

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування та спеціалізованих комп’ютерних систем

**Лабораторна робота № 1**

з дисципліни

**«Бази даних та засоби управління»**

*Тема: «***Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL***»*

Виконав: студент ІІІ курсу

ФПМ групи КВ-93

Зеленяк І.О.

Перевірив:

Київ, 2021

## Розробка ER-моделі роботи парковки

### Опис обраної предметної галузі

В даній лабораторній буде розглядатися принцип роботи парковки з розділеними персональними місцями для кожної автівки. Для того, щоб особа користувалася парковкою необхідно підписати договір на окреме паркувальне місце.

Вважаємо що в одного користувача може бути доступ до декількох автівок (декілька договорів).

Одному паркомісцю відповідає тільки одна машина яка може там стояти (за договором), якщо воно орендоване, інакше жодна машина не може там стояти.

В результаті розробки бази даних буде можливо реалізувати контроль роботи парковки.

### Перелік сутностей з описом їх призначення

**Автівка** фізичний автомобіль який ідентифікується за державним номером та перепусткою та має атрибути:

* ***Державний номер*** - номер автівки виданий державним органом
* ***Тип*** - тип автомобіля
* ***Виробник*** - виробник автомобіля

Власник авто особа яка є власником авто:

* **Прізвище , Ім’я , По-батькові**
* **Номер телефону** - список номерів телефону

**Паркомісце -** фізичне місце для парковки автомобіля в межах паркувального майданчика тамає атрибути:

* ***Номер*** - унікальний номер паркомісця який не повторюється

### Концептуальна модель «сутність-зв’язок»



[Посилання на діаграму](https://lucid.app/lucidchart/1369ec07-2d40-43f5-9ef6-ad393ed7d69a/edit?invitationId=inv_4983367c-0fe9-400b-8fe2-440fcdc16171)

Обрана система нотацій Пітера Чена

## Логічна модель даних

Концептуальна ER-модель описує взаємодію сутностей на самому високому рівні та не відображає всю інформацію, що необхідна для збереження даних про сутності та не презентує як саме буде зберігатись інформація. Тому необхідно виконати перетворення концептуальної   
ER-моделі у логічну модель.

Під час такого перетворення отримаємо наступні сутності та зв’язки:

**Автівка -** фізичний автомобіль який ідентифікується за державним номером та перепусткою та має атрибути:

* ***Державний номер*** - номер автівки виданий державним органом
* ***Тип*** - ідентифікатор типа автомобіля
* ***Виробник*** – ідентифікатор виробника автомобіля

Під час перетворення концептуальної ER-моделі у логічну, атрибути автівки ***Тип*** та ***Виробник*** утворили відповідні сутності:

**Тип –** сутність яка відображає інформацію про тип автівки такі як легковий, вантажний тощо, та має атрибути:

* ***Ідентифікатор типу***
* ***Назва типу***

**Виробник** – сутність, що містить назву виробника автівки, та має атрибути:

* ***Ідентифікатор марки авто***
* ***Назва марки***
* ***Опис***

**Власник авто** особа яка є власником авто:

* ***Код*** – ідентифікаційний код особи
* ***Прізвище***
* ***Ім’я***
* ***По-батькові***

Під час перетворення концептуальної ER-моделі у логічну (поліпшену ER-модель), атрибут сутності **Власник авто** ***Номер телефону*** утворює відповідну сутність **Номер телефону**, а атрибут **Прізвище , Ім’я , По-батькові** був розділений на окремі атрибути.

