Приложение Б. Скрипт базы данных

АННОТАЦИЯ

В данном программном документе приведен скрипт базы данных для информационной системы автосервиса.

1. СКРИПТ БАЗЫ ДАННЫХ
   1. Наименование скрипта

Наименование – «AutoService.sql».

* 1. Область применения скрипта

Скрипт предназначен для программы «Информационная система автосервис».

* 1. Скрипт

CREATE DATABASE "AutoService"

WITH

OWNER = postgres

ENCODING = 'UTF8'

CONNECTION LIMIT = 10;

create table RoleUser

(

RoleName varchar(30) not null constraint PK\_RoleNumber primary key unique

);

create or replace procedure RoleUser\_Insert(p\_RoleName varchar(30))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM RoleUser

where RoleName = p\_RoleName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into RoleUser(RoleName)

values (p\_RoleName);

end if;

end;

$$;

call RoleUser\_Insert('Администратор');

call RoleUser\_Insert('Сотрудник отдела закупок');

call RoleUser\_Insert('Клиент');

call RoleUser\_Insert('Сотрудник склада');

call RoleUser\_Insert('Сотрудник отдела продаж');

call RoleUser\_Insert('Сотрудник ремонтного отдела');

create or replace procedure RoleUser\_Update(p\_RoleName varchar(30), p\_newRoleName varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM RoleUser

where RoleName = p\_newRoleName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update RoleUser set

RoleName = p\_newRoleName

where

RoleName = p\_RoleName;

end if;

end;

$$;

create or replace procedure RoleUser\_Delete(p\_RoleName varchar(30))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where RoleName = p\_RoleName;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from RoleUser

where

RoleName = p\_RoleName;

--end if;

end;

$$;

create table Users

(

Email varchar(50) not null constraint PK\_Email primary key unique,

Password varchar(30) not null,

LastName varchar(30) not null,

FirstName varchar(30) not null,

MiddleName varchar(30) not null,

RegistrationDate date not null default CURRENT\_DATE,

RoleName varchar(30) not null,

foreign key (RoleName) references RoleUser (RoleName) on update cascade on delete cascade,

life boolean not null default True

);

INSERT INTO Users VALUES ('diagram@mail.ru', 'qwe123', 'Special','For','Diagramm','2019-10-10','Сотрудник склада');

INSERT INTO Users VALUES ('diagram2@mail.ru', 'qwe123', 'Special','For','Diagramm','2019-10-10','Сотрудник склада');

INSERT INTO Users VALUES ('diagram3@mail.ru', 'qwe123', 'Special','For','Diagramm','2019-10-10','Сотрудник отдела закупок');

INSERT INTO Users VALUES ('diagram4@mail.ru', 'qwe123', 'Special','For','Diagramm','2019-10-10','Сотрудник склада');

INSERT INTO Users VALUES ('diagram5@mail.ru', 'qwe123', 'Special','For','Diagramm','2019-10-10','Сотрудник отдела закупок');

INSERT INTO Users VALUES ('diagram6@mail.ru', 'qwe123', 'Special','For','Diagramm','2020-10-10','Клиент');

INSERT INTO Users VALUES ('diagram7@mail.ru', 'qwe123', 'Special','For','Diagramm','2020-10-10','Клиент');

INSERT INTO Users VALUES ('diagram8@mail.ru', 'qwe123', 'Special','For','Diagramm','2020-10-10','Клиент');

INSERT INTO Users VALUES ('diagram9@mail.ru', 'qwe123', 'Special','For','Diagramm','2020-10-10','Клиент');

INSERT INTO Users VALUES ('diagram10@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник отдела продаж');

INSERT INTO Users VALUES ('diagram11@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник отдела продаж');

INSERT INTO Users VALUES ('diagram12@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник отдела продаж');

INSERT INTO Users VALUES ('diagram13@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник отдела продаж');

INSERT INTO Users VALUES ('diagram14@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник отдела продаж');

INSERT INTO Users VALUES ('diagram15@mail.ru', 'qwe123', 'Special','For','Diagramm','2021-10-10','Сотрудник ремонтного отдела');

create or replace view Users\_View ("Email", "Пароль", "Фамилия", "Имя", "Отчество", "Дата регистрации","Роль")

as

select

email,

password,

lastname,

firstname,

middlename,

RegistrationDate,

rolename

from Users where life = true;

create or replace procedure Users\_Insert(p\_Email varchar(50), p\_Password varchar(30), p\_LastName varchar(30), p\_FirstName varchar(30), p\_MiddleName varchar(30), p\_RoleName varchar(30))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Users

where Email = p\_Email and Password = p\_Password and LastName = p\_LastName and FirstName = p\_FirstName and MiddleName = p\_MiddleName and RoleName = p\_RoleName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Users(Email,Password,LastName,FirstName,MiddleName,RoleName)

