



KASPI

KZ

**ESSAN**

**DBMS 2.0**

**GROUP PROJECT**



# WHO WE ARE?

YERASSYL - TABLES, TUPLES

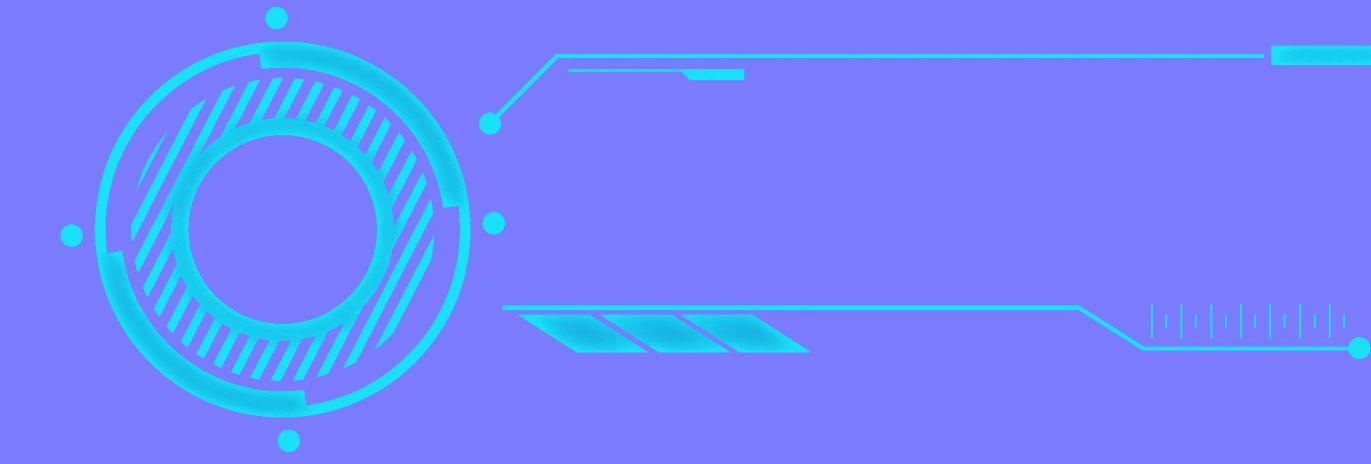
SHADIYAR - NF

SHYNGYS - ERD

ANNUR - ERD

NURDAULET - TEAM LEADER, NF

EXCEPT THESE MAIN DUTIES WE ARE ALL TRIED TO  
HELP EACH OTHER, AND WORKED WELL



YERASSYL - PROCEDURE WHICH USES SQL%ROWCOUNT TO DETERMINE THE NUMBER OF ROWS AFFECTED  
SHADIYAR - FUNCTION WHICH COUNTS THE NUMBER OF RECORDS  
SHYNGYS - PROCEDURE WHICH DOES GROUP BY INFORMATION  
ANNUR - ADD USER-DEFINED EXCEPTION WHICH DISALLOWS TO ENTER TITLE OF ITEM (E.G. BOOK) TO BE LESS THAN 5 CHARACTERS  
NURDAULET - CREATE A TRIGGER BEFORE INSERT ON ANY ENTITY WHICH WILL SHOW THE CURRENT NUMBER OF ROWS IN THE TABLE



# TRIGGER

# Creating the trigger

Language SQL  Rows 10

↻ C | Q | 🔧 A::

```
1 CREATE OR REPLACE TRIGGER PRODUCT_PRICE_TRIG
2 BEFORE INSERT ON PRODUCT
3 FOR EACH ROW
4 DECLARE
5   n NUMBER;
6 BEGIN
7   SELECT COUNT(*) INTO n FROM PRODUCT;
8   DBMS_OUTPUT.PUT_LINE('Before insert product table has ' || n || ' values.');
9 END;
10
```

Results Explain Describe Saved SQL History

Trigger created.

