Государственный Университет Молдовы

Факультет Математики и Информатики

Департамент Информатики

Индивидуальная работа No.3

“Системы управления базами данных”

Проверил: Светлана Бодруг

Выполнил: Чобану Артём

Группа: i1902

Кишинев 2021

Задание:

Создать БД для индивидуальной темы:

1. Использовать Primary Key, Foreign key – 10
2. Нарисовать схему отношений – 10
3. Заполнить – 10
4. Выполнить 10 запросов в PL/SQL:

Используйте анонимные блоки 10, процедур 10, функции 10

1. Используйте Exceptions 10
2. Используйте Insert, update 10, delete 10,
3. Используйте if 10, cursor 2 типов 20

|  |  |
| --- | --- |
| 5 | 40 |
| 6 | 50 |
| 7 | 60 |
| 8 | 70 |
| 9 | 80 |
| 10 | 100 |

Предметная область:

# **Страховая компания**

Вы работаете в страховой компании. Вашей задачей является отслеживание ее финансовой деятельности.

Компания имеет различные филиалы по всей стране. Каждый филиал характеризуется названием, адресом и телефоном.

Деятельность компании организована следующим образом: к вам обращаются различные лица с целью заключения договора о страховании.

В зависимости от принимаемых на страхование объектов и страхуемых рисков договор заключается по определенному виду страхования (например, страхование автотранспорта от угона, страхование домашнего имущества, добровольное медицинское страхование). При заключении договора вы фиксируете дату заключения, страховую сумму, вид страхования, тарифную ставку и филиал, в котором заключался договор.

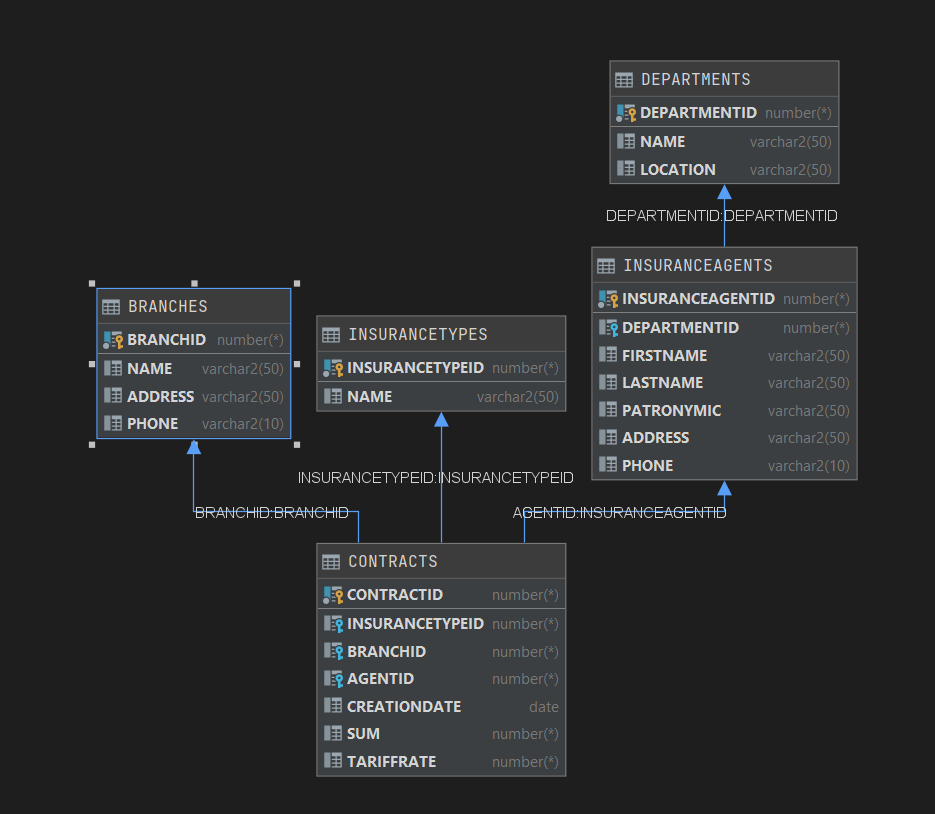
**Таблицы:**

**Договоры** (Номер договора, Дата заключения, Страховая сумма, Тарифная ставка, Код филиала, Код вида страхования).

**Вид страхования** (Код вида страхования, Наименование).

**Филиал** (Код филиала, Наименование филиала, Адрес, Телефон).

Диаграмма отношений таблиц:



Создание таблиц:

create table InsuranceTypes  
(  
 InsuranceTypeID int generated always as identity (start with 1 increment by 1),  
 Name varchar(50),  
  
 constraint PK\_InsuranceType primary key (InsuranceTypeID)  
);  
create table Branches  
(  
 BranchID int generated always as identity (start with 1 increment by 1),  
 Name varchar(50),  
 Address varchar(50),  
 Phone varchar(10),  
  
 constraint PK\_Branch primary key (BranchID)  
);  
create table Departments  
(  
 DepartmentID int generated always as identity (start with 1 increment by 1),  
 Name varchar(50),  
 Location varchar(50),  
  
 constraint PK\_Department primary key (DepartmentID)  
);  
create table InsuranceAgents  
(  
 InsuranceAgentID int generated always as identity (start with 1 increment by 1),  
 DepartmentID int,  
 FirstName varchar(50),  
 LastName varchar(50),  
 Patronymic varchar(50),  
 Address varchar(50),  
 Phone varchar(10),  
  
 constraint PK\_InsuranceAgent primary key (InsuranceAgentID),  
 constraint FK\_DepartmentID foreign key (DepartmentID) references Departments(DepartmentID)  
);  
create table Contracts  
(  
 ContractID int generated always as identity (start with 1 increment by 1),  
 InsuranceTypeID int,  
 BranchID int,  
 AgentID int,  
 CreationDate DATE,  
 Sum int,  
 TariffRate int,  
  
 constraint PK\_ContractID primary key (ContractID),  
 constraint FK\_InsuranceTypeID foreign key (InsuranceTypeID) references InsuranceTypes (InsuranceTypeID),  
 constraint FK\_BranchID foreign key (BranchID) references Branches (BranchID),  
 constraint FK\_AgentID foreign key (AgentID) references InsuranceAgents (InsuranceAgentID)  
);

Заполнение таблиц:

insert into INSURANCETYPES(Name)  
VALUES ('ThirdInsuranceType');

insert into BRANCHES(NAME, ADDRESS, PHONE)  
values ('FirstBranch', 'FirstAddress', 'FirstPhone');

insert into DEPARTMENTS(NAME, LOCATION)  
values ('Departmenttwo', 'LocationTwo');

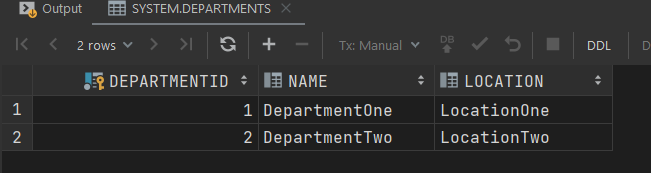
insert into INSURANCEAGENTS

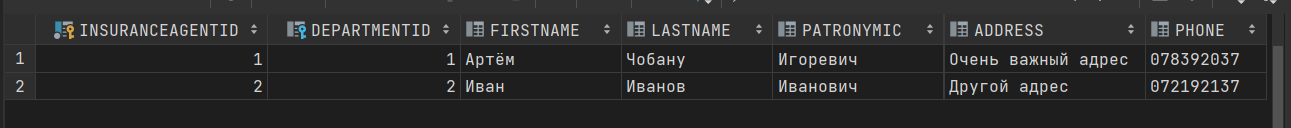
(DEPARTMENTID, FIRSTNAME, LASTNAME, PATRONYMIC, ADDRESS, PHONE)  
values

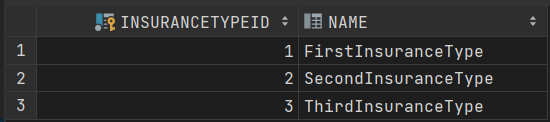
(1,'Артём', 'Чобану', 'Игоревич', 'Очень важный адрес', '078392037');

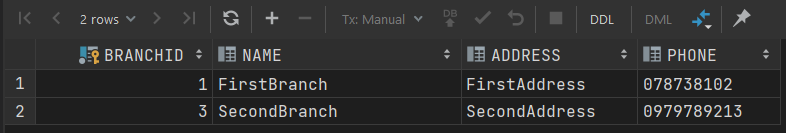
insert into CONTRACTS(INSURANCETYPEID, BRANCHID, AGENTID, CREATIONDATE, SUM, TARIFFRATE)  
values (1, 1, 1, to\_date('02/04/2020', 'DD/MM/YYYY'), 25000, 2);

