Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра информатики

Отчёт

Лабораторная работа №2

Выполнил:

студент группы №853504

Кузьма В.В.

Проверил:

Чащин С.В.

Минск 2021

ЗАДАНИЕ 1.

1. Построить две таблицы STUDENTS и GROUPS реализующих соответственно справочник студентов и справочник групп

Поле	Тип	комментарий
STUDENTS		
ID	Number	Код студента
NAME	VARCHAR2	Имя студента
GROUP_ID	Number	Код группы
GROUPS		
ID	Number	Код группы
NAME	VARCHAR2	Название группы
C_VAL	Number	Количество студентов в группе

```
CREATE TABLE STUDENTS(
   ID NUMBER,
   NAME VARCHAR2(100),
   GROUP_ID NUMBER
);

CREATE TABLE GROUPS(
   ID NUMBER,
   NAME VARCHAR2(100),
   C_VAL NUMBER
);
```

```
Table STUDENTS created.


Table GROUPS created.
```

ЗАДАНИЕ 2.

2. Реализовать триггеры для таблиц задания 1 проверку целостности (проверка на уникальность полей ID), генерацию автоинкрементного ключа и проверку уникальности для поля GROUP.NAME

```
CREATE SEQUENCE students_seq;
CREATE OR REPLACE TRIGGER unique_student
BEFORE INSERT
   ON STUDENTS
   FOR EACH ROW
DECLARE
   counter number;
BEGIN
   IF :new.id IS NOT NULL THEN
      select count(*) as c into counter
```

```
            from students
            where id = :new.id;
        IF counter > 0 THEN
            raise_application_error(-20000, 'Неверный id. Он должен быть уникальным. Попробуйте другой.');
        END IF;
    ELSE
        counter := 1;
        while counter > 0
        LOOP
            SELECT students_seq.nextval
            INTO :new.id
            FROM dual;
            select count(*) as c into counter
            from students
            where id = :new.id;
        END LOOP;
    END IF;

END;
```

ТЕСТ 1.

Добавление обычной записи

INSERT INTO students(id, name, group_id) VALUES (1, 'asd', 855);

INSERT INTO students(id, name, group_id) VALUES (3, 'aqsd', 8455);

SELECT * FROM students;

```
1 row inserted.


        ID NAME                                                                                                   GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                         855
         3 aqsd                                                                                                       8455
```

ТЕСТ 2.

Добавление записи с существующим id.

INSERT INTO students(id, name, group_id) VALUES (3, 'aqsd', 8455);

SELECT * FROM students;

```
Error starting at line : 33 in command -
INSERT INTO students(id, name, group_id) VALUES (3, 'aqsd', 8455)
Error report -
ORA-20000: Неверный id. Он должен быть уникальным. Попробуйте другой.
ORA-06512: на  "SYSTEM.UNIQUE_STUDENT", line 9
ORA-04088: ошибка во время выполнения триггера 'SYSTEM.UNIQUE_STUDENT'


        ID NAME                                                                                                   GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                         855
         3 aqsd                                                                                                       8455
```

ТЕСТ 3.

Добавление двух записей с автоинкрементом.

```
INSERT INTO students(name, group_id) VALUES ('aqsd', 8455);
SELECT * FROM students;
```

```
1 row inserted.


        ID NAME                                                                                                 GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                       855
         3 asd                                                                                                       855
         2 aqsd                                                                                                     8455


1 row inserted.


        ID NAME                                                                                                 GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                       855
         3 asd                                                                                                       855
         2 aqsd                                                                                                     8455
         4 aqsd                                                                                                     8455
```

```
CREATE SEQUENCE groups_seq;
CREATE OR REPLACE TRIGGER unique_group
BEFORE INSERT
   ON GROUPS
   FOR EACH ROW
DECLARE
   counter number;
BEGIN
   select count(*) as c into counter
         from groups
         where name = :new.name;
      IF counter > 0 THEN
         raise_application_error(-20000, 'Такая группа уже существует. Название
группы должно быть уникальным. Попробуйте другое.');
      END IF;
   IF :new.id IS NOT NULL THEN
      select count(*) as c into counter
         from groups
         where id = :new.id;
      IF counter > 0 THEN
         raise_application_error(-20000, 'Неверный id. Он должен быть
уникальным. Попробуйте другой.');
      END IF;
   ELSE
      counter := 1;
      while counter > 0
      LOOP
         SELECT groups_seq.nextval
         INTO :new.id
         FROM dual;
```

```
            select count(*) as c into counter
            from students
            where id = :new.id;
        END LOOP;
    END IF;

END;
```

ТЕСТ 4.

Добавление обычных записей

INSERT INTO groups(id, name, c_val) VALUES (1, 'asd', 8155);

INSERT INTO groups(id, name, c_val) VALUES (3, 'aqsd', 8455);

select * from groups;

```
1 row inserted.


1 row inserted.


        ID NAME                                                                                                      C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                        8155
         3 aqsd                                                                                                       8455
```

Тест 5.

