Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

Индивидуальная работа No.2

“Системы управления базами данных”

Проверил: Светлана Бодруг

Выполнил: Чобану Артём

Группа: i1902

Кишинев 2021

Задание:

Индивидуальное задание

Создать объектную БД

Создание типов 5

Создание объектных таблиц (таблица хранит объект целиком) 5

Создать связи между объектами 5 (REF)

Хранить объекты таблицах в виде атрибутов (ЗАПОЛНЕНИЕ 5, ВЫБОРКА 5) 5

Атрибуты в виде коллекций (таблицы 5, массивы 5) (ЗАПОЛНЕНИЕ 5, ВЫБОРКА 5)

Использовать операцию BULK 5

Создать иерархию типов 5 (перегружать 5 и/или переопределять методы 5, вызов методов 5)

Использовать различные типы выборки информации:

VALUE 5, DEREF 5, IS OF 5, иерархия 5 (разнотипная инф)

95

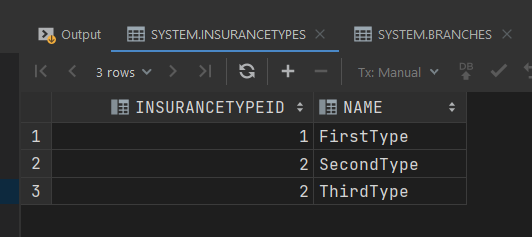
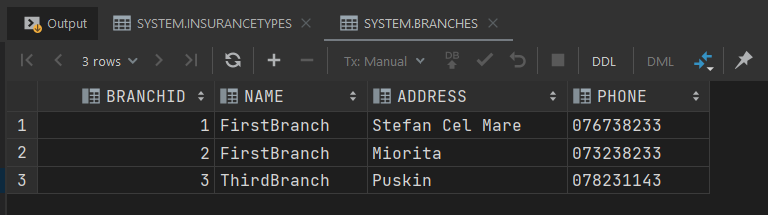
|  |  |
| --- | --- |
| 5 | 30 |
| 6 | 45 |
| 7 | 55 |
| 8 | 65 |
| 9 | 75 |
| 10 | 90 |

create or replace type InsuranceAgent as object  
(  
 InsuranceAgentID int,  
 FirstName varchar(50),  
 LastName varchar(50),  
 Patronymic varchar(50),  
 Address varchar(50),  
 Phone varchar(10),  
  
 AgentBranch ref Branch  
);  
  
create or replace type Contract as object  
(  
 ContractID int,  
 CreationDate DATE,  
 ContractSum int,  
 TariffRate int,  
  
 ContractInsuranceType ref InsuranceType,  
 ContractBranch ref Branch,  
  
  
 constructor function Contract(  
 contractID int,  
 creationDate DATE,  
 contractSum int,  
 tariffRate int) return self as result,  
  
 constructor function Contract(  
 contractID int,  
 creationDate DATE,  
 contractSum int,  
 tariffRate int,  
 insurance ref InsuranceType,  
 branch ref Branch) return self as result  
);  
  
create or replace type body Contract as  
 constructor function Contract(  
 contractID int,  
 creationDate DATE,  
 contractSum int,  
 tariffRate int) return self as result  
 is  
 begin  
 SELF.CONTRACTID := contractID;  
 SELF.CREATIONDATE := creationDate;  
 SELF.TARIFFRATE := tariffRate;  
 SELF.CONTRACTSUM := contractSum;  
 return;  
 end;  
  
 constructor function Contract(  
 contractID int,  
 creationDate DATE,  
 contractSum int,  
 tariffRate int,  
 insurance ref InsuranceType,  
 branch ref Branch) return self as result  
 is  
 begin  
 SELF.CONTRACTID := contractID;  
 SELF.CREATIONDATE := creationDate;  
 SELF.TARIFFRATE := tariffRate;  
 SELF.CONTRACTSUM := contractSum;  
 SELF.CONTRACTINSURANCETYPE := insurance;  
 SELF.CONTRACTBRANCH := branch;  
 return;  
 end;  
end;  
  
  
create or replace type InsuranceType as object  
(  
 InsuranceTypeID int,  
 Name varchar(50)  
);  
  
create or replace type Branch as object  
(  
 BranchID int,  
 Name varchar(50),  
 Address varchar(50),  
 Phone varchar(10)  
);  
  
create table InsuranceAgents of InsuranceAgent;  
create table Contracts of Contract;  
create table InsuranceTypes of InsuranceType;  
create table Branches of BRANCH;

Для типа Contract было создано 2 конструктора.

insert into BRANCHES  
values (3, 'ThirdBranch', 'Puskin', '075221999')

insert into InsuranceTypes  
values (1, 'FirstInsuranceType')



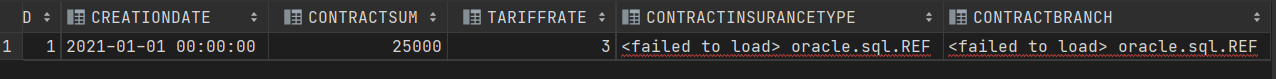
declare  
 insType ref INSURANCETYPE;  
 bran ref BRANCH;  
 branchName varchar(50);  
begin  
  
 select ref(IT)  
 into insType  
 from INSURANCETYPES IT  
 where IT.NAME like '%First%';  
  
 select REF(B)  
 into bran  
 from BRANCHES B  
 where B.BRANCHID = 1;  
  
 select B.NAME  
 into branchName  
 from BRANCHES B  
 where B.BRANCHID = 1;  
  