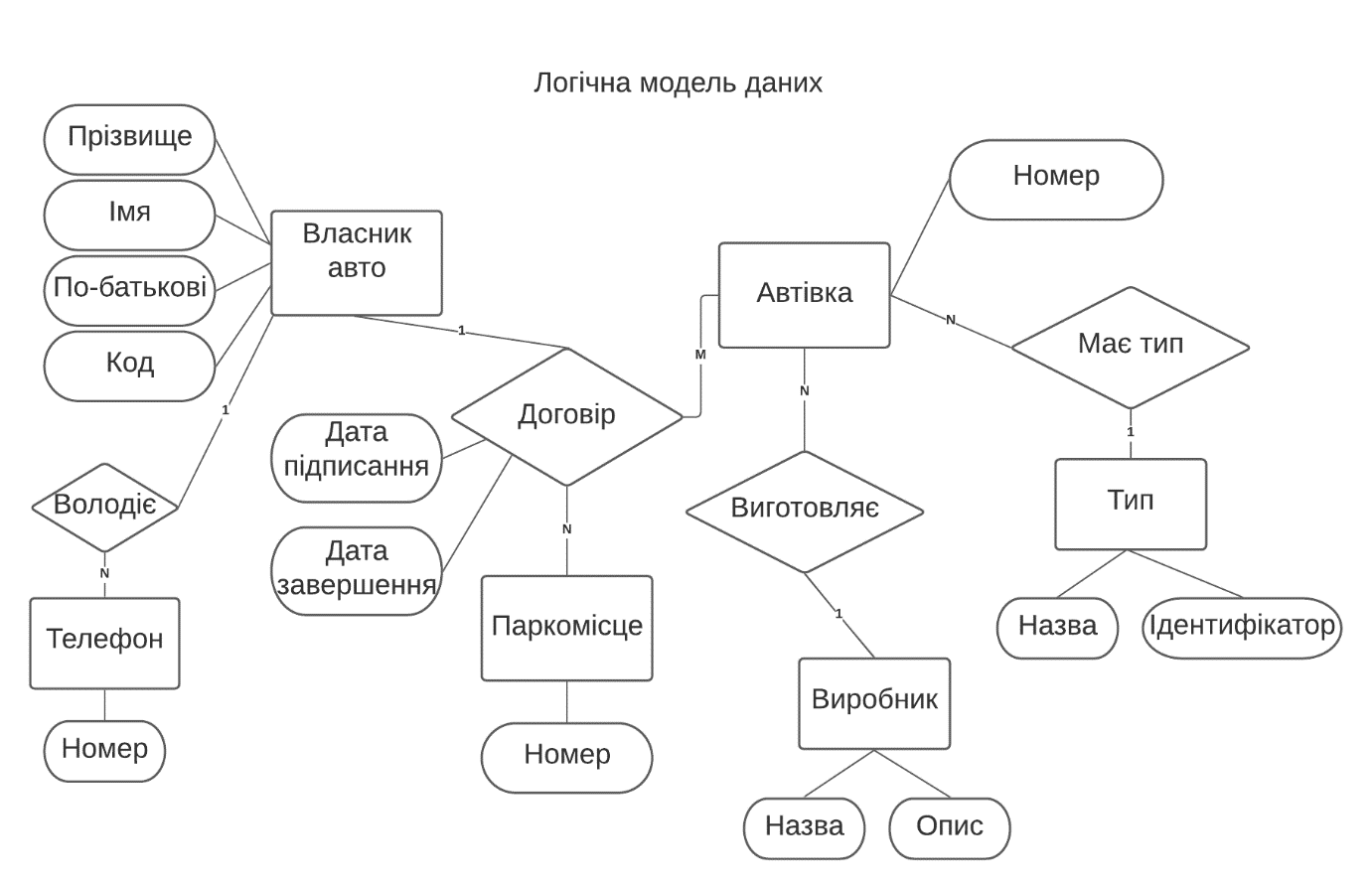
**Номер телефону** - номер телефону, що має атрибути:

* ***Номер телефону***
* Ідентифікаційний код **Власника авто**

**Паркомісце -** фізичне місце для парковки автомобіля в межах паркувального майданчика тамає атрибути:

* ***Номер*** - унікальний номер паркомісця який не повторюється

Таким чином Логічна модель даних може бути подана у наступному вигляді:



[Посилання на діаграму](https://lucid.app/lucidchart/780aa54b-80a6-41ad-ae9f-375ca9f053aa/edit?viewport_loc=-1048%2C-132%2C4170%2C1479%2C0_0&invitationId=inv_8600e318-71bb-430b-8d39-b639c771b42f)

## Перетворення Логічної моделі даних в таблиці бази данних

Сутність **Власник** авто було перетворено в таблицю ***Persons.***

Сутність **Телефон** було перетворено таблицю ***Phones*** яка утворює із таблицею ***Persons*** зв’язок 1:N

Сутність **Автівка** було перетворено в таблицю ***Cars****.*

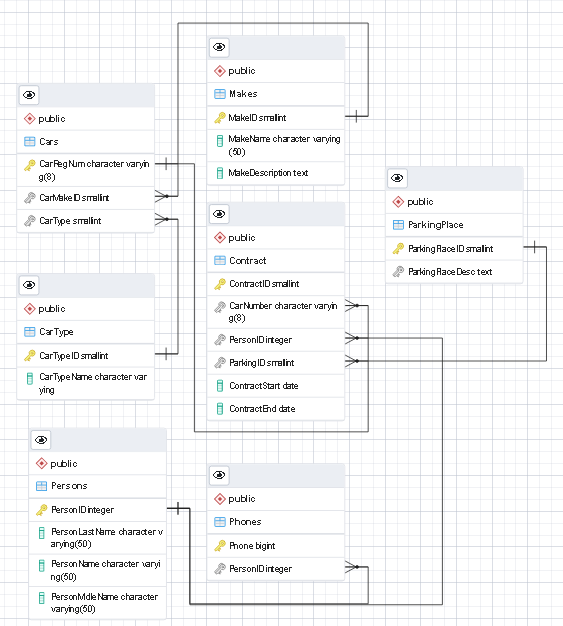
Сутність **Виробник** було перетворено в таблицю ***Makes*** яка утворює із таблицею ***Cars*** зв’язок 1:N.

Сутність **Тип** було перетворено в таблицю ***CarType*** яка утворює із таблицею ***Cars*** зв’язок 1:N.

Сутність **Паркомісце** було перетворено у таблицю ***ParkingPlace***

Сутність-зв'язок **Договір** було перетворено в таблицю ***Contracts*** яка містить зовнішні ключі на таблиці ***Cars*** поле ***CarRegNum, Persons*** поле ***PersonID, ParkingPlace*** поле ***ParkingPlaceID.***

Сутність-зв'язок ***Договір***  реалізує зв’язки M:N, 1:М та 1:1. Тобто у 1-го Власника багато Договорів (1:М), а у 1-го Авто і 1-го Місця 1-н Договір (1:1). В той самий час історично між багатьма авто та пакувальними місцями з часом реалізований зв’язок M:N



### Нормалізація таблиць баз данних

При перетворенні логічної моделі у таблиці баз даних, схема бази даних знаходиться в третій нормальній формі. А саме:

Схема відповідає 1НФ, тому що:

* В таблицях немає дубльованих рядків.
* В кожній комірці зберігається атомарне (скалярне) значення.
* В кожному стовпці зберігаються дані одного типу.