# Empty Product Table

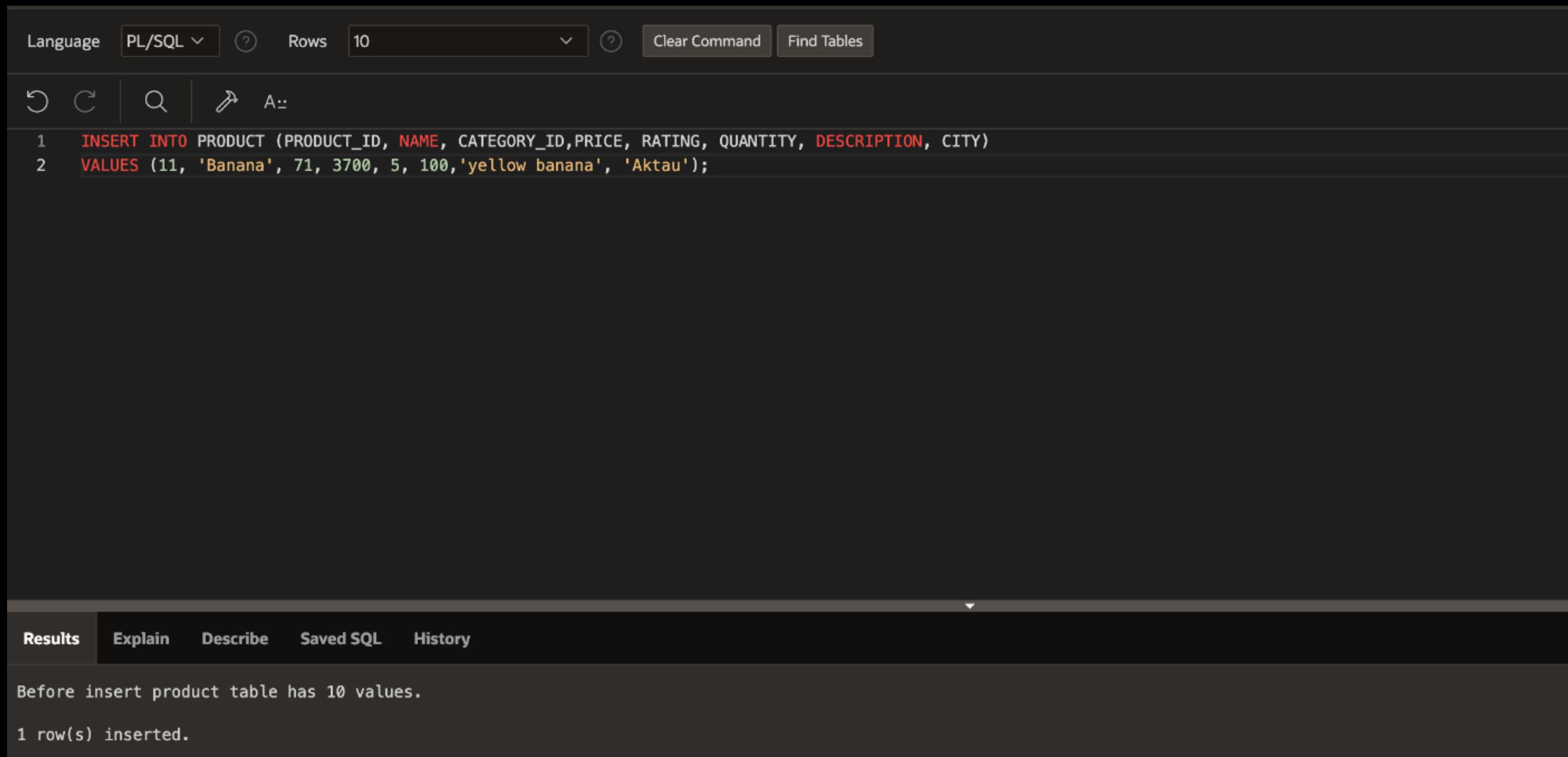
PRODUCT

Columns Data Indexes Constraints Grants Statistics Triggers Dependencies DDL Sample Queries

+ Insert Row Columns... Filter... Count Rows Load Data Download Refresh

	PRODUCT_ID	NAME	QUANTITY	DESCRIPTION	CITY	PRICE	RATING	CATEGORY_ID
	31	Eazzy	90	quam nec dui luctus	Rawa	29614	1	61
	32	Trupe	147	eleifend quam a o...	Curahuasi	27652	4	62
	33	Snaptags	186	arcu libero rutrum ...	Ejidal	32321	1	63
	34	Yacero	135	odio curabitur con...	Novi Slankamen	37195	1	64
	35	Oyondu	53	et ultrices posuere ...	Shazi	25000	2	65
	36	Twitterbeat	58	sed justo pellentes...	Hanjiaji	33628	1	66
	37	Blogtags	200	tortor id nulla ultri...	Olivia	10176	5	67
	38	Linkbridge	140	nunc rhoncus dui ...	Oyskhara	28164	3	68
	39	Twitterbridge	86	eu interdum eu tin...	Butel	19427	2	69
	40	Realmix	96	sed ante vivamus t...	Soanindrariny	39913	4	70

# Inserting a new row into Product Table



The screenshot shows a dark-themed database interface. At the top, there are buttons for 'Language' (set to 'PL/SQL'), 'Rows' (set to '10'), 'Clear Command', and 'Find Tables'. Below the toolbar are icons for back, forward, search, and edit. The main area contains the following SQL code:

```
1  INSERT INTO PRODUCT (PRODUCT_ID, NAME, CATEGORY_ID, PRICE, RATING, QUANTITY, DESCRIPTION, CITY)
2  VALUES (11, 'Banana', 71, 3700, 5, 100, 'yellow banana', 'Aktau');
```

At the bottom, there are tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is selected. The output section displays:

Before insert product table has 10 values.  
1 row(s) inserted.