values (p\_Email,p\_Password,p\_LastName,p\_FirstName,p\_MiddleName,p\_RoleName);

end if;

end;

$$;

call Users\_insert('Admin', 'Admin', 'Admin', 'Admin', 'Admin', 'Администратор');

create or replace procedure Users\_Update(p\_Email varchar(50), p\_Password varchar(30), p\_LastName varchar(30), p\_FirstName varchar(30), p\_MiddleName varchar(30), p\_RoleName varchar(30), p\_newEmail varchar(50) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Users

where Email = p\_newEmail and Password = p\_Password and LastName = p\_LastName and FirstName = p\_FirstName and MiddleName = p\_MiddleName and RoleName = p\_RoleName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Users set

Password = p\_Password,

LastName = p\_LastName,

FirstName = p\_FirstName,

MiddleName = p\_MiddleName,

RoleName = p\_RoleName,

Email = p\_newEmail

where

Email = p\_Email;

end if;

end;

$$;

create or replace procedure Users\_Delete(p\_Email varchar(50))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where Email = p\_Email;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Users

where

Email = p\_Email;

--end if;

end;

$$;

create table Client

(

ClientNumber varchar(10) not null constraint PK\_ClientNumber primary key unique,

PassportSeries varchar(4) not null,

PassportNumber varchar(6) not null,

Email varchar(50) not null,

foreign key (Email) references Users (Email) on update cascade on delete cascade

);

create or replace procedure Client\_Insert(p\_ClientNumber varchar(10), p\_PassportSeries varchar(4), p\_PassportNumber varchar(6), p\_Email varchar(50))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Client

where ClientNumber = p\_ClientNumber and PassportSeries = p\_PassportSeries and PassportNumber = p\_PassportNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Client(ClientNumber,PassportSeries,PassportNumber,Email)

values (p\_ClientNumber,p\_PassportSeries,p\_PassportNumber,p\_Email);

end if;

end;

$$;

create or replace procedure Client\_Update(p\_ClientNumber varchar(10), p\_PassportSeries varchar(4), p\_PassportNumber varchar(6), p\_Email varchar(50), p\_newClientNumber varchar(10) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Client

where ClientNumber = p\_newClientNumber and PassportSeries = p\_PassportSeries and PassportNumber = p\_PassportNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Client set

PassportSeries = p\_PassportSeries,

PassportNumber = p\_PassportNumber,

Email = p\_Email,

ClientNumber = p\_newClientNumber

where

ClientNumber = p\_ClientNumber;

end if;

end;

$$;

create or replace procedure Client\_Delete(p\_ClientNumber varchar(10))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where ClientNumber = p\_ClientNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Client

where

ClientNumber = p\_ClientNumber;

--end if;

end;

$$;

create or replace function Sign\_IN(login varchar(30), pass varchar(30))

returns varchar(30)

language plpgsql

as

$$

DECLARE have\_record int := count (\*) FROM Users

where email = login and password = pass;

begin

if have\_record>0 then

return rolename from users where email = login and password = pass;

else

return 'No';

end if;

end;

$$

create table PurchaseOrder

(

PurchaseOrderNumber varchar(35) not null constraint PK\_PurchaseOrderNumber primary key unique,

PurchaseOrderDate date not null,

Quantity int not null,

RotalPrice decimal(38,2) not null,

Article varchar(35) not null

);

call PurchaseOrder\_Insert('test2', '2022-10-10', '33', '38.2', '123456')

create or replace procedure PurchaseOrder\_Insert(p\_PurchaseOrderNumber varchar(35), p\_PurchaseOrderDate date, p\_Quantity int, p\_RotalPrice decimal(38,2), p\_Article varchar(35))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM PurchaseOrder

where PurchaseOrderNumber = p\_PurchaseOrderNumber;

begin

if have\_record>0 then

raise exception 'Такой номер договора на поставку уже есть!';

else

insert into PurchaseOrder(PurchaseOrderNumber,PurchaseOrderDate,Quantity,RotalPrice,Article)

values (p\_PurchaseOrderNumber,p\_PurchaseOrderDate,p\_Quantity,p\_RotalPrice,p\_Article);

end if;

end;

$$;

create or replace procedure PurchaseOrder\_Update(p\_PurchaseOrderNumber varchar(35), p\_PurchaseOrderDate date, p\_Quantity int, p\_RotalPrice decimal(38,2), p\_Article varchar(35), p\_newPurchaseOrderNumber varchar(35) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM PurchaseOrder

where PurchaseOrderNumber = p\_PurchaseOrderNumber and PurchaseOrderDate = p\_PurchaseOrderDate and Quantity = p\_Quantity and RotalPrice = p\_RotalPrice and Article = p\_Article;

begin

if have\_record>0 then

raise exception 'Такой договор на поставку уже есть!';

else

update PurchaseOrder set

PurchaseOrderDate = p\_PurchaseOrderDate,

Quantity = p\_Quantity,

RotalPrice = p\_RotalPrice,

Article = p\_Article,

PurchaseOrderNumber = p\_newPurchaseOrderNumber

where

PurchaseOrderNumber = p\_PurchaseOrderNumber;

end if;

end;

$$;

create or replace procedure PurchaseOrder\_Delete(p\_PurchaseOrderNumber varchar(35))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where PurchaseOrderNumber = p\_PurchaseOrderNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from PurchaseOrder

where

PurchaseOrderNumber = p\_PurchaseOrderNumber;

