

МІНІСТЕРСТВО ОСВІТИ І НАУКИ  
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА  
ПОЛІТЕХНІКА»

Кафедра систем штучного інтелекту



Лабораторна робота №6  
З курсу “Організація баз даних та знань”

Виконав:  
студент групи КН-210  
Марій Павло

Викладач:  
Мельникова Наталя Іванівна

**Тема:** Виконання реляційних операцій реляційної алгебри засобами SQL.

**Мета:** Розробити SQL запити для виконання операцій реляційної алгебри: проекції, селекції, натурального з'єднання, умовного з'єднання.

### Теоретичні відомості

В реляційну алгебру крім теоретико-множинних операцій входять ще й реляційні операції над відношеннями. Зокрема проекція, селекція, натуральне та умовне з'єднання.

Проекцією відношення на задану підмножину множини атрибутів називають множину проекцій кортежів відношення на ці атрибути за вилученням повторень. Тобто операція створення проекції створює нову таблицю шляхом виключення певних стовпців з існуючої таблиці. Для створення проекції – реляційної таблиці, що складається лише з деяких визначених стовпців іншої реляційної таблиці – ми просто вказуємо початкову таблицю, а далі перелічуємо ті стовпці, які хочемо залишити.

Результатом операції селекції деякого відношення за заданим критерієм є нове відношення, яке утворюється з тих кортежів, значення атрибутів яких роблять істинною умову, сформульовану критерієм. Критерій селекції – це логічний вираз, який порівнює значення атрибутів кортежу з деякими заданими величинами. Вимоги до значень атрибутів критерію формуються через порівняння значень ( $=$ ,  $>$ ,  $<$ ,  $>=$ ,  $<=$  тощо).

Операція натурального з'єднання визначається для двох відношень, де відношення мають однакові атрибути. Результатом операції є нове відношення, множина атрибутів якого є об'єднанням множин атрибутів першого та другого відношень, а кожен кортеж утворюється шляхом об'єднання тих кортежів відношень, в яких значення спільних атрибутів співпадають. Дана операція призначена для утворення більш крупних відношень з більш дрібних.

Результатом умовного з'єднання двох відношень є нове відношення, множина атрибутів якого є об'єднанням множини атрибутів першого та другого відношень, а кожен кортеж утворюється шляхом об'єднання тих кортежів відношень, для яких виконується критерій умовного з'єднання за атрибутами. Для утворення умовного з'єднання необхідно визначити критерій або умову порівняння атрибутів з вказаним виразом або між собою.

Для створення проєкції на мові SQL можна використовувати директиву створення віртуальних таблиць **CREATE VIEW**:

**CREATE VIEW** ім'я\_проєкції [(перелік\_полів)]

**AS SELECT DISTINCT** (перелік\_полів) **FROM** ім'я\_таблиці

## Хід роботи

Таблиця user:

	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"
✱	NULL	NULL	NULL	NULL	NULL

Таблиця advertisement:

	id	price	mileage	color	condition	pubdate	address	year_auto	text	auto_id	user_id
▶	1	2000	1000	"color1"	"normal"	2020-05-20 10:10:10	"address1"	2001	"text1"	1	5
	2	3000	2000	"color2"	"bad"	2020-05-21 21:21:22	"address2"	2002	"text2"	1	3
	3	10999	9999	"color3"	"good"	2020-05-22 21:21:23	"address3"	2003	"text3"	3	3
	4	5000	4000	"color4"	"brilliant"	2020-05-23 21:21:24	"address4"	2004	"text4"	3	1
	5	6000	5000	"color5"	"normal"	2020-05-24 21:21:25	"address5"	2005	"text5"	5	1
✱	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

### 1. Запит на виконання проєкції:

Код скрипта (поле 'condition' задане в лапках, тому що назва поля співпадає з ключовим словом, яке використовується в SQL):

**CREATE VIEW** mydb.advertisementVIEW **AS SELECT DISTINCT** id, price, 'condition', auto\_id, pubdate

**FROM** mydb.advertisement;

Результат:

	id	price	condition	auto_id	pubdate
▶	1	2000	condition	1	2020-05-20 10:10:10
	2	3000	condition	1	2020-05-21 21:21:22
	3	10999	condition	3	2020-05-22 21:21:23
	4	5000	condition	3	2020-05-23 21:21:24
	5	6000	condition	5	2020-05-24 21:21:25

2. Запит на виконання селекції оголошень за певний час.