Message that proofs that trigger is active

# Product table with a inserted new row

PRODUCT

Columns Data Indexes Constraints Grants Statistics Triggers Dependencies DDL Sample Queries

+ Insert Row Columns... Filter... Count Rows Load Data Download Refresh

	ID	USER_ID	PRODUCT_ID	NAME	QUANTITY	DESCRIPTION	CITY	PRICE	RATING	
	1	1	31	Eazzy	90	quam nec dui l...	Rawa	29614	1	
	2	2	32	Trupe	147	eleifend quam ...	Curahuasi	27652	4	
	3	3	33	Snaptags	186	arcu libero rutr...	Ejidal	32321	1	
	4	4	34	Yacero	135	odio curabitur ...	Novi Slankamen	37195	1	
	5	5	35	Oyondu	53	et ultrices pos...	Shazi	25000	2	
	6	6	36	Twitterbeat	58	sed justo pelle...	Hanjiaji	33628	1	
	7	7	37	Blogtags	200	tortor id nulla ...	Olivia	10176	5	
	8	8	38	Linkbridge	140	nunc rhoncus ...	Oyskhara	28164	3	
	9	9	39	Twitterbridge	86	eu interdum e...	Butel	19427	2	
	10	10	40	Realmix	96	sed ante vivam...	Soanindrariny	39913	4	
	21		11	Banana	100	yellow banana	Aktau	3700	5	

**PROCEDURE WHICH DOES GROUP  
BY INFORMATION**

# Creating the procedure

The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop, Team Development, and Gallery, along with a search bar and user profile information. The main area is titled "SQL Commands" and displays a PL/SQL code editor. The code is a procedure named "esepte\_products\_with\_rating\_1" that selects cities with a rating of 1 and prints them to the screen. The code is numbered from 1 to 13. Below the code editor are tabs for Results, Explain, Describe, Saved SQL, and History. The results section shows the message "Procedure created." and a execution time of "0.04 seconds".

```
1 CREATE OR REPLACE PROCEDURE esepte_products_with_rating_1 IS
2   CURSOR qwe_product_ratings IS
3     SELECT city, COUNT(*) AS num_products
4     FROM product
5     WHERE rating = 1
6     GROUP BY city;
7 BEGIN
8   FOR ss_product_ratings IN qwe_product_ratings LOOP
9     DBMS_OUTPUT.PUT_LINE(ss_product_ratings.city || ':' || ss_product_ratings.num_products);
10  END LOOP;
11 END;
12
13 |
```

Results Explain Describe Saved SQL History

Procedure created.

0.04 seconds

# Empty Product Table

PRODUCT

Columns Data Indexes Constraints Grants Statistics Triggers Dependencies DDL Sample Queries

+ Insert Row Columns... Filter... Count Rows Load Data Download Refresh

	PRODUCT_ID	NAME	QUANTITY	DESCRIPTION	CITY	PRICE	RATING	CATEGORY_ID
	31	Eazzy	90	quam nec dui luctus	Rawa	29614	1	61
	32	Trupe	147	eleifend quam a o...	Curahuasi	27652	4	62
	33	Snaptags	186	arcu libero rutrum ...	Ejidal	32321	1	63
	34	Yacero	135	odio curabitur con...	Novi Slankamen	37195	1	64
	35	Oyondu	53	et ultrices posuere ...	Shazi	25000	2	65
	36	Twitterbeat	58	sed justo pellentes...	Hanjiaji	33628	1	66
	37	Blogtags	200	tortor id nulla ultri...	Olivia	10176	5	67
	38	Linkbridge	140	nunc rhoncus dui ...	Oyskhara	28164	3	68
	39	Twitterbridge	86	eu interdum eu tin...	Butel	19427	2	69
	40	Realmix	96	sed ante vivamus t...	Soanindrariny	39913	4	70

# In result we will see this

The screenshot shows a database query editor interface. At the top, there's a header with "SQL Commands" and a schema dropdown set to "WKSP\_SSLMZHN". Below the header are buttons for "Language" (set to "PL/SQL"), "Rows" (set to 10), "Clear Command", and "Find Tables". On the right side of the header are "Save" and "Run" buttons. The main area contains a code editor with the following PL/SQL code:

```
1 BEGIN
2   septe_products_with_rating_1;
3 END;
```

Below the code editor, there are icons for refresh, copy, search, and other actions. The bottom navigation bar includes tabs for "Results", "Explain", "Describe", "Saved SQL", and "History", with "Results" being the active tab. The results pane displays the output of the executed query:

```
Ejidal: 1
Rawa: 1
Novi Slankamen: 1
Hanjiaji: 1

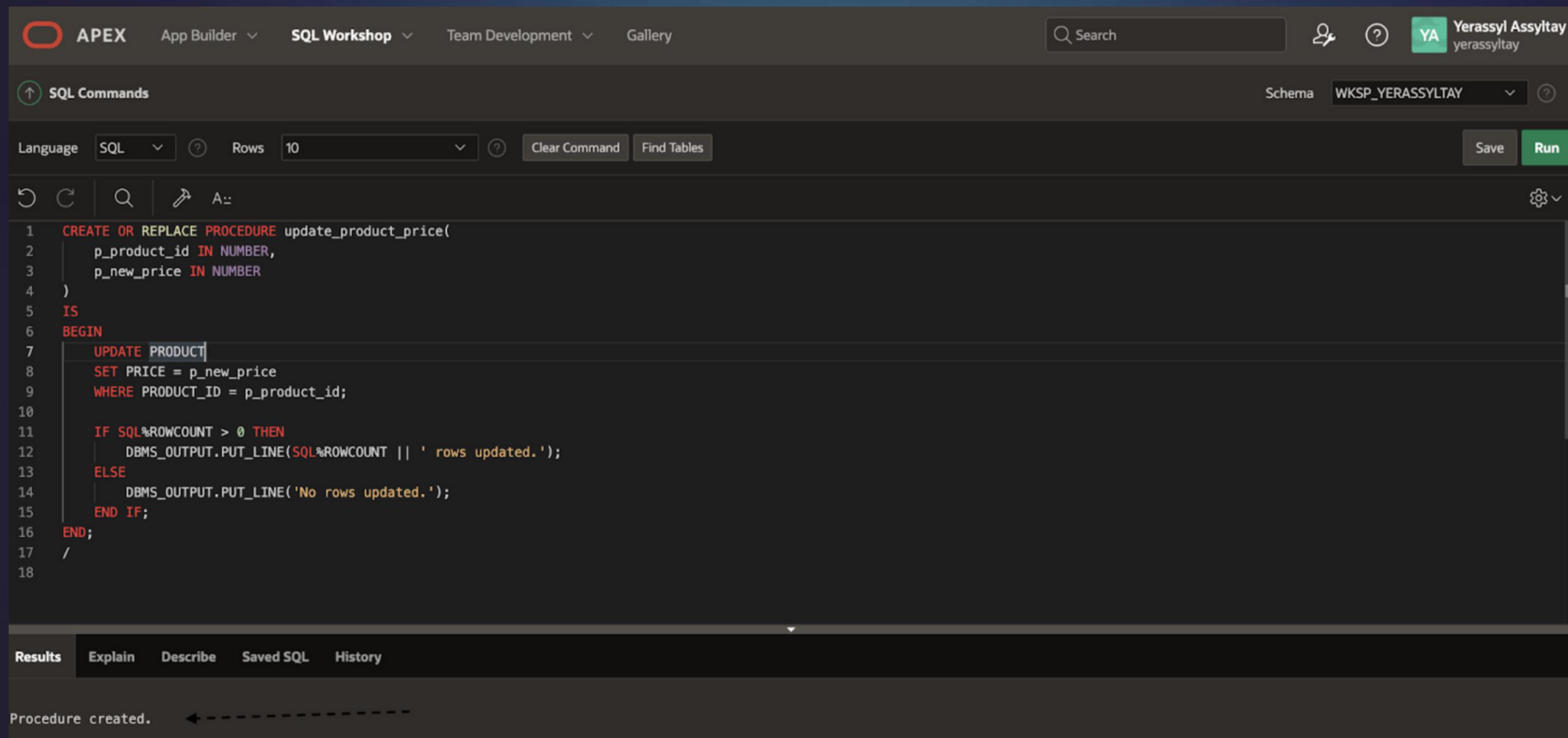
Statement processed.
```

OKAY, NOW WE WILL LOOK AT  
ONE OF THE BASIC CONSTRUCTS  
IN PL/SQL PROCEDURES.

PL/  
SQL



IN THIS EXAMPLE, THE PROCEDURE TAKES A PRODUCT ID AND A NEW PRICE AS INPUT PARAMETERS AND UPDATES THE PRICES OF THE CORRESPONDING PRODUCT IN THE "PRODUCT" TABLE.



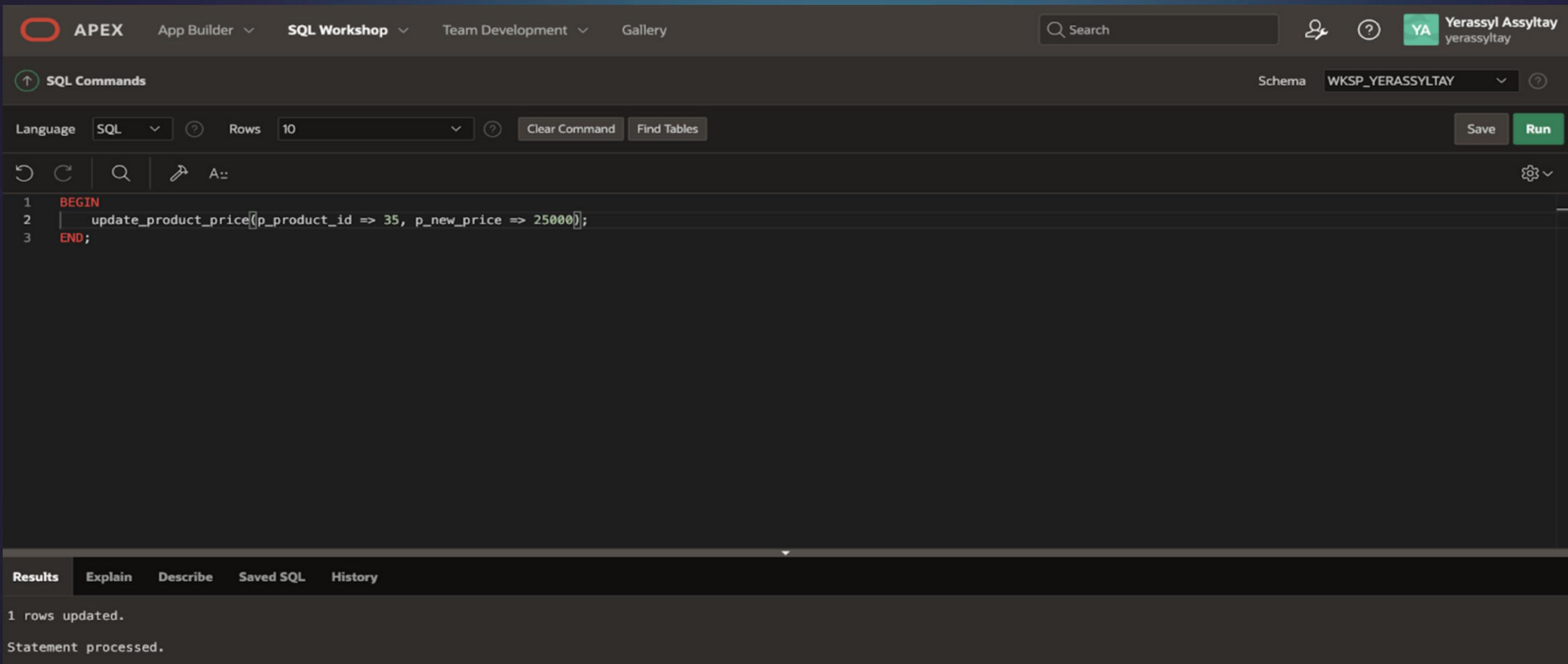
The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes links for APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side of the header shows a user profile for 'Yerassyl Assyltay' (YA). The main workspace is titled 'SQL Commands' and contains the following PL/SQL code:

```
1 CREATE OR REPLACE PROCEDURE update_product_price(
2     p_product_id IN NUMBER,
3     p_new_price IN NUMBER
4 )
5 IS
6 BEGIN
7     UPDATE PRODUCT|
8     SET PRICE = p_new_price
9     WHERE PRODUCT_ID = p_product_id;
10
11    IF SQL%ROWCOUNT > 0 THEN
12        DBMS_OUTPUT.PUT_LINE(SQL%ROWCOUNT || ' rows updated.');
13    ELSE
14        DBMS_OUTPUT.PUT_LINE('No rows updated.');
15    END IF;
16
17 /
18
```

The code is syntax-highlighted, with keywords in red and identifiers in blue. The 'Run' button is visible in the toolbar above the code editor. At the bottom of the screen, a message 'Procedure created.' is displayed next to a dashed arrow pointing left.

THE SQL%ROWCOUNT ATTRIBUTE IS USED TO DETERMINE THE NUMBER OF ROWS UPDATED BY THE UPDATE STATEMENT.

and we will write requests



The screenshot shows the Oracle SQL Workshop interface. The top navigation bar includes APEX, App Builder, SQL Workshop (selected), Team Development, and Gallery. The right side shows a user profile for Yerassyl Assyltay (yerassyltay). The main area is titled "SQL Commands". The language is set to SQL, and the number of rows is set to 10. The command entered is:

```
1 BEGIN
2 |   update_product_price(p_product_id => 35, p_new_price => 25000);
3 END;
```

The results section at the bottom shows the output of the executed command:

```
1 rows updated.
Statement processed.
```

