**Міністерство освіти та науки України**

**Державний вищий навчальний заклад “Київський національний економічний університет ім. Вадима Гетьмана”**

**Кафедра інформаційних систем в економіці**

**Дисципліна “Адміністрування та програмування баз даних в штучному інтелекті”**

**ЗВІТ**

**з лабораторної роботи №4**

**«SQL Essential»**

**Підготував:**

студент 5 курсу гр. ІШІ-501

спеціальності «8і09»

Донєв Данило Романович

**Прийняла:**

доц. Добролюбова М.В.

Київ-2023

***ВІДПОВІДІ НА ПИТАННЯ***

# Що таке база даних та СУБД?

**База даних** (сукупність зв’язаних даних) організована за певними правилами та незалежна від прикладних програм. Правила організації передбачають загальні принципи опису, збереження та маніпулювання даними.

**Система управління базами даних** – система програмного забезпечення, яка дозволяє обробляти звернення до бази даних, що надходять від прикладних програм.

# Що таке нормалізація? Які нормальні форми та їх обмеження вам відомі?

Нормалізація в базах даних визначає набір методів, які використовуються для створення та контролю за структурою БД. Це процес перетворення складних даних в більш прості або зрозумілі структури даних.

**Основні нормальні форми:**

* Перша нормальна форма (1NF): Забезпечує, що кожне поле містить лише атомарні (неділимі) значення. Кожне сутність (таблиця) має унікальний ключ.
* Друга нормальна форма (2NF): Усі вимоги першої нормальної форми виконані, а також кожен не-ключовий атрибут повинен повністю залежати від основного ключа.
* Третя нормальна форма (3NF): Усі вимоги 2NF виконані, і всі стовпець, що не є ключовим, взаємно незалежні (тобто їх значення не залежать від інших не ключових стовпців).
* Нормальна форма Бойса-Кодда (BCNF): Усі вимоги 3NF виконані, і для будь-якої залежності X -> Y, X є суперключем.
* Четверта нормальна форма (4NF): Основна мета – видалити багатозначні залежності.
* П'ята нормальна форма (5NF або PJ/NF – від Project Join Normal Form): Забезпечує розподіл великих таблиць на менші, відокремлює структури даних і відновлює їх без втрат.
* Шоста нормальна форма в основному використовується для оптимізації продуктивності баз даних, які зберігають або впроваджують дати й час. Це досягається шляхом видалення часових дублікатів, а також впровадженням більш ефективної структури для зберігання часових записів.

**Обмеження:**

* Перша нормальна форма (1NF):

Кожне поле містить лише примітивні атомарні типи;

Заборона на повторювані поля або поля, які містять кілька значень;

Відсутнє забезпечення унікальності запису.

* Друга нормальна форма (2NF):

Не враховується структура даних, що може містити додаткові залежності від ключа;

Може виникнути необхідність перерозподілу даних між таблицями.

* Третя нормальна форма (3NF):

Часом може призвести до надлишкової або непотрібної розбитості даних.

* Четверта нормальна форма (4NF):

Потенційні недоліки з реалізацією змін в структурі даних;

Може призвести до єще більшої розбитості даних.

* П’ята нормальна форма (5NF):

Додаткова розбитість та складність в управлінні таблицями.

* Шоста нормальна форма (6NF):

Більш складне управління структурою за рахунок додаткової складності схеми.

# Які види зовнішніх з’єднань вам відомі?

LEFT OUTER JOIN (ліве зовнішнє об’єднання) – зовнішнє об’єднання, при якому в запиті всі записи з таблиці на лівому боці операції LEFT JOIN в інструкції SQL додаються в результуючий набір записів, навіть якщо в таблиці на правому боці відсутні співпадаючі значення в зв’язаних полях.

RIGHT OUTER JOIN (праве зовнішнє об’єднання) – зовнішнє об’єднання, при якому в запиті всі записи з таблиці на правому боці операції RIGHT JOIN в інструкції SQL додаються в результуючий набір записів, навіть якщо в таблиці на лівому боці відсутні співпадаючі значення в зв’язаних полях.

