Федеральное агентство связи

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра ПМиК

Практическое задание №6

по дисциплине «Сетевые базы данных»

“Коллекции”

Выполнил:

студент гр. МГ-211 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Бурдуковский И.А./

подпись

Проверил:

Доцент

кафедры ПМиК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Приставка П.А./

Новосибирск 2023 г.

Оглавление

[Задание 3](#_Toc96693268)

[Выполнение 4](#_Toc96693269)

# Задание

Ассоциативные массивы

1. Создать таблицу БД, содержащую информацию об автомобилях, допускаемых к въезду на территорию парковки организации (уникальный признак – гос. номер). Заполнить ее тестовыми данными. Реализовать кэширующую в коллекцию типа ассоциативный массив процедуру поиска въезжающего автомобиля. Коллекция должна обеспечивать учет статусов хранящихся в ней автомобилей – «въехал» и «выехал». Реализовать процедуру поиска автомобиля по гос. номеру, предоставляющую информацию о том въезжал ли искомый автомобиль на территорию в отчетном периоде (период сессии) и его текущем статусе. (Указания: Определение типа коллекции и объявление ассоциативного массива реализовать внутри пакета. Автомобиль считается въезжавшим на территорию, если соответствующий элемент находится в коллекции).

Массивы перменной длины (VARRAY)

2. Определить хранимый в БД тип коллекции (тип уровня схемы) соответствующий массиву переменной длины. Создать в БД таблицу, хранящую информацию о маршруте: название, список населенных пунктов, входящих в маршрут, в порядке следования. Список населенных пунктов хранить в поле таблицы в виде коллекции типа VARRAY. Заполнить таблицу тестовыми данными.Реализовать процедуру добавления нового населенного пункта в заданный маршрут.

Вложенные таблицы (NESTED TABLES)

3. Определить хранимый в БД тип коллекции (тип уровня схемы) соответствующий вложенной таблице. Создать в БД таблицу, содержащую информацию о стране и цветах ее флага. Цвета флага хранить коллекции типа NESTED TABLES непосредственно в поле таблицы. Заполнить ее тестовыми данными (названия стран). Реализовать процедуру внесения цветовой гаммы для заданной страны.

Операции с мультимножествами

4. Реализовать функцию, которая для заданных стран из таблицы п. 3 возвращает список общих цветов. Реализовать процедуру распечатку списка общих цветов.

# Выполнение

Ассоциативные массивы

1. Ассоциативные массивы. Создать таблицу БД, содержащую информацию об автомобилях, допускаемых к въезду на территорию парковки организации (уникальный признак – гос. номер). Заполнить ее тестовыми данными.

Реализовать кэширующую в коллекцию типа ассоциативный массив процедуру поиска въезжающего автомобиля. Коллекция должна обеспечивать учет статусов хранящихся в ней автомобилей – «въехал» и «выехал».

Реализовать процедуру поиска автомобиля по гос. номеру, предоставляющую информацию о том въезжал ли искомый автомобиль на территорию в отчетном периоде (период сессии) и его текущем статусе. (Указания: Определение типа коллекции и объявление ассоциативного массива реализовать внутри пакета. Автомобиль считается въезжавшим на территорию, если соответствующий элемент находится в коллекции).

Создание таблицы:

CREATE TABLE car (numer VARCHAR2(100) );

ALTER TABLE car

ADD (CONSTRAINT car\_pk\_numer PRIMARY KEY (numer));

Заполнение таблицы:

INSERT INTO car VALUES (‘B555PX39’);

INSERT INTO car VALUES ('C976MM78');

INSERT INTO car VALUES ('K976MM78');

INSERT INTO car VALUES ('M976MM77');

INSERT INTO car VALUES ('O000OO19');

INSERT INTO car VALUES ('X777OY17');

Реализация задания:

CREATE OR REPLACE PACKAGE parking

IS

PROCEDURE set\_status

(num IN car.pnum%TYPE, value\_status IN BOOLEAN);

PROCEDURE search\_by\_number

(num IN car.pnum%TYPE);

END parking;

