
16   (5,'Laptop',20,40000,'Electronics','2024-03-03'),
17   (6,'Laptop Charger',16,4000,'Electronics','2024-10-07'),
18   (7,'Pen drive',4,400,'Electronics','2024-11-20');
19 • select * from Products;

```

Automatic context help disabled. Use the toolbar manually get help for the current caret position or toggle automatic help

Context Help Snippets

Result Grid

Product_Id	Product_Name	Qty	Price	Category	Product_Purchased_Date
1	Mobile	7	15000	Electronics	2021-12-11
2	Mobile Charger	5	1000	Electronics	2021-10-17
3	Office Chair	10	3000	Furniture	2022-02-15
4	Office Table	15	6000	Furniture	2023-07-05
5	Laptop	20	40000	Electronics	2024-03-03
6	Laptop Charger	16	4000	Electronics	2024-10-07
7	Pen drive	4	400	Electronics	2024-11-20

Products 1 x Apply Revert Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
3	21:52:38	create table Products(Product_Id int primary key, Product_Name varchar(25), Qty i...	0 row(s) affected	0.297 sec
4	21:52:38	insert into Products values (1,'Mobile',7,15000,'Electronics','2021-12-11'), (2,'Mobile ...	7 row(s) affected Records: 7 Duplicates: 0 Warnings: 0	0.000 sec
5	21:55:09	select * from Products LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec

Question 1

Create procedure that takes product id as input and returns details of that product details

The screenshot shows the SQL Developer interface with a PL/SQL script editor. The script defines a procedure named `getDetailsOfProducts` that takes an integer `Id` as input. The procedure uses a `select` statement to retrieve details from the `products` table where `Product_Id = Id`. The script also includes a `call` statement to execute the procedure with the value `2`.

```
53  
54 delimiter $$  
55 • create procedure getDetailsOfProducts(IN Id int)  
56 • begin  
57 • select * from products where Product_Id = Id;  
58 • end; $$  
59 delimiter ;  
60  
61 • call getDetailsOfProducts(2);  
62
```

The **Result Grid** shows the output of the procedure call:

Product_Id	Product_Name	Qty	Price	Category	Product_Purchased_Date
2	Mobile Charger	5	1000	Electronics	2021-10-17

The **Output** pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
5	21:55:09	select * from Products LIMIT 0, 1000	7 row(s) returned	0.000 sec / 0.000 sec
6	21:58:12	create procedure getDetailsOfProducts(IN Id int) begin select * from products where...	0 row(s) affected	0.015 sec
7	21:58:17	call getDetailsOfProducts(2)	1 row(s) returned	0.000 sec / 0.000 sec

Question 2

Create function takes price as an argument and returns details of the product whose price is greater than the price input parameter

The screenshot shows the SQL Developer interface with a PL/SQL script editor. The script defines a function named `getDetails` that takes an integer `prod_price` as input. The function uses a `select` statement to retrieve details from the `products` table where `price > prod_price` and limits the result to 1 row. The script also includes a `SELECT` statement to execute the function with the value `400`.

```
57 delimiter $$  
58 • create function getDetails(prod_price int)  
59 • returns varchar(200)  
60 • deterministic  
61 • begin  
62 • declare productDetails varchar(300);  
63 • select concat(Product_Id, ' ', Product_Name, ' ', Qty, ' ', Price, ' ', Category, ' ', Product_Purchased_Date)  
64 • into productDetails from products where price > prod_price limit 1;  
65 • return productDetails;  
66 • end $$  
67 delimiter ;  
68 • SELECT getDetails(400);
```

The **Result Grid** shows the output of the function call:

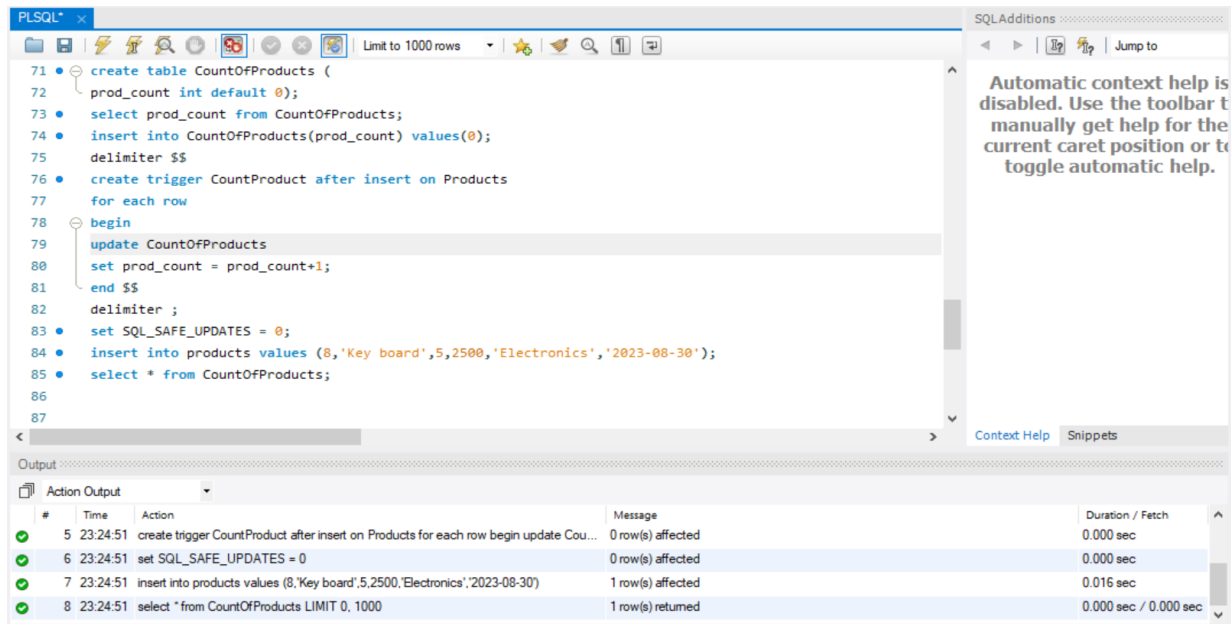
getDetails(400)
1 Mobile 7 15000 Electronics 2021-12-11

The **Output** pane shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	22:06:11	use tasks	0 row(s) affected	0.000 sec
2	22:06:24	create function getDetails(prod_price int) returns varchar(200) deterministic begin	0 row(s) affected	0.000 sec

Question 3

Create a trigger that displays the count of rows after inserting a new record in products table

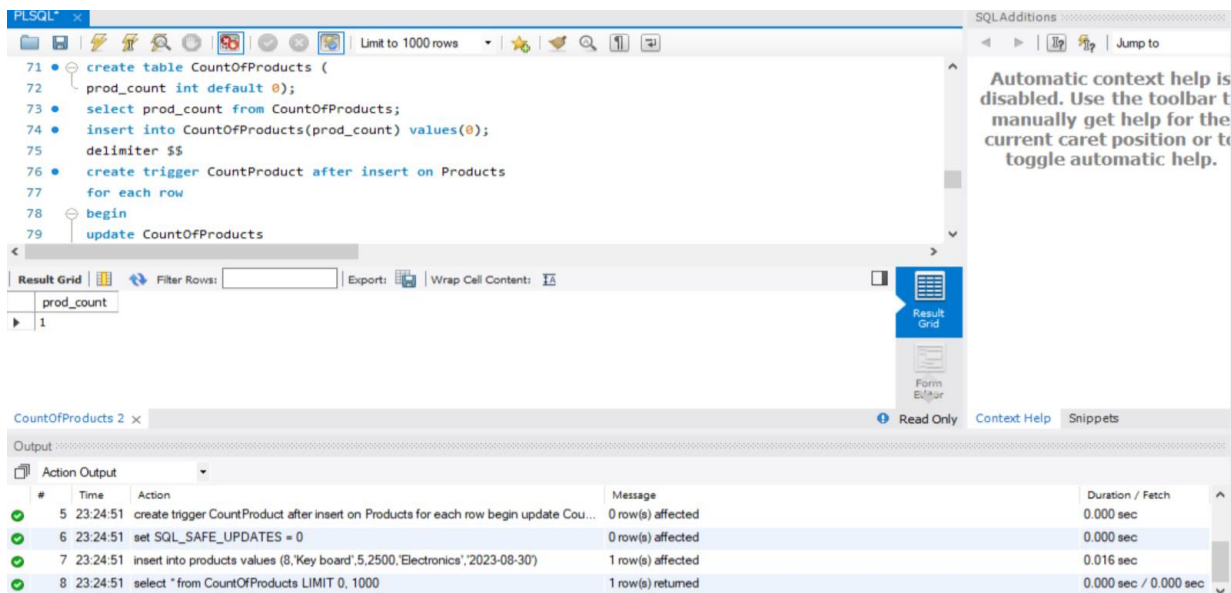


The screenshot shows the SQL Developer interface with the following SQL code:

```
71 • create table CountOfProducts (  
72   prod_count int default 0);  
73 • select prod_count from CountOfProducts;  
74 • insert into CountOfProducts(prod_count) values(0);  
75   delimiter $$  
76 • create trigger CountProduct after insert on Products  
77   for each row  
78   begin  
79     update CountOfProducts  
80     set prod_count = prod_count+1;  
81   end $$  
82   delimiter ;  
83 • set SQL_SAFE_UPDATES = 0;  
84 • insert into products values (8,'Key board',5,2500,'Electronics','2023-08-30');  
85 • select * from CountOfProducts;  
86  
87
```

The output window shows the following results:

#	Time	Action	Message	Duration / Fetch
5	23:24:51	create trigger CountProduct after insert on Products for each row begin update Cou...	0 row(s) affected	0.000 sec
6	23:24:51	set SQL_SAFE_UPDATES = 0	0 row(s) affected	0.000 sec
7	23:24:51	insert into products values (8,'Key board',5,2500,'Electronics','2023-08-30')	1 row(s) affected	0.016 sec
8	23:24:51	select * from CountOfProducts LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec



The screenshot shows the SQL Developer interface with the following SQL code:

```
71 • create table CountOfProducts (  
72   prod_count int default 0);  
73 • select prod_count from CountOfProducts;  
74 • insert into CountOfProducts(prod_count) values(0);  
75   delimiter $$  
76 • create trigger CountProduct after insert on Products  
77   for each row  
78   begin  
79     update CountOfProducts
```

The output window shows the following results:

#	Time	Action	Message	Duration / Fetch
5	23:24:51	create trigger CountProduct after insert on Products for each row begin update Cou...	0 row(s) affected	0.000 sec
6	23:24:51	set SQL_SAFE_UPDATES = 0	0 row(s) affected	0.000 sec
7	23:24:51	insert into products values (8,'Key board',5,2500,'Electronics','2023-08-30')	1 row(s) affected	0.016 sec
8	23:24:51	select * from CountOfProducts LIMIT 0, 1000	1 row(s) returned	0.000 sec / 0.000 sec

Question 4

Create a view that displays the details of Products from Furniture Category and price greater than 1000

The screenshot shows the SQL Developer interface. The SQL Editor contains the following code:

```

87
88 • create view Category as
89   select * from Products where Price > 1000 and Category = 'Furniture';
90   select * from Category;
91
92
93
94
95

```

The Results window displays the output of the second query, showing two rows of furniture products:

Product_Id	Product_Name	Qty	Price	Category	Product_Purchased_Date
3	Office Chair	10	3000	Furniture	2022-02-15
4	Office Table	15	6000	Furniture	2023-07-05

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	23:32:09	use tasks	0 row(s) affected	0.000 sec
2	23:32:18	create view Category as select * from Products where Price > 1000 and Category = 'Furniture';	0 row(s) affected	0.047 sec
3	23:32:18	select * from Category LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec

Question 5

Write a procedure that returns the details of products which will be purchased in this year

The screenshot shows the SQL Developer interface. The SQL Editor contains the following code:

```

70 delimiter $$
71 • create procedure PurchasedDate(in purYear int)
72   begin
73     select * from Products where year(Product_Purchased_Date) = purYear;
74   end $$
75 delimiter ;
76
77 • call PurchasedDate(2024);
78

```

The Results window displays the output of the third query, showing three rows of products purchased in 2024:

Product_Id	Product_Name	Qty	Price	Category	Product_Purchased_Date
5	Laptop	20	40000	Electronics	2024-03-03
6	Laptop Charger	16	4000	Electronics	2024-10-07
7	Pen drive	4	400	Electronics	2024-11-20

The Output window shows the execution log:

#	Time	Action	Message	Duration / Fetch
1	23:03:39	use tasks	0 row(s) affected	0.000 sec
2	23:03:47	create procedure PurchasedDate(in purYear int) begin select * from Products where year(Product_Purchased_Date) = purYear; end	0 row(s) affected	0.032 sec
3	23:03:52	call PurchasedDate(2024)	3 row(s) returned	0.000 sec / 0.000 sec

Question 6

Create a view that displays the details of highest product price

PLSQL

```
68
69
70 -- create a view that displays the details of highest product price
71
72 • create view HighestPrice as
73 select * from Products where Price = (select max(Price) from Products);
74
75 • select * from HighestPrice;
76
```

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

Product_Id	Product_Name	Qty	Price	Category	Product_Purchased_Date
5	Laptop	20	40000	Electronics	2024-03-03

HighestPrice 1

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	22:54:29	use tasks	0 row(s) affected	0.000 sec
2	22:54:35	create view HighestPrice as select * from Products where Price = (select max(Price) f...	0 row(s) affected	0.032 sec
3	22:54:40	select * from HighestPrice LIMIT 0, 1000	1 row(s) returned	0.016 sec / 0.000 sec