--end if;

end;

$$;

create table Suplier

(

INN varchar(10) not null constraint PK\_INN primary key unique,

SuplierName varchar(30) not null,

PaymentAccount varchar(20) not null

);

call Suplier\_Insert('1234567890', 'Petrovuch', '12345678901234567890')

create or replace procedure Suplier\_Insert(p\_INN varchar(10), p\_SuplierName varchar(30), p\_PaymentAccount varchar(20))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Suplier

where INN = p\_INN or SuplierName = p\_SuplierName or PaymentAccount = p\_PaymentAccount ;

begin

if have\_record>0 then

raise exception 'Введенные данные поставщика не уникальны';

else

insert into Suplier(INN,SuplierName,PaymentAccount)

values (p\_INN,p\_SuplierName,p\_PaymentAccount);

end if;

end;

$$;

create or replace procedure Suplier\_Update(p\_INN varchar(10), p\_SuplierName varchar(30), p\_PaymentAccount varchar(20), p\_newINN varchar(10) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Suplier

where INN = p\_newINN and SuplierName = p\_SuplierName and PaymentAccount = p\_PaymentAccount ;

begin

if have\_record>0 then

raise exception 'Введенные данные поставщика не уникальны';

else

update Suplier set

SuplierName = p\_SuplierName,

PaymentAccount = p\_PaymentAccount,

INN = p\_newINN

where

INN = p\_INN;

end if;

end;

$$;

create or replace procedure Suplier\_Delete(p\_INN varchar(10))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where INN = p\_INN;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Suplier

where

INN = p\_INN;

--end if;

end;

$$;

create table PurchaseContract

(

PurchaseContractNumber varchar(35) not null constraint PK\_PurchaseContractNumber primary key unique,

PurchaseContractDate date not null,

PurchaseOrderNumber varchar(35) not null,

foreign key (PurchaseOrderNumber) references PurchaseOrder (PurchaseOrderNumber) on update cascade on delete cascade,

INN varchar(10) not null,

foreign key (INN) references Suplier (INN) on update cascade on delete cascade,

Email varchar(50) not null,

foreign key (Email) references Users (Email) on update cascade on delete cascade

);

call PurchaseContract\_Insert('1345gfge', '2022-10-10', 'test2', '1234567890', 'Zak@mail.ru')

create or replace procedure PurchaseContract\_Insert(p\_PurchaseContractNumber varchar(35), p\_PurchaseContractDate date, p\_PurchaseOrderNumber varchar(35), p\_INN varchar(10), p\_Email varchar(50))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM PurchaseContract

where PurchaseContractNumber = p\_PurchaseContractNumber and PurchaseContractDate = p\_PurchaseContractDate and PurchaseOrderNumber = p\_PurchaseOrderNumber and INN = p\_INN;

begin

if have\_record>0 then

raise exception 'Такой Контракт на поставку уже есть';

else

insert into PurchaseContract(PurchaseContractNumber,PurchaseContractDate,PurchaseOrderNumber,INN,Email)

values (p\_PurchaseContractNumber,p\_PurchaseContractDate,p\_PurchaseOrderNumber,p\_INN,p\_Email);

end if;

end;

$$;

select \* from Users where (lastname ||' '|| firstname||' '|| middlename) = 'Special For Diagramm'

create or replace procedure PurchaseContract\_Update(p\_PurchaseContractNumber varchar(35), p\_PurchaseContractDate date, p\_PurchaseOrderNumber varchar(35), p\_INN varchar(10), p\_Email varchar(50), p\_newPurchaseContractNumber varchar(35) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM PurchaseContract

where PurchaseContractNumber = p\_PurchaseContractNumber ;

begin

if have\_record>0 then

raise exception 'Контракт на поставку с таким номером уже есть';

else

update PurchaseContract set

PurchaseContractDate = p\_PurchaseContractDate,

PurchaseOrderNumber = p\_PurchaseOrderNumber,

INN = p\_INN,

Email = p\_Email,

PurchaseContractNumber = p\_newPurchaseContractNumber

where

PurchaseContractNumber = p\_PurchaseContractNumber;

end if;

end;

$$;

create or replace procedure PurchaseContract\_Delete(p\_PurchaseContractNumber varchar(35))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where PurchaseContractNumber = p\_PurchaseContractNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from PurchaseContract

where

PurchaseContractNumber = p\_PurchaseContractNumber;