/

CREATE OR REPLACE PACKAGE BODY parking

IS

TYPE carType IS TABLE OF BOOLEAN

INDEX BY car.pnum%TYPE;

incomers carType;

PROCEDURE set\_status

(num IN car.pnum%TYPE, value\_status IN BOOLEAN)

IS

FUNCTION check\_car RETURN BOOLEAN IS

value\_number car.pnum%TYPE;

BEGIN

IF incomers.EXISTS(num) THEN

RETURN TRUE;

END IF;

SELECT pnum INTO value\_number

FROM car WHERE pnum = num;

RETURN TRUE;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

END;

BEGIN

IF check\_car THEN

incomers(num) := value\_status;

IF NOT incomers.EXISTS(num) THEN

DBMS\_OUTPUT.PUT\_LINE('Въезд разрешен, но у водителя не получилось заехать: ' || num || '.');

ELSE

IF value\_status THEN

DBMS\_OUTPUT.PUT\_LINE('Заехал: ' || num || '.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Выехал: ' || num || '.');

END IF;

END IF;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Въезд запрещен: ' || num || '.');

END IF;

END set\_status;

PROCEDURE search\_by\_number (num IN car.pnum%TYPE)

IS

BEGIN

IF incomers.EXISTS(num) THEN

IF incomers(num) THEN

DBMS\_OUTPUT.PUT\_LINE('Автомобиль ' || num || ' находится на парковке.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Автомобиль ' || num || ' выехал с парковки.');

END IF;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Автомобиль ' || num || ' никогда не был на этой парковке.');

END IF;

END search\_by\_number;

END parking;

Проверка:

BEGIN

parking.set\_status('K976MM78', TRUE);

parking.set\_status('B555PX39', TRUE);

parking.set\_status('K976MM78', FALSE);

parking.search\_by\_number('K976MM78');

parking.search\_by\_number('B555PX39');

parking.search\_by\_number('C976MM78');

END;

Вывод:

Заехал: K976MM78.  
Заехал: B555PX39.  
Выехал: K976MM78.  
Автомобиль K976MM78 выехал с парковки.  
Автомобиль B555PX39 находится на парковке.  
Автомобиль C976MM78 никогда не был на этой парковке.

Массивы перменной длины (VARRAY).

1. Определить хранимый в БД тип коллекции (тип уровня схемы) соответствующий массиву переменной длины.  
   Создать в БД таблицу, хранящую информацию о маршруте: название, список населенных пунктов, входящих в маршрут, в порядке следования. Список населенных пунктов хранить в поле таблицы в виде коллекции типа VARRAY. Заполнить таблицу тестовыми данными. Реализовать процедуру добавления нового населенного пункта в заданный маршрут.

Создание типа и таблицы:

CREATE OR REPLACE TYPE LOCALITY\_T IS VARRAY(50) OF VARCHAR2(50);

DROP TABLE routes;

CREATE TABLE routes (

title VARCHAR2(100),

locality LOCALITY\_T

);

Заполнение таблицы:

BEGIN

INSERT INTO routes VALUES ('Новосибирск-Томск',

LOCALITY\_T('Новосибирск', 'Юрга', 'Томск'));

INSERT INTO routes VALUES ('Новосибирск-Барнаул',

LOCALITY\_T('Новосибирск', 'Бердск', 'Черепаново', 'Барнаул'));

INSERT INTO routes VALUES ('Омск-Красноярск',

LOCALITY\_T('Омск', 'Татарск', 'Барабинск', 'Кемерово', 'Красноярск'));

END;

Скрипт:

CREATE OR REPLACE PROCEDURE add\_locality (value\_in IN VARCHAR2, title\_in

IN VARCHAR2, pos IN INTEGER)

IS

lists LOCALITY\_T;

current VARCHAR2(50);

next VARCHAR2(50);

BEGIN

SELECT locality INTO lists FROM routes WHERE title = title\_in;

DBMS\_OUTPUT.PUT\_LINE('Старый маршрут:');

FOR element IN 1 .. lists.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE(lists(element));

END LOOP;

IF lists.COUNT < 50 AND (pos >= 1 AND pos <= lists.COUNT + 1) THEN

lists.EXTEND;

