МІНІСТЕРСТВО ОСВІТИ І НАУКИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»
Кафедра систем штучного інтелекту

Лабораторна робота №5
З курсу “Організація баз даних та знань”

Виконав:
студент групи КН-210
Марій Павло

Викладач:
Мельникова Наталя Іванівна

Львів – 2020

Тема: Виконання теоретико-множинних операцій реляційної алгебри засобами SQL.

Мета: Розробити SQL запити для виконання операцій реляційної алгебри: об’єднання, перетину, різниці, декартового добутку.

Теоретичні відомості

Реляційна алгебра – це множина операцій, що виконуються над відношеннями і мають за мету утворення нових відношень або їх станів. Реляційна алгебра визначає операції, які однаковим чином реалізуються в усіх базах даних реляційного типу, незалежно від їх змісту і технологій, за допомогою яких вони реалізовані. Тобто реляційна алгебра представляє собою процедурну мову обробки реляційних таблиць.

Реляційна алгебра складається з таких операцій: об’єднання, перетин, різниця, декартовий добуток, проекція, селекція, натуральне з’єднання, умовне з’єднання, а також операції включення/вилучення кортежу з відношень, включення/вилучення атрибуту з відношення, зміни параметрів атрибуту.

Перші чотири операції взяті з математичної теорії множин і практично співпадають з операціями над множинами. Це зручно, оскільки реляційні таблиці є множинами, і цілком природно застосовувати до них операції над множинами.

Об’єднанням двох відношень R та S з відповідними множинами атрибутів називається відношення T , що має ту саму множину атрибутів, а його інформаційне наповнення утворюється кортежами першого та другого відношень за вилученням повторень.

Об’єднання дозволяє нам комбінувати дані з двох таблиць з однаковими множинами атрибутів. Однакові множини атрибутів потрібні для того, щоб результатом виконання операції об’єднання була реляційна таблиця.

Перетином двох відношень R та S з відповідними множинами атрибутів називається відношення T , що має ту саму множину атрибутів, а його інформаційне наповнення утворюється кортежами, які є спільними для цих двох відношень. Операція перетину дозволяє нам ідентифікувати рядки, спільні для двох таблиць.

Різницею двох відношень R та S з відповідними множинами атрибутів називається відношення T , що має ту саму множину атрибутів, а його інформаційне наповнення утворюється кортежами першого відношення за вилученням кортежів, які є спільними з другим відношенням. Операція різниці дозволяє ідентифікувати ті рядки, які є в одній таблиці, але відсутні в

іншій.

Декартовим добутком двох відношень R та S з відповідними множинами атрибутів називається нове відношення T , множина атрибутів якого складається з об’єднання множини атрибутів двох відношень, а кожен кортеж інформаційного наповнення утворюється шляхом конкатенації (сполучення) кожного кортежу першого відношення з кожним кортежем другого відношення.

Для реалізації теоретико-множинних операцій на мові SQL використовують директиву **SELECT**, спрощений опис якої наведено далі, а також функції роботи з множинами значень **IN**(), **NOT IN**().

SELECT

[*ALL* | *DISTINCT* | *DISTINCTROW*]

елемент_вибірки [, *елемент_вибірки*]

[**FROM** *перелік_таблиць*]

[**WHERE** *умова_відбору*]

елемент_вибірки

Вираз, або назва поля, значення якого потрібно вибрати. Символ «*» позначає всі поля.

перелік_таблиць

Назва таблиці, з якої здійснюється вибір значень.

умова_відбору

Вказує умови відбору потрібних записів.

DISTINCT | *DISTINCTROW*

Видалення з результату рядків-дублікатів. За замовчуванням вибираються всі рядки.

Для того, щоб виконати операцію об’єднання таблиць, потрібно за допомогою команди **UNION** об’єднати результати вибору рядків з двох, або більше, таблиць. Наведемо синтаксис команди.

SELECT ...

UNION [ALL | DISTINCT] SELECT ...

[UNION [ALL | DISTINCT] SELECT ...]

