

PLSQL

STORED PROCEDURE AND FUNCTIONS

1. Stored Procedure:

The Stored Procedure is which performs one or more specific tasks. It is just like procedures in other programming languages.

Syntax:

```
DELIMITER //
```

```
CREATE [OR REPLACE] PROCEDURE procedure name
```

```
IS
```

```
    declaration section
```

```
BEGIN
```

```
    executable section
```

```
END;
```

```
DELIMITER//
```

Create a Stored Procedure to display the whole table

```
create database products;
```

```
use products;
```

```
create table Allproducts(
```

```
    productID INT primary key,
```

```
    productName varchar(255),
```

```
    productprice decimal (10,2)
```

```
);
```

```
insert into Allproducts (productID,productName,productprice)
```

```
values (1,'laptop',1000.00),
```

```
(2,'smartphone',1000.00),
```

```
(3,'Earphone',50.00),
```

```
(4,'Airpods',500.00);
```

```
*/ Stored Procedure /*
```

```
DELIMITER $$
```

```
Create Procedure GetEntireTable()
```

```
Begin
```

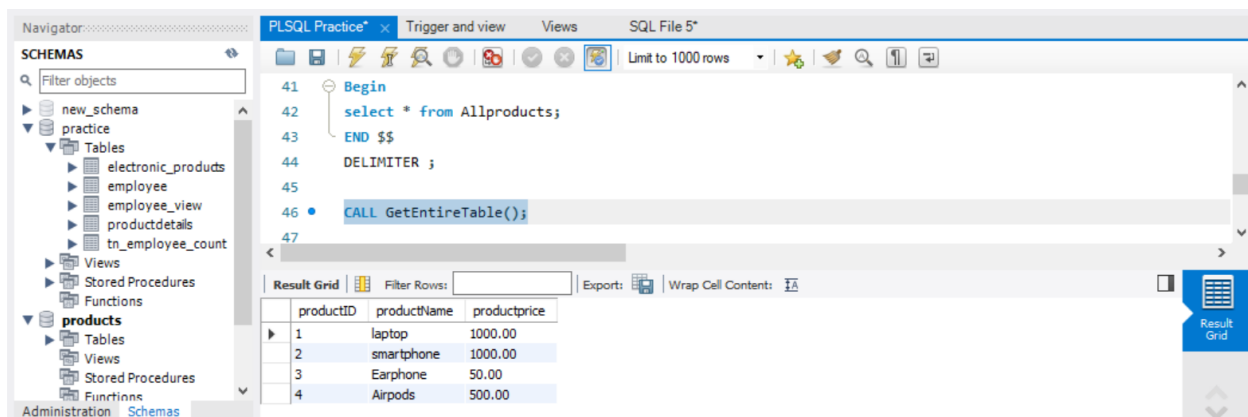
```
select * from Allproducts;
```

```
END $$
```

```
DELIMITER ;
```

```
CALL GetEntireTable();
```

OUTPUT:



The screenshot shows a SQL IDE interface. On the left is a 'Navigator' pane with a tree view of database objects including 'Schemas', 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The 'products' schema is expanded, showing 'Tables' and 'Stored Procedures'. The main editor pane shows a SQL script with the following lines: 41 Begin, 42 select * from Allproducts;, 43 END \$\$, 44 DELIMITER ;, 45, 46 CALL GetEntireTable();, 47. The 'Result Grid' at the bottom displays the output of the query, showing a table with columns 'productID', 'productName', and 'productprice'. The table contains four rows of data.

productID	productName	productprice
1	laptop	1000.00
2	smartphone	1000.00
3	Earphone	50.00
4	Airpods	500.00

Create a Stored Procedure that takes a productID as Input parameter and returns the detail of the product

```
DELIMITER $$
```

```
Create procedure product_ID_filter(in product_ID int)
```

```
begin
```

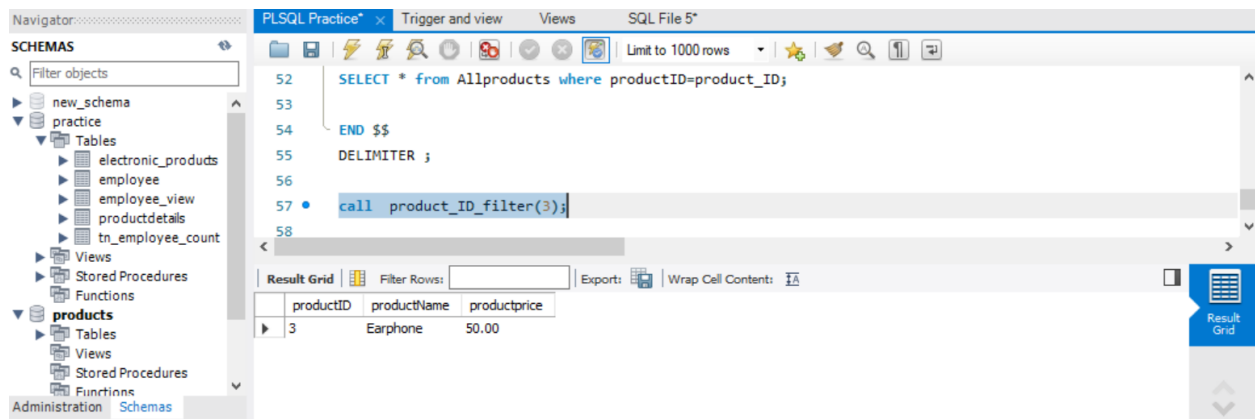
```
SELECT * from Allproducts where productID=product_ID;
```

```
END $$
```

```
DELIMITER ;
```

```
call product_ID_filter(3);
```

OUTPUT:



Create a Function that takes a price as input parameter and returns the detail of the product whose price > argument

DELIMITER \$\$

CREATE FUNCTION get_details()

returns varchar(255)

deterministic

begin

declare product_list varchar(255);

select group_concat(productName separator ',') into product_list

from Allproducts

where productprice > 100;

return product_list;

END \$\$

DELIMITER ;

select get_details();

OUTPUT

The screenshot shows a SQL IDE interface with the following components:

- Navigator:** Displays a tree view of the database schema. The 'products' schema is expanded, showing tables like 'laptop', 'smartphone', and 'airpods'.
- SQL Editor:** Contains a PL/SQL script with the following code:


```

29 where productprice >100;
30
31 return product_list;
32 END $$
33 DELIMITER ;
34 select get_details();
35

```
- Execution Results:** The 'Result Grid' shows the output of the 'get_details()' function call, displaying a list of products: 'laptop,smartphone,Airpods'.
- Toolbar:** Includes icons for file operations, execution, and search. A 'Limit to 1000 rows' dropdown is visible.
- Status Bar:** At the bottom, it indicates 'Result 7' and provides a 'Read Only' status.