



**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
ІМЕНІ ІГОРЯ СІКОРСЬКОГО”**

**Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем**

**ЛАБОРