Рогова Ксения
гр. 3530202/70201

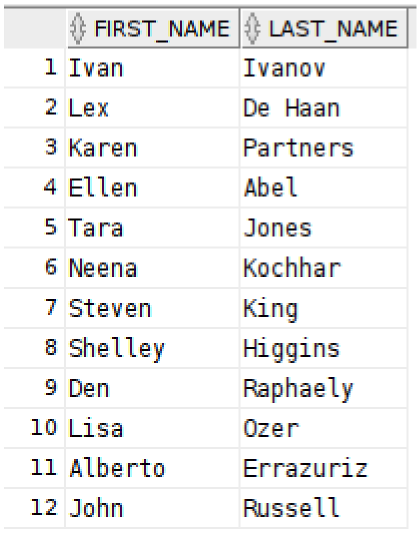
Отчет по лабораторной работе №3

Представления

1. Создать представление, отображающее всех служащих из отделов, которые работали над проектами в заданный интервал времени.

```
CREATE VIEW EMPLOYEEES_BY_DATE AS
  SELECT FIRST_NAME, LAST_NAME FROM EMPLOYEES
    LEFT JOIN DEPARTMENTS_EMPLOYEES ON (DEPARTMENTS_EMPLOYEES.EMPLOYEE_ID =
EMPLOYEES.ID)
    LEFT JOIN DEPARTMENTS ON (DEPARTMENTS.ID = DEPARTMENTS_EMPLOYEES.DEPARTMENT_ID)
    LEFT JOIN PROJECTS ON (PROJECTS.DEPARTMENT_ID = DEPARTMENTS.ID)
  WHERE PROJECTS.DATE_END >= '01.02.20' AND PROJECTS.DATE_BEG <= '11.02.20'
  GROUP BY FIRST_NAME, LAST_NAME;
```

Результат:

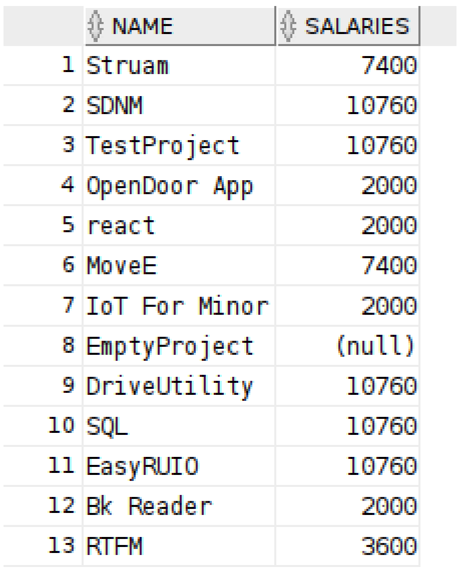


	FIRST_NAME	LAST_NAME
1	Ivan	Ivanov
2	Lex	De Haan
3	Karen	Partners
4	Ellen	Abel
5	Tara	Jones
6	Neena	Kochhar
7	Steven	King
8	Shelley	Higgins
9	Den	Raphaely
10	Lisa	Ozer
11	Alberto	Errazuriz
12	John	Russell

2. Создать представление, отображающее все проекты и затраты на их реализацию за месяц.

```
CREATE VIEW COSTS_PER_MONTH AS
  SELECT PROJECTS.NAME, SUM(EMPLOYEES.SALARY) AS SALARIES FROM PROJECTS
    LEFT JOIN DEPARTMENTS  ON (DEPARTMENTS.ID = PROJECTS.DEPARTMENT_ID)
    LEFT JOIN DEPARTMENTS_EMPLOYEES ON (DEPARTMENTS_EMPLOYEES.DEPARTMENT_ID =
DEPARTMENTS.ID)
    LEFT JOIN EMPLOYEES ON (EMPLOYEES.ID = DEPARTMENTS_EMPLOYEES.EMPLOYEE_ID)
  GROUP BY PROJECTS.NAME;
```

Результат:



	NAME	SALARIES
1	Struam	7400
2	SDNM	10760
3	TestProject	10760
4	OpenDoor App	2000
5	react	2000
6	MoveE	7400
7	IoT For Minor	2000
8	EmptyProject	(null)
9	DriveUtility	10760
10	SQL	10760
11	EasyRUIO	10760
12	Bk Reader	2000
13	RTFM	3600