Код скрипта:

```
SELECT * FROM mydb.advertisement
```

```
WHERE pubdate >= '2020-05-21' AND pubdate <= '2020-05-23';
```

Результат:

	id	price	mileage	color	condition	pubdate	address	year_auto	text	auto_id	user_id
▶	2	3000	2000	"color2"	"bad"	2020-05-21 21:21:22	"address2"	2002	"text2"	1	3
	3	10999	9999	"color3"	"good"	2020-05-22 21:21:23	"address3"	2003	"text3"	3	3
*		NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3. Запит на виконання натурального з'єднання advertisement і user.

Код скрипта:

```
SELECT mydb.advertisement.id, mydb.advertisement.price,
```

```
mydb.advertisement.auto_id, mydb.advertisement.mileage,
```

```
mydb.advertisement.user_id, mydb.user.name, mydb.user.surname
```

```
FROM mydb.advertisement, mydb.user WHERE mydb.user.id =
```

```
mydb.advertisement.user_id;
```

Результат: показуються оголошення, а також ім'я і прізвище автора оголошення в одній таблиці за допомогою з'єднань:

	id	price	auto_id	mileage	user_id	name	surname
▶	1	2000	1	1000	5	"name5"	"surname5"
	2	3000	1	2000	3	"name3"	"surname3"
	3	10999	3	9999	3	"name3"	"surname3"
	4	5000	3	4000	1	"name1"	"surname1"
	5	6000	5	5000	1	"name1"	"surname1"

4. Запит на виконання умовного з'єднання. Додатковою умовою тут буде перевірка ціни оголошення відносно пробігу авто. Якщо ціна більша ніж пробіг авто, завищена ціна. Таким чином, ми знайдемо оголошення, в яких ціна завищена, і відразу отримаємо ім'я та прізвище користувача, який розмістив це оголошення.

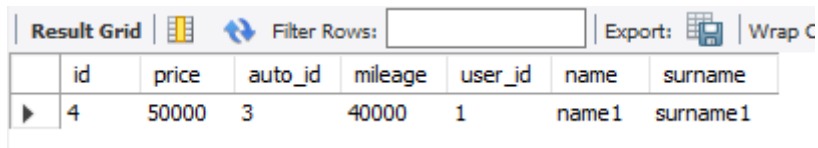
Для цього спершу внесу зміни в таблицю оголошень:

	id	price	mileage	color	condition	pubdate	address	year_auto	text	auto_id	user_id
▶	1	2000	10000	color1	normal	2020-05-20 10:10:10	address1	2001	text1	1	5
	2	3000	20000	color2	bad	2020-05-21 21:21:22	address2	2002	text2	1	3
	3	1000	30000	color3	good	2020-05-22 21:21:23	address3	2003	text3	3	3
	4	50000	40000	color4	brilliant	2020-05-23 21:21:24	address4	2004	text4	3	1
	5	7000	50000	color5	normal	2020-05-24 21:21:25	address5	2005	text5	5	1
*		NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Код скрипта:

```
SELECT mydb.advertisement.id, mydb.advertisement.price,  
mydb.advertisement.auto_id, mydb.advertisement.mileage,  
mydb.advertisement.user_id, mydb.user.name, mydb.user.surname  
FROM mydb.advertisement, mydb.user  
WHERE mydb.user.id = mydb.advertisement.user_id  
AND mydb.advertisement.price > mydb.advertisement.mileage;
```

Результат: одне оголошення.



The screenshot shows a database interface with a 'Result Grid' tab. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap C' option. The grid displays a single row of data with the following values:

	id	price	auto_id	mileage	user_id	name	surname
▶	4	50000	3	40000	1	name1	surname1

**Висновок:** на цій лабораторній роботі було розглянуто операції реляційної алгебри та здійснено проекцію, селекцію, натуральне та умовне з'єднання таблиць.