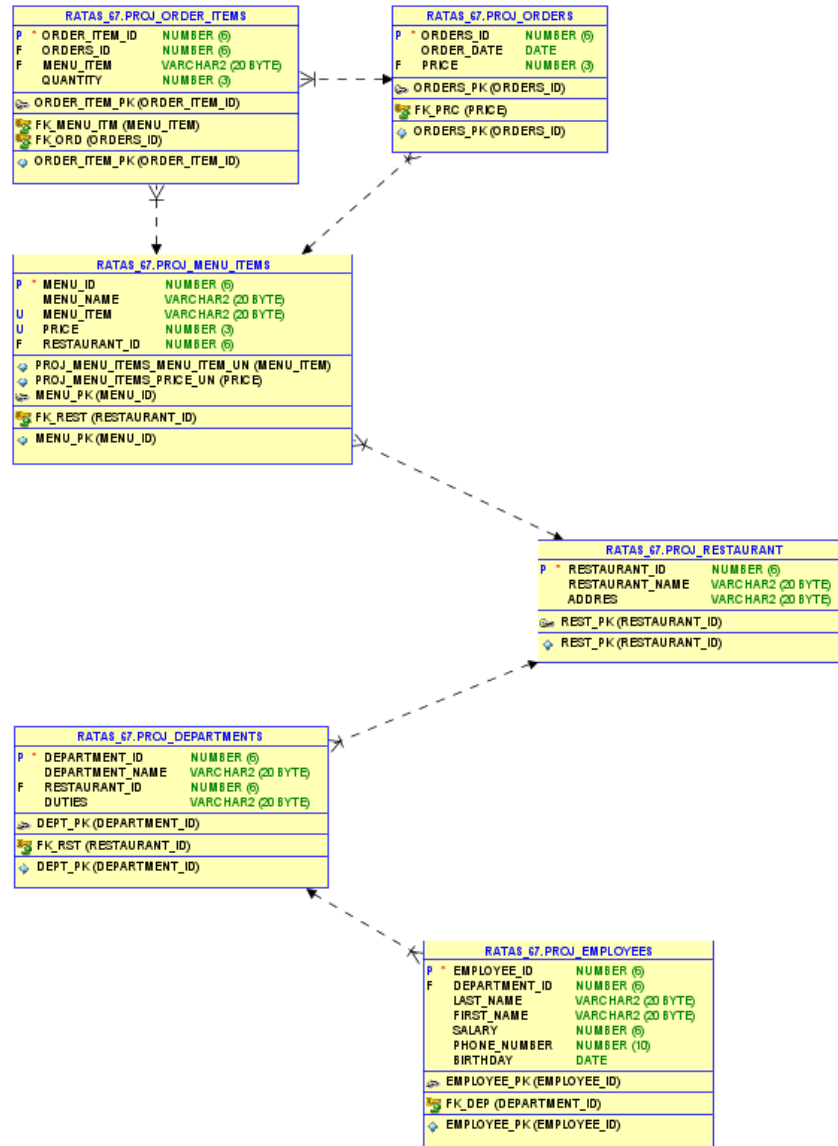
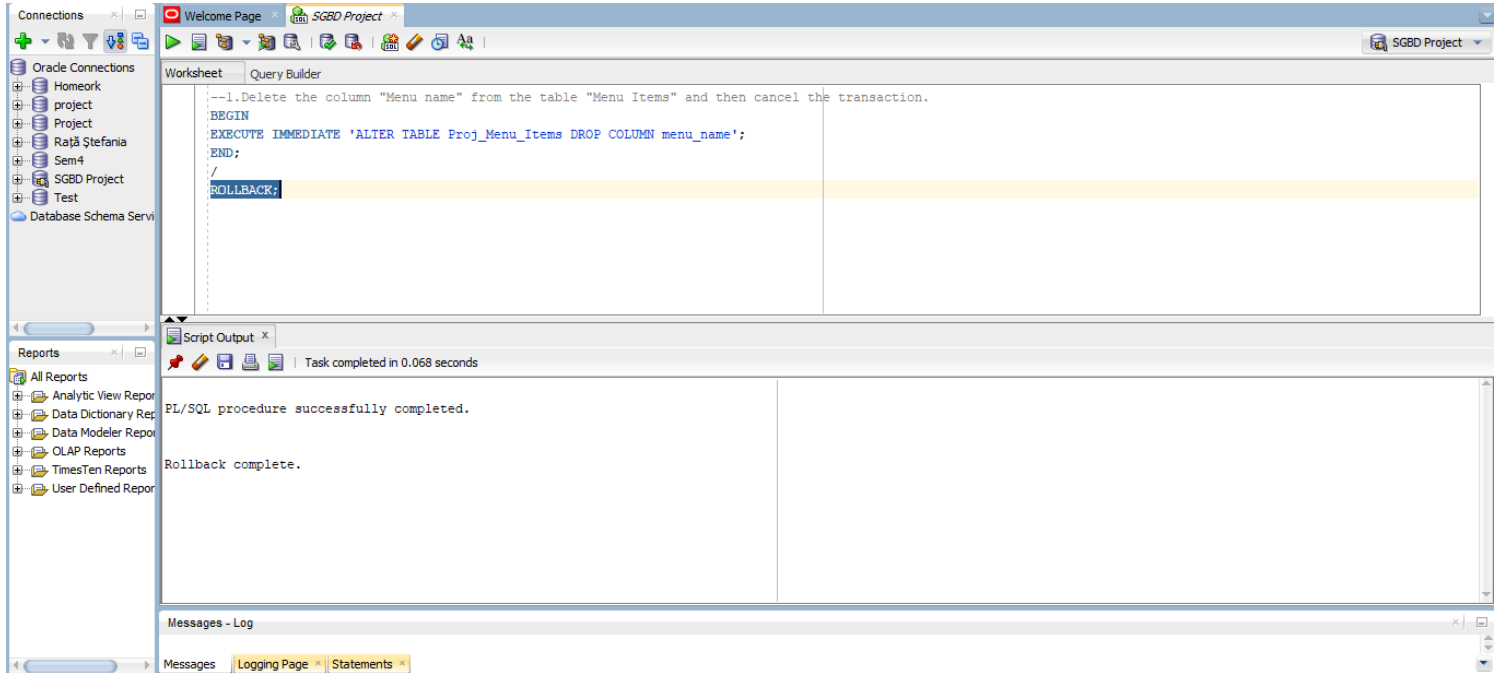