FULL OUTER JOIN (повне об’єднання) – зовнішнє об’єднання, при якому в запиті всі записи з таблиці на лівому та правому боці операції FULL JOIN додаються в результуючий набір записів, при відповідності умові значень у зв’язаних полях, а також:

1) значення з правої таблиці, що не мають відповідностей в лівій таблиці;

2) значення з лівої таблиці, що не мають відповідностей в правій таблиці.

# Що таке транзакція та відкат транзакції?

**Транзакція** – це група послідовно виконуємих операторів SQL, які або повинні бути виконані всі, або не повинен бути виконаний жоден з них.

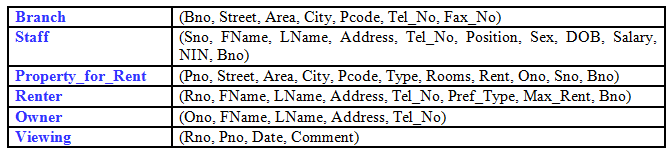
**Відкат тразакції** – дія, що забезпечує анулювання всіх змін даних, які були зроблені в тілі поточної незавершеної транзакції.

ПОСИЛАННЯ НА GITHUB

<https://github.com/doniev-danylo/kneu_cs/tree/main/lab4>

***Завдання 1***

Створіть базу даних з ім’ям «DreamHomeDB». У створеній БД DreamHomeDB створіть 5 таблиць: Staff, Property\_for\_Rent, Renter, Qwner, Viewing.



У створеній БД DreamHomeDB створити вибірку за допомогою JOIN’s для наступних завдань:

1) Скласти список імен всіх клієнтів, які вже оглянули хоча б один об'єкт, що здається в оренду, і повідомили свою думку з цього приводу.

2) Перелічити відділення компанії і об'єкти, що здаються в оренду, які розташовані в одному і тому ж місті, а також інші відділення компанії.

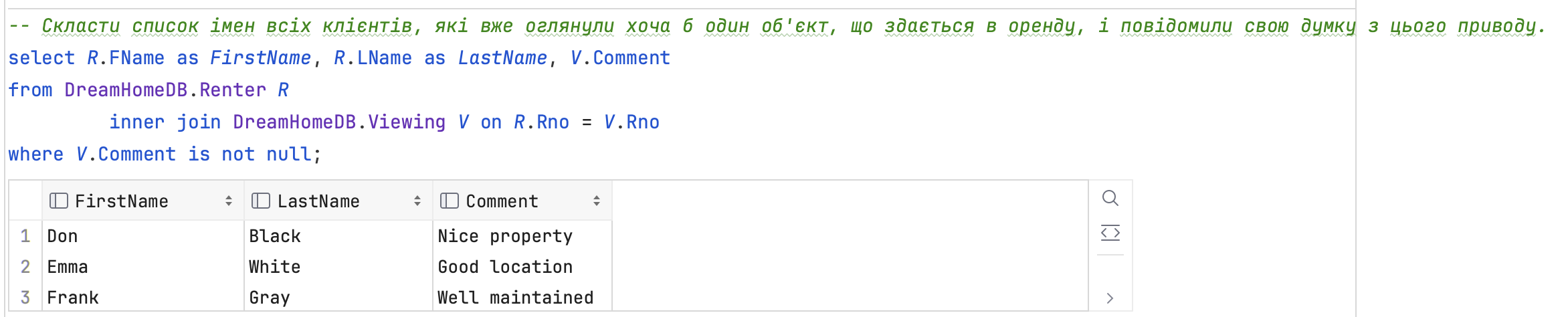
3) Перелічити відділення компанії і об'єкти, що здаються в оренду, які розташовані в одному і тому ж місті, а також всі інші об'єкти.

