

EXERCISE-17

TRIGGER

Name: Vedhasree S

Register Number: 240701580

Department: CSE

PROGRAM 1

Write a code in PL/SQL to develop a trigger that enforces referential integrity by preventing the deletion of a parent record if child records exist.

The screenshot displays the Oracle APEX SQL Workshop interface. At the top, the navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is active, showing a language dropdown set to 'PL/SQL' and a 'Rows' limit of 20. The main editor contains the following PL/SQL code:

```
1 CREATE OR REPLACE TRIGGER prevent_parent_delete
2 BEFORE DELETE ON departments
3 FOR EACH ROW
4 DECLARE
5     v_count NUMBER;
6 BEGIN
7     SELECT COUNT(*)
8     INTO v_count
9     FROM employees
10    WHERE department_id = :OLD.department_id;
11    IF v_count > 0 THEN
12        RAISE_APPLICATION_ERROR(-20001, 'Cannot delete department. Employees exist in this department.');

Below the editor, the 'Results' tab is selected, showing the message 'Trigger created.' and the execution time '0.07 seconds'.


```

APEX

App Builder

SQL Workshop

Team Development

Gallery

Q Search

↑ SQL Commands

Schema WKSP_DBMS05

Language PL/SQL Rows 20 Clear Command Find Tables

A:

1 DELETE FROM departments WHERE department_id = 80;

Results

Explain

Describe

Saved SQL

History

ORA-20001: Cannot delete department. Employees exist in this department.
ORA-06512: at "WKSP_DBMS05.PREVENT_PARENT_DELETE", line 12
ORA-04088: error during execution of trigger "WKSP_DBMS05.PREVENT_PARENT_DELETE"

1. DELETE FROM departments WHERE department_id = 80;

0.02 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language PL/SQL Rows 20 Clear Command Find Tables

A:

1 DELETE FROM departments WHERE department_id = 20;

Results

Explain

Describe

Saved SQL

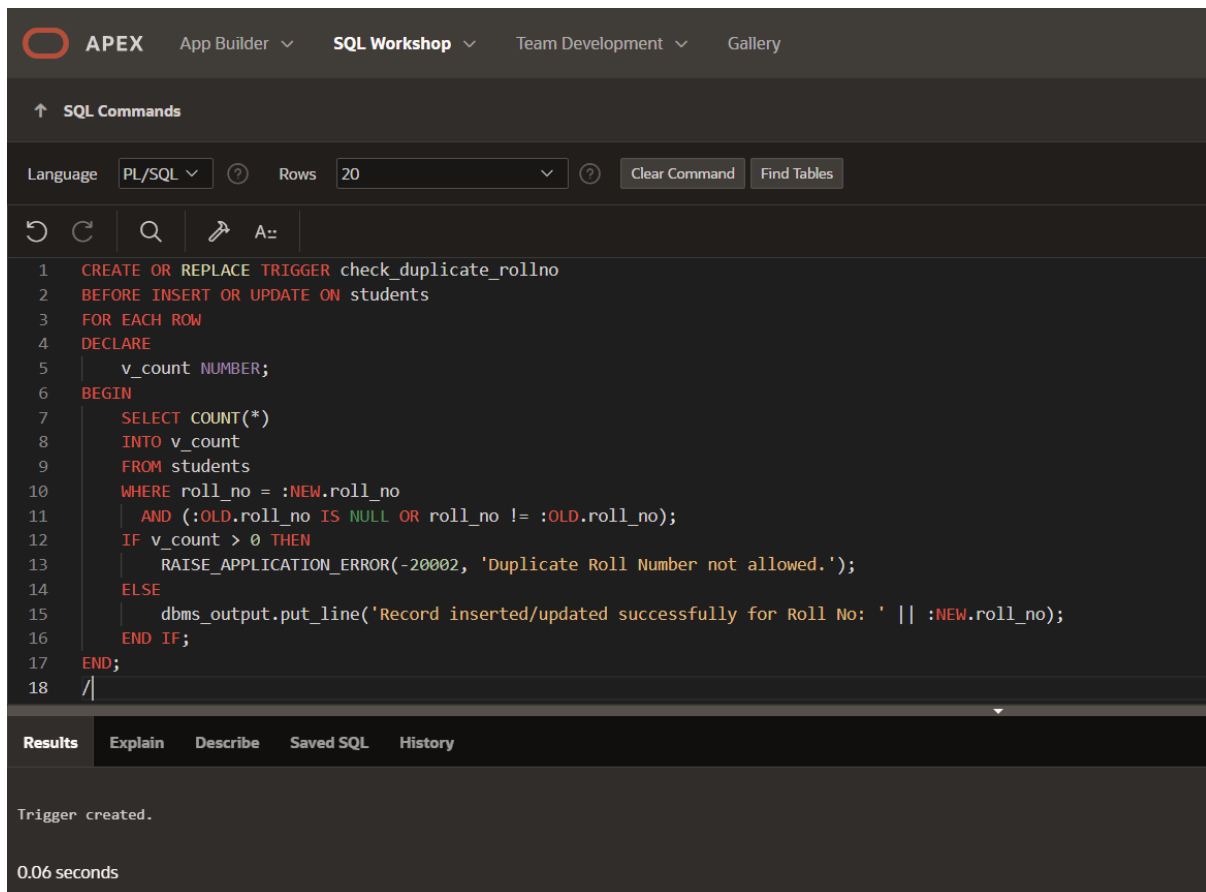
History

1 row(s) deleted.