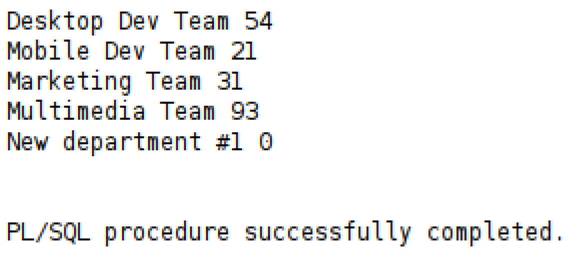
Хранимые процедуры

- без параметров:

Создать хранимую процедуру, выводящую все отделы и среднее время их работы над проектами.

```
SET SERVEROUTPUT ON
CREATE OR REPLACE PROCEDURE DEPARTMENTS_WITH_AVERAGE_TIME
IS
BEGIN
  FOR I IN
    (SELECT DEPARTMENTS.NAME AS depName, NVL(ROUND(AVG(PROJECTS.DATE_END -
PROJECTS.DATE_BEG), 0), 0) AS avgTime FROM DEPARTMENTS
      LEFT JOIN PROJECTS ON (PROJECTS.DEPARTMENT_ID = DEPARTMENTS.ID)
      GROUP BY DEPARTMENTS.NAME)
  LOOP
    DBMS_OUTPUT.PUT_LINE(I.depName || ' ' || I.avgTime);
  END LOOP;
END DEPARTMENTS_WITH_AVERAGE_TIME;
```

Результат:



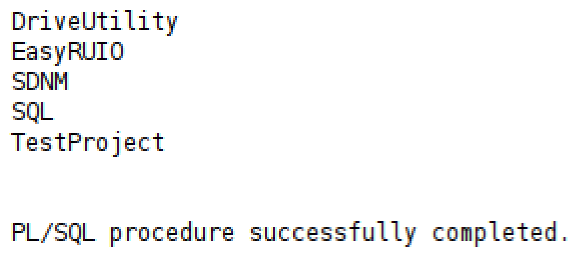
```
Desktop Dev Team 54
Mobile Dev Team 21
Marketing Team 31
Multimedia Team 93
New department #1 0


PL/SQL procedure successfully completed.
```

- с входными параметрами:

Создать хранимую процедуру, имеющую два параметра «служащий1» и «служащий2». Она должна возвращать проекты, в которых эти два служащих работали одновременно.

```
CREATE OR REPLACE PROCEDURE PROJECTS_BY_PAIR (EMPL_1 IN VARCHAR2, EMPL_2 IN VARCHAR2)
IS
BEGIN
  FOR TEMP IN
    ((SELECT PROJECTS.NAME AS projName FROM PROJECTS
      JOIN DEPARTMENTS_EMPLOYEES ON DEPARTMENTS_EMPLOYEES.DEPARTMENT_ID =
PROJECTS.DEPARTMENT_ID
      WHERE DEPARTMENTS_EMPLOYEES.EMPLOYEE_ID = EMPL_1)
    INTERSECT
     (SELECT PROJECTS.NAME AS projName FROM PROJECTS
      JOIN DEPARTMENTS_EMPLOYEES ON DEPARTMENTS_EMPLOYEES.DEPARTMENT_ID =
PROJECTS.DEPARTMENT_ID
      WHERE DEPARTMENTS_EMPLOYEES.EMPLOYEE_ID = EMPL_2))
  LOOP
    DBMS_OUTPUT.PUT_LINE(TEMP.projName);
  END LOOP;
END PROJECTS_BY_PAIR;
```

Результат:



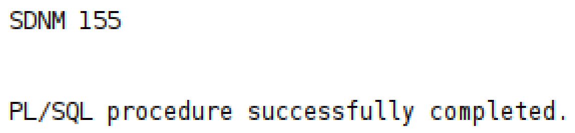
```
DriveUtility
EasyRUIO
SDNM
SQL
TestProject


PL/SQL procedure successfully completed.
```

- с выходными параметрами:

    Создать хранимую процедуру с входным параметром «отдел» и двумя выходными параметрами, возвращающими самое большое время, которое потребовалось для реализации проекта и сам проект, поставивший рекорд.

```
CREATE OR REPLACE PROCEDURE LONGEST_PROJECT_BY_DEP (DEP_NAME IN VARCHAR2, PROJ_NAME
OUT VARCHAR2, LEN OUT NUMBER)
IS
  depID NUMBER;
  projName VARCHAR(20);
  projLen NUMBER;
BEGIN
  SELECT ID INTO depID FROM DEPARTMENTS
    WHERE NAME = DEP_NAME;
  SELECT ROUND(MAX(DATE_END - DATE_BEG),0) INTO projLen FROM PROJECTS
    WHERE PROJECTS.DEPARTMENT_ID = depID;
  SELECT NAME INTO projName FROM PROJECTS
    WHERE ROUND((DATE_END - DATE_BEG), 0) = projLen;
  DBMS_OUTPUT.PUT_LINE(projName || ' ' || projLen);
END LONGEST_PROJECT_BY_DEP;
```