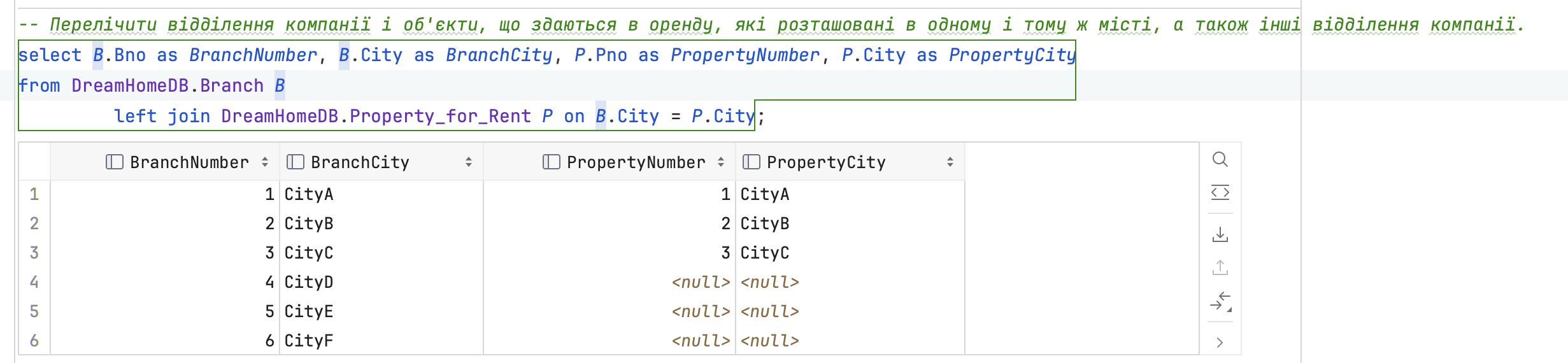
4) Перелічити відділення компанії і об'єкти, що здаються в оренду, розташовані в одному і тому ж місті, а також всі інші відділення і об'єкти.