FOR element IN 1 .. lists.COUNT

LOOP

IF element = pos THEN

current := lists (element);

lists (element) := value\_in;

ELSIF element > pos THEN

next := lists (element);

lists (element) := current;

current := next;

END IF;

END LOOP;

UPDATE routes SET locality = lists WHERE title = title\_in;

END IF;

SELECT locality INTO lists FROM routes WHERE title = title\_in;

DBMS\_OUTPUT.PUT\_LINE('Новый маршрут:');

FOR element IN 1 .. lists.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE(lists(element));

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Маршрут с названием ' || title\_in || ' отсутствует');

END add\_locality;/

Проверка:

BEGIN

add\_locality ('Новосибирск', 'Омск-Красноярск', 4);

END;

Вывод:

Старый маршрут:

Омск

Татарск

Барабинск

Кемерово

Красноярск

Новый маршрут:  
Омск

Татарск

Барабинск

Новосибирск

Кемерово

Красноярск

Вложенные таблицы (NESTED TABLES).

1. Определить хранимый в БД тип коллекции (тип уровня схемы) соответствующий вложенной таблице. Создать в БД таблицу, содержащую информацию о стране и цветах ее флага. Цвета флага хранить коллекции типа NESTED TABLES непосредственно в поле таблицы. Заполнить ее тестовыми данными (названия стран). Реализовать процедуру внесения цветовой гаммы для заданной страны.

Создание типа и таблицы:

CREATE OR REPLACE TYPE COLORS\_T AS TABLE OF VARCHAR2(50);

DROP TABLE country;

CREATE TABLE country (

title VARCHAR2(100),

colors COLORS\_T

) NESTED TABLE colors STORE AS colors\_model\_tab;

Заполнение таблицы:

BEGIN

INSERT INTO country VALUES ('Россия', colors\_t('белый', 'синий', 'красный'));

INSERT INTO country VALUES ('Китай', colors\_t('красный', 'желтый'));

END;

Скрипт:

CREATE OR REPLACE PROCEDURE set\_color\_to\_flag (title\_in IN VARCHAR2,

colors\_in IN COLORS\_T)

IS

colors COLORS\_T;

BEGIN

SELECT colors INTO colors FROM country WHERE title = title\_in;

DBMS\_OUTPUT.PUT\_LINE('Старый флаг:');

FOR element IN 1 .. colors.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE(colors (element));

END LOOP;

UPDATE country SET colors = colors\_in WHERE title = title\_in;

colors := colors\_in;

DBMS\_OUTPUT.PUT\_LINE('Новый флаг:');

FOR element IN 1 .. colors.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE(colors (element));

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Нет страны ' || title\_in || '!');

END set\_color\_to\_flag;

/

Проверка:

BEGIN

set\_color\_to\_flag('Китай', COLORS\_T('желтый', 'синий', 'красный')); END;

Вывод:

Старый флаг:  
красный  
желтый  
Новый флаг:  
желтый  
синий

красный

Операции с мультимножествами.

1. Реализовать функцию, которая для заданных стран из таблицы п. 3 возвращает список общих цветов. Реализовать процедуру распечатку списка общих цветов.

Скрипт:

CREATE OR REPLACE FUNCTION get\_general\_colors

RETURN COLORS\_T

AS

general\_colors COLORS\_T := COLORS\_T ();

first\_colors COLORS\_T;

BEGIN

FOR rec IN (SELECT title from country)

LOOP

SELECT colors INTO first\_colors FROM country WHERE title = rec.title;

IF general\_colors.COUNT = 0 THEN

general\_colors := first\_colors;

ELSE

general\_colors := general\_colors MULTISET INTERSECT first\_colors;

END IF;

END LOOP;

RETURN general\_colors;

END;

/

CREATE OR REPLACE PROCEDURE print\_general\_colors (list\_colors IN COLORS\_T )

AS

BEGIN

FOR element IN 1 .. list\_colors.COUNT

LOOP

DBMS\_OUTPUT.PUT\_LINE(list\_colors (element));

END LOOP;

END;

/

Проверка:

BEGIN

print\_general\_colors (get\_general\_colors);

END;

Вывод:

красный