Хід роботи

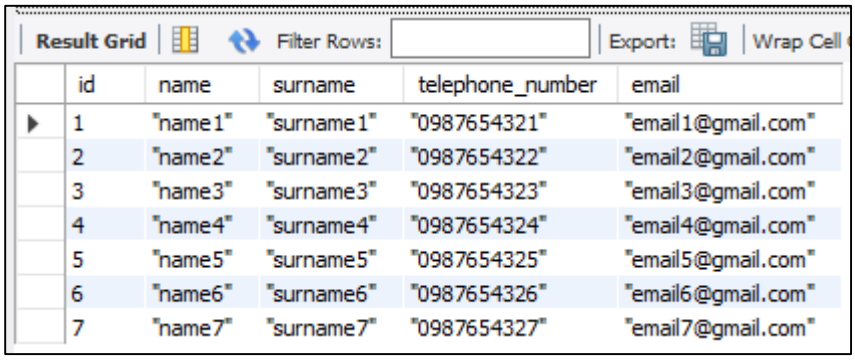
Перед виконанням завдання, потрібно сформувати дві таблиці з однаковими множинами атрибутів. Візьмемо за основу таблицю користувачів user і виконаємо вибір двох множин записів, які перетинаються. Результат збережемо в таблицях user1 і user2.

Код скрипта:

```
CREATE TABLE mydb.user1
AS SELECT id, name, surname, telephone_number, email FROM mydb.user
WHERE id <= 7;
CREATE TABLE mydb.user2
AS SELECT id, name, surname, telephone_number, email FROM mydb.user
WHERE id >= 4;
```

Результат:

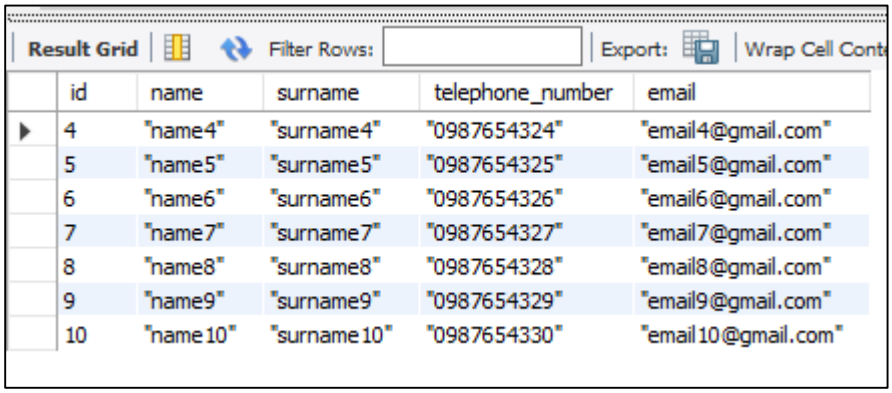
Таблиця user1:



Result Grid | Filter Rows: | Export: | Wrap Cell

id	name	surname	telephone_number	email
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"
4	"name4"	"surname4"	"0987654324"	"email4@gmail.com"
5	"name5"	"surname5"	"0987654325"	"email5@gmail.com"
6	"name6"	"surname6"	"0987654326"	"email6@gmail.com"
7	"name7"	"surname7"	"0987654327"	"email7@gmail.com"

Таблиця user2:



Result Grid | Filter Rows: | Export: | Wrap Cell Cont

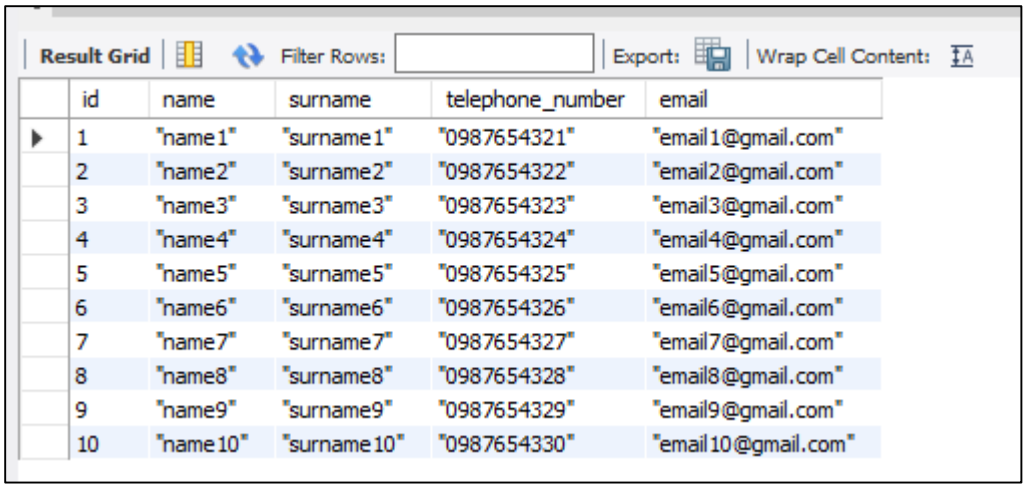
id	name	surname	telephone_number	email
4	"name4"	"surname4"	"0987654324"	"email4@gmail.com"
5	"name5"	"surname5"	"0987654325"	"email5@gmail.com"
6	"name6"	"surname6"	"0987654326"	"email6@gmail.com"
7	"name7"	"surname7"	"0987654327"	"email7@gmail.com"
8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"