0.01 seconds

PROGRAM 2

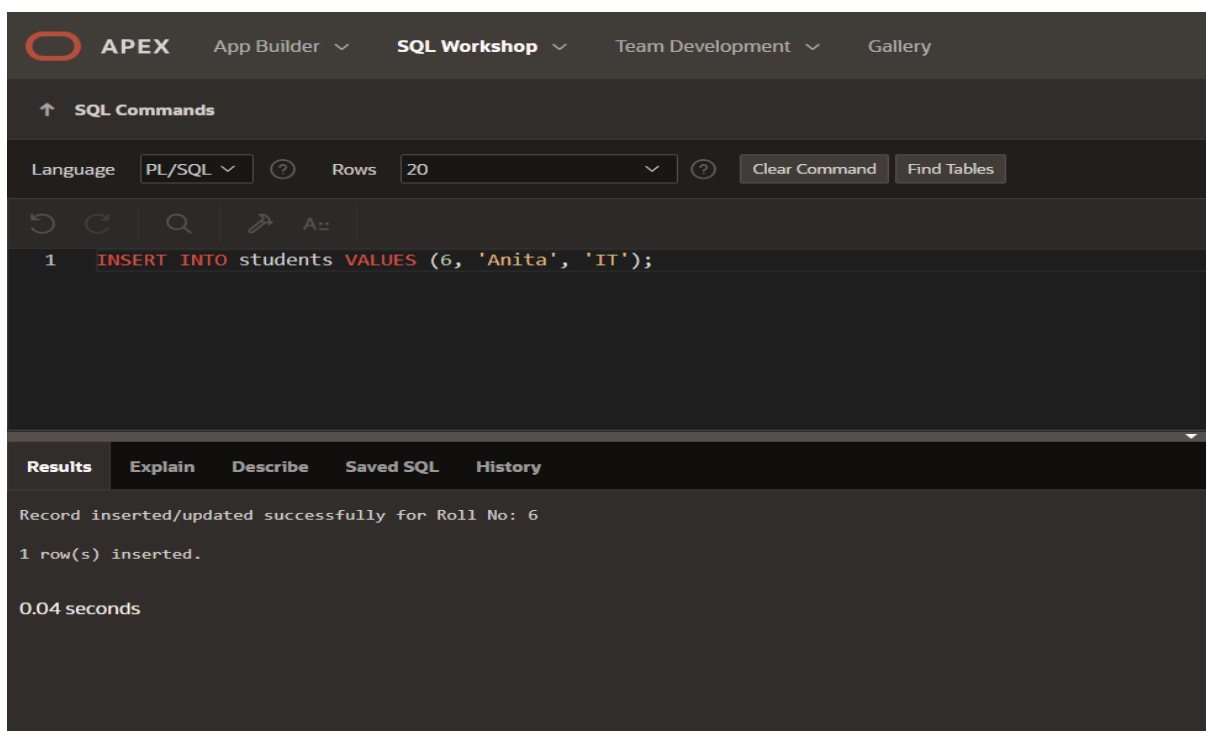
Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' tab is active, showing a PL/SQL code editor with the following code:

```
1 CREATE OR REPLACE TRIGGER check_duplicate_rollno
2 BEFORE INSERT OR UPDATE ON students
3 FOR EACH ROW
4 DECLARE
5     v_count NUMBER;
6 BEGIN
7     SELECT COUNT(*)
8     INTO v_count
9     FROM students
10    WHERE roll_no = :NEW.roll_no
11    AND (:OLD.roll_no IS NULL OR roll_no != :OLD.roll_no);
12    IF v_count > 0 THEN
13        RAISE_APPLICATION_ERROR(-20002, 'Duplicate Roll Number not allowed.');
```

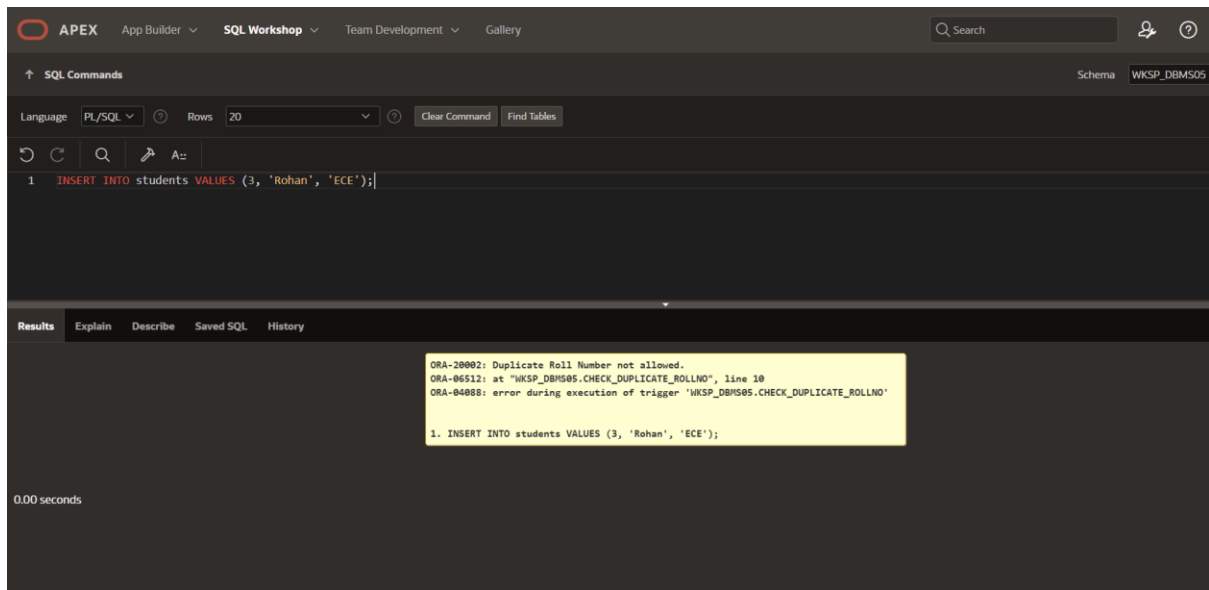
The code continues with an ELSE block that outputs a success message, followed by END IF, END, and a slash. The 'Results' tab is selected, displaying 'Trigger created.' and '0.06 seconds'.



The screenshot shows the APEX SQL Workshop interface with the 'SQL Commands' tab active. The code editor contains a single SQL statement:

```
1 INSERT INTO students VALUES (6, 'Anita', 'IT');
```