Схема відповідає 2НФ, тому що:

* Вона відповідає 1НФ.
* Має первинний ключ, а всі не ключові стовпці таблиці залежать від первинного ключа.

Схема відповідає 3НФ, тому що:

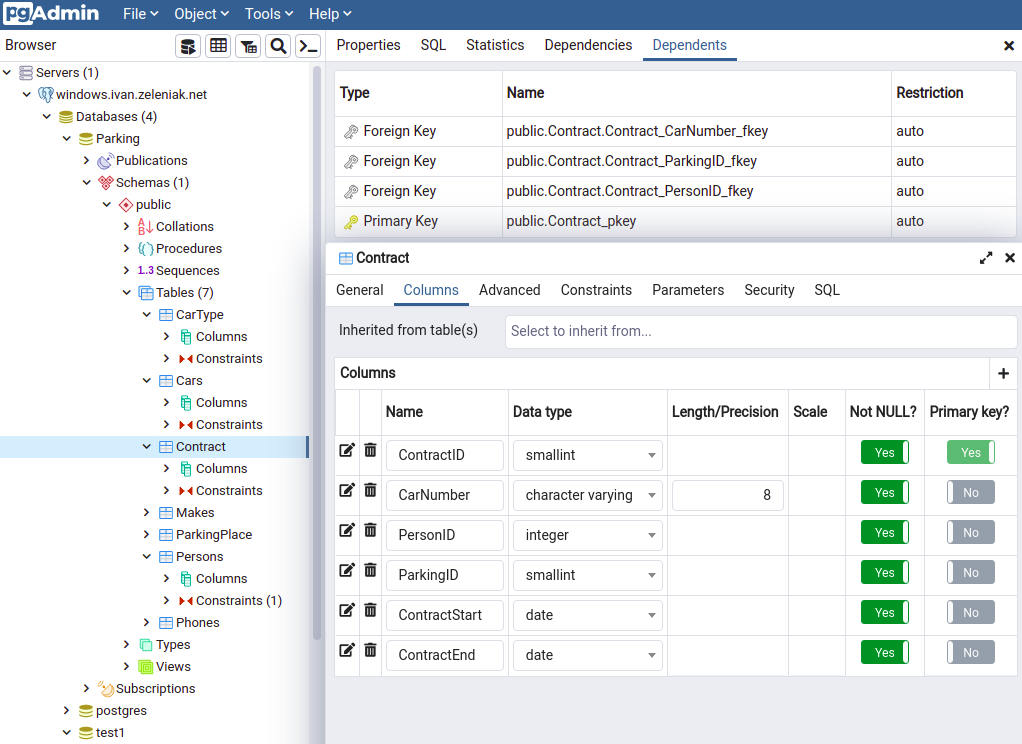
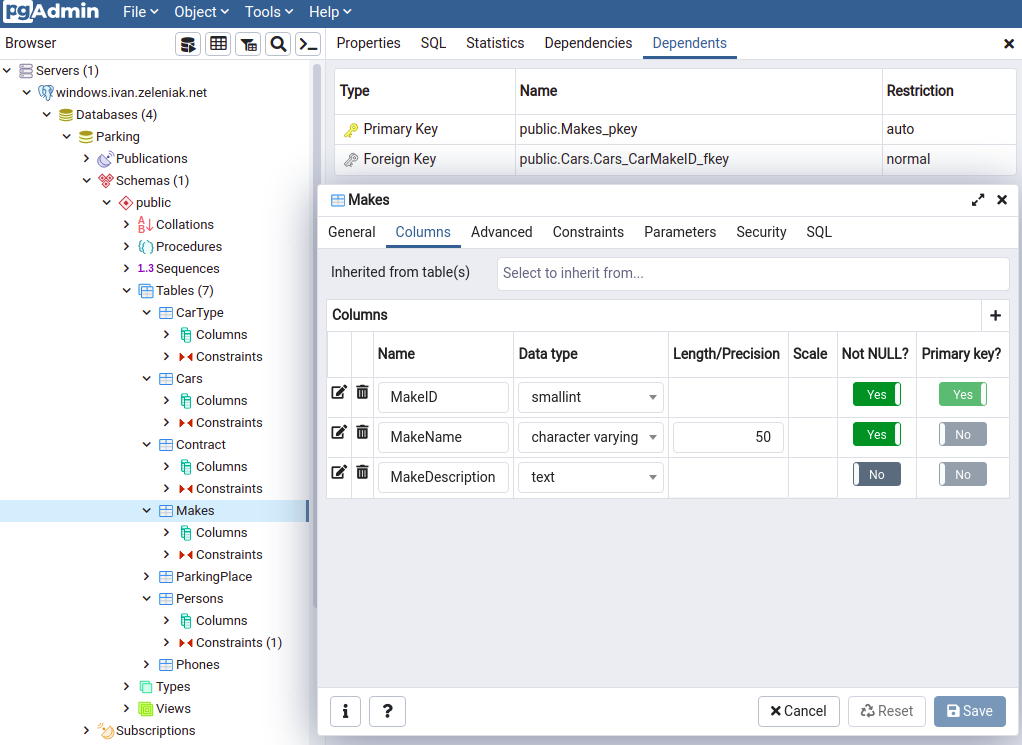
* Вона відповідає 2НФ.
* Всі не ключові атрибути таблиці залежать винятково від усього первинного ключа, а не його частини. Тобто кожен неключових атрибут нетранзитивно (без посередника) залежить від первинного ключа.