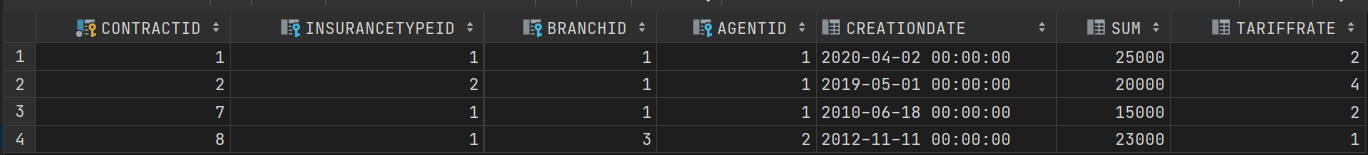
Результат заполнения:











Обновление таблиц:

update BRANCHES  
SET PHONE = '078738102'  
where BRANCHID = 1

update DEPARTMENTS  
set NAME = 'DepartmentTwo'  
where DEPARTMENTID = 2

Функция, вычисляющая тариф:

create or replace function CountTarrif(contractSum int, rate int)  
 return float  
as  
 result float;  
begin  
 result := contractSum / 100 \* rate;  
  
 return result;  
end;

Функция, выводящая тарифы всех

create or replace procedure PrintAllInsuranceTypesForDepartment(department int)  
as  
 cursor insurance\_cursor is  
 select IT.INSURANCETYPEID,  
 IT.NAME,  
 C.CONTRACTID  
 from DEPARTMENTS D  
 join INSURANCEAGENTS I on D.DEPARTMENTID = I.DEPARTMENTID  
 join CONTRACTS C on I.INSURANCEAGENTID = C.AGENTID  
 join INSURANCETYPES IT on IT.INSURANCETYPEID = C.INSURANCETYPEID  
 where D.DEPARTMENTID = department;  
 Name varchar(50);  
 ID int;  
 contract int;  
begin  
 open insurance\_cursor;  
 loop  
 fetch insurance\_cursor into ID, Name, contract;  
 DBMS\_OUTPUT.PUT\_LINE('Имя: ' || Name || ', ID: ' || ID || ', ID контракта: ' || contract);  
  
 exit when insurance\_cursor%NotFound;  
 end loop;  
  
 close insurance\_cursor;  
end;

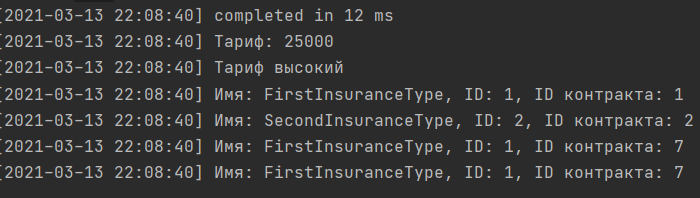
Удаление:

delete from CONTRACTS  
where INSURANCETYPEID = 2;

Анонимный блок:

declare  
 tarrif float;  
begin  
 tarrif := CountTarrif(25000, 100);  
 DBMS\_OUTPUT.PUT\_LINE('Тариф: ' || tarrif);  
  
 if tarrif > 10000 then  
 DBMS\_OUTPUT.PUT\_LINE('Тариф высокий');  
 else  
 DBMS\_OUTPUT.PUT\_LINE('Тариф низкий');  
 end if;  
  
 PRINTALLINSURANCETYPESFORDEPARTMENT(1);  
end;

Вывод:



Модифицируем функцию под проверку на деление на ноль:

create or replace function CountTarrif(contractSum int, rate int)  
 return float  
as  
 result float;  
begin  
 if rate = 0 then  
 raise ZERO\_DIVIDE;  
 end if;  
  
 result := contractSum / 100 \* rate;  
 return result;  
end;

begin  
 DBMS\_OUTPUT.PUT\_LINE('Тариф: ' || CountTarrif(25000, 0));  
  
exception  
 when Zero\_divide then  
 DBMS\_OUTPUT.PUT\_LINE('ДЕЛЕНИЕ НА 0!');  
end;