Результат:



```
SDNM 155


PL/SQL procedure successfully completed.
```

Триггера

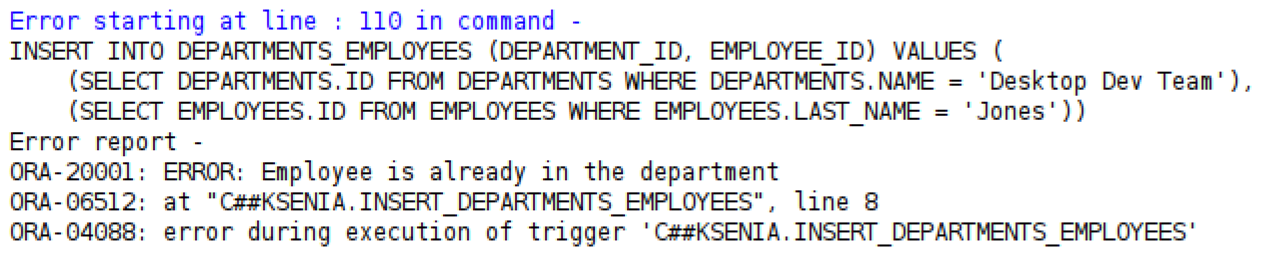
- Триггера на вставку:

    Создать триггер, который не позволяет добавить в отдел служащего, если он там уже есть.

```
CREATE OR REPLACE TRIGGER INSERT_DEPARTMENTS_EMPLOYEES
  BEFORE INSERT ON DEPARTMENTS_EMPLOYEES
  FOR EACH ROW
```

```
DECLARE
  emplID NUMBER;
BEGIN
  SELECT COUNT(*) INTO emplID FROM DEPARTMENTS_EMPLOYEES
    WHERE EMPLOYEE_ID = :NEW.EMPLOYEE_ID AND DEPARTMENT_ID = :NEW.DEPARTMENT_ID;
  IF emplID > 0
  THEN
    RAISE_APPLICATION_ERROR(-20001, 'ERROR: Employee is already in the department');
  END IF;
END INSERT_DEPARTMENTS_EMPLOYEES;
```

Результат:



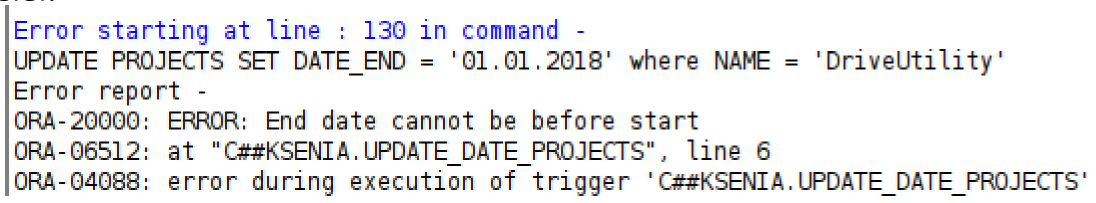
```
Error starting at line : 110 in command -
INSERT INTO DEPARTMENTS_EMPLOYEES (DEPARTMENT_ID, EMPLOYEE_ID) VALUES (
    (SELECT DEPARTMENTS.ID FROM DEPARTMENTS WHERE DEPARTMENTS.NAME = 'Desktop Dev Team'),
    (SELECT EMPLOYEES.ID FROM EMPLOYEES WHERE EMPLOYEES.LAST_NAME = 'Jones'))
Error report -
ORA-20001: ERROR: Employee is already in the department
ORA-06512: at "C##KSENIA.INSERT_DEPARTMENTS_EMPLOYEES", line 8
ORA-04088: error during execution of trigger 'C##KSENIA.INSERT_DEPARTMENTS_EMPLOYEES'
```

- Триггера на модификацию:

  Создать триггер, который не позволяет установить дату окончания проекта меньше, чем дата начала.

```
CREATE OR REPLACE TRIGGER UPDATE_DATE_PROJECTS
  BEFORE UPDATE ON PROJECTS
  FOR EACH ROW
DECLARE
  endDate DATE;
BEGIN
  IF :NEW.DATE_BEG > :NEW.DATE_END
    OR :NEW.DATE_BEG > :OLD.DATE_END
    OR :NEW.DATE_END < :OLD.DATE_BEG
  THEN
    RAISE_APPLICATION_ERROR(-20002, 'ERROR: End date cannot be before start');
  END IF;
END UPDATE_DATE_PROJECTS;
```

Результат:



```
Error starting at line : 130 in command -
UPDATE PROJECTS SET DATE_END = '01.01.2018' where NAME = 'DriveUtility'
Error report -
ORA-20000: ERROR: End date cannot be before start
ORA-06512: at "C##KSENIA.UPDATE_DATE_PROJECTS", line 6
ORA-04088: error during execution of trigger 'C##KSENIA.UPDATE_DATE_PROJECTS'
```

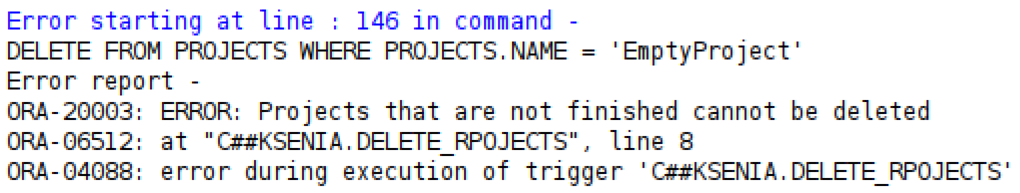
- Триггера на удаление:

  Создать триггер, который при удалении проекта в случае, если проект не завершен, откатывает транзакцию.

```
CREATE OR REPLACE TRIGGER DELETE_RPOJECTS
  BEFORE DELETE ON PROJECTS
  FOR EACH ROW
DECLARE
  endDate DATE;
```

```
BEGIN
  SELECT DATE_END_REAL INTO endDate FROM PROJECTS;
  IF endDate IS NULL
  THEN
    RAISE_APPLICATION_ERROR(-20003, 'ERROR: Projects that are not finished cannot be deleted');
  END IF;
END DELETE_RPOJECTS;
```

Результат:



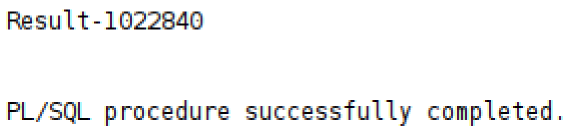
```
Error starting at line : 146 in command -
DELETE FROM PROJECTS WHERE PROJECTS.NAME = 'EmptyProject'
Error report -
ORA-20003: ERROR: Projects that are not finished cannot be deleted
ORA-06512: at "C##KSENIA.DELETE_RPOJECTS", line 8
ORA-04088: error during execution of trigger 'C##KSENIA.DELETE_RPOJECTS'
```

Курсоры

Хранимая процедура для расчета суммы прибыли от завершенных к настоящему времени проектов за период:

```
CREATE OR REPLACE PROCEDURE CALC_PROFIT (IN_DATE IN DATE, RES OUT NUMBER) AS
  CURSOR PROJ_CUR
  IS
  SELECT PROJECTS.ID,
    ROUND(PROJECTS.COST - NVL(SUM(EMPLOYEES.SALARY * TRUNC(PROJECTS.DATE_END_REAL -
PROJECTS.DATE_BEG )), 0), 2) AS PROFIT
  FROM PROJECTS
    JOIN DEPARTMENTS_EMPLOYEES ON PROJECTS.DEPARTMENT_ID =
DEPARTMENTS_EMPLOYEES.DEPARTMENT_ID
    JOIN EMPLOYEES ON DEPARTMENTS_EMPLOYEES.EMPLOYEE_ID = EMPLOYEES.ID
  WHERE PROJECTS.DATE_END_REAL < CURRENT_DATE AND PROJECTS.DATE_END_REAL > IN_DATE
  GROUP BY PROJECTS.ID, PROJECTS.COST, PROJECTS.DATE_END_REAL, PROJECTS.DATE_BEG;

BEGIN
  RES := 0;
  FOR I IN
    PROJ_CUR
  LOOP
    RES := RES + I.PROFIT;
  END LOOP;
END;
```

Результат:



```
Result-1022840


PL/SQL procedure successfully completed.
```