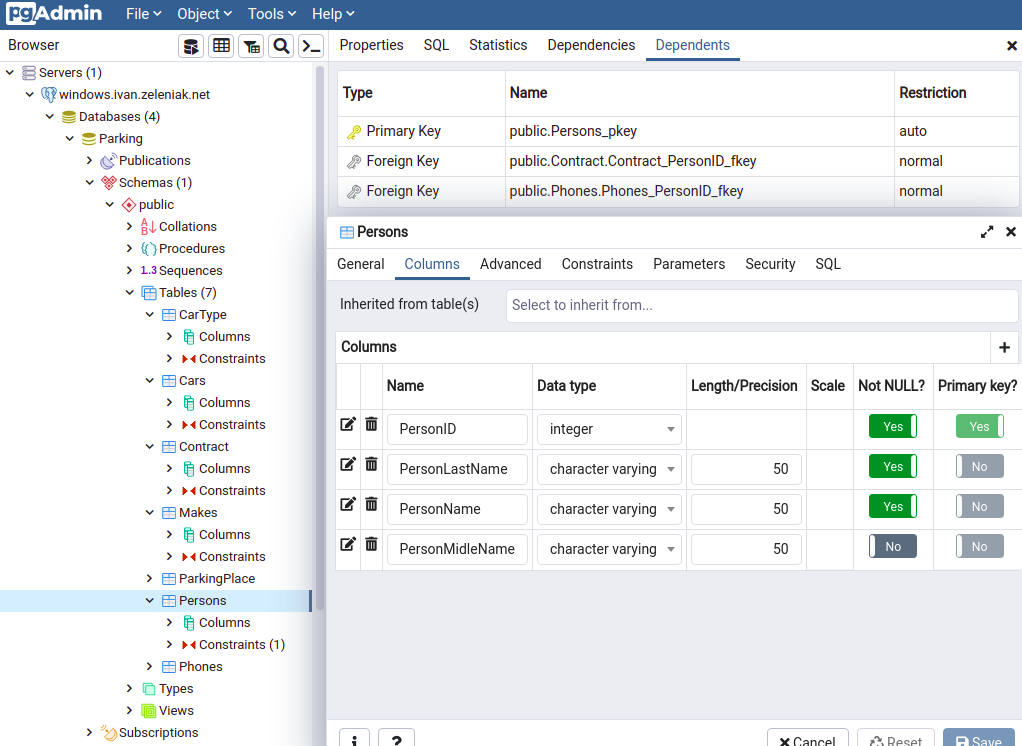
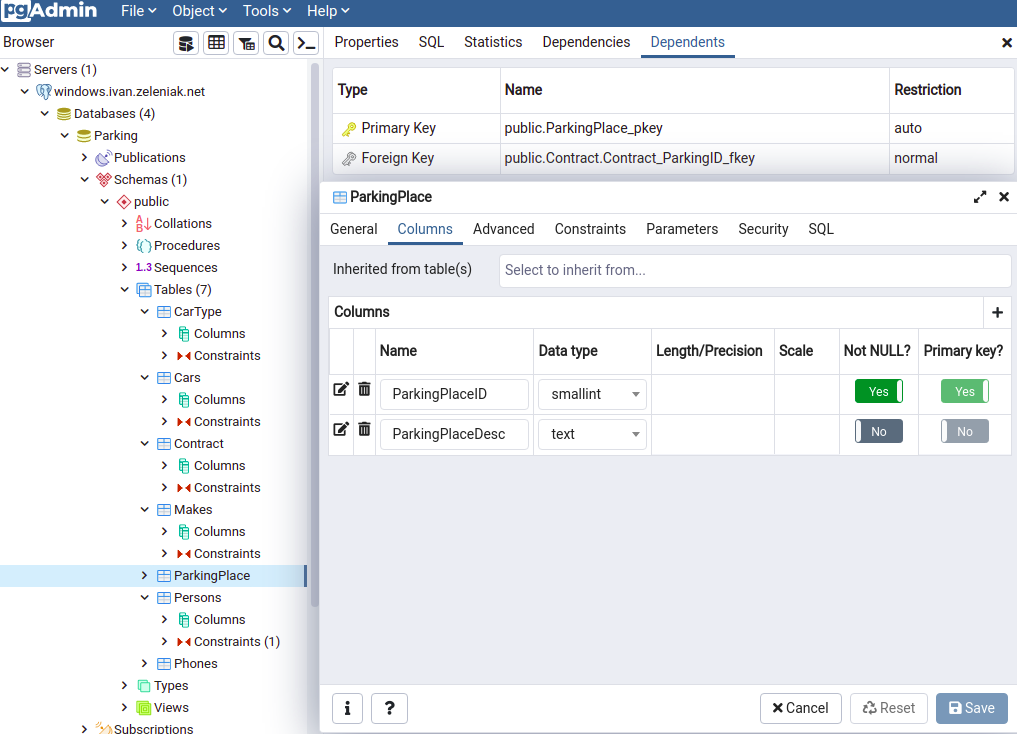
### Опис стовбців їх данних типів