--end if;

end;

$$;

create or replace view PurchaseContract\_View ("purchasecontractnumber", "purchasecontractdate", "purchaseordernumber", "inn", "email")

as

select

purchasecontractnumber,

purchasecontractdate,

purchaseordernumber,

supliername,

lastname ||' '|| firstname||' '|| middlename

from PurchaseContract

inner join Suplier on PurchaseContract.inn = Suplier.inn

inner join Users on PurchaseContract.email = Users.email;

create table Disposable

(

SeriesNumber varchar(35) not null constraint PK\_SeriesNumber primary key unique,

DisposableName varchar(35) not null,

DisposableWeight decimal(38,2) not null,

DisposableColor varchar(20) not null,

DisposableMaterial varchar(35) not null,

DisposableSertificate varchar(35) not null,

PurchaseContractNumber varchar(35) not null,

foreign key (PurchaseContractNumber) references PurchaseContract (PurchaseContractNumber) on update cascade on delete cascade

);

call Disposable\_Insert('iurrtuigjv8634673', 'iueryugeig', '45.6', 'Red', 'Materiable', 'oureouviu87er87','1345gfge')

create or replace procedure Disposable\_Insert(p\_SeriesNumber varchar(35), p\_DisposableName varchar(35), p\_DisposableWeight decimal(38,2), p\_DisposableColor varchar(20), p\_DisposableMaterial varchar(35), p\_DisposableSertificate varchar(35), p\_PurchaseContractNumber varchar(35))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Disposable

where SeriesNumber = p\_SeriesNumber and DisposableSertificate = p\_DisposableSertificate and PurchaseContractNumber = p\_PurchaseContractNumber ;

begin

if have\_record>0 then

raise exception 'Расходник с таким серийным номером или сертификатом или контрактом на поставку уже есть';

else

insert into Disposable(SeriesNumber,DisposableName,DisposableWeight,DisposableColor,DisposableMaterial,DisposableSertificate,PurchaseContractNumber)

values (p\_SeriesNumber,p\_DisposableName,p\_DisposableWeight,p\_DisposableColor,p\_DisposableMaterial,p\_DisposableSertificate,p\_PurchaseContractNumber);

end if;

end;

$$;

create or replace procedure Disposable\_Update(p\_SeriesNumber varchar(35), p\_DisposableName varchar(35), p\_DisposableWeight decimal(38,2), p\_DisposableColor varchar(20), p\_DisposableMaterial varchar(35), p\_DisposableSertificate varchar(35), p\_PurchaseContractNumber varchar(35), p\_newSeriesNumber varchar(35) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Disposable

where SeriesNumber = p\_SeriesNumber and DisposableSertificate = p\_DisposableSertificate and PurchaseContractNumber = p\_PurchaseContractNumber ;

begin

if have\_record>0 then

raise exception 'Расходник с таким серийным номером или сертификатом или контрактом на поставку уже есть';

else

update Disposable set

DisposableName = p\_DisposableName,

DisposableWeight = p\_DisposableWeight,

DisposableColor = p\_DisposableColor,

DisposableMaterial = p\_DisposableMaterial,

DisposableSertificate = p\_DisposableSertificate,

PurchaseContractNumber = p\_PurchaseContractNumber,

SeriesNumber = p\_newSeriesNumber

where

SeriesNumber = p\_SeriesNumber;

end if;

end;

$$;

create or replace procedure Disposable\_Delete(p\_SeriesNumber varchar(35))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where SeriesNumber = p\_SeriesNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Disposable

where

SeriesNumber = p\_SeriesNumber;

--end if;

end;

$$;

create table Warehouse

(

WarehouseName varchar(30) not null constraint PK\_WarehouseName primary key unique,

WarehouseAddress varchar(60) not null

);

create or replace procedure Warehouse\_Insert(p\_WarehouseName varchar(30), p\_WarehouseAddress varchar(60))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Warehouse

where WarehouseName = p\_WarehouseName and WarehouseAddress = p\_WarehouseAddress;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Warehouse(WarehouseName,WarehouseAddress)

values (p\_WarehouseName,p\_WarehouseAddress);

end if;

end;

$$;

create or replace procedure Warehouse\_Update(p\_WarehouseName varchar(30), p\_WarehouseAddress varchar(60), p\_newWarehouseName varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Warehouse

where WarehouseName = p\_newWarehouseName and WarehouseAddress = p\_WarehouseAddress;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Warehouse set

WarehouseAddress = p\_WarehouseAddress,

WarehouseName = p\_newWarehouseName

where

WarehouseName = p\_WarehouseName;

end if;

end;

$$;

create or replace procedure Warehouse\_Delete(p\_WarehouseName varchar(30))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where WarehouseName = p\_WarehouseName;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Warehouse

where

WarehouseName = p\_WarehouseName;

--end if;

end;

$$;

create table Cell

(

CellNumber varchar(5) not null constraint PK\_CellNumber primary key unique,

WarehouseName varchar(30) not null,

foreign key (WarehouseName) references Warehouse (WarehouseName) on update cascade on delete cascade

);

call Cell\_Insert('345', 'lhloh')

create or replace procedure Cell\_Insert(p\_CellNumber varchar(5), p\_WarehouseName varchar(30))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Cell

where CellNumber = p\_CellNumber and WarehouseName = p\_WarehouseName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Cell(CellNumber,WarehouseName)

values (p\_CellNumber,p\_WarehouseName);

end if;

end;

$$;

create or replace procedure Cell\_Update(p\_CellNumber varchar(5), p\_WarehouseName varchar(30), p\_newCellNumber varchar(5) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Cell

where CellNumber = p\_newCellNumber and WarehouseName = p\_WarehouseName ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Cell set

WarehouseName = p\_WarehouseName,

CellNumber = p\_newCellNumber

where

CellNumber = p\_CellNumber;

end if;

end;

$$;

create or replace procedure Cell\_Delete(p\_CellNumber varchar(5))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where CellNumber varchar(5) = p\_CellNumber varchar(5);

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Cell

where

CellNumber = p\_CellNumber;

--end if;

end;

$$;

create table Invoice

(

InvoiceNmber varchar(35) not null constraint PK\_InvoiceNmber primary key unique ,

InvoiceDate date not null,

Quantity int not null,

SeriesNumber varchar(35) not null,

foreign key (SeriesNumber) references Disposable (SeriesNumber) on update cascade on delete cascade,

CellNumber varchar(5) not null,

foreign key (cellnumber) references Cell (cellnumber) on update cascade on delete cascade,

Email varchar(50) not null,

foreign key (Email) references Users (Email) on update cascade on delete cascade

);

create or replace procedure Invoice\_Insert(p\_InvoiceNmber varchar(35), p\_InvoiceDate date, p\_Quantity int, p\_SeriesNumber varchar(35), p\_CellNumber varchar(5), p\_Email varchar(50))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Invoice

where InvoiceNmber = p\_InvoiceNmber and InvoiceDate = p\_InvoiceDate and Quantity = p\_Quantity and SeriesNumber = p\_SeriesNumber and CellNumber = p\_CellNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Введеная накладная уже есть';

else

insert into Invoice(InvoiceNmber,InvoiceDate,Quantity,SeriesNumber,CellNumber,Email)

values (p\_InvoiceNmber,p\_InvoiceDate,p\_Quantity,p\_SeriesNumber,p\_CellNumber,p\_Email);

end if;

end;

$$;

create or replace procedure Invoice\_Update(p\_InvoiceNmber varchar(35), p\_InvoiceDate date, p\_Quantity int, p\_SeriesNumber varchar(35), p\_CellNumber varchar(5), p\_Email varchar(50), p\_newInvoiceNmber varchar(35) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Invoice

where InvoiceNmber = p\_newInvoiceNmber and InvoiceDate = p\_InvoiceDate and Quantity = p\_Quantity and SeriesNumber = p\_SeriesNumber and CellNumber = p\_CellNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Invoice set

InvoiceDate = p\_InvoiceDate,

Quantity = p\_Quantity,

SeriesNumber = p\_SeriesNumber,

CellNumber = p\_CellNumber,

Email = p\_Email,

InvoiceNmber = p\_newInvoiceNmber

where

InvoiceNmber = p\_InvoiceNmber;

end if;

end;

$$;

create or replace procedure Invoice\_Delete(p\_InvoiceNmber varchar(35))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where InvoiceNmber = p\_InvoiceNmber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Invoice

where

InvoiceNmber = p\_InvoiceNmber;

--end if;

end;

$$;

create or replace view Invoice\_View ("invoicenmber", "invoicedate", "quantity", "seriesnumber", "cellnumber", "email")

as

select

invoicenmber,

invoicedate,

quantity,

Disposable.seriesnumber,

Cell.warehousename || ' - '|| Cell.cellnumber,

Users.email

from Invoice

inner join Disposable on Invoice.seriesnumber = Disposable.seriesnumber

inner join Cell on Invoice.cellnumber = Cell.cellnumber

inner join Users on Invoice.email = Users.email;

create table Car

(

CarNumber varchar(6) not null constraint PK\_CarNumber primary key unique,

CarBrand varchar(35) not null,

CarModel varchar(50) not null,

STS varchar(10) not null,

VIN varchar(17) not null,

ClientNumber varchar(10) not null,

foreign key (ClientNumber) references Client (ClientNumber) on update cascade on delete cascade