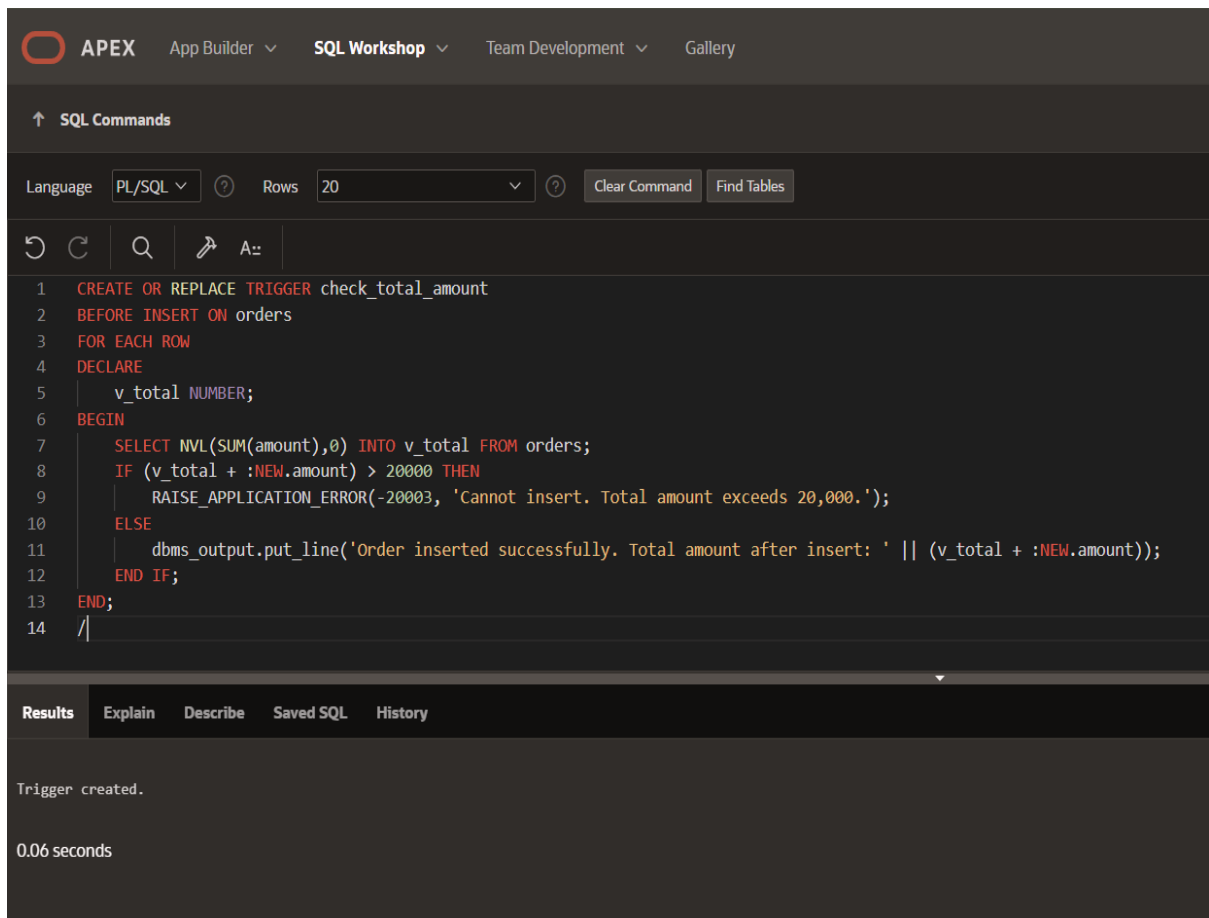
The 'Results' tab is selected, displaying the output: 'Record inserted/updated successfully for Roll No: 6', '1 row(s) inserted.', and '0.04 seconds'.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. A search bar is on the right. Below the navigation bar, the 'SQL Commands' section is active, showing a single command: `1 INSERT INTO students VALUES (3, 'Rohan', 'ECE');`. The 'Results' tab is selected, displaying an error message in a yellow box:
ORA-20002: Duplicate Roll Number not allowed.
ORA-06512: at "WKSP_DBMS05.CHECK_DUPLICATE_ROLLNO", line 10
ORA-04088: error during execution of trigger 'WKSP_DBMS05.CHECK_DUPLICATE_ROLLNO'
1. INSERT INTO students VALUES (3, 'Rohan', 'ECE');

PROGRAM 3

Write a code in PL/SQL to create a trigger that restricts the insertion of new rows if the total of a column's values exceeds a certain threshold.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, the 'SQL Commands' section is active, showing a PL/SQL block:
1 CREATE OR REPLACE TRIGGER check_total_amount
2 BEFORE INSERT ON orders
3 FOR EACH ROW
4 DECLARE
5 v_total NUMBER;
6 BEGIN
7 SELECT NVL(SUM(amount),0) INTO v_total FROM orders;
8 IF (v_total + :NEW.amount) > 20000 THEN
9 RAISE_APPLICATION_ERROR(-20003, 'Cannot insert. Total amount exceeds 20,000.');10 ELSE
11 dbms_output.put_line('Order inserted successfully. Total amount after insert: ' || (v_total + :NEW.amount));
12 END IF;
13 END;
14 /

The 'Results' tab is selected, displaying the message:
Trigger created.
0.06 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

↺

↻

🔍

🔗

A=

1 `INSERT INTO orders VALUES (6, 'Anita', 2000);`

Results

Explain

Describe

Saved SQL

History

Order inserted successfully. Total amount after insert: 17000

1 row(s) inserted.

0.04 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

🔍 Search

👤

?