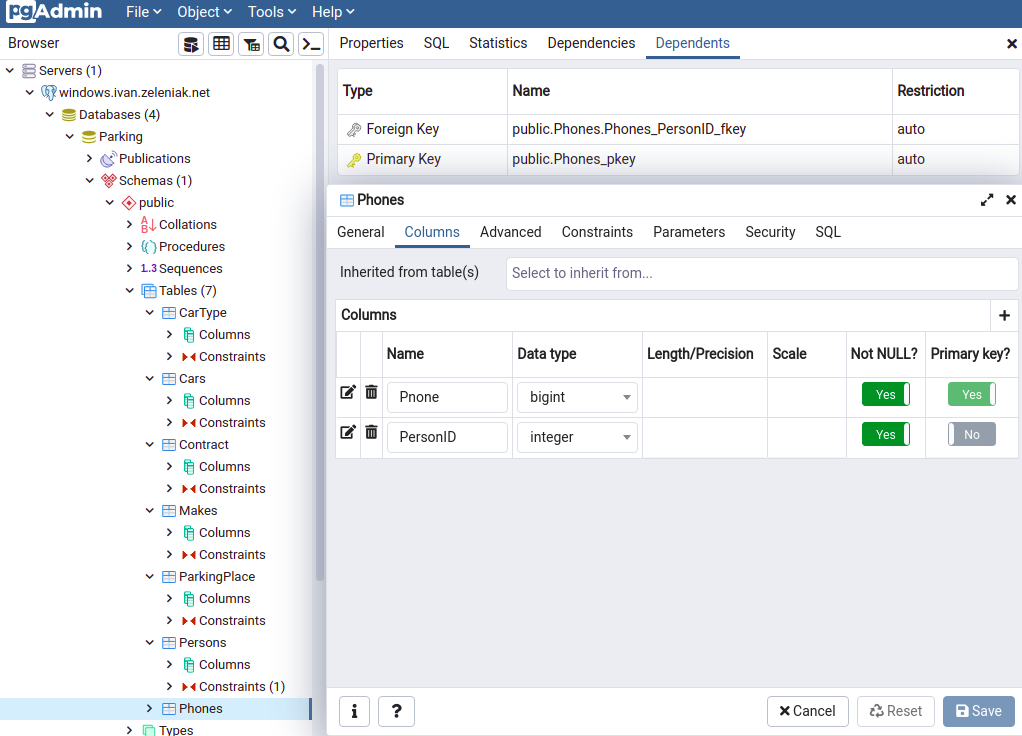
|  |  |  |  |
| --- | --- | --- | --- |
| **Назва таблиці/поля** | **тип**  **даних** | **обов’язково** | **опис** |
| **Таблиця "Cars"** |  |  |  |
| *"CarRegNum"* | text | Так | Номер автівки |
| *"CarMakeID"* | smallint | Так | Ідентифікатор виробника автівки |
|  |  |  |  |
| **Таблиця "Makes"** |  |  |  |
| *"MakeID"* | smallint | Так | ідентифікатор марки авто |
| *"MakeName"* | text | Так | назва марки |
| *"MakeDescription"* | text | Ні | опис |
|  |  |  |  |
| **Таблиця "ParkingPlace"** |  |  |  |
| *"ParkingPlaceID"* | smallint | Так | ідентифікатор парковки |
| *"ParkingPlaceDesc"* | text | Ні | опис |
|  |  |  |  |
| **Таблиця "Persons"** |  |  |  |
| *"PersonID"* | bigint | Так | ідентифікатор персони |
| *"PersonLastName"* | text | Так | фамілія персони |
| *"PersonName"* | text | Так | ім’я персони |
| *"PersonMidleName"* | text | Ні | по батькові персони |
|  |  |  |  |
| **Таблиця "Phones"** |  |  |  |
| *"Pnone"* | bigint | Так | номер телефону |
| *"PersonID"* | bigint | Так | ідентифікатор персони |
|  |  |  |  |
| **Таблиця "CarType"** |  |  |  |
| *"CarTypeID"* | smallint | Так | ідентифікатор типу |
| *"CarTypeName"* | text | Так | назва типу |
|  |  |  |  |
| **Таблиця "Contract"** |  |  |  |
| *"ContractID"* | smallint | Так | номер договору |
| *"CarNumber"* | text | Так | Номер автівки |
| *"PersonID"* | bigint | Так | ідентифікатор персони |
| "ParkingID" | smallint | Так | ідентифікатор парковки |
| "ContractStart" | date | Так | Дата підписання договору |
| *"*ContractEnd*"* | date | Так | Дата завершення договору |