Taste of Paradise

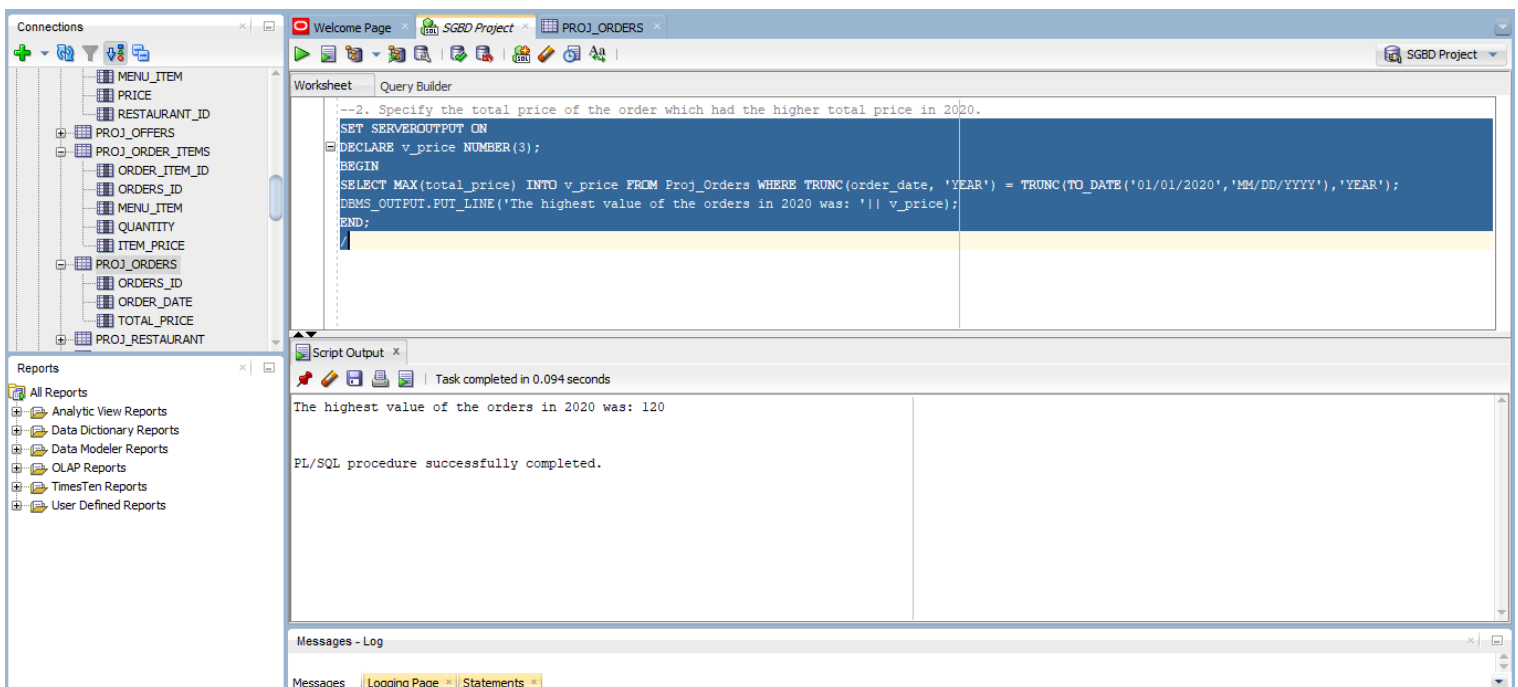


This is the database schema for my project. As a topic, I chose a restaurant called Taste of Paradise.