↑ SQL Commands

Schema WKSP_DBMS05

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

↺

↻

🔍

🔗

A=

1 `INSERT INTO orders VALUES (7, 'Rohan', 4000);`

Results

Explain

Describe

Saved SQL

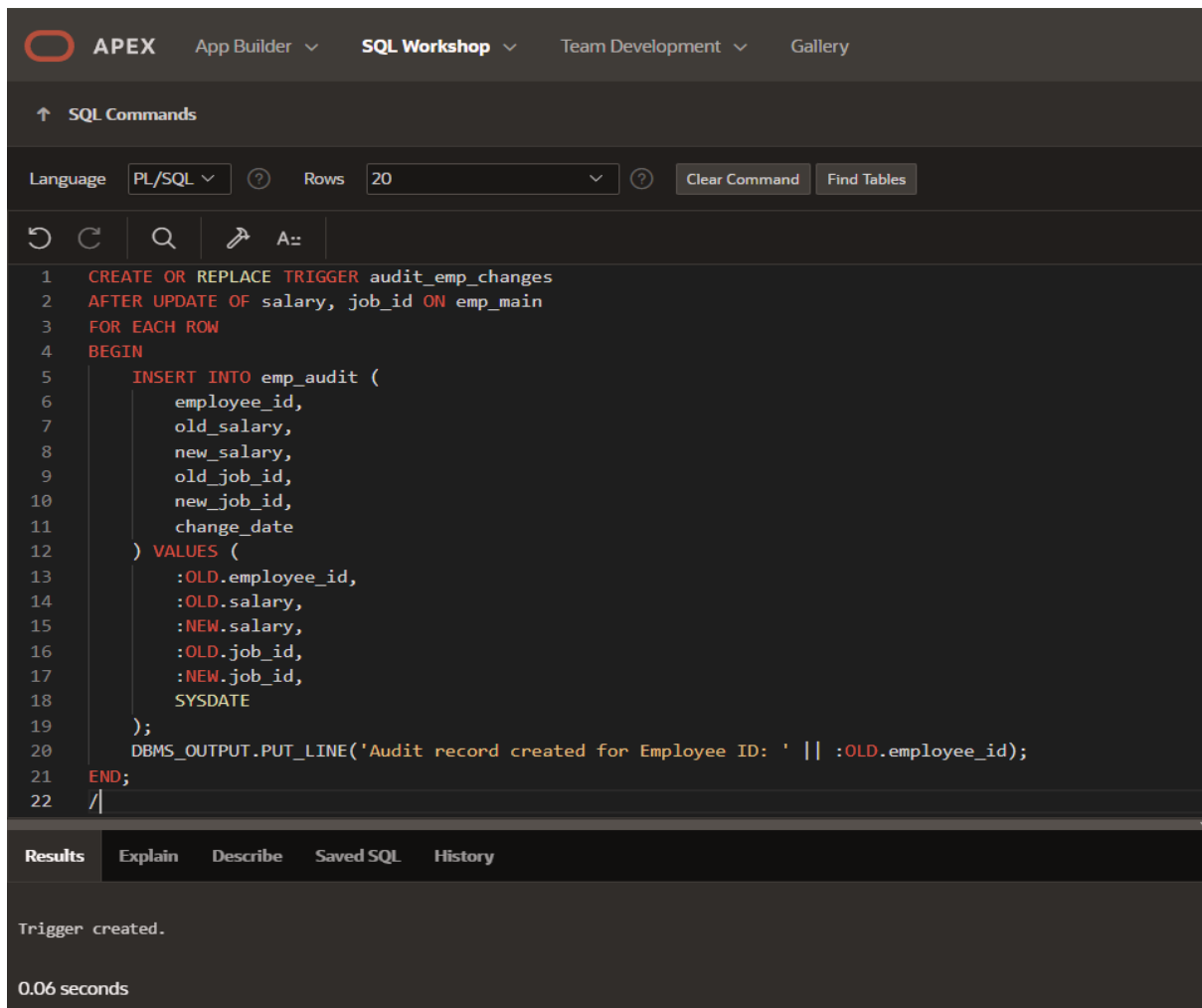
History

ORA-20003: Cannot insert. Total amount exceeds 20,000.
ORA-06512: at "WKSP_DBMS05.CHECK_TOTAL_AMOUNT", line 6
ORA-04088: error during execution of trigger 'WKSP_DBMS05.CHECK_TOTAL_AMOUNT'

0.00 seconds

PROGRAM 4

Write a code in PL/SQL to design a trigger that captures changes made to specific columns and logs them in an audit table.



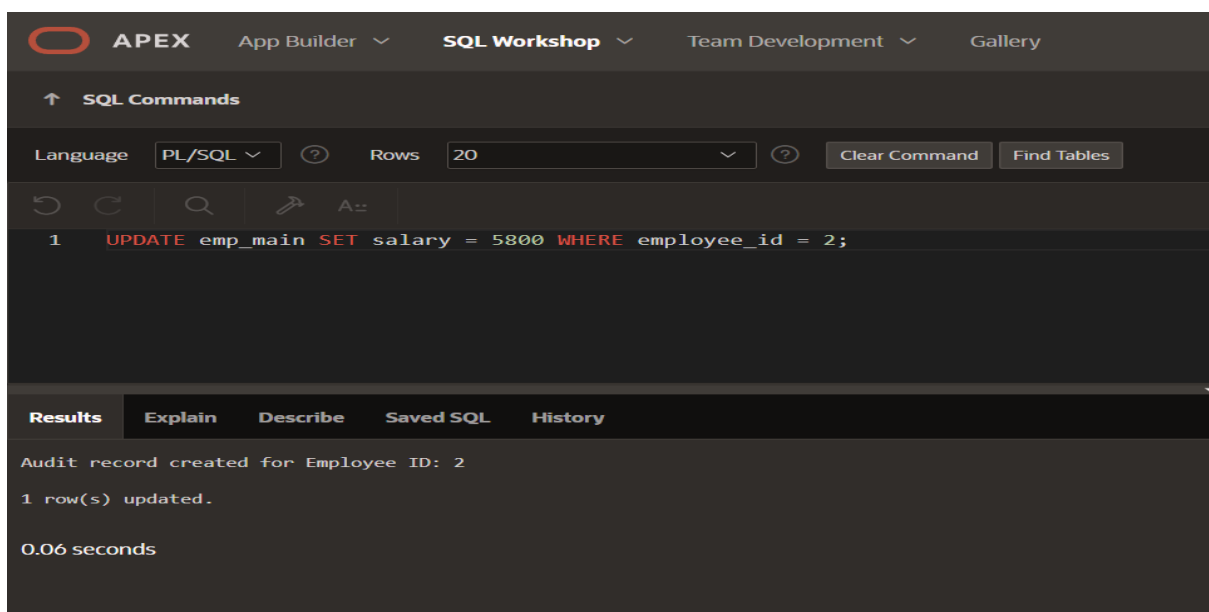
The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a PL/SQL command. The command is a trigger named 'audit_emp_changes' that fires after an update on the 'salary' column of the 'emp_main' table. The trigger body inserts a record into the 'emp_audit' table with columns: employee_id, old_salary, new_salary, old_job_id, new_job_id, and change_date. The change_date is set to SYSDATE. A message 'Audit record created for Employee ID: ' || :OLD.employee_id;' is displayed. The command is executed, and the 'Results' tab shows 'Trigger created.' and '0.06 seconds'.

```
1 CREATE OR REPLACE TRIGGER audit_emp_changes
2 AFTER UPDATE OF salary, job_id ON emp_main
3 FOR EACH ROW
4 BEGIN
5     INSERT INTO emp_audit (
6         employee_id,
7         old_salary,
8         new_salary,
9         old_job_id,
10        new_job_id,
11        change_date
12    ) VALUES (
13        :OLD.employee_id,
14        :OLD.salary,
15        :NEW.salary,
16        :OLD.job_id,
17        :NEW.job_id,
18        SYSDATE
19    );
20    DBMS_OUTPUT.PUT_LINE('Audit record created for Employee ID: ' || :OLD.employee_id);
21 END;
22 /
```

Results Explain Describe Saved SQL History

Trigger created.

0.06 seconds



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a PL/SQL command. The command is an update query that sets the 'salary' column to 5800 for the row where 'employee_id' is 2. The command is executed, and the 'Results' tab shows 'Audit record created for Employee ID: 2' and '1 row(s) updated.' and '0.06 seconds'.

```
1 UPDATE emp_main SET salary = 5800 WHERE employee_id = 2;
```

Results Explain Describe Saved SQL History

Audit record created for Employee ID: 2

1 row(s) updated.

0.06 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

↶ ↷ 🔍 ↵ A::

1 UPDATE emp_main SET job_id = 'CSE_PROG' WHERE employee_id = 3;

Results

Explain

Describe

Saved SQL

History

Audit record created for Employee ID: 3

1 row(s) updated.