## Знімки екрану зі стовпцями та залежностями

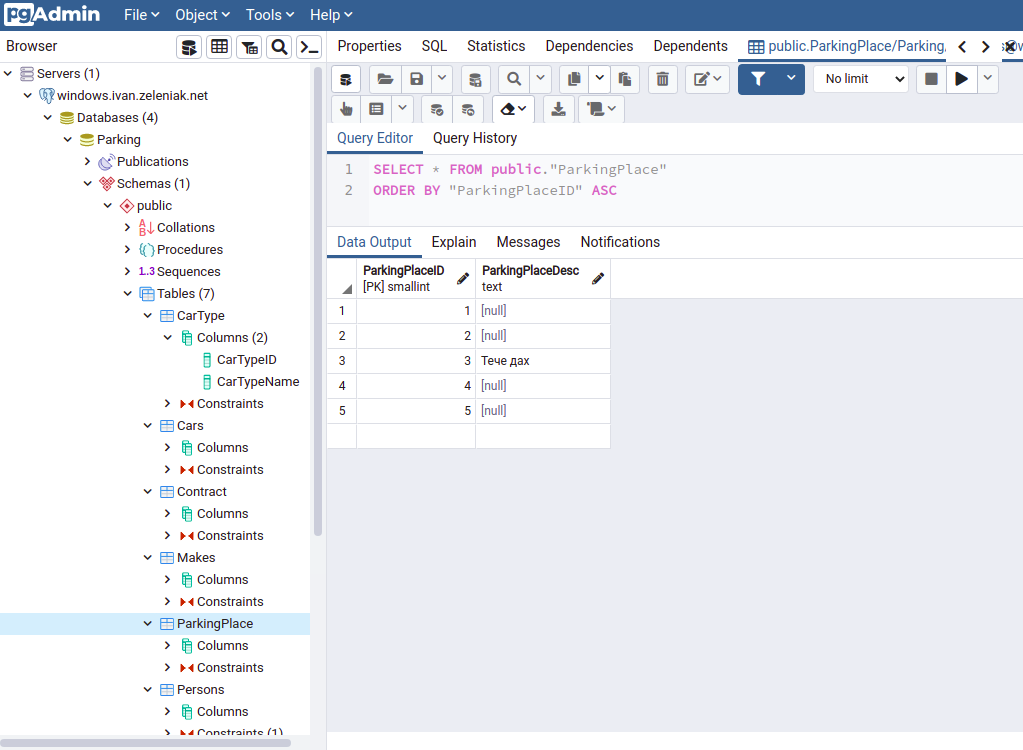
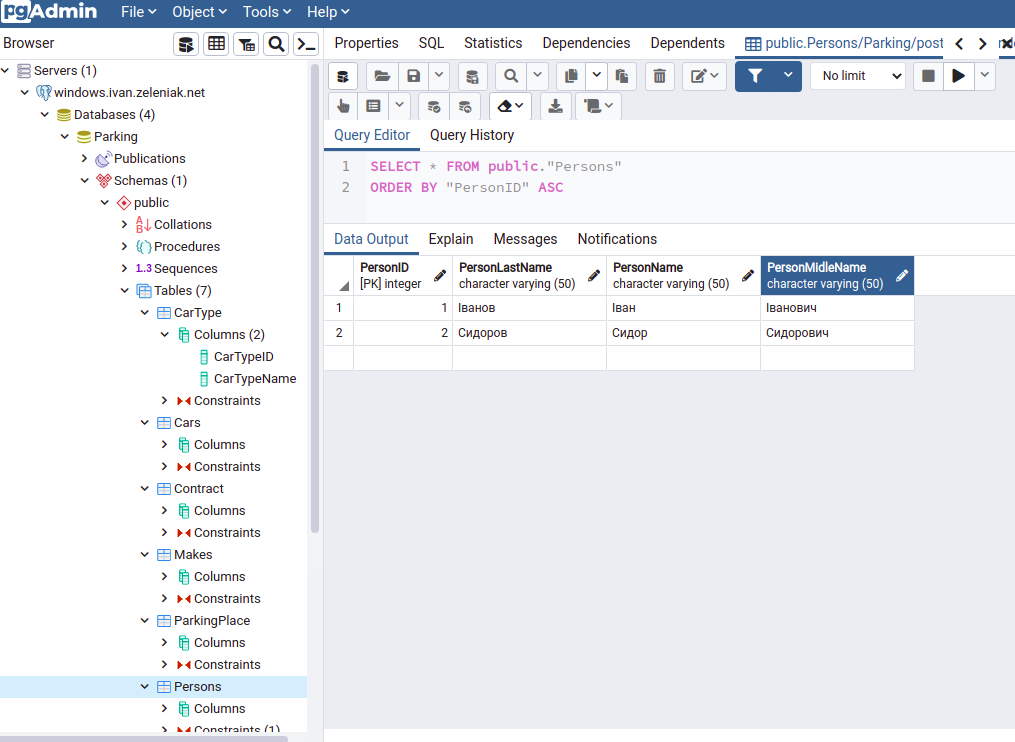
## 

