**Нуритдинходжаева Алина С20-712**

**Лабораторная работа №5**

**Вариант №17**

**--Задание 0**

```
CREATE OR REPLACE TRIGGER check_film
BEFORE INSERT OR UPDATE ON timetables
FOR EACH ROW
DECLARE
CURSOR cur IS SELECT * FROM timetables;
r timetables%ROWTYPE;
ol integer;
nl integer;
BEGIN
OPEN cur;
FETCH cur INTO r;
SELECT time INTO nl FROM films WHERE id=:new.films_id;
WHILE cur%FOUND LOOP
SELECT time INTO ol FROM films WHERE id=r.films_id;
IF (r.hall = :new.hall) AND
(((r.date_time < :new.date_time) AND ((r.date_time + (ol/1440)) >
:new.date_time)) OR
((r.date_time < (:new.date_time + (nl/1440)) AND
((r.date_time + (ol/1440)) > (:new.date_time + (nl/1440))))
OR (r.date_time > :new.date_time) AND ((r.date_time + (ol/1440)) >
(:new.date_time + (nl/1440)))))
THEN
raise_application_error(-20000,'Время недопустимо для заданного кинозала');
END IF;
FETCH cur INTO r;
```

```
END LOOP;
CLOSE cur;
END;

--для проверки
INSERT INTO timetables
VALUES ('29.04.2022', 'Над уровнем неба', 1, 'м.Киевская, "Европейский"', 6);
```

**--Задание 1**

```
CREATE OR REPLACE FUNCTION Ccinemas (filmname IN varchar2)
return number IS
filmid integer;
newdate integer;
cnt number;
BEGIN
SELECT id INTO filmid FROM films WHERE movie_title = filmname;
SELECT time INTO newdate FROM films WHERE movie_title = filmname;
SELECT COUNT(cinemas_address) INTO cnt FROM timetables INNER JOIN
films
ON (films.id = timetables.films_id) WHERE (timetables.films_id = filmid)
AND (timetables.date_time < sysdate) OR (sysdate <
(timetables.date_time+newdate/1440));
return cnt;
END;

--для проверки
SELECT DISTINCT Ccinemas('Титаник'), cinemas_address FROM timetables
WHERE  film ='Титаник';
```

**--Задание 2**

```
CREATE OR REPLACE TYPE Func AS object
(
numb number,
static function ODCIAggregateInitialize(sctx IN OUT Func)
return number,
member function ODCIAggregateIterate(self IN OUT Func, value IN varchar2)
return number,
member function ODCIAggregateTerminate(self IN Func, returnValue OUT
number, flags IN number)
return number,
member function ODCIAggregateMerge(self IN OUT Func, ctx2 IN Func)
return number);
/

CREATE OR REPLACE TYPE BODY Func is
static function ODCIAggregateInitialize(sctx IN OUT Func)
return number IS
BEGIN
sctx := Func(0);
return ODCIConst.Success;
END;

member function ODCIAggregateIterate(self IN OUT Func , value IN varchar2)
return number IS
d_p number;
```

```
SELECT COUNT(movie_title) INTO d_p FROM films WHERE film_company =
value;
self.numb := d_p + self.numb;
return ODCIConst.Success;
END;

member function ODCIAggregateTerminate(self IN Func, returnValue OUT
number, flags IN number)
return number IS
BEGIN
returnValue := self.numb;
return ODCIConst.Success;
END;

member function ODCIAggregateMerge(self IN OUT Func, ctx2 IN Func)
return number IS
BEGIN
self.numb := self.numb + ctx2.numb;
return ODCIConst.Success;
END;
END;
/

CREATE OR REPLACE FUNCTION CountFilms(input varchar2) RETURN
number
PARALLEL_ENABLE AGGREGATE USING Func;
/

SELECT CountFilms('A24') AS "Всего фильмов" FROM films;
```

**--Задача 3**

```
CREATE OR REPLACE VIEW NewView AS
SELECT cinemas_address, film, date_time FROM timetables;

CREATE OR REPLACE TRIGGER T3
INSTEAD OF UPDATE ON NewView
FOR EACH ROW
BEGIN
UPDATE timetables t SET date_time = :new.date_time
WHERE date_time = :old.date_time ;
--AND film_id = (SELECT id FROM films WHERE films.id = t.films_id AND movie_title = :old.movie_title);
END;

UPDATE NewView SET date_time = '05.05.2021' WHERE cinemas_address ='м.Автозаводская, ТРЦ "Ревьера"'
AND film ='Брат';
```