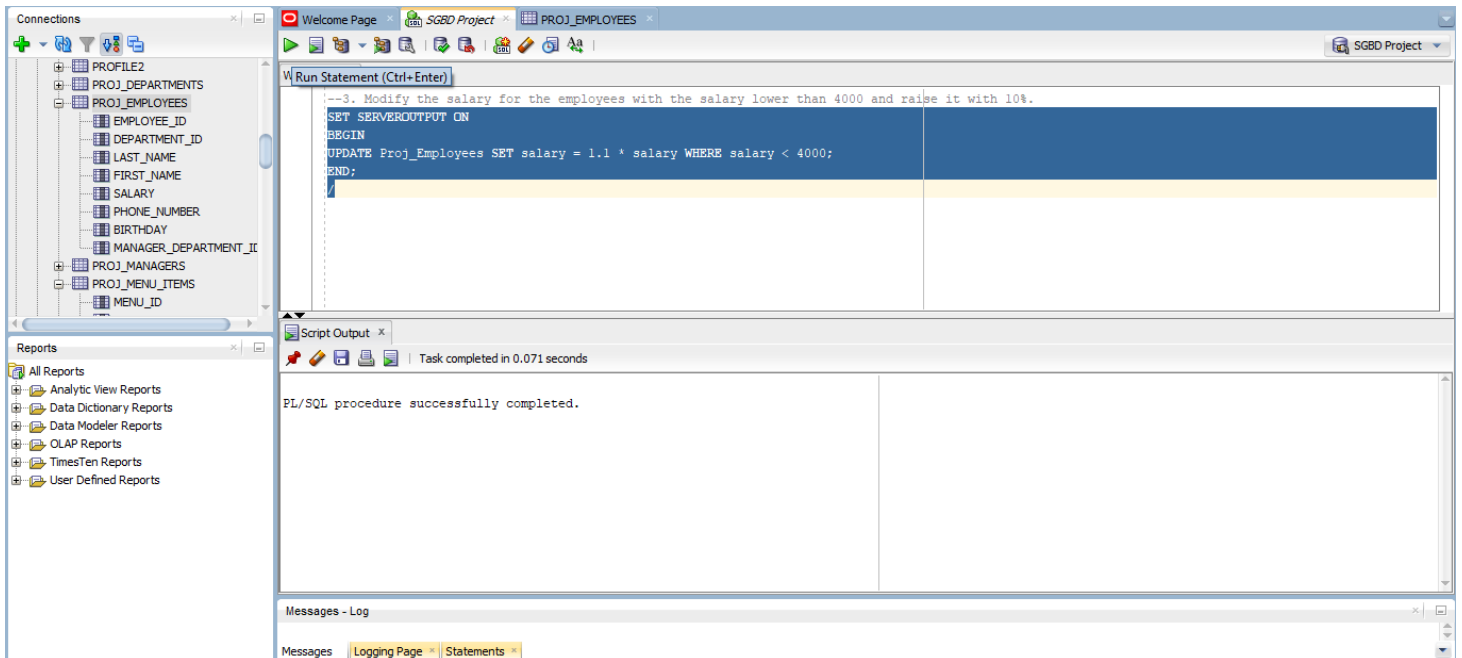
- Cancelling a transaction



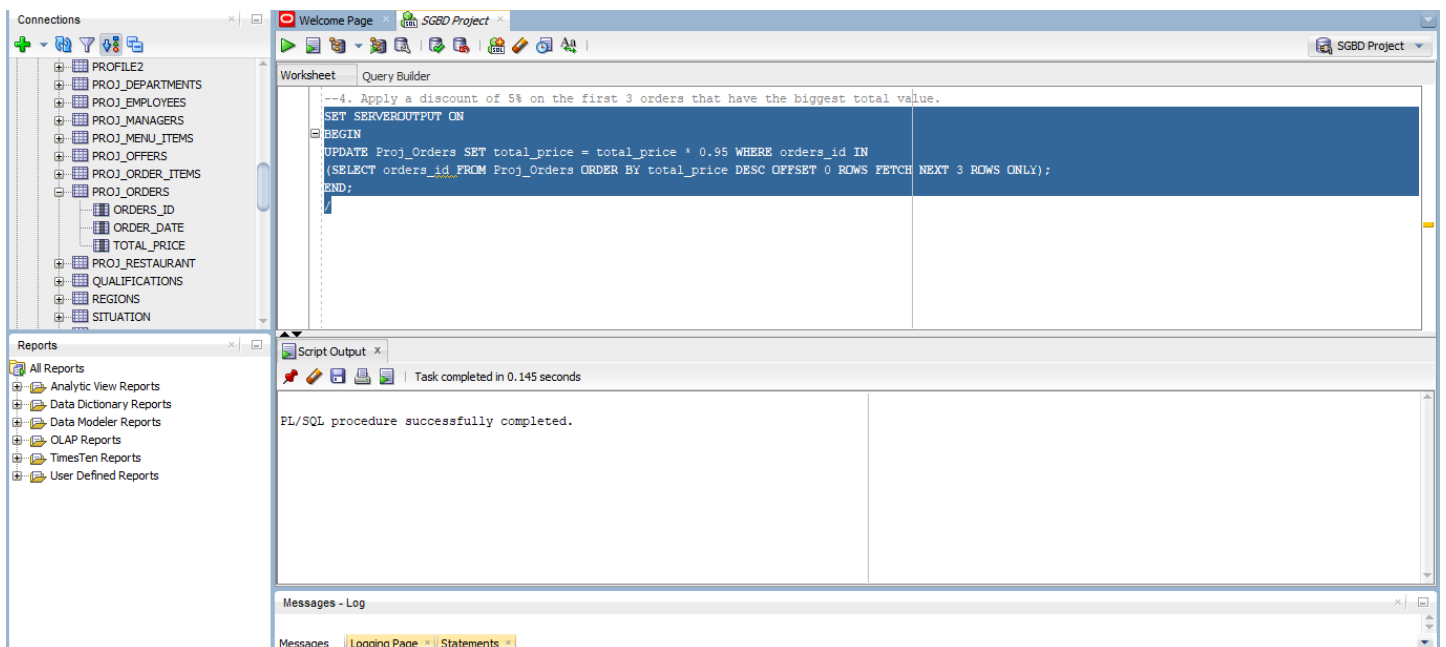
- Group functions



- CRUD commands



- SELECT statements



- Index-by tables

The screenshot shows the SQL Developer interface with a project named 'SGBD Project'. The 'Connections' pane on the left shows a tree of database objects, including 'PRODUCT_INFORMATION', 'PROFILE1', 'PROFILE2', 'PROJ_DEPARTMENTS', 'PROJ_EMPLOYEES', 'PROJ_MANAGERS', 'PROJ_MENU_ITEMS', 'PROJ_OFFERS', 'PROJ_ORDER_ITEMS', 'PROJ_ORDERS', and 'PROJ_RESTAURANT'. The 'Reports' pane shows 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', 'Data Modeler Reports', 'OLAP Reports', 'TimesTen Reports', and 'User Defined Reports'. The 'Worksheet' pane contains a PL/SQL procedure that uses an indexed table to loop through menu items and their prices. The 'Script Output' pane shows the results of the procedure, which are displayed as text lines. The 'Messages - Log' pane at the bottom shows the status of the procedure.

```
--5. Display the items in the menu and their prices using an indexed table and a for loop.
SET SERVEROUTPUT ON
DECLARE
  TYPE tab_ind IS TABLE OF Proj_Menu_Items%ROWTYPE INDEX BY PLS_INTEGER;
  v_tab tab_ind;
BEGIN
  FOR i IN 76833..76850 LOOP
    SELECT * INTO v_tab(i) FROM Proj_Menu_Items WHERE menu_id=i;
  END LOOP;
  FOR i in v_tab.FIRST..v_tab.LAST LOOP
    DBMS_OUTPUT.PUT_LINE('The product '||v_tab(i).menu_item||' has the price '||v_tab(i).price);
  END LOOP;
END;
```

Task completed in 0.084 seconds

The product Piept de curcan has the price 30
The product Steak de vita has the price 50
The product Ciorba de vacuta has the price 25
The product Supa de pui has the price 23
The product Ciorba de fasole has the price 20

PL/SQL procedure successfully completed.