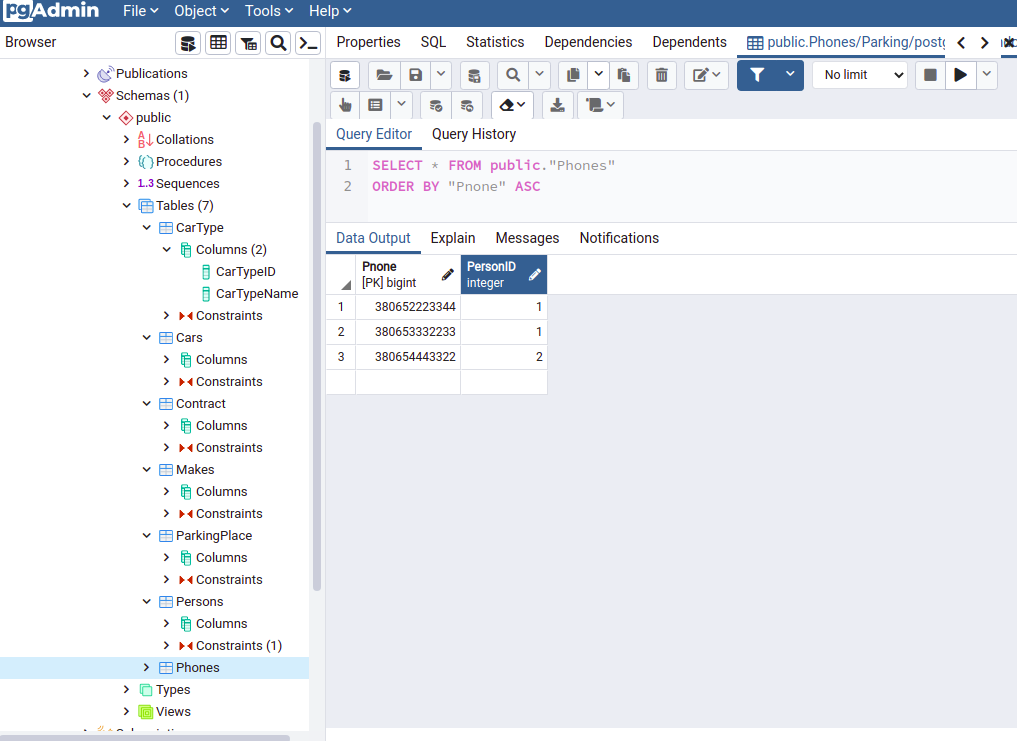
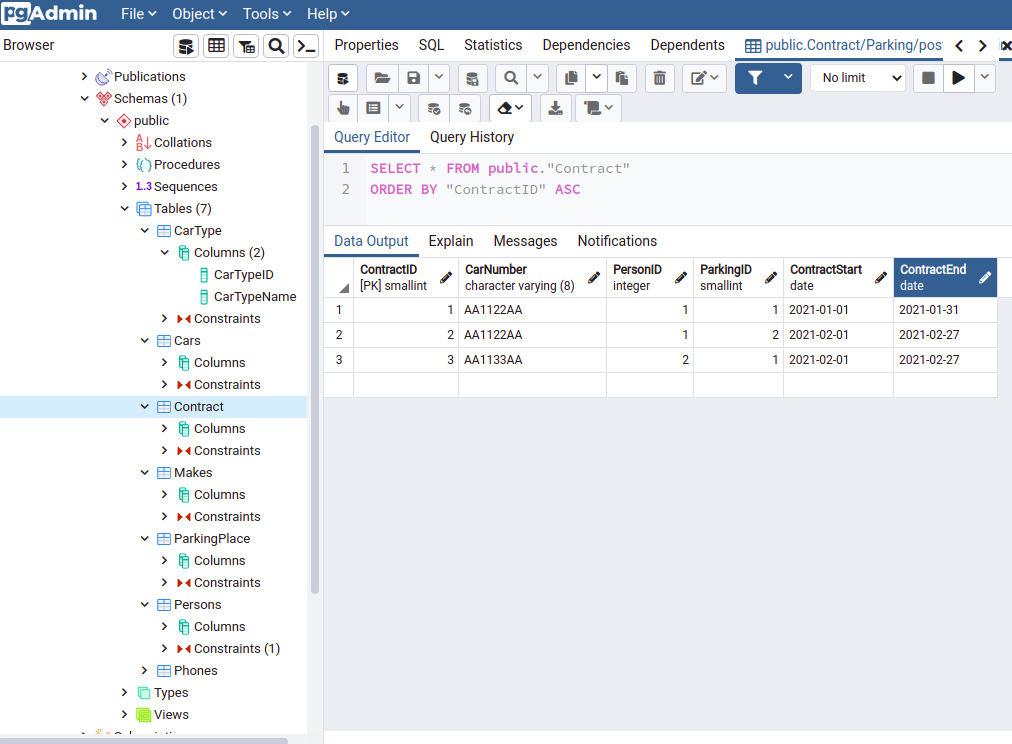