**--Задание 4**

```
SET SERVEROUTPUT ON;
CREATE TABLE stack (
 ordinal_num INTEGER NOT NULL UNIQUE,
 string varchar2(64 CHAR)
);
/

CREATE OR REPLACE FUNCTION check_stack
RETURN integer IS
```

```
 st_exist numeric(1);
 cnt integer;
BEGIN

--Проверка на инициализацию
SELECT CASE
WHEN EXISTS(SELECT * FROM stack) THEN 1
ELSE 0
END
INTO st_exist FROM dual;

 --проверка на наличие значений
 IF (st_exist = 1) THEN
 SELECT COUNT(*) INTO cnt FROM stack;
 IF (cnt > 1) THEN --ЕСТЬ ХОТЯ БЫ 1 ЗНАЧЕНИЕ
 RETURN 2;
 ELSE            --ПРОИНИЦИАЛИЗИРОВАН, НО НЕТ НИ ОДНОГО
ЗНАЧЕНИЯ
 RETURN 1;
 END IF;
 ELSE
 RETURN 0;       --НЕ ПРОИНИЦИАЛИЗИРОВАН
 END IF;
END;
/

CREATE OR REPLACE PROCEDURE PUSH (str IN varchar2) IS
 ord INTEGER;
 check_val INTEGER;
```

```
BEGIN
 check_val := CHECK_STACK();
 IF (check_val = 1 OR check_val = 2) THEN
 SELECT MAX(ordinal_num) + 1 INTO ord FROM stack;
 INSERT INTO stack(ordinal_num, string) VALUES(ord, str);
 DBMS_OUTPUT.put_line('Значение добавлено на вершину стека.');
 ELSE
 RAISE_APPLICATION_ERROR(-20000, 'СТЕК НЕ
ПРОИНИЦИАЛИЗИРОВАН!');
 END IF;
END;
/

CREATE OR REPLACE FUNCTION POP
RETURN varchar2 IS
 str varchar2(64 CHAR);
 check_val INTEGER;
BEGIN
 check_val := CHECK_STACK();
 IF (check_val = 1) THEN
 DBMS_OUTPUT.put_line('Стек пуст!');
 RETURN '';
 ELSIF(check_val = 2) THEN
 SELECT string INTO str FROM stack WHERE ordinal_num = (SELECT
MAX(ordinal_num) FROM stack);
 DELETE FROM stack WHERE EXISTS(SELECT * FROM stack) AND
ordinal_num = (SELECT MAX(ordinal_num) FROM stack);
 RETURN str;
 ELSE
```

```
 RAISE_APPLICATION_ERROR(-20000, 'СТЕК НЕ
ПРОИНИЦИАЛИЗИРОВАН!');
 END IF;
END;
/

CREATE OR REPLACE PROCEDURE INIT
IS
 check_val INTEGER;
BEGIN
 check_val := check_stack();
 IF (check_val = 1 OR check_val = 2) THEN
 DBMS_OUTPUT.put_line('Стек уже проинициализирован!');
 ELSE
 INSERT INTO stack VALUES(0, null);
 DBMS_OUTPUT.put_line('Инициализация прошла успешно!');
 END IF;
END;
/

CREATE OR REPLACE PROCEDURE EMPTYy IS
BEGIN
 DELETE FROM stack;
 DBMS_OUTPUT.put_line('Стек очищен!');
 INIT();
END;
/

CREATE OR REPLACE FUNCTION TOP
```

```
RETURN varchar2 IS
 str varchar2(64 CHAR) := null;
 check_val INTEGER;
BEGIN
 check_val := CHECK_STACK();
 IF (check_val = 1) THEN
 DBMS_OUTPUT.put_line('Стек пустой!');
 RETURN '';
 ELSIF (check_val = 2) THEN
 SELECT string INTO str FROM stack WHERE ordinal_num = (SELECT
MAX(ordinal_num) FROM stack);
 RETURN str;
 ELSE
 RAISE_APPLICATION_ERROR(-20000, 'СТЕК НЕ
ПРОИНИЦИАЛИЗИРОВАН!');
 END IF;
END;
/


BEGIN
 INIT();
END;


BEGIN
 DBMS_OUTPUT.put_line(TOP());
END;


BEGIN
 PUSH('a');
```

```
 PUSH('б');
 PUSH('в');
 PUSH('г');
 PUSH('д');
 PUSH('е');
END;

BEGIN
DBMS_OUTPUT.put_line(POP());
END;

BEGIN
EMPTYy();
END;
/
```