0.03 seconds

↑ SQL Commands

Schema

WKSP_DBMS05

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

Save

Run

↶ ↷ 🔍 ↵ A:: ⚙

1 SELECT * FROM emp_audit;

Results

Explain

Describe

Saved SQL

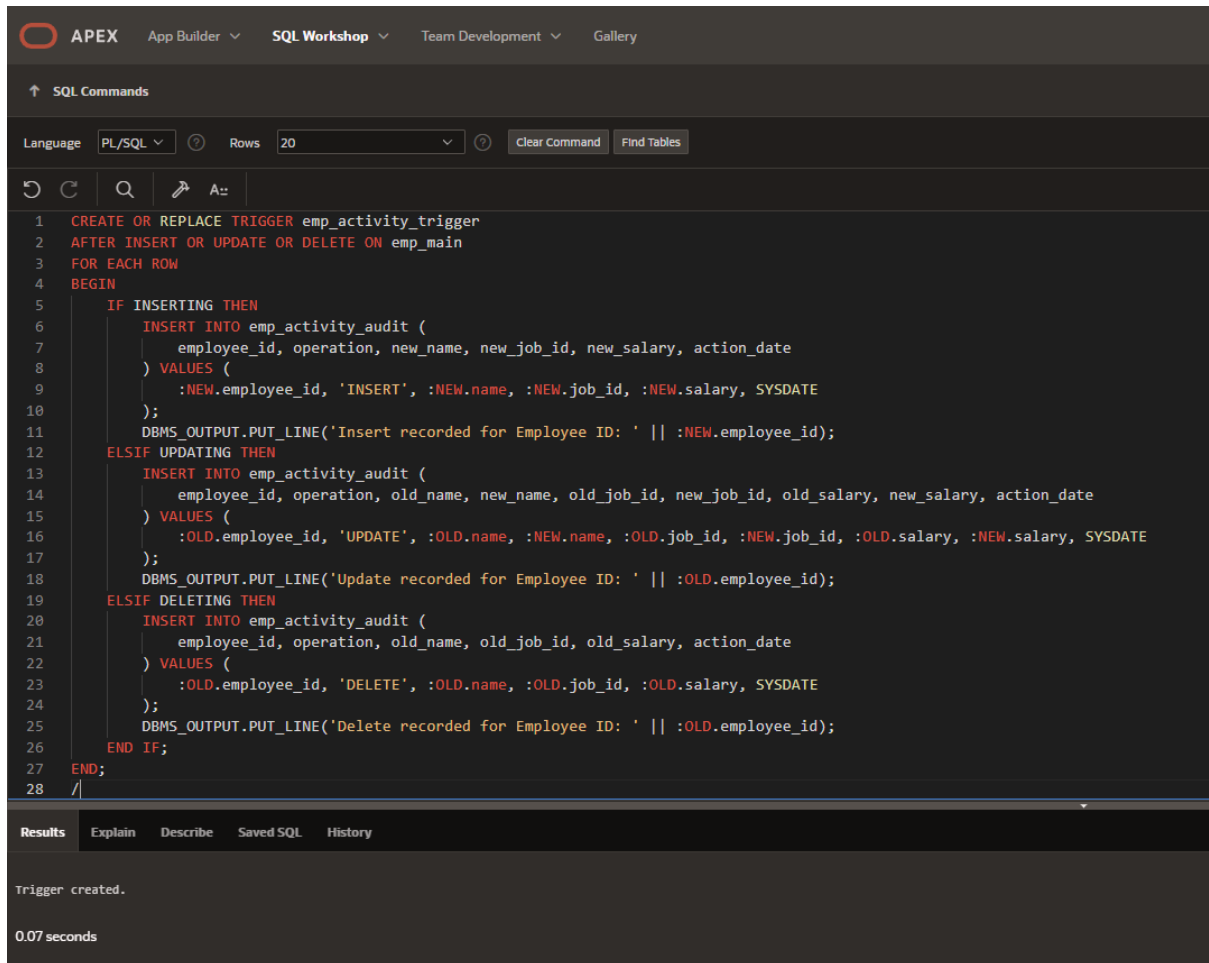
History

AUDIT_ID	EMPLOYEE_ID	OLD_SALARY	NEW_SALARY	OLD_JOB_ID	NEW_JOB_ID	CHANGE_DATE
1	2	6000	5800	IT_PROG	IT_PROG	10/26/2025
21	3	7000	7000	ECE_PROG	CSE_PROG	10/26/2025

2 rows returned in 0.02 seconds [Download](#)

PROGRAM 5

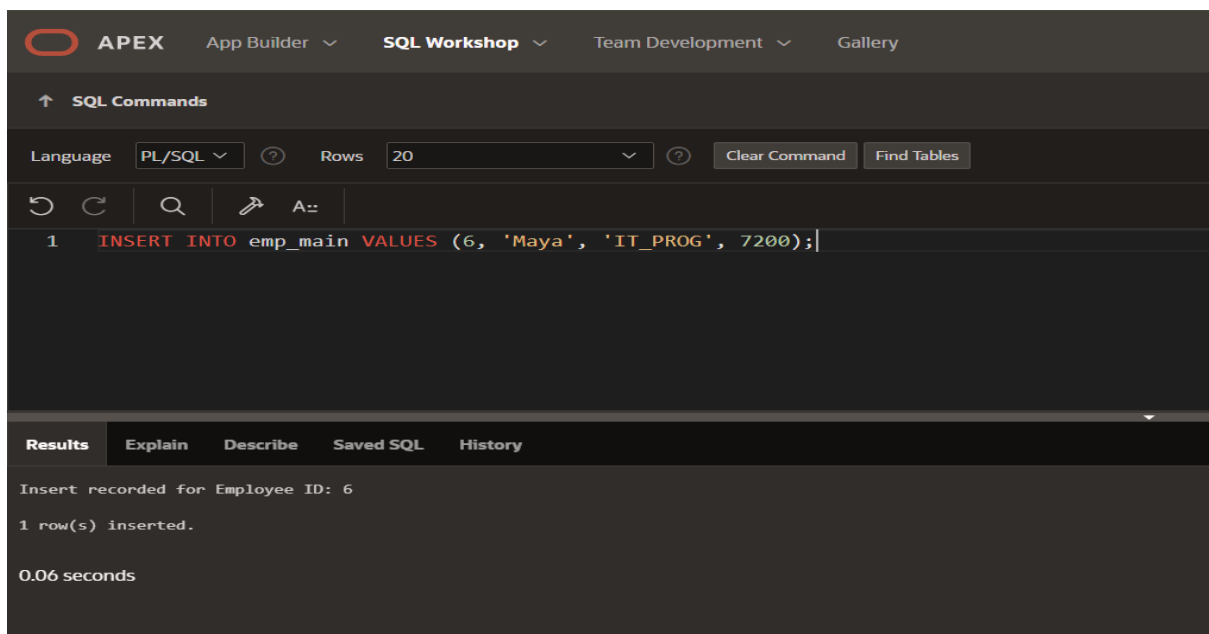
Write a code in PL/SQL to implement a trigger that records user activity (inserts, updates, deletes) in an audit log for a given set of tables.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, the 'SQL Commands' section is active. The language is set to 'PL/SQL', and the number of rows is set to '20'. The command area contains the following PL/SQL code:

```
1 CREATE OR REPLACE TRIGGER emp_activity_trigger
2 AFTER INSERT OR UPDATE OR DELETE ON emp_main
3 FOR EACH ROW
4 BEGIN
5     IF INSERTING THEN
6         INSERT INTO emp_activity_audit (
7             employee_id, operation, new_name, new_job_id, new_salary, action_date
8         ) VALUES (
9             :NEW.employee_id, 'INSERT', :NEW.name, :NEW.job_id, :NEW.salary, SYSDATE
10        );
11        DBMS_OUTPUT.PUT_LINE('Insert recorded for Employee ID: ' || :NEW.employee_id);
12    ELSIF UPDATING THEN
13        INSERT INTO emp_activity_audit (
14            employee_id, operation, old_name, new_name, old_job_id, new_job_id, old_salary, new_salary, action_date
15        ) VALUES (
16            :OLD.employee_id, 'UPDATE', :OLD.name, :NEW.name, :OLD.job_id, :NEW.job_id, :OLD.salary, :NEW.salary, SYSDATE
17        );
18        DBMS_OUTPUT.PUT_LINE('Update recorded for Employee ID: ' || :OLD.employee_id);
19    ELSIF DELETING THEN
20        INSERT INTO emp_activity_audit (
21            employee_id, operation, old_name, old_job_id, old_salary, action_date
22        ) VALUES (
23            :OLD.employee_id, 'DELETE', :OLD.name, :OLD.job_id, :OLD.salary, SYSDATE
24        );
25        DBMS_OUTPUT.PUT_LINE('Delete recorded for Employee ID: ' || :OLD.employee_id);
26    END IF;
27 END;
28 /
```