);

create or replace procedure Car\_Insert(p\_CarNumber varchar(6), p\_CarBrand varchar(35), p\_CarModel varchar(50), p\_STS varchar(10), p\_VIN varchar(17), p\_ClientNumber varchar(10))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Car

where CarNumber = p\_CarNumber and CarBrand = p\_CarBrand and CarModel = p\_CarModel and STS = p\_STS and VIN = p\_VIN and ClientNumber = p\_ClientNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Car(CarNumber,CarBrand,CarModel,STS,VIN,ClientNumber)

values (p\_CarNumber,p\_CarBrand,p\_CarModel,p\_STS,p\_VIN,p\_ClientNumber);

end if;

end;

$$;

create or replace procedure Car\_Update(p\_CarNumber varchar(6), p\_CarBrand varchar(35), p\_CarModel varchar(50), p\_STS varchar(10), p\_VIN varchar(17), p\_ClientNumber varchar(10), p\_newCarNumber varchar(6) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Car

where CarNumber = p\_newCarNumber and CarBrand = p\_CarBrand and CarModel = p\_CarModel and STS = p\_STS and VIN = p\_VIN and ClientNumber = p\_ClientNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Car set

CarBrand = p\_CarBrand,

CarModel = p\_CarModel,

STS = p\_STS,

VIN = p\_VIN,

ClientNumber = p\_ClientNumber,

CarNumber = p\_newCarNumber

where

CarNumber = p\_CarNumber;

end if;

end;

$$;

create or replace procedure Car\_Delete(p\_CarNumber varchar(6))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where CarNumber = p\_CarNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Car

where

CarNumber = p\_CarNumber;

--end if;

end;

$$;

create table OrderClient

(

OrderNumber varchar(35) not null constraint PK\_OrderNumber primary key unique,

OrderSum decimal(38,2) not null,

OrderDate date not null,

CarNumber varchar(6) not null,

foreign key (CarNumber) references Car (CarNumber) on update cascade on delete cascade

);

create or replace procedure OrderClient\_Insert(p\_OrderNumber varchar(35), p\_OrderSum decimal(38,2), p\_OrderDate date, p\_ClientNumber varchar(10))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM OrderClient

where OrderNumber = p\_OrderNumber and OrderSum = p\_OrderSum and OrderDate = p\_OrderDate and CarNumber = p\_ClientNumber ;

begin

if have\_record>0 then

raise exception 'Уже записан';

else

insert into OrderClient(OrderNumber,OrderSum,OrderDate,CarNumber)

values (p\_OrderNumber,p\_OrderSum,p\_OrderDate,p\_ClientNumber);

end if;

end;

$$;

create or replace procedure OrderClient\_Update(p\_OrderNumber varchar(35), p\_OrderSum decimal(38,2), p\_OrderDate date, p\_ClientNumber varchar(10), p\_newOrderNumber varchar(35) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM OrderClient

where OrderNumber = p\_newOrderNumber and OrderSum = p\_OrderSum and OrderDate = p\_OrderDate and CarNumber = p\_ClientNumber ;

begin

if have\_record>0 then

raise exception 'Уже записан';

else

update OrderClient set

OrderSum = p\_OrderSum,

OrderDate = p\_OrderDate,

CarNumber = p\_ClientNumber,

OrderNumber = p\_newOrderNumber

where

OrderNumber = p\_OrderNumber;

end if;

end;

$$;

create or replace procedure OrderClient\_Delete(p\_OrderNumber varchar(35))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where OrderNumber = p\_OrderNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from OrderClient

where

OrderNumber = p\_OrderNumber;

--end if;

end;

$$;

create table Service

(

ServiceName varchar(30) not null constraint PK\_ServiceName primary key unique,

Price decimal(38,2) not null,

ServiceDescription text not null

);

call Service\_Insert('Ремонт КПП', '45800', 'Если коробка переключения передач выходит из строя, не стоит даже пробовать выполнить ремонт своими руками, такой сложный агрегат, требует исключительную точность. Диагностика и ремонт. Мы проводим точную предварительную диагностику, что намного повышает качество ремонтных работ. Наши специалисты быстро и по доступной цене отремонтируют КПП')

create or replace procedure Service\_Insert(p\_ServiceName varchar(30), p\_Price decimal(38,2), p\_ServiceDescription text)

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Service

where ServiceName = p\_ServiceName and Price = p\_Price and ServiceDescription = p\_ServiceDescription ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Service(ServiceName,Price,ServiceDescription)

values (p\_ServiceName,p\_Price,p\_ServiceDescription);

end if;

end;

$$;

create or replace procedure Service\_Update(p\_ServiceName varchar(30), p\_Price decimal(38,2), p\_ServiceDescription text, p\_newServiceName varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Service

where ServiceName = p\_newServiceName and Price = p\_Price and ServiceDescription = p\_ServiceDescription ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Service set