Добавление записи с существующим id

INSERT INTO groups(id, name, c_val) VALUES (3, 'aqusd', 8455);

select * from groups;

```
Error starting at line : 38 in command -
INSERT INTO groups(id, name, c_val) VALUES (3, 'aqusd', 8455)
Error report -
ORA-20000: Неверный id. Он должен быть уникальным. Попробуйте другой.
ORA-06512: на  "SYSTEM.UNIQUE_GROUP", line 15
ORA-04088: ошибка во время выполнения триггера 'SYSTEM.UNIQUE_GROUP'


        ID NAME                                                                                                      C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                        8155
         3 aqsd                                                                                                       8455
```

Тест 6.

Добавление записи с существующим name

INSERT INTO groups(id, name, c_val) VALUES (4, 'aqsd', 8455);

select * from groups;

```
INSERT INTO groups(id, name, c_val) VALUES (4, 'aqsd', 8455)
Error report -
ORA-20000: Такая группа уже существует. Название группы должно быть уникальным. Попробуйте другое.
ORA-06512: на  "SYSTEM.UNIQUE_GROUP", line 8
ORA-04088: ошибка во время выполнения триггера 'SYSTEM.UNIQUE_GROUP'


        ID NAME                                                                                                      C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                        8155
         3 aqsd                                                                                                       8455
```

Тест 7.

Добавление двух записей с автоинкрементом.

INSERT INTO groups(name, c_val) VALUES ('aaqsd', 8355);
INSERT INTO groups(name, c_val) VALUES ('asqsd', 8655);
select * from groups;

```
1 row inserted.


1 row inserted.


        ID NAME                                                                                                    C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
         1 asd                                                                                                       8155
         3 aqsd                                                                                                      8455
         7 aaqsd                                                                                                     8355
         8 asqsd                                                                                                     8655
```

ЗАДАНИЕ 3.

3. Реализовать триггер реализующий Foreign Key с каскадным удалением между таблицами STUDENTS и GROUPS

```
CREATE OR REPLACE TRIGGER cascade_student
   AFTER DELETE
   ON GROUPS
   FOR EACH ROW
BEGIN
   DELETE FROM STUDENTS WHERE group_id = :old.id;
END;
```

ТЕСТ

Наполним таблицы данными

```
1 row inserted.


1 row inserted.


1 row inserted.


        ID NAME                                                                                                    C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
        24 001                                                                                                          0
        25 002                                                                                                          0
        26 003                                                                                                          0
```

```
        ID NAME                                                                                                 GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
        27 001                                                                                                         24
        28 002                                                                                                         24
        29 003                                                                                                         25
        30 004                                                                                                         25
        31 005                                                                                                         26
        32 006                                                                                                         26
```

Удалим группу 001
delete from groups where name = '001';
select * from students;
select * from groups;

```
1 row deleted.


        ID NAME                                                                                                   GROUP_ID
---------- ---------------------------------------------------------------------------------------------------- ----------
        29 003                                                                                                          25
        30 004                                                                                                          25
        31 005                                                                                                          26
        32 006                                                                                                          26


        ID NAME                                                                                                      C_VAL
---------- ---------------------------------------------------------------------------------------------------- ----------
        25 002                                                                                                           0
        26 003                                                                                                           0
```

ЗАДАНИЕ 4.

4. Реализовать триггер реализующий журналирование всех действий над данными таблицы STUDENTS

```
CREATE TABLE LOGGING_ACTIONS(
   ID NUMBER,
   operation VARCHAR2(10),
   ex_time TIMESTAMP,
   stud_id NUMBER,
   stud_name VARCHAR2(10),
   stud_group_id NUMBER,
   nstud_id NUMBER,
   nstud_name VARCHAR2(10),
   nstud_group_id NUMBER
);
CREATE SEQUENCE logger_seq;
CREATE OR replace trigger students_logger
   AFTER INSERT OR UPDATE OR DELETE
   ON STUDENTS
   FOR EACH ROW
DECLARE
   TEMP_ID NUMBER;
BEGIN
   SELECT logger_seq.NEXTVAL into TEMP_ID FROM dual;
   IF INSERTING THEN
      INSERT INTO LOGGING_ACTIONS VALUES(TEMP_ID, 'INSERT', SYSTIMESTAMP,
:new.id, :new.name, :new.group_id, NULL, NULL, NULL);
   END IF;

   IF UPDATING THEN
      INSERT INTO LOGGING_ACTIONS VALUES(TEMP_ID, 'UPDATE', SYSTIMESTAMP,
:old.id, :old.name, :old.group_id, :new.id, :new.name, :new.group_id);
   END IF;

   IF DELETING THEN
      INSERT INTO LOGGING_ACTIONS VALUES(TEMP_ID, 'delete', SYSTIMESTAMP, :old.id,
:old.name, :old.group_id, NULL, NULL, NULL);
   END IF;

END;
```