 DBMS\_OUTPUT.PUT\_LINE(branchName);  
end;

Вывод:

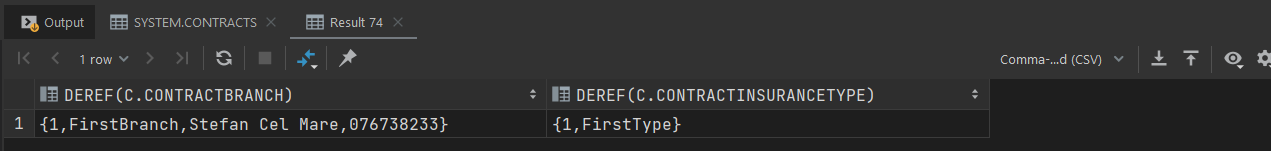


Добавление нового контракта:

insert into CONTRACTS C  
values (1,  
 TO\_DATE('01/01/2021', 'DD/MM/YYYY'),  
 25000,  
 3,  
 insType,  
 bran)  
returning REF(C) into contr;



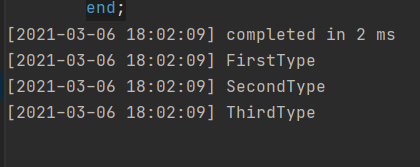
select deref(C.CONTRACTBRANCH), deref(c.CONTRACTINSURANCETYPE)  
from CONTRACTS C;



Bulk collect:

Команда увеличивает производительность, помещая данные в коллекцию.

declare  
 type InsuranceTable is table of INSURANCETYPES.NAME%type;  
 names InsuranceTable;  
begin  
  
 select (IT.Name) bulk collect  
 into names  
 from InsuranceTypes IT;  
  
 for nameIndex in names.FIRST .. names.LAST  
 loop  
 DBMS\_OUTPUT.PUT\_LINE(names(nameIndex));  
 end loop;  
  
 end;



Второй вариант с явным использованием курсора:

declare  
 type InsuranceTable is table of INSURANCETYPES.NAME%type;  
 names InsuranceTable;  
 cursor names\_cursor is select Name  
 from InsuranceTypes;  
begin  
  
 open names\_cursor;  
 fetch names\_cursor bulk collect into names;  
 close names\_cursor;  
  
 for nameIndex in names.FIRST .. names.LAST  
 loop  
 DBMS\_OUTPUT.PUT\_LINE(names(nameIndex));  
 end loop;  
  
end;

declare  
 it INSURANCETYPE;  
 cursor branches\_cursor is select deref(C.CONTRACTINSURANCETYPE)  
 from CONTRACTS C where C.CONTRACTID = 1;  
begin  
 open branches\_cursor;  
 fetch branches\_cursor into it;  
 close branches\_cursor;  
 DBMS\_OUTPUT.PUT\_LINE(it.NAME);  
end;

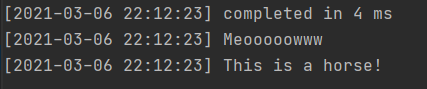
Наследование, иерархия типов:

create or replace type Animal as object  
(  
 Name varchar(50),  
 Age int,  
  
 member procedure SaySomething  
) not final instantiable;  
create or replace type Cat under Animal  
(  
 WhiskersLength int, --Длина усов  
  
 overriding  
 member procedure SaySomething  
);  
create or replace type body Cat as  
 overriding  
 member procedure SaySomething  
 is  
 begin  
 DBMS\_OUTPUT.PUT\_LINE('Meooooowww');  
 end;  
end;  
create or replace type Horse under Animal  
(  
 HoovesThickness int, --Количество копыт  
  
 overriding  
 member procedure SaySomething  
);  
create or replace type body Horse as  
 overriding  
 member procedure SaySomething  
 is  
 begin  
 DBMS\_OUTPUT.PUT\_LINE('This is a horse!');  
 end;  
end;

Использование типов в качестве базового:

declare  
 someAnimal Animal;  
begin  
 someAnimal := Cat('Кот Кузилий', 5, 4);  
 someAnimal.SAYSOMETHING;  
  
 someAnimal := Horse('Плотва', 4, 25);  
 someAnimal.SAYSOMETHING;  
end;

Вывод:

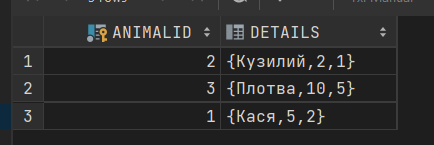


Создадим таблицу класса Animal:

create table Animals  
(  
 AnimalID int primary key,  
 Details ANIMAL  
);  
  
insert into Animals  
values (1, Cat('Кася', 5, 6));

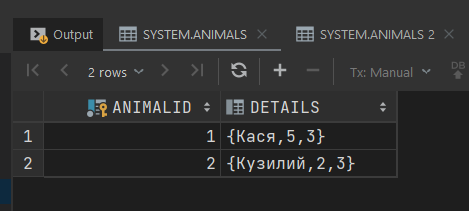
insert into Animals  
values (3, Horse('Плотва', 10, 5));

Итоговая таблица:



Вывод только кошек:

select \* from Animals  
where Details IS OF (Cat);



Второй вариант создания подобной таблицы:

create table Animals of Animal;  
  
insert into Animals  
values (Cat('Кася', 5, 3));

select Value(A)  
from Animals A  
where VALUE(A) IS OF (Cat);