The 'Results' tab is selected, showing the message 'Trigger created.' and a duration of '0.07 seconds'.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, the 'SQL Commands' section is active. The language is set to 'PL/SQL', and the number of rows is set to '20'. The command area contains the following SQL statement:

```
1 INSERT INTO emp_main VALUES (6, 'Maya', 'IT_PROG', 7200);
```

The 'Results' tab is selected, showing the message 'Insert recorded for Employee ID: 6' and '1 row(s) inserted.' with a duration of '0.06 seconds'.

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

Rows

20

Clear Command

Find Tables

↶

↷

🔍

🔗

A≐

1 UPDATE emp_main SET salary = 8000 WHERE employee_id = 2;

Results

Explain

Describe

Saved SQL

History

Update recorded for Employee ID: 2

Audit record created for Employee ID: 2

1 row(s) updated.

0.02 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

Rows

20

Clear Command

Find Tables

↶

↷

🔍

🔗

A≐

1 DELETE FROM emp_main WHERE employee_id = 5;

Results

Explain

Describe

Saved SQL

History

Delete recorded for Employee ID: 5

1 row(s) deleted.

0.05 seconds

↑ SQL Commands

Schema

WKSP_DBMS05

Language

PL/SQL

Rows

20

Clear Command

Find Tables

Save

Run

↶

↷

🔍

🔗

A≐

⚙️

1 SELECT * FROM emp_activity_audit ORDER BY audit_id;

Results

Explain

Describe

Saved SQL

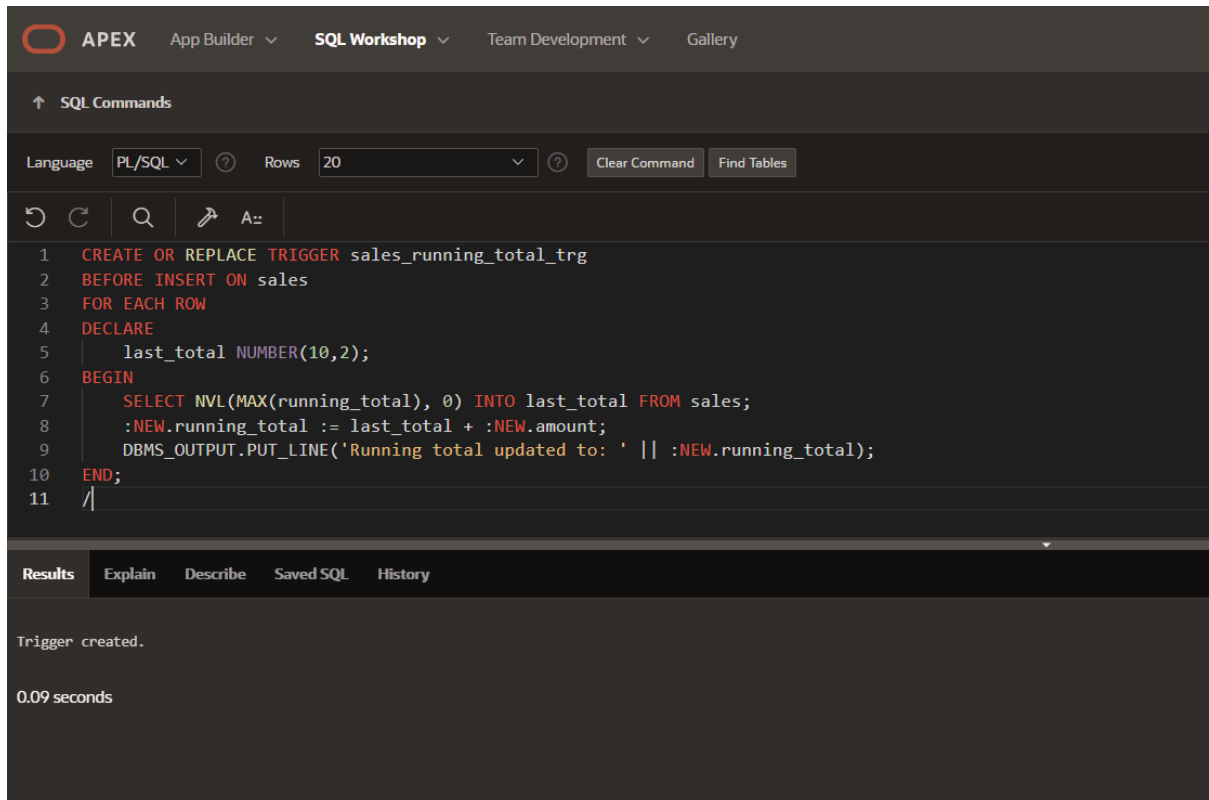
History

AUDIT_ID	EMPLOYEE_ID	OPERATION	OLD_NAME	NEW_NAME	OLD_JOB_ID	NEW_JOB_ID	OLD_SALARY	NEW_SALARY	ACTION_DATE
1	6	INSERT	-	Maya	-	IT_PROG	-	7200	10/26/2025
2	2	UPDATE	Rahul	Rahul	IT_PROG	IT_PROG	5800	8000	10/26/2025
21	5	DELETE	Arun	-	IT_PROG	-	6500	-	10/26/2025

3 rows returned in 0.03 seconds [Download](#)

PROGRAM 6

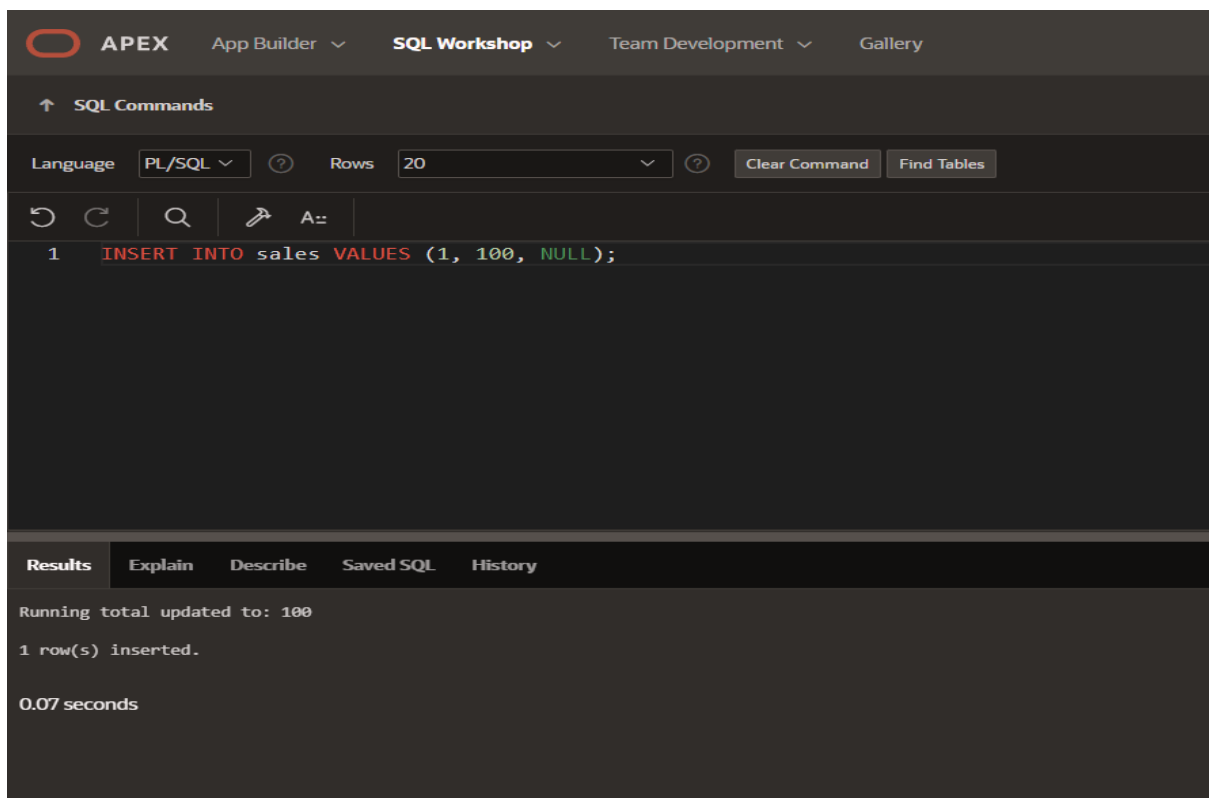
Write a code in PL/SQL to implement a trigger that automatically calculates and updates a running total column for a table whenever new rows are inserted.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, showing a language dropdown set to 'PL/SQL' and a 'Rows' limit of '20'. The SQL editor contains the following PL/SQL code:

```
1 CREATE OR REPLACE TRIGGER sales_running_total_trg
2 BEFORE INSERT ON sales
3 FOR EACH ROW
4 DECLARE
5     last_total NUMBER(10,2);
6 BEGIN
7     SELECT NVL(MAX(running_total), 0) INTO last_total FROM sales;
8     :NEW.running_total := last_total + :NEW.amount;
9     DBMS_OUTPUT.PUT_LINE('Running total updated to: ' || :NEW.running_total);
10 END;
11 /
```

The 'Results' tab is selected, displaying the message 'Trigger created.' and the execution time '0.09 seconds'.



The screenshot shows the APEX SQL Workshop interface with the same top navigation bar. The 'SQL Commands' section is active, with the language set to 'PL/SQL' and 'Rows' limit of '20'. The SQL editor contains the following SQL statement:

```
1 INSERT INTO sales VALUES (1, 100, NULL);
```

The 'Results' tab is selected, displaying the output 'Running total updated to: 100' and '1 row(s) inserted.' The execution time is '0.07 seconds'.

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

↶ ↷ 🔍 📌 A=

1 INSERT INTO sales VALUES (2, 200, NULL);

Results Explain Describe Saved SQL History

Running total updated to: 300
1 row(s) inserted.

0.04 seconds

APEX

App Builder

SQL Workshop

Team Development

Gallery

↑ SQL Commands

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

↶ ↷ 🔍 📌 A=

1 INSERT INTO sales VALUES (3, 150, NULL);

Results Explain Describe Saved SQL History

Running total updated to: 450
1 row(s) inserted.

0.01 seconds

↑ SQL Commands

Schema WKSP_DEMO505

Language

PL/SQL

 Rows

20

Clear Command

Find Tables

Save

Run

↶ ↷ 🔍 📌 A=

1 SELECT * FROM sales;

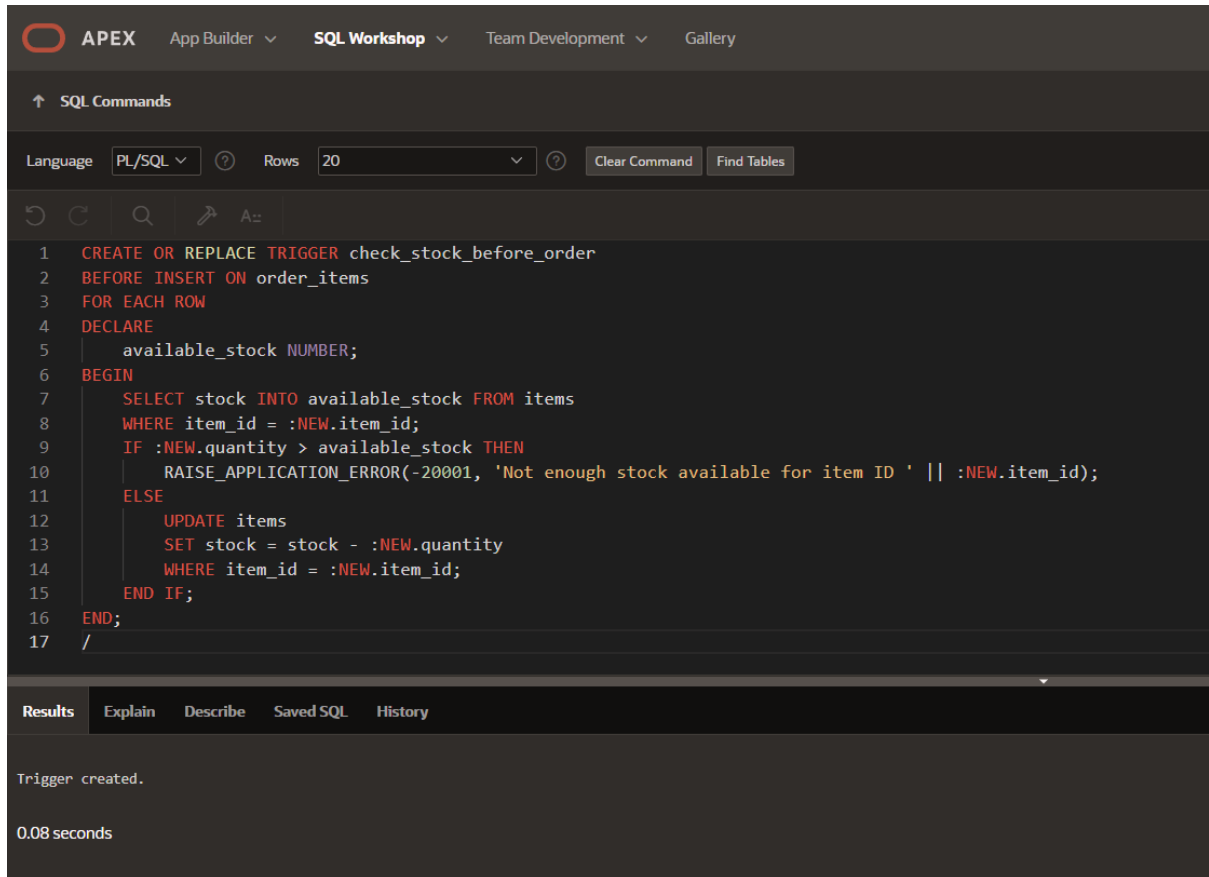
Results Explain Describe Saved SQL History