Price = p\_Price,

ServiceDescription = p\_ServiceDescription,

ServiceName = p\_newServiceName

where

ServiceName = p\_ServiceName;

end if;

end;

$$;

create or replace procedure Service\_Delete(p\_ServiceName varchar(30))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where ServiceName = p\_ServiceName;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Service

where

ServiceName = p\_ServiceName;

--end if;

end;

$$;

create table ServiceOrder

(

ID\_ServiceOrder SERIAL PRIMARY KEY,

ServiceName varchar(30) not null,

foreign key (ServiceName) references Service (ServiceName) on update cascade on delete cascade,

OrderNumber varchar(35) not null,

foreign key (OrderNumber) references OrderClient (OrderNumber) on update cascade on delete cascade

);

create or replace procedure ServiceOrder\_Insert(p\_ServiceName varchar(30), p\_OrderNumber varchar(35))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM ServiceOrder

where ServiceName = p\_ServiceName and OrderNumber = p\_OrderNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into ServiceOrder(ServiceName,OrderNumber)

values (p\_ServiceName,p\_OrderNumber);

end if;

end;

$$;

create or replace procedure ServiceOrder\_Update(p\_ID\_ServiceOrder int, p\_ServiceName varchar(30), p\_OrderNumber varchar(35), p\_newID\_ServiceOrder int )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM ServiceOrder

where ServiceName = p\_ServiceName and OrderNumber = p\_OrderNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update ServiceOrder set

ServiceName = p\_ServiceName,

OrderNumber = p\_OrderNumber

where

ID\_ServiceOrder = p\_ID\_ServiceOrder;

end if;

end;

$$;

create or replace procedure ServiceOrder\_Delete(p\_ID\_ServiceOrder int)

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where ID\_ServiceOrder = p\_ID\_ServiceOrder;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from ServiceOrder

where

ID\_ServiceOrder = p\_ID\_ServiceOrder;

--end if;

end;

$$;

create table Contract

(

ContractNumber varchar(30) not null constraint PK\_ContractNumber primary key unique,

DateComclusion date not null,

OrderNumber varchar(35) not null,

foreign key (OrderNumber) references orderclient (OrderNumber) on update cascade on delete cascade,

ClientNumber varchar(10) not null,

foreign key (ClientNumber) references client (ClientNumber) on update cascade on delete cascade,

Email varchar(50) not null,

foreign key (Email) references Users (Email) on update cascade on delete cascade

);

create or replace procedure Contract\_Insert(p\_ContractNumber varchar(30), p\_DateComclusion date, p\_OrderNumber varchar(35), p\_ClientNumber varchar(10), p\_Email varchar(50))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Contract

where ContractNumber = p\_ContractNumber and DateComclusion = p\_DateComclusion and OrderNumber = p\_OrderNumber and ClientNumber = p\_ClientNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Contract(ContractNumber,DateComclusion,OrderNumber,ClientNumber,Email)

values (p\_ContractNumber,p\_DateComclusion,p\_OrderNumber,p\_ClientNumber,p\_Email);

end if;

end;

$$;

create or replace procedure Contract\_Update(p\_ContractNumber varchar(30), p\_DateComclusion date, p\_OrderNumber varchar(35), p\_ClientNumber varchar(10), p\_Email varchar(50), p\_newContractNumber varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Contract

where ContractNumber = p\_newContractNumber and DateComclusion = p\_DateComclusion and OrderNumber = p\_OrderNumber and ClientNumber = p\_ClientNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Contract set

DateComclusion = p\_DateComclusion,

OrderNumber = p\_OrderNumber,

ClientNumber = p\_ClientNumber,

Email = p\_Email,

ContractNumber = p\_newContractNumber

where

ContractNumber = p\_ContractNumber;

end if;

end;

$$;

create or replace procedure Contract\_Delete(p\_ContractNumber varchar(30))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where ContractNumber = p\_ContractNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Contract

where

ContractNumber = p\_ContractNumber;

--end if;

end;

$$;

create table Diagnostics

(

DiagnosticsNumber varchar(30) not null constraint PK\_DiagnosticsNumber primary key unique,

DiagnosticsDate date not null,

DiagnosticsResults text not null,

DiagnosticsDescription text not null,

CarNumber varchar(6) not null,

foreign key (CarNumber) references car (CarNumber) on update cascade on delete cascade,

ContractNumber varchar(30) not null,

foreign key (ContractNumber) references Contract (ContractNumber) on update cascade on delete cascade,