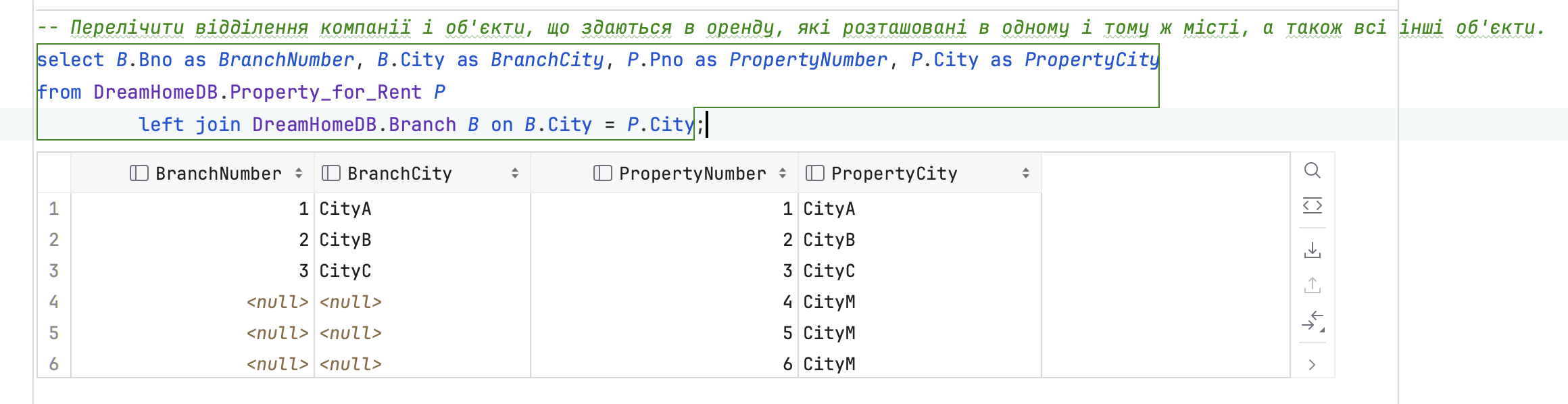
КОД ПРОГРАМИ

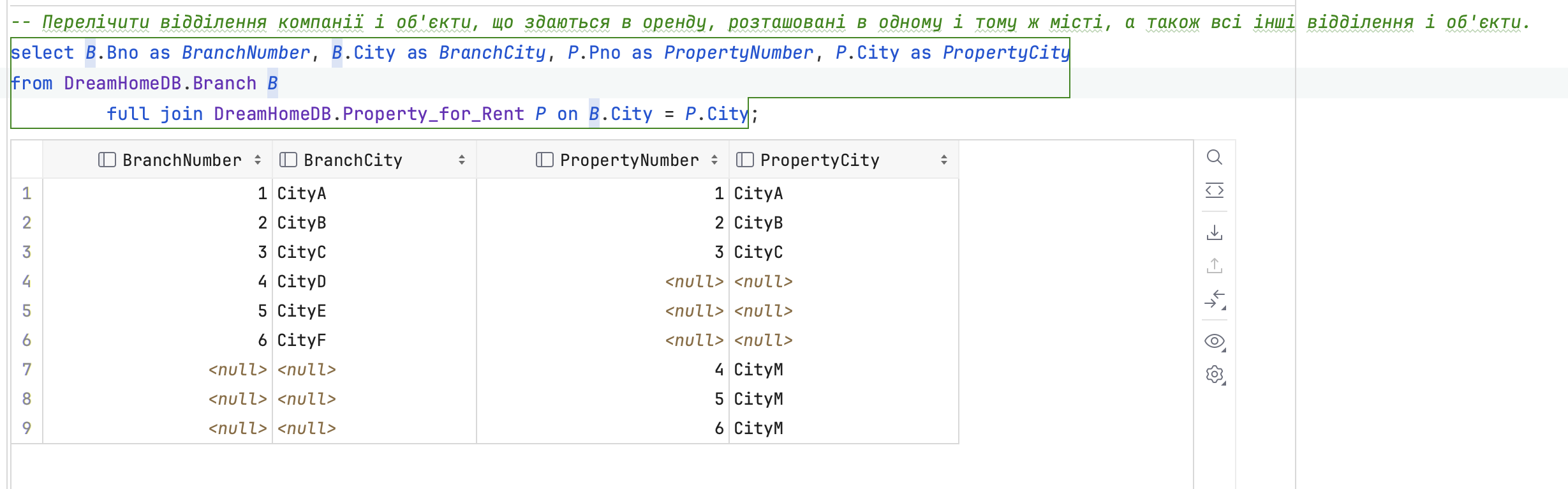
*-- create schema*create schema DreamHomeDB  
go  
  
exec sp\_addextendedproperty 'MS\_Description', 'lab4\_apbd\_kneu', 'SCHEMA', 'DreamHomeDB'  
go  
;  
  
*-- create tables*create table DreamHomeDB.Branch  
(  
 Bno **int**,  
 Street **varchar**(32),  
 Area **varchar**(32),  
 City **varchar**(32),  
 Pcode **varchar**(32),  
 Tel\_No **varchar**(32),  
 Fax\_No **varchar**(32),  
)  
go  
  