SALE_ID	AMOUNT	RUNNING_TOTAL
3	150	450
2	200	300
1	100	100

3 rows returned in 0.01 seconds [Download](#)

PROGRAM 7

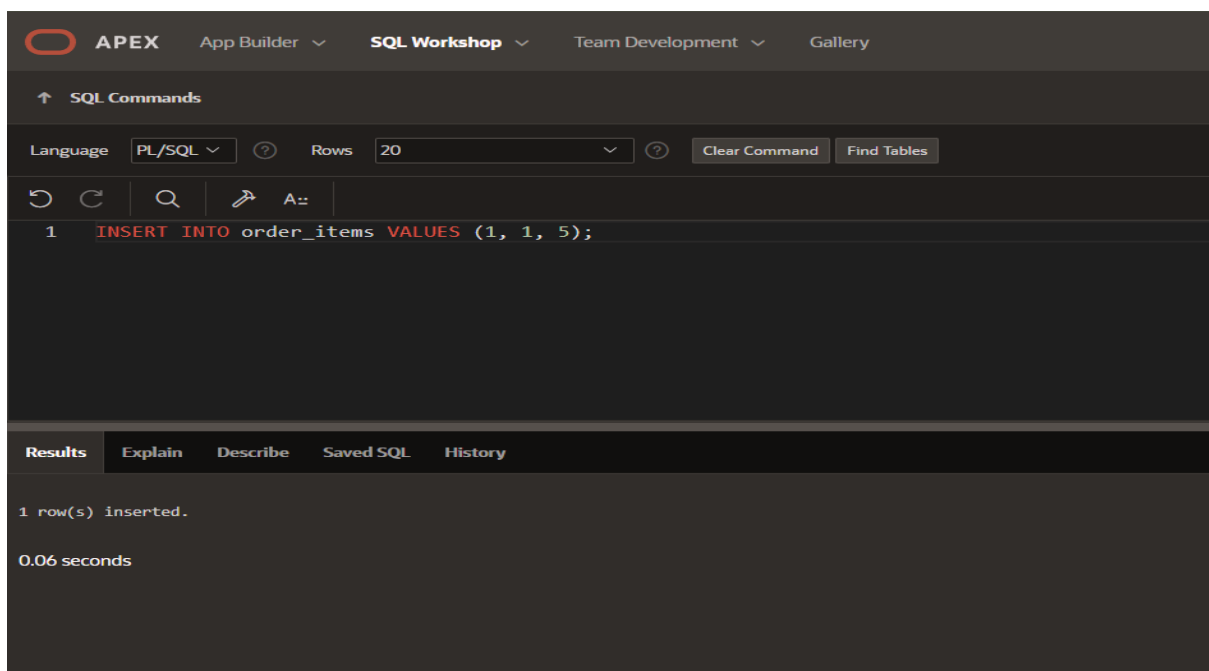
Write a code in PL/SQL to create a trigger that validates the availability of items before allowing an order to be placed, considering stock levels and pending orders.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, there's a section for 'SQL Commands' with a language dropdown set to 'PL/SQL', a 'Rows' dropdown set to '20', and buttons for 'Clear Command' and 'Find Tables'. The main editor area contains the following PL/SQL code:

```
1 CREATE OR REPLACE TRIGGER check_stock_before_order
2 BEFORE INSERT ON order_items
3 FOR EACH ROW
4 DECLARE
5     available_stock NUMBER;
6 BEGIN
7     SELECT stock INTO available_stock FROM items
8     WHERE item_id = :NEW.item_id;
9     IF :NEW.quantity > available_stock THEN
10         RAISE_APPLICATION_ERROR(-20001, 'Not enough stock available for item ID ' || :NEW.item_id);
11     ELSE
12         UPDATE items
13         SET stock = stock - :NEW.quantity
14         WHERE item_id = :NEW.item_id;
15     END IF;
16 END;
17 /
```

Below the code editor, there's a 'Results' tab selected, showing the message 'Trigger created.' and the execution time '0.08 seconds'.



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below the navigation bar, there's a section for 'SQL Commands' with a language dropdown set to 'PL/SQL', a 'Rows' dropdown set to '20', and buttons for 'Clear Command' and 'Find Tables'. The main editor area contains the following SQL statement:

```
1 INSERT INTO order_items VALUES (1, 1, 5);
```

Below the code editor, there's a 'Results' tab selected, showing the message '1 row(s) inserted.' and the execution time '0.06 seconds'.

APEXApp BuilderSQL WorkshopTeam DevelopmentGallery

Search

SQL CommandsSchemaWKSP_DBMS05

LanguagePL/SQLRows20Clear CommandFind Tables

1

INSERT INTO order_items VALUES (2, 1, 6);

Results

ExplainDescribeSaved SQLHistory

ORA-20001: Not enough stock available for item ID 1
ORA-06512: at "WKSP_DBMS05.CHECK_STOCK_BEFORE_ORDER", line 7
ORA-04088: error during execution of trigger "WKSP_DBMS05.CHECK_STOCK_BEFORE_ORDER"

0.01seconds

SQL CommandsSchemaWKSP_DBMS05

LanguagePL/SQLRows20Clear CommandFind TablesSaveRun

1

SELECT * FROM items;

Results

ExplainDescribeSaved SQLHistory

ITEM_ID	ITEM_NAME	STOCK
3	Keyboard	20
4	Monitor	15
5	USB Cable	100
1	Laptop	5
2	Mouse	50

5 rows returned in 0.01 secondsDownload

SQL CommandsSchemaWKSP_DBMS05

LanguagePL/SQLRows20Clear CommandFind TablesSaveRun

1

SELECT * FROM order_items;

Results

ExplainDescribeSaved SQLHistory

ORDER_ID	ITEM_ID	QUANTITY
1	1	5

1 rows returned in 0.02 secondsDownload