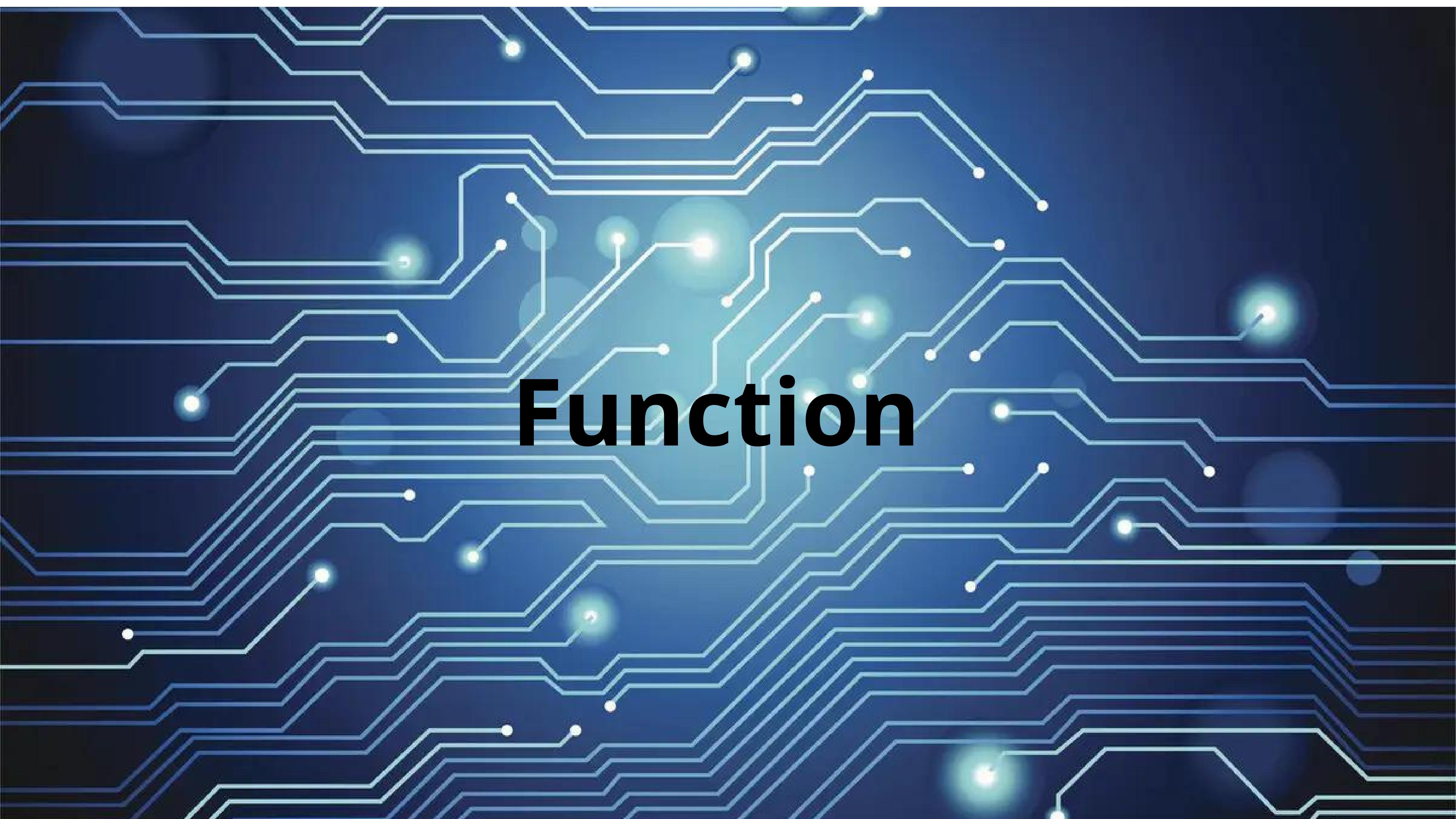
SO WE SEE HERE THE PRICE HAS BEEN  
CHANGED  
AND OUR CODE WORKED

**PRODUCT**

Columns Data Indexes Constraints Grants Statistics Triggers Dependencies DDL Sample Queries

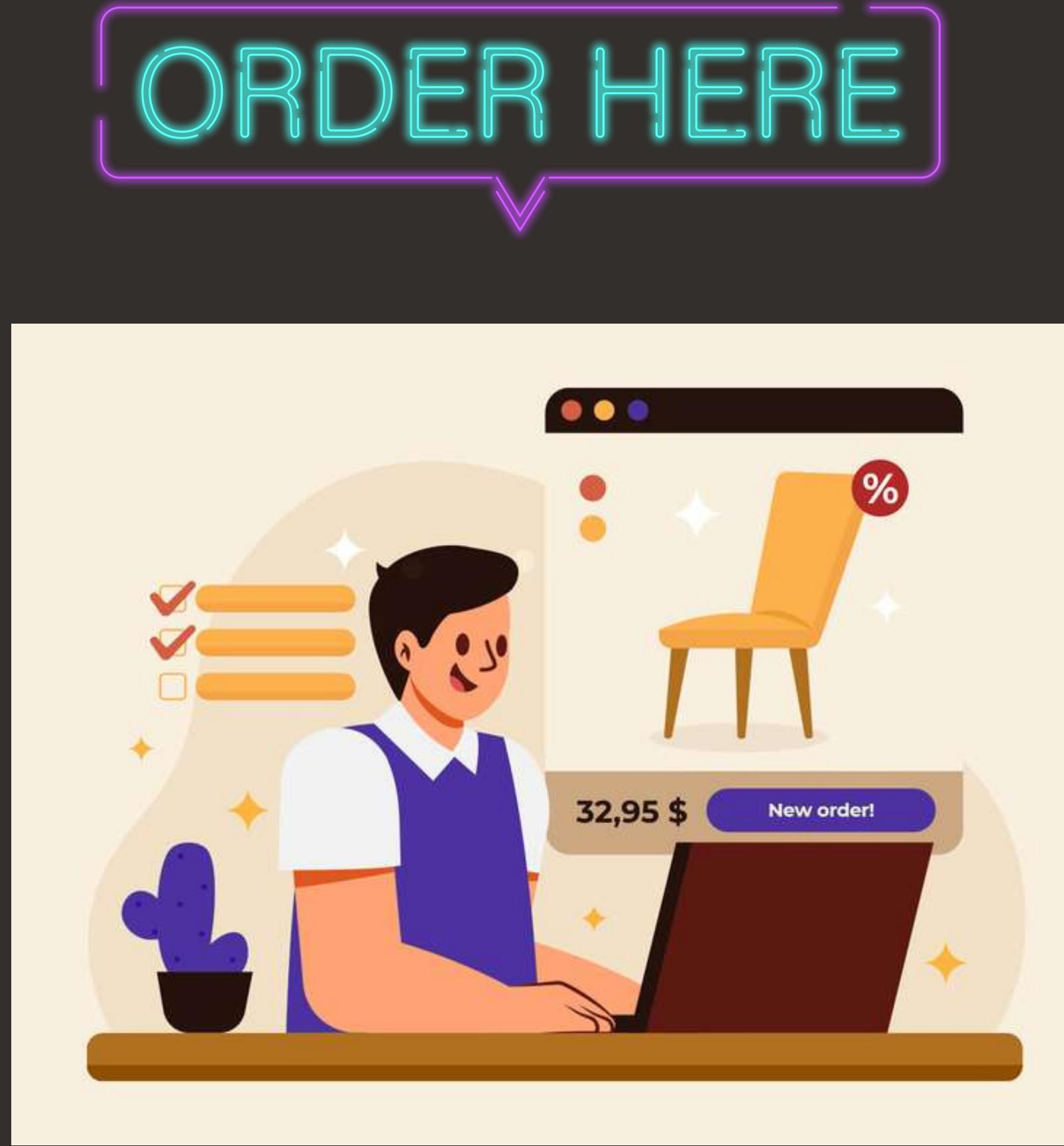
+ Insert Row Columns... Filter... Count Rows Load Data Download Refresh