create table DreamHomeDB.Staff  
(  
 Sno **int**,  
 FName **nvarchar**(50),  
 LName **nvarchar**(50),  
 Address **nvarchar**(100),  
 Tel\_No **nvarchar**(15),  
 Position **nvarchar**(50),  
 Sex **char**(1),  
 DOB **date**,  
 Salary **decimal**(10, 2),  
 NIN **nvarchar**(15),  
 Bno **int**)  
go  
create table DreamHomeDB.Property\_for\_Rent  
(  
 Pno **int**,  
 Street **nvarchar**(100),  
 Area **nvarchar**(50),  
 City **nvarchar**(50),  
 Pcode **nvarchar**(10),  
 Type **nvarchar**(50),  
 Rooms **int**,  
 Rent **decimal**(10, 2),  
 Ono **int**,  
 Sno **int**,  
 Bno **int**)  
go  
create table DreamHomeDB.Renter  
(  
 Rno **int**,  
 FName **nvarchar**(50),  
 LName **nvarchar**(50),  
 Address **nvarchar**(100),  
 Tel\_No **nvarchar**(15),  
 Pref\_Type **nvarchar**(50),  
 Max\_Rent **decimal**(10, 2),  
 Bno **int**)  
go  
create table DreamHomeDB.Owner  
(  
 Ono **int**,  
 FName **nvarchar**(50),  
 LName **nvarchar**(50),  
 Address **nvarchar**(100),  
 Tel\_No **nvarchar**(15)  
)  
go  
create table DreamHomeDB.Viewing  
(  
 Rno **int**,  
 Pno **int**,  
 [Date] **date**,  
 Comment **nvarchar**(256)  
)  
go  
;  
*-- set primary keys*alter table DreamHomeDB.Branch  
 alter column Bno **int** not null  
go  
  
alter table DreamHomeDB.Branch  
 add constraint Branch\_pk  
 primary key (Bno)  
go  
  
alter table DreamHomeDB.Owner  
 alter column Ono **int** not null  
go  
  
alter table DreamHomeDB.Owner  
 add constraint Owner\_pk  
 primary key (Ono)  
go  
  
alter table DreamHomeDB.Property\_for\_Rent  
 alter column Pno **int** not null  
go  
  
alter table DreamHomeDB.Property\_for\_Rent  
 add constraint Property\_for\_Rent\_pk  
 primary key (Pno)  
go  
  
alter table DreamHomeDB.Renter  
 alter column Rno **int** not null  
go  
  
alter table DreamHomeDB.Renter  
 add constraint Renter\_pk  
 primary key (Rno)  
go  
  
alter table DreamHomeDB.Staff  
 alter column Sno **int** not null  
go  
  
alter table DreamHomeDB.Staff  
 add constraint Staff\_pk  
 primary key (Sno)  
go  
;  
*-- insert test values   
-- Inserting data into DreamHomeDB.Branch table*insert into DreamHomeDB.Branch(Bno, Street, Area, City, Pcode, Tel\_No, Fax\_No)  
values (1, '123 St', '#01', 'CityA', 'PC1', '1234567890', '0987654321'),  
 (2, '456 St', '#02', 'CityB', 'PC2', '2345678901', '1098765432'),  
 (3, '789 St', '#03', 'CityC', 'PC3', '3456789012', '2109876543')  
  
  
*-- Inserting data into DreamHomeDB.Staff table*insert into DreamHomeDB.Staff(Sno, FName, LName, Address, Tel\_No, Position, Sex, DOB, Salary, NIN, Bno)  
values (1, 'Alice', 'Brown', 'Street #1', '1235486789', 'Manager', 'F', '1991-01-01', 5000.00, 'AB123456C', 1),  
 (2, 'Bob', 'Smith', 'Street #2', '3456789101', 'Assistant', 'M', '1992-02-02', 3000.00, 'BS234567D', 2),  
 (3, 'Charlie', 'Green', 'Street #3', '5678910234', 'Clerk', 'M', '1993-03-03', 2000.00, 'CG345678E', 3)  
  
  
*-- Inserting data into DreamHomeDB.Property\_for\_Rent table*insert into DreamHomeDB.Property\_for\_Rent(Pno, Street, Area, City, Pcode, Type, Rooms, Rent, Ono, Sno, Bno)  
values (1, 'Street #1', 'Area1', 'CityA', 'PC1', 'Flat', 2, 200.00, 1, 1, 1),  
 (2, 'Street #2', 'Area2', 'CityB', 'PC2', 'House', 3, 300.00, 2, 2, 2),  
 (3, 'Street #3', 'Area3', 'CityC', 'PC3', 'Bungalow', 4, 400.00, 3, 3, 3)  
  