- Exceptions

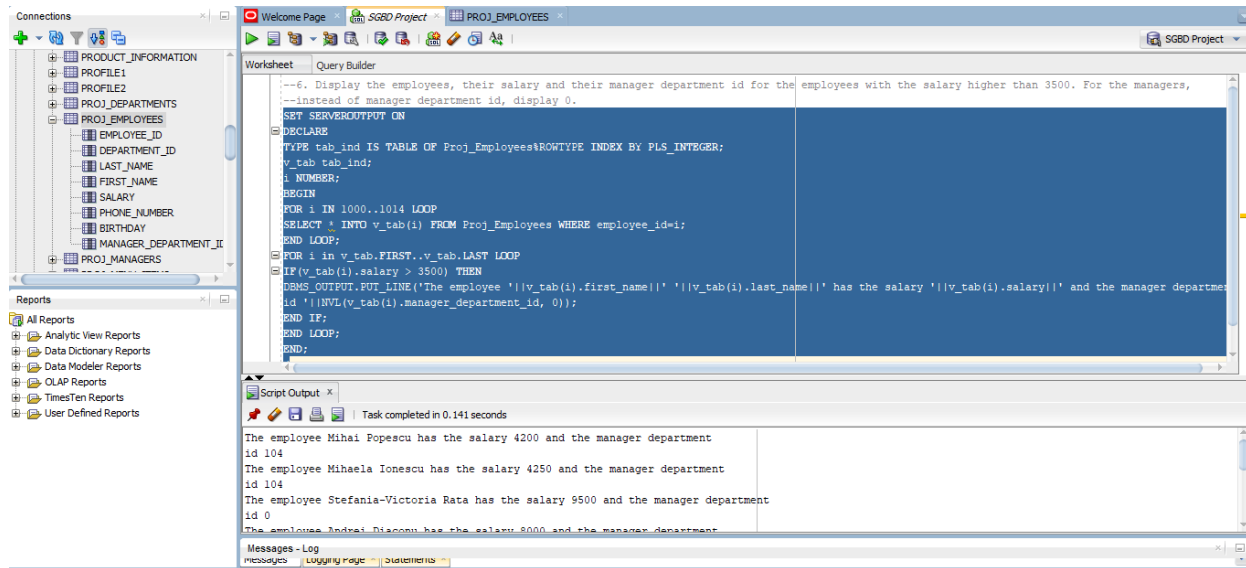
The screenshot shows the SQL Developer interface with a project named 'SGBD Project'. The 'Connections' pane on the left shows a tree of database objects, including 'BONUSES', 'CHECK_PERSON', 'CLIENTS', 'COUNTRIES', 'CUSTOMERS', 'DEPARTMENTS', 'EMP_SAL', 'EMPLOYEES', 'EX', 'EX10', 'JOB_HISTORY', 'JOBS', 'JOBS_HISTORY', 'LOCATIONS', 'MESSAGES', and 'PROJ_MENU_ITEMS'. The 'Reports' pane shows 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', 'Data Modeler Reports', 'OLAP Reports', 'TimesTen Reports', and 'User Defined Reports'. The 'Worksheet' pane contains a PL/SQL procedure that uses an exception handler to handle a case where an employee does not exist. The 'Script Output' pane shows the results of the procedure, which are displayed as text lines. The 'Messages - Log' pane at the bottom shows the status of the procedure.

```
--6. Display the name and the salary for the employee with the id 34. Handle the case when the employee does not exist.
SET SERVEROUTPUT ON
DECLARE
  v_name VARCHAR(2);
  v_salary NUMBER;
BEGIN
  SELECT last_name||' '||first_name INTO v_name FROM Employees WHERE employee_id = 34;
  SELECT salary INTO v_salary FROM Employees WHERE employee_id = 34;
  DBMS_OUTPUT.PUT_LINE('The employee '||v_name||' has the salary '||v_salary);
  EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('There is no employee with the given id!');
END;
```

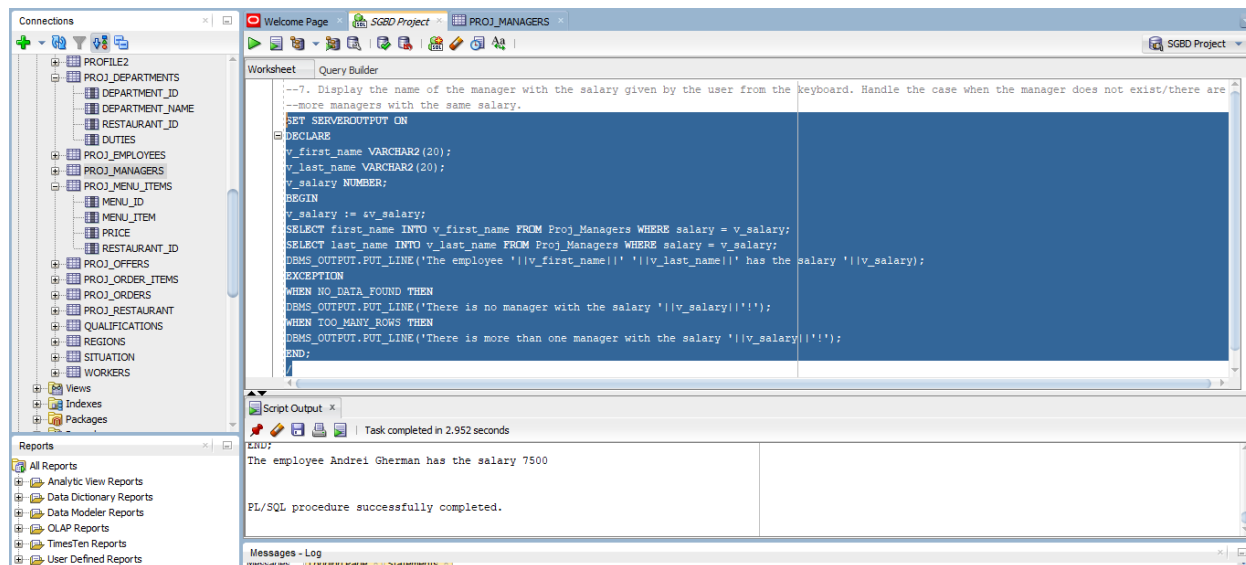
Task completed in 0.124 seconds

There is no employee with the given id!

PL/SQL procedure successfully completed.