	PRODUCT_ID	NAME	QUANTITY	DESCRIPTION	CITY	PRICE	RATING	CATEGORY_ID
	31	Eazzy	90	quam nec dui luctus	Rawa	29614	1	61
	32	Trupe	147	eleifend quam a o...	Curahuasi	27652	4	62
	33	Snaptags	186	arcu libero rutrum...	Ejidal	32321	1	63
	34	Yacero	135	odio curabitur con...	Novi Slankamen	37195	1	64
	35	Oyondu	53	et ultrices posuer...	Shazi	25000	2	65
	36	Twitterbeat	58	sed justo pellente...	Hanjiaji	33628	1	66
	37	Blogtags	200	tortor id nulla ultri...	Olivia	10176	5	67
	38	Linkbridge	140	nunc rhoncus dui ...	Oyskhara	28164	3	68
	39	Twitterbridge	86	eu interdum eu ti...	Butel	19427	2	69
	40	Realmix	96	sed ante vivamus ...	Soanindrariny	39913	4	70



The background features a complex network of blue and white circuit board traces on a dark blue gradient. Numerous glowing white and yellow circular nodes are scattered across the board, connected by the traces. Some nodes are larger and more prominent, while others are smaller. The overall effect is a futuristic and technological feel.

# Function



```
1 create or replace FUNCTION count_orders
2   RETURN NUMBER
3 IS
4   count_order NUMBER;
5 BEGIN
6   SELECT COUNT(order_id) INTO count_order FROM orders;
7   RETURN count_order;
8 END;
9 /
```

Results Explain Describe Saved SQL History

Function created.

```
1 select count_orders from dual
```

Results

Explain

Describe

Saved SQL

History

COUNT_ORDERS
10



```
1  create or replace FUNCTION get_avg_rating(rating IN VARCHAR2)
2  RETURN NUMBER
3  IS
4      l_avg_rating NUMBER;
5  BEGIN
6      SELECT AVG(TO_NUMBER(rating))
7      INTO l_avg_rating
8      FROM product;
9      RETURN l_avg_rating;
10 EXCEPTION
11     WHEN NO_DATA_FOUND THEN
12         RETURN NULL;
13 END;
14 /
```