Email varchar(50) not null,

foreign key (Email) references users (Email) on update cascade on delete cascade

);

call Diagnostics\_Insert('jkdfjkfdjk', '2022.01.01', 'kjevjnrjr', 'knerjknvkj', '344343','398883312', 'diagram15@mail.ru')

select \* from diagnostics where carnumber = '344343'

create or replace procedure Diagnostics\_Insert(p\_DiagnosticsNumber varchar(30), p\_DiagnosticsDate date, p\_DiagnosticsResults text, p\_DiagnosticsDescription text, p\_CarNumber varchar(6), p\_ContractNumber varchar(30), p\_Email varchar(50))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Diagnostics

where DiagnosticsNumber = p\_DiagnosticsNumber and DiagnosticsDate = p\_DiagnosticsDate and DiagnosticsResults = p\_DiagnosticsResults and DiagnosticsDescription = p\_DiagnosticsDescription and CarNumber = p\_CarNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into Diagnostics(DiagnosticsNumber,DiagnosticsDate,DiagnosticsResults,DiagnosticsDescription,CarNumber,ContractNumber,Email)

values (p\_DiagnosticsNumber,p\_DiagnosticsDate,p\_DiagnosticsResults,p\_DiagnosticsDescription,p\_CarNumber,p\_ContractNumber,p\_Email);

end if;

end;

$$;

create or replace procedure Diagnostics\_Update(p\_DiagnosticsNumber varchar(30), p\_DiagnosticsDate date, p\_DiagnosticsResults text, p\_DiagnosticsDescription text, p\_CarNumber varchar(6), p\_ContractNumber varchar(30),p\_Email varchar(50), p\_newDiagnosticsNumber varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM Diagnostics

where DiagnosticsNumber = p\_newDiagnosticsNumber and DiagnosticsDate = p\_DiagnosticsDate and DiagnosticsResults = p\_DiagnosticsResults and DiagnosticsDescription = p\_DiagnosticsDescription and CarNumber = p\_CarNumber and Email = p\_Email ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update Diagnostics set

DiagnosticsDate = p\_DiagnosticsDate,

DiagnosticsResults = p\_DiagnosticsResults,

DiagnosticsDescription = p\_DiagnosticsDescription,

CarNumber = p\_CarNumber,

ContractNumber = p\_ContractNumber,

Email = p\_Email,

DiagnosticsNumber = p\_newDiagnosticsNumber

where

DiagnosticsNumber = p\_DiagnosticsNumber;

end if;

end;

$$;

create or replace procedure Diagnostics\_Delete(p\_DiagnosticsNumber varchar(30))

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where DiagnosticsNumber = p\_DiagnosticsNumber;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from Diagnostics

where

DiagnosticsNumber = p\_DiagnosticsNumber;

--end if;

end;

$$;

create table StatusCar

(

ID\_StatusCar SERIAL PRIMARY KEY,

StatusCar varchar(30) not null,

StatusTime timestamp not null,

DiagnosticsNumber varchar(30) not null,

foreign key (DiagnosticsNumber) references Diagnostics (DiagnosticsNumber) on update cascade on delete cascade

);

create or replace procedure StatusCar\_Insert(p\_StatusCar varchar(30), p\_StatusTime timestamp, p\_DiagnosticsNumber varchar(30))

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM StatusCar

where StatusCar = p\_StatusCar and StatusTime = p\_StatusTime and DiagnosticsNumber = p\_DiagnosticsNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

insert into StatusCar(StatusCar,StatusTime,DiagnosticsNumber)

values (p\_StatusCar,p\_StatusTime,p\_DiagnosticsNumber);

end if;

end;

$$;

create or replace procedure StatusCar\_Update(p\_ID\_StatusCar int, p\_StatusCar varchar(30), p\_StatusTime timestamp, p\_DiagnosticsNumber varchar(30) )

language plpgsql

as $$

DECLARE have\_record int := count (\*) FROM StatusCar

where StatusCar = p\_StatusCar and StatusTime = p\_StatusTime and DiagnosticsNumber = p\_DiagnosticsNumber ;

begin

if have\_record>0 then

raise exception 'Уже существует';

else

update StatusCar set

StatusCar = p\_StatusCar,

StatusTime = p\_StatusTime,

DiagnosticsNumber = p\_DiagnosticsNumber

where

ID\_StatusCar = p\_ID\_StatusCar;

end if;

end;

$$;

create or replace procedure StatusCar\_Delete(p\_ID\_StatusCar int)

language plpgsql

as $$

--DECLARE have\_record int := count (\*) FROM Таблица где используется данная таблица

--where ID\_StatusCar = p\_ID\_StatusCar;

begin

--if have\_record>0 then raise exception 'Данные нельзя удалить, так как они уже задействованы';

--else

delete from StatusCar

where

ID\_StatusCar = p\_ID\_StatusCar;

--end if;

end;

$$;