- Substitution variables



Rață Ștefania-Victoria
Group 1105, Year 3
Business Informatics

The screenshot shows the SQL Developer interface with the 'PROJ_MANAGERS' table selected in the schema browser. The 'Worksheet' tab contains the following PL/SQL code:

```
--more managers with the same salary.  
  
SET SERVEROUTPUT ON  
  
DECLARE  
  v_name VARCHAR(2);  
  v_salary NUMBER;  
BEGIN  
  v_salary := 456;  
  SELECT last_name||' '||first_name INTO v_name FROM Proj_Managers WHERE salary = v_salary;  
  DBMS_OUTPUT.PUT_LINE('The employee '||v_name||' has the salary '||v_salary);  
EXCEPTION  
  WHEN NO_DATA_FOUND THEN  
    DBMS_OUTPUT.PUT_LINE('There is no manager with the salary '||v_salary||'');  
  WHEN TOO_MANY_ROWS THEN  
    DBMS_OUTPUT.PUT_LINE('There is more than one manager with the salary '||v_salary||'');  
END;
```

The 'Script Output' tab shows the execution results:

```
END:  
There is no manager with the salary 456!  
  
PL/SQL procedure successfully completed.
```

The 'Messages - Log' tab shows the execution status: 'Task completed in 2.762 seconds'.

The screenshot shows the SQL Developer interface with the 'PROJ_MANAGERS' table selected in the schema browser. The 'Worksheet' tab contains the same PL/SQL code as the first screenshot, but with the salary value changed to 8000:

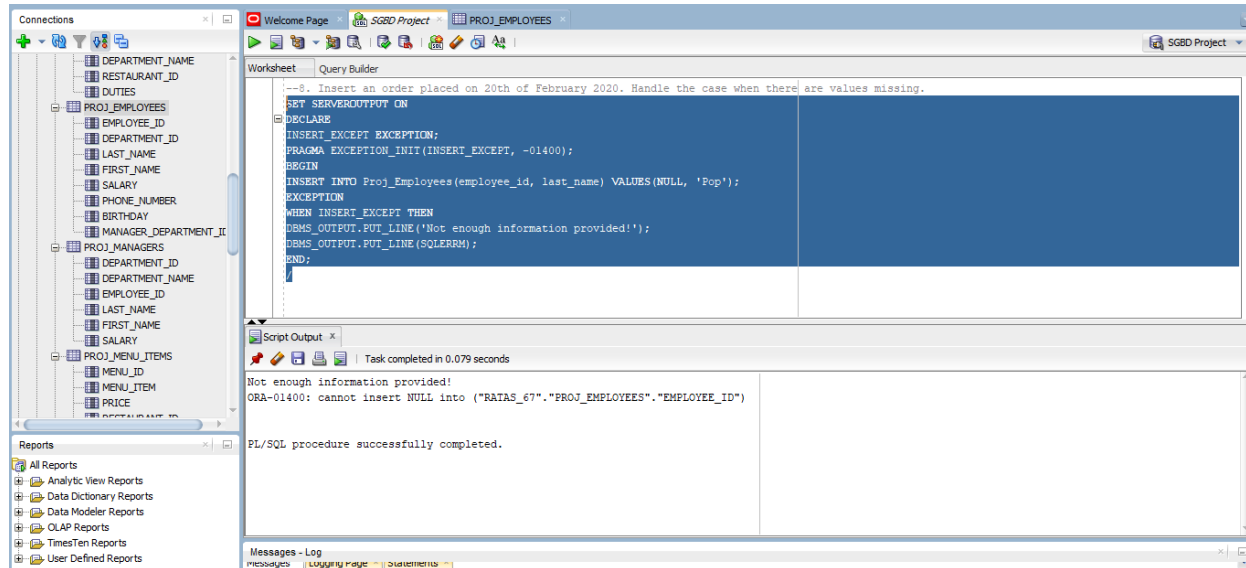
```
--more managers with the same salary.  
  
SET SERVEROUTPUT ON  
  
DECLARE  
  v_name VARCHAR(2);  
  v_salary NUMBER;  
BEGIN  
  v_salary := 8000;  
  SELECT last_name||' '||first_name INTO v_name FROM Proj_Managers WHERE salary = v_salary;  
  DBMS_OUTPUT.PUT_LINE('The employee '||v_name||' has the salary '||v_salary);  
EXCEPTION  
  WHEN NO_DATA_FOUND THEN  
    DBMS_OUTPUT.PUT_LINE('There is no manager with the salary '||v_salary||'');  
  WHEN TOO_MANY_ROWS THEN  
    DBMS_OUTPUT.PUT_LINE('There is more than one manager with the salary '||v_salary||'');  
END;
```

The 'Script Output' tab shows the execution results:

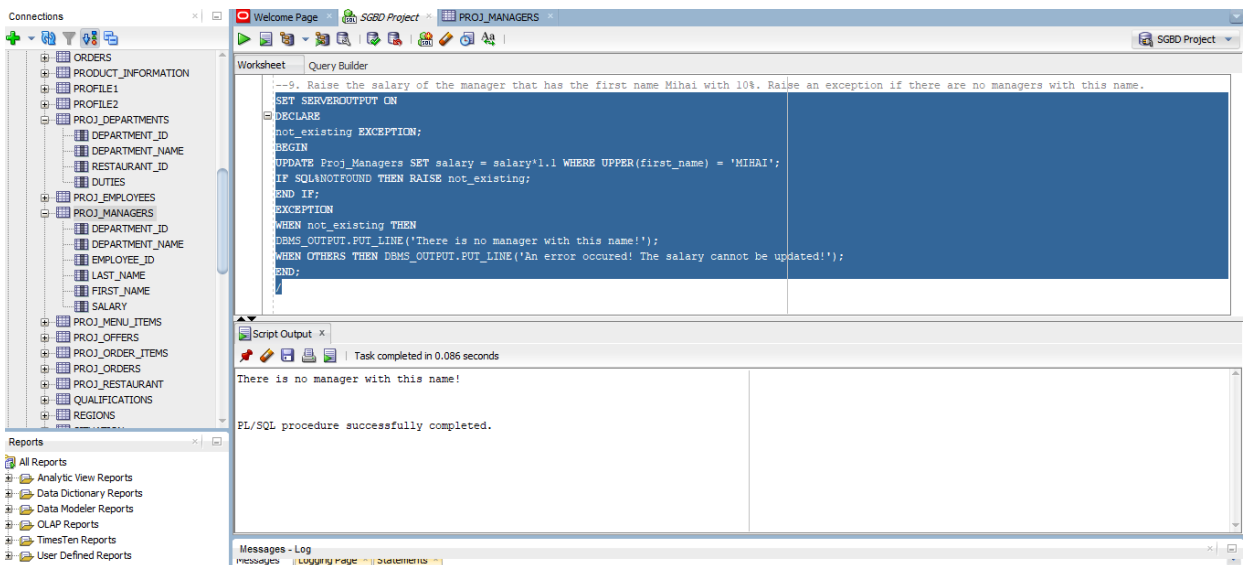
```
END:  
There is more than one manager with the salary 8000!  
  
PL/SQL procedure successfully completed.
```

The 'Messages - Log' tab shows the execution status: 'Task completed in 2.878 seconds'.