1. Запит на виконання об’єднання user1 і user2:

Код скрипта:

```
SELECT * FROM mydb.user1
UNION SELECT * FROM mydb.user2;
```

Результат:

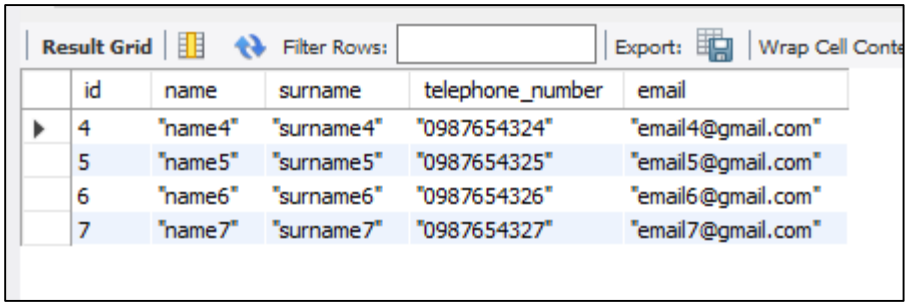


id	name	surname	telephone_number	email
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"
4	"name4"	"surname4"	"0987654324"	"email4@gmail.com"
5	"name5"	"surname5"	"0987654325"	"email5@gmail.com"
6	"name6"	"surname6"	"0987654326"	"email6@gmail.com"
7	"name7"	"surname7"	"0987654327"	"email7@gmail.com"
8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"

2. Запит на виконання перетину.
Код скрипта:

```
SELECT * FROM mydb.user1
WHERE id IN (SELECT id FROM mydb.user2);
```

Результат:

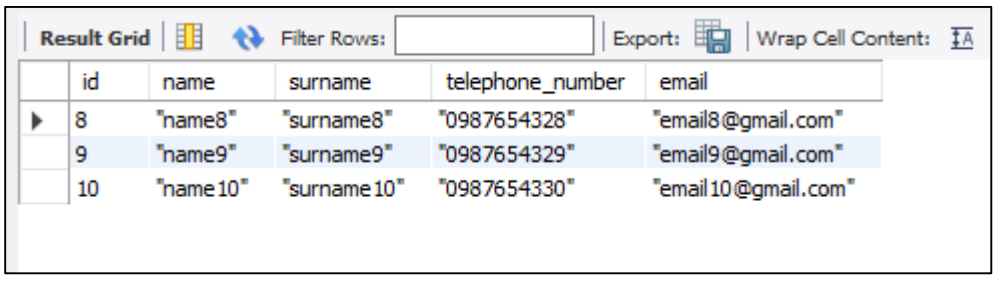


id	name	surname	telephone_number	email
4	"name4"	"surname4"	"0987654324"	"email4@gmail.com"
5	"name5"	"surname5"	"0987654325"	"email5@gmail.com"
6	"name6"	"surname6"	"0987654326"	"email6@gmail.com"
7	"name7"	"surname7"	"0987654327"	"email7@gmail.com"

3. Запит на виконання різниці user2 і user1.
Код скрипта:

```
SELECT * FROM mydb.user2
WHERE id NOT IN (SELECT id FROM mydb.user1);
```