Results

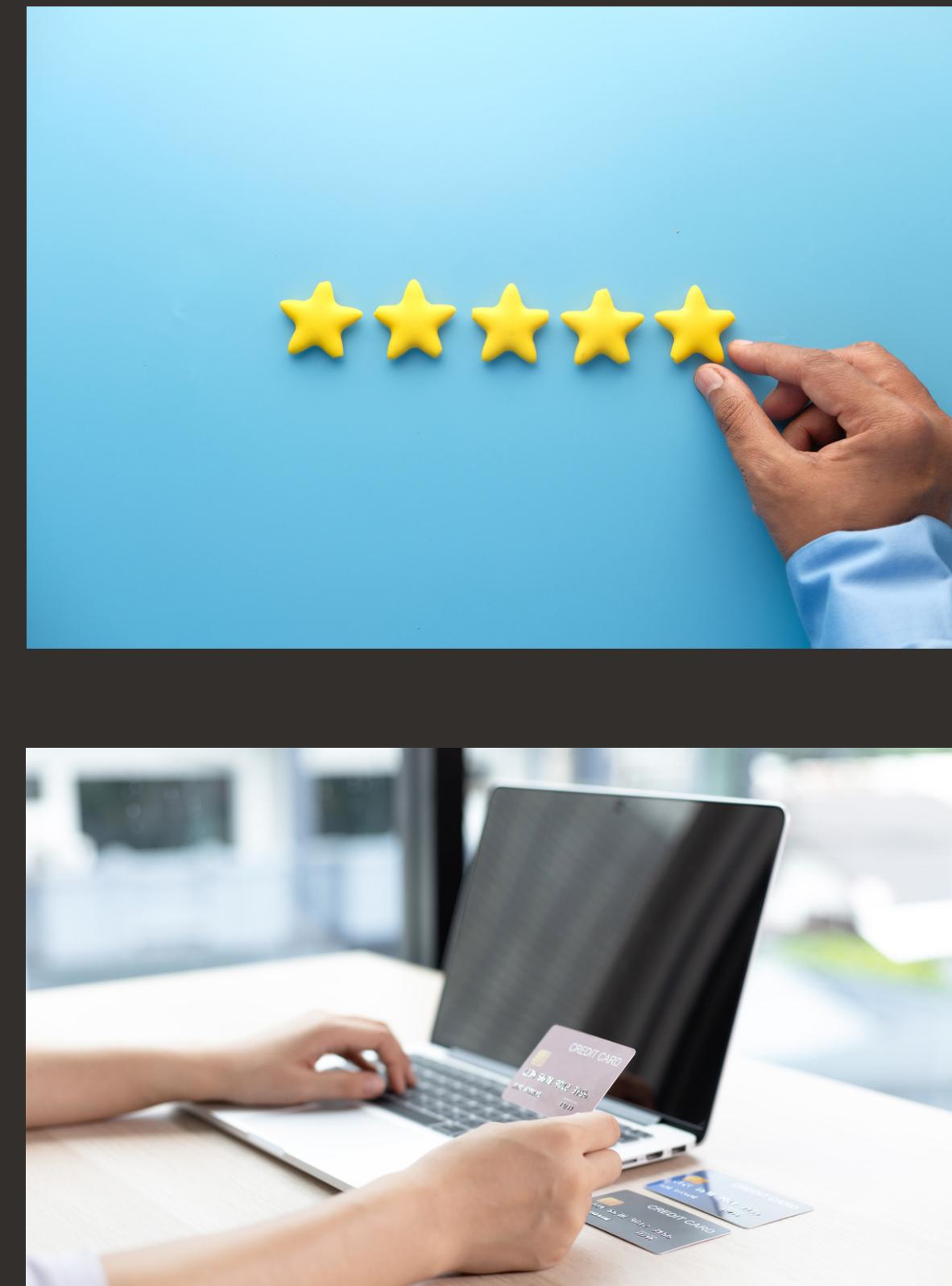
Explain

Describe

Saved SQL

History

Function created.



```
1 select get_avg_rating ('rating') as av_rating  
2 from dual;
```

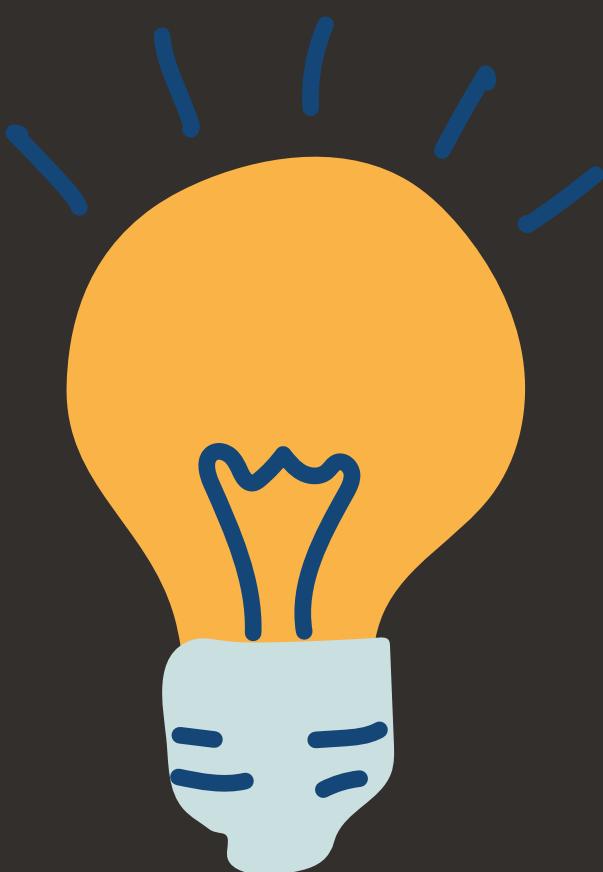
Results Explain Describe Saved SQL History

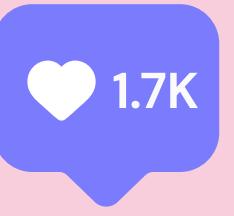
2.4

1 rows returned in 0.12 seconds

[Download](#)

AV\_RATING





**THANK YOU FOR  
YOUR ATTENTION**