  
*-- Inserting data into DreamHomeDB.Renter table*insert into DreamHomeDB.Renter(Rno, FName, LName, Address, Tel\_No, Pref\_Type, Max\_Rent, Bno)  
values (1, 'Don', 'Black', 'Street #4', '7891023456', 'Flat', 500.00, 1),  
 (2, 'Emma', 'White', 'Street #5', '9102345678', 'House', 600.00, 2),  
 (3, 'Frank', 'Gray', 'Street #6', '0123456789', 'Bungalow', 700.00, 3)  
  
  
*-- Inserting data into DreamHomeDB.Owner table*insert into DreamHomeDB.Owner(Ono, FName, LName, Address, Tel\_No)  
values (1, 'Gary', 'Purple', 'Street #7', '2345678901'),  
 (2, 'Helen', 'Pink', 'Street #8', '5678901234'),  
 (3, 'Ian', 'Orange', 'Street #9', '9012345678')  
  
  
*-- Inserting data into DreamHomeDB.Viewing table*insert into DreamHomeDB.Viewing(Rno, Pno, [Date], Comment)  
values (1, 1, '2020-01-01', 'Nice property'),  
 (2, 2, '2020-02-02', 'Good location'),  
 (3, 3, '2020-03-03', 'Well maintained')  
  
  
go;  
  
  
*-- Скласти список імен всіх клієнтів, які вже оглянули хоча б один об'єкт, що здається в оренду, і повідомили свою думку з цього приводу.*select *R*.FName as *FirstName*, *R*.LName as *LastName*, *V*.Comment  
from DreamHomeDB.Renter *R* inner join DreamHomeDB.Viewing *V* on *R*.Rno = *V*.Rno  
where *V*.Comment is not null;  
  
*-- Перелічити відділення компанії і об'єкти, що здаються в оренду, які розташовані в одному і тому ж місті, а також інші відділення компанії.*select *B*.Bno as *BranchNumber*, *B*.City as *BranchCity*, *P*.Pno as *PropertyNumber*, *P*.City as *PropertyCity*from DreamHomeDB.Branch *B* left join DreamHomeDB.Property\_for\_Rent *P* on *B*.City = *P*.City;  
  
*-- Перелічити відділення компанії і об'єкти, що здаються в оренду, які розташовані в одному і тому ж місті, а також всі інші об'єкти.*select *B*.Bno as *BranchNumber*, *B*.City as *BranchCity*, *P*.Pno as *PropertyNumber*, *P*.City as *PropertyCity*from DreamHomeDB.Property\_for\_Rent *P* left join DreamHomeDB.Branch *B* on *B*.City = *P*.City;  
  
*-- Перелічити відділення компанії і об'єкти, що здаються в оренду, розташовані в одному і тому ж місті, а також всі інші відділення і об'єкти.*select *B*.Bno as *BranchNumber*, *B*.City as *BranchCity*, *P*.Pno as *PropertyNumber*, *P*.City as *PropertyCity*from DreamHomeDB.Branch *B* full join DreamHomeDB.Property\_for\_Rent *P* on *B*.City = *P*.City;

РЕЗУЛЬТАТ РОБОТИ ПРОГРАМИ









В даній роботі не додавалися foreign keys, оскільки це не обов’язково.

Також не робився Primary key autoincrement, оскільки це не обов’язково.

**Варіант 5**

***Предметна область:*** Сесія

***ER-діаграма:***



*Категорія дисципліни – гуманітарна, комп’ютерна, математична, загальноінженерна тощо. Вид контроля – залік, екзамен.*

1. Записати рядок в таблиці Група, Сесія, Дисципліна за допомогою оператора INSERT.

2. Вибрати всі дані таблиці Група. Вибрати дані стовпця Назва.

3. Вибрати дані стовпців Назва, Обсяг годин таблиці Дисципліна, де дані впорядковані по стовпцю Обсяг годин за зростанням.

4. Визначити дисципліну з кількістю годин, рівною 150.