ТЕСТ

```
INSERT INTO students(name, group_id) values('001', 1);
INSERT INTO students(name, group_id) values('002', 2);
UPDATE students set name = '007' where name = '001';
delete from students where name = '002';

select * from LOGGING_ACTIONS;
```

ID	OPERATION	EX_TIME	STUD_ID	STUD_NAME	STUD_GROUP_ID	NSTUD_ID	NSTUD_NAME	NSTUD_GROUP_ID
25	INSERT	12.03.21 21:58:49,476000000	63	001	1			
26	INSERT	12.03.21 21:58:49,481000000	64	002	2			
27	UPDATE	12.03.21 21:58:49,486000000	63	001	1	63	007	1
28	delete	12.03.21 21:58:49,491000000	64	002	2			

ЗАДАНИЕ 5.

5. Исходя из данных предыдущей задачи, реализовать процедуру для восстановления информации на указанный временной момент и на временное смещение

```
CREATE OR REPLACE PROCEDURE restore_students(r_time in TIMESTAMP) IS
BEGIN
   FOR action IN (
   SELECT * FROM logging_actions WHERE r_time <= ex_time ORDER BY id DESC)
   LOOP
   IF action.operation = 'INSERT' THEN
      DELETE students WHERE id = action.stud_id;
   END IF;

   IF action.operation = 'UPDATE' THEN
      UPDATE students SET
      id = action.stud_id,
      name = action.stud_name,
      group_id = action.stud_group_id
      WHERE id = action.nstud_id;
   END IF;

   IF action.operation = 'DELETE' THEN
     INSERT INTO students VALUES (action.stud_id, action.stud_name, action.stud_group_id);
   END IF;
   END LOOP;
END;
```

ТЕСТ

```
SET SERVEROUTPUT ON
DECLARE
   status1 TIMESTAMP;
BEGIN

   INSERT INTO students(name, group_id) values('001', 1);
   INSERT INTO students(name, group_id) values('002', 2);
```

```
    status1 := SYSTIMESTAMP;
    for student in (select * from students)
    LOOP
        DBMS_OUTPUT.PUT_LINE('id = ' || student.id || ', name = ' || student.name || ', group_id =
' || student.group_id);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('___');
    UPDATE students set name = '007' where name = '001';
    DELETE students where name = '002';
    for student in (select * from students)
    LOOP
        DBMS_OUTPUT.PUT_LINE('id = ' || student.id || ', name = ' || student.name || ', group_id =
' || student.group_id);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('___');
    restore_students(status1);
    for student in (select * from students)
    LOOP
        DBMS_OUTPUT.PUT_LINE('id = ' || student.id || ', name = ' || student.name || ', group_id =
' || student.group_id);
    END LOOP;
    DBMS_OUTPUT.PUT_LINE('___');

END;
```

```
id = 125, name = 001, group_id = 1
id = 126, name = 002, group_id = 2
___
id = 125, name = 007, group_id = 1
___
id = 125, name = 001, group_id = 1
id = 126, name = 002, group_id = 2
___


PL/SQL procedure successfully completed.
```

ЗАДАНИЕ 6.

6. Реализовать триггер, который в случае изменения данных в таблице STUDENTS будет соответственно обновлять информацию C_VAL таблицы GROUPS

```
CREATE OR REPLACE TRIGGER c_val_update
    AFTER INSERT OR UPDATE OR DELETE
    ON students
    for each row
BEGIN
    IF INSERTING THEN
        UPDATE groups
        SET C_VAL = C_VAL + 1
```

```
    WHERE id = :new.group_id;
  END IF;

  IF INSERTING THEN
    UPDATE groups
    SET C_VAL = C_VAL - 1
    WHERE id = :old.group_id;

    UPDATE groups
    SET C_VAL = C_VAL + 1
    WHERE id = :new.group_id;
  END IF;

  IF DELETING THEN
    UPDATE groups
    SET C_VAL = C_VAL - 1
    WHERE id = :old.group_id;
  END IF;
END;
```

ТЕСТ

Наполним таблицу данными.

ID	NAME	C_VAL
1	001	0
25	002	0
26	003	0

```
INSERT INTO STUDENTS(name, group_id) values('01', 2);
INSERT INTO STUDENTS(name, group_id) values('02', 2);
INSERT INTO STUDENTS(name, group_id) values('03', 3);
INSERT INTO STUDENTS(name, group_id) values('04', 3);
INSERT INTO STUDENTS(name, group_id) values('05', 4);
INSERT INTO STUDENTS(name, group_id) values('06', 4);
select * from groups;
```

ID	NAME	C_VAL
2	001	2
3	002	2
4	003	2

```
UPDATE students
set group_id = 3
where name = '01';
select * from groups;
```

```
1 row updated.
```

ID	NAME	C_VAL
2	001	1
3	002	3
4	003	2

DELETE from students
where group_id = 4;
select * from groups;

```
2 rows deleted.
```

ID	NAME	C_VAL
2	001	1
3	002	3
4	003	0