- Custom error codes



- Standard exceptions



Connections

- EMPLOYEES
- EX
- EX10
- JOB_HISTORY
- JOB_HISTORY
- LOCATIONS
- MESSAGES
- ORDER_ITEMS
- ORDERS
- PRODUCT_INFORMATION
 - PRODUCT_ID
 - PRODUCT_NAME
 - PRODUCT_DESCRIPTION
 - CATEGORY_ID
 - WEIGHT_CLASS
 - SUPPLIER_ID
 - PRODUCT_STATUS
 - LIST_PRICE
 - MIN_PRICE
 - CATALOG_URL
- PROFILE1
- PROFILE2
- PROJ_DEPARTMENTS

Worksheet | Query Builder

```
--10. Raise the salary of the employee that has the first name David with 10%. Raise an application exception if there are no managers with this name.
--Create a table called 'Errors' and display the error in it.

CREATE TABLE ERRORS(
  username varchar2(32),
  data_exc date,
  code_exc number(7),
  msg_exc varchar2(128));

SET SERVEROUTPUT ON

DECLARE
  code NUMBER(7);
  message VARCHAR2(255);
  not_existing EXCEPTION;
  PRAGMA EXCEPTION_INIT(not_existing,-20999);

BEGIN
  UPDATE Proj_Employees
  SET salary = salary * 1.1
  WHERE UPPER(first_name) ='DAVID';

  IF SQLNOTFOUND THEN
    RAISE_APPLICATION_ERROR (-20999,'No employee with this name');
```

Script Output | Query Result | Query Result 1

SQL | All Rows Fetched: 1 in 0.007 seconds

USERNAME	DATA_EXC	CODE_EXC	MSG_EXC
1 RATAS_67	23-MAY-23	-20999	ORA-20999: No employee with this name

Messages - Log

messages | Logging page | Statements

Connections

- EMPLOYEES
- EX
- EX10
- JOB_HISTORY
- JOB_HISTORY
- LOCATIONS
- MESSAGES
- ORDER_ITEMS
- ORDERS
- PRODUCT_INFORMATION
 - PRODUCT_ID
 - PRODUCT_NAME
 - PRODUCT_DESCRIPTION
 - CATEGORY_ID
 - WEIGHT_CLASS
 - SUPPLIER_ID
 - PRODUCT_STATUS
 - LIST_PRICE
 - MIN_PRICE
 - CATALOG_URL
- PROFILE1
- PROFILE2
- PROJ_DEPARTMENTS

Worksheet | Query Builder

```
IF SQLNOTFOUND THEN
  RAISE_APPLICATION_ERROR (-20999,'No employee with this name');
END IF;

EXCEPTION
  WHEN not_existing THEN
    DBMS_OUTPUT.PUT_LINE('This employee does not exist');
    code:=SQLCODE;
    message:=SQLERRM;
    INSERT INTO ERRORS VALUES(USER, SYSDATE, code, message);
END;

SELECT * FROM ERRORS;
```

Script Output | Query Result | Query Result 1

Task completed in 0.583 seconds

*Cause:
*Action:
This employee does not exist

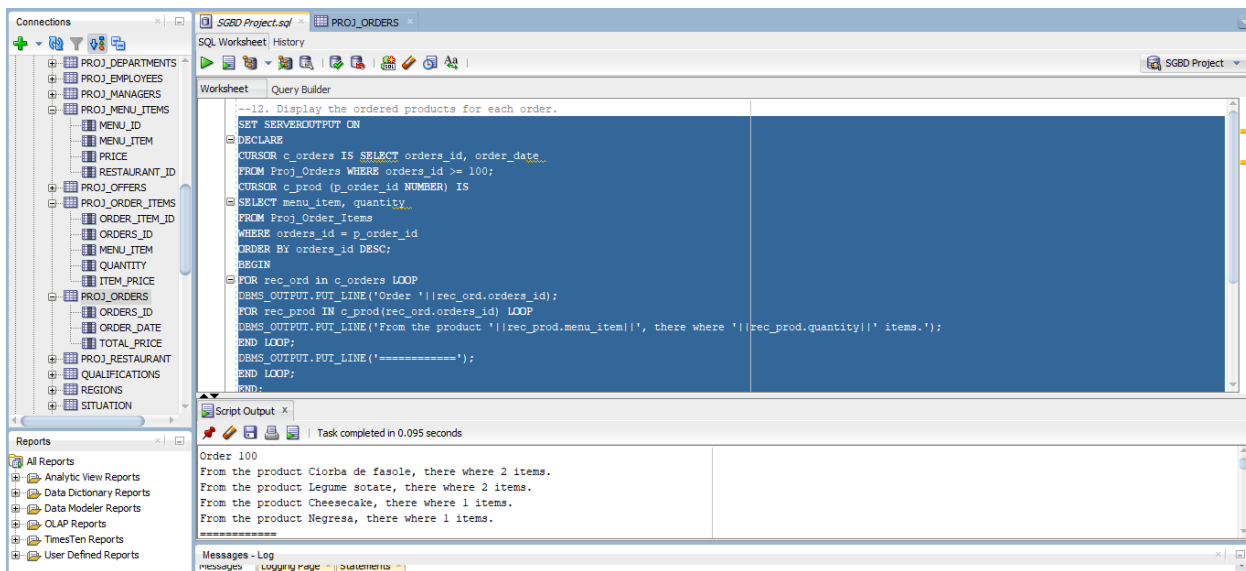
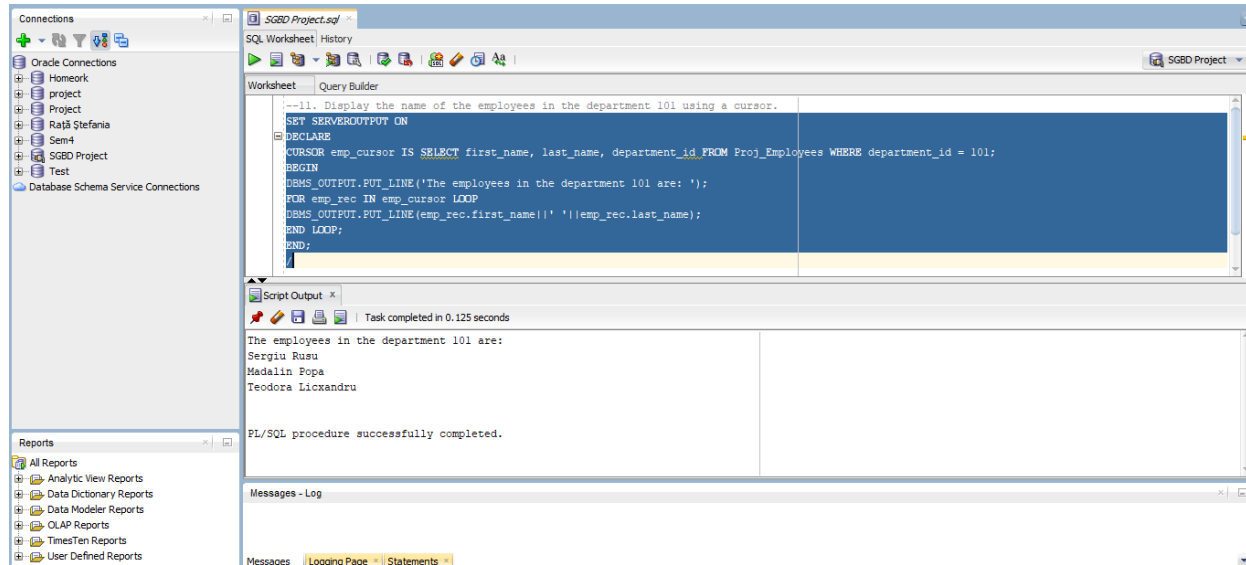
PL/SQL procedure successfully completed.

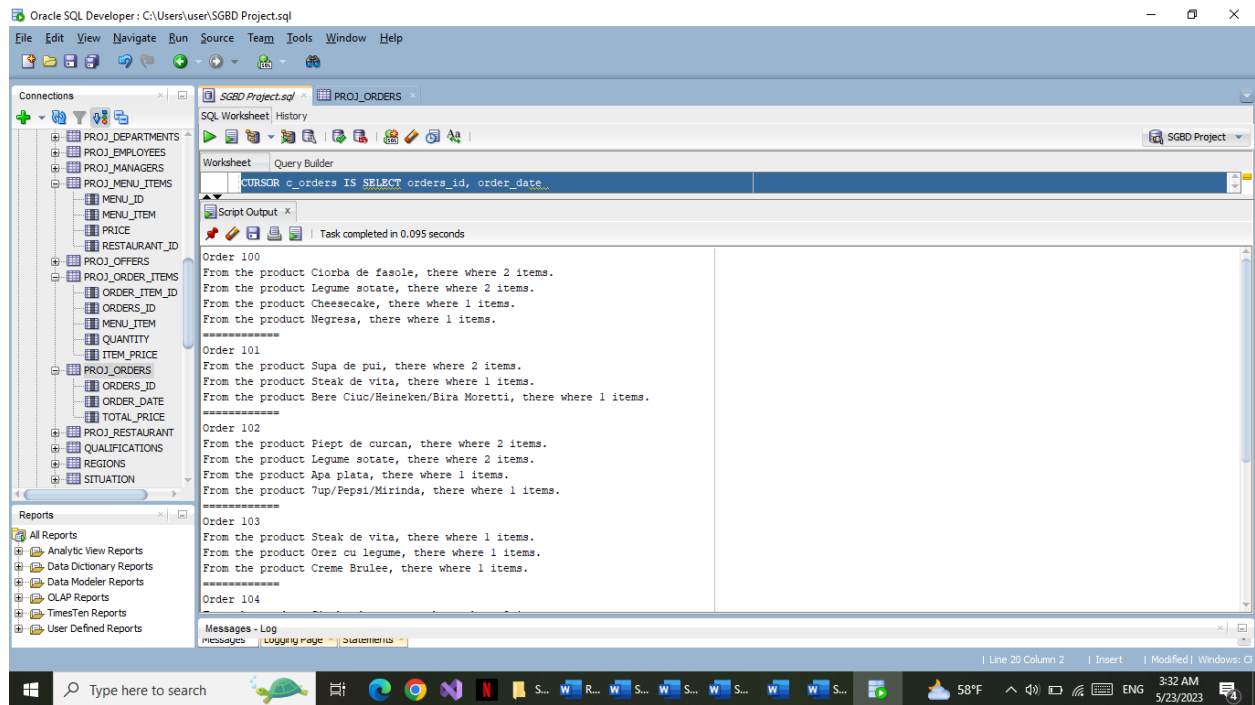
>>Query Run In:Query Result 1

Messages - Log

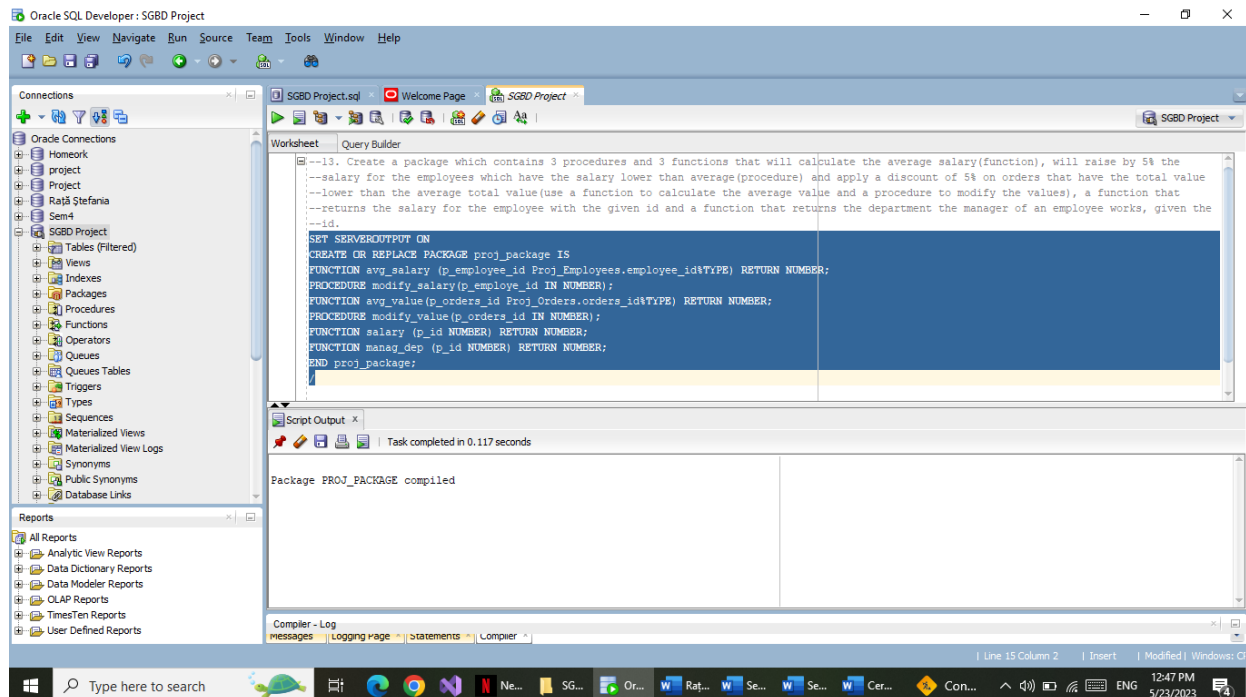
messages | Logging page | Statements

- Cursors





- Functions and procedures



Rață Ștefania-Victoria
Group 1105, Year 3
Business Informatics

The screenshot displays the SQL Developer interface. On the left, the 'Connections' pane shows a tree view of the 'PROJ' schema, including tables like PROJ_EMPLOYEES, PROJ_ORDER_ITEMS, and PROJ_RESTAURANT. The main 'Worksheet' pane contains the following SQL code:

```
END IF;
EXCEPTION
WHEN exceptionl THEN DBMS_OUTPUT.PUT_LINE('There is no employee with this id!');
END;

FUNCTION manag_dep (p_id NUMBER) RETURN NUMBER
IS
p_manag_dep NUMBER;
null_except EXCEPTION;
PRAGMA EXCEPTION_INIT(null_except, -02290);
BEGIN
SELECT manager_department_id INTO p_manag_dep FROM Proj_Employees WHERE employee_id = p_id;
EXCEPTION
WHEN null_except THEN DBMS_OUTPUT.PUT_LINE('This employee is a manager!');
DBMS_OUTPUT.PUT_LINE(SQLERRM);
END;

END proj_package;
```

Below the code, the 'Script Output' pane shows the message: 'Task completed in 0.163 seconds'. The 'Reports' pane at the bottom lists various report types, including 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', 'Data Modeler Reports', 'OLAP Reports', 'TimesTen Reports', and 'User Defined Reports'.

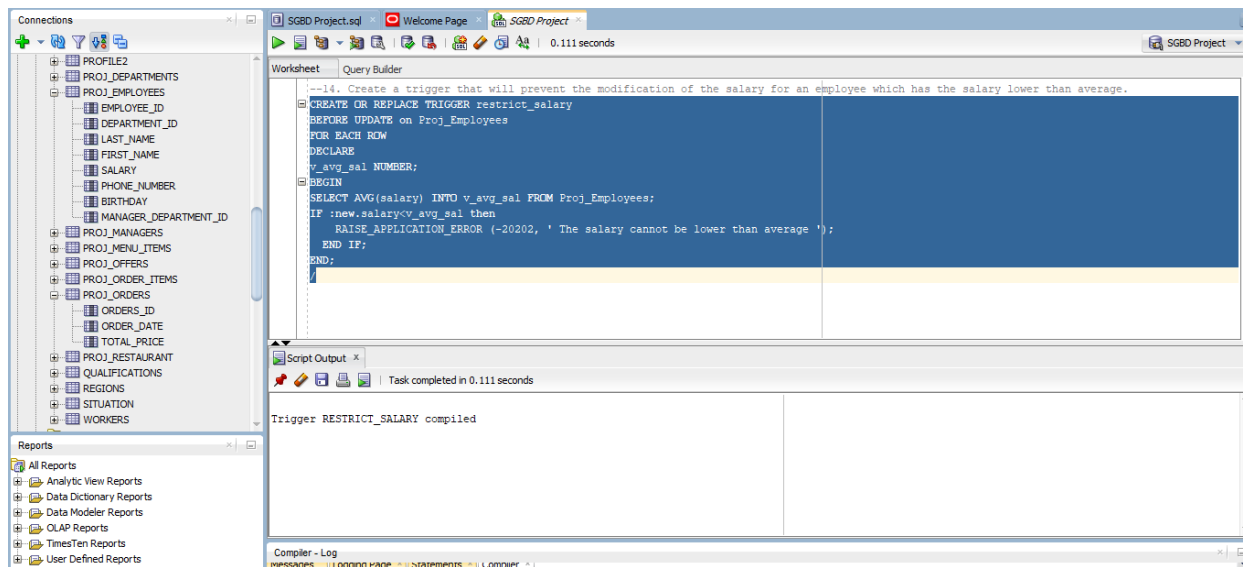
The screenshot displays the SQL Developer interface with a different SQL script in the 'Worksheet' pane. The 'Connections' pane on the left is identical to the previous screenshot. The SQL code in the main pane is:

```
SELECT AVG(total_price) INTO p_avg_value FROM Proj_Orders;
RETURN p_avg_value;
END;

PROCEDURE modify_value(p_orders_id IN NUMBER)
IS
BEGIN
UPDATE Proj_Orders SET total_price = total_price * 0.95 WHERE total_price > avg_value(p_orders_id);
END;

FUNCTION salary (p_id NUMBER) RETURN NUMBER
IS
p_salary NUMBER;
exceptionl EXCEPTION;
BEGIN
SELECT salary INTO p_salary FROM Proj_Employees WHERE employee_id = p_id;
IF SQLROWCOUNT = 0 THEN RAISE exceptionl;
END IF;
EXCEPTION
WHEN exceptionl THEN DBMS_OUTPUT.PUT_LINE('There is no employee with this id!');
```

The 'Script Output' pane shows 'Task completed in 0.163 seconds'. The 'Reports' pane at the bottom is also identical to the previous screenshot.



15.