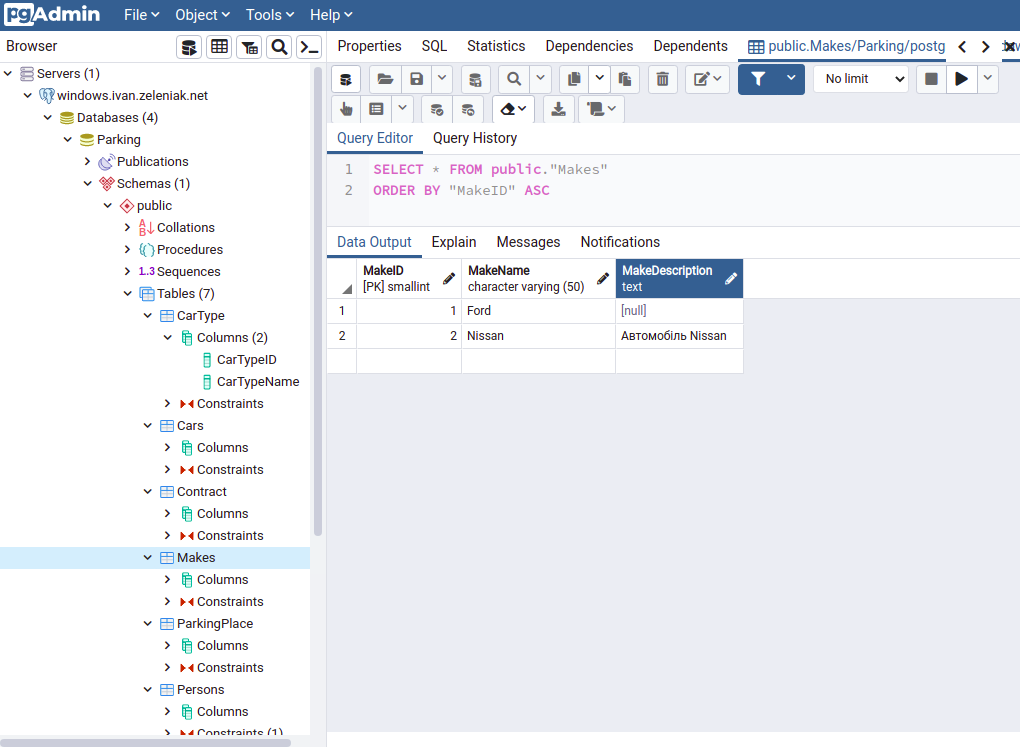




## Знімки екрану з даними з таблиць







# Додаток 1 скрипт sql

-- This script was generated by a beta version of the ERD tool in pgAdmin 4.

-- Please log an issue at https://redmine.postgresql.org/projects/pgadmin4/issues/new if you find any bugs, including reproduction steps.

BEGIN;

CREATE TABLE IF NOT EXISTS public."CarType"

(

"CarTypeID" smallint NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 32767 CACHE 1 ),

"CarTypeName" character varying NOT NULL,

PRIMARY KEY ("CarTypeID")

);

CREATE TABLE IF NOT EXISTS public."Cars"

(

"CarRegNum" character varying(8) NOT NULL,

"CarMakeID" smallint NOT NULL,

"CarType" smallint,

PRIMARY KEY ("CarRegNum")

);

CREATE TABLE IF NOT EXISTS public."Contract"

(

"ContractID" smallint NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 32767 CACHE 1 ),

"CarNumber" character varying(8) NOT NULL,

"PersonID" bigint NOT NULL,

"ParkingID" smallint NOT NULL,

"ContractStart" date NOT NULL,

"ContractEnd" date NOT NULL,

PRIMARY KEY ("ContractID")

);

CREATE TABLE IF NOT EXISTS public."Makes"

(

"MakeID" smallint NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 32767 CACHE 1 ),

"MakeName" character varying(50) NOT NULL,

"MakeDescription" text,

PRIMARY KEY ("MakeID")

);

CREATE TABLE IF NOT EXISTS public."ParkingPlace"

(

"ParkingPlaceID" smallint NOT NULL GENERATED BY DEFAULT AS IDENTITY ( INCREMENT 1 START 1 MINVALUE 1 MAXVALUE 50 CACHE 1 ),

"ParkingPlaceDesc" text,

PRIMARY KEY ("ParkingPlaceID")

);

CREATE TABLE IF NOT EXISTS public."Persons"

(

"PersonID" bigint NOT NULL,

"PersonLastName" character varying(50) NOT NULL,

"PersonName" character varying(50) NOT NULL,

"PersonMidleName" character varying(50),

PRIMARY KEY ("PersonID")

);

CREATE TABLE IF NOT EXISTS public."Phones"

(

"Pnone" bigint NOT NULL,

"PersonID" bigint NOT NULL,

PRIMARY KEY ("Pnone")

);

ALTER TABLE IF EXISTS public."Cars"

ADD FOREIGN KEY ("CarMakeID")

REFERENCES public."Makes" ("MakeID")

NOT VALID;

ALTER TABLE IF EXISTS public."Cars"

ADD FOREIGN KEY ("CarType")

REFERENCES public."CarType" ("CarTypeID")

NOT VALID;

ALTER TABLE IF EXISTS public."Contract"

ADD FOREIGN KEY ("CarNumber")

REFERENCES public."Cars" ("CarRegNum")

NOT VALID;

ALTER TABLE IF EXISTS public."Contract"

ADD FOREIGN KEY ("ParkingID")

REFERENCES public."ParkingPlace" ("ParkingPlaceID")

NOT VALID;

ALTER TABLE IF EXISTS public."Contract"

ADD FOREIGN KEY ("PersonID")

REFERENCES public."Persons" ("PersonID")

NOT VALID;

ALTER TABLE IF EXISTS public."Phones"

ADD FOREIGN KEY ("PersonID")

REFERENCES public."Persons" ("PersonID")

NOT VALID;

END;

Файли що стосуються цієї лабораторної роботи завантажені в репозиторій git за адресою <https://github.com/AmiriVANYCH/DataBase/tree/main/Lab1>