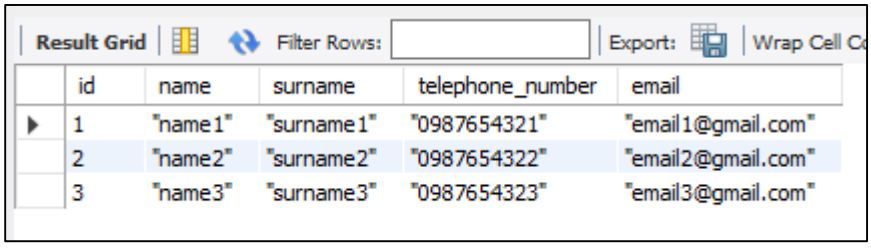
Результат:



id	name	surname	telephone_number	email
8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"

4. Запит на виконання декартового добутку двох таблиць. Для цього спершу видалимо деякі записи в таблицях, щоб таблиці стали меншими, і можна було наглядно бачити результат.

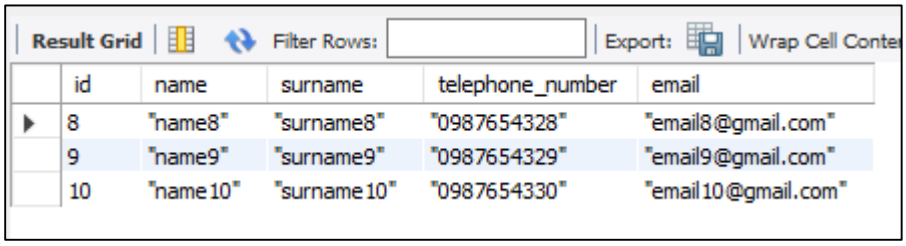
Таблиця user1:



Result Grid | Filter Rows: | Export: | Wrap Cell Co

id	name	surname	telephone_number	email
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"

Таблиця user2:



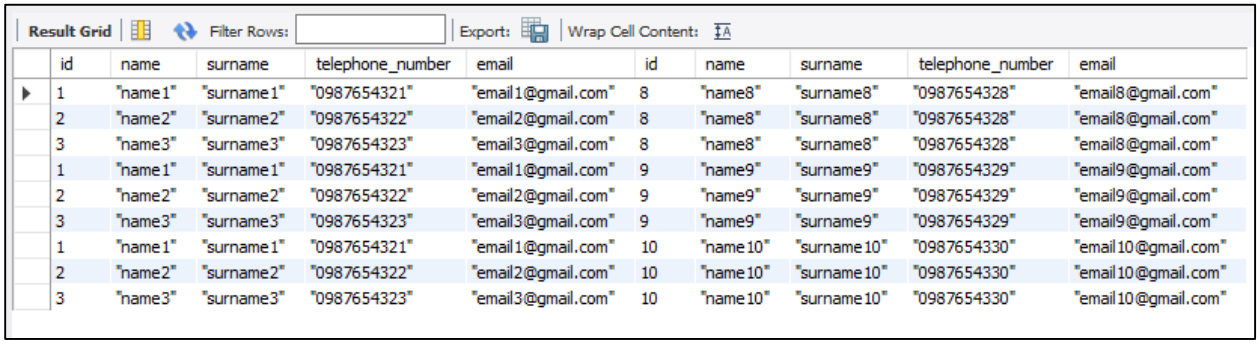
Result Grid | Filter Rows: | Export: | Wrap Cell Conte

id	name	surname	telephone_number	email
8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"

Код скрипта:

```
SELECT * FROM mydb.user1, mydb.user2;
```

Результат:



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

id	name	surname	telephone_number	email	id	name	surname	telephone_number	email
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"	8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"	8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"	8	"name8"	"surname8"	"0987654328"	"email8@gmail.com"
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"	9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"	9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"	9	"name9"	"surname9"	"0987654329"	"email9@gmail.com"
1	"name1"	"surname1"	"0987654321"	"email1@gmail.com"	10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"
2	"name2"	"surname2"	"0987654322"	"email2@gmail.com"	10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"
3	"name3"	"surname3"	"0987654323"	"email3@gmail.com"	10	"name10"	"surname10"	"0987654330"	"email10@gmail.com"

Висновок: На цій лабораторній роботі було розглянуто операції реляційної алгебри та їх реалізація на мові SQL. Здійснено об’єднання, перетин, різницю та декартовий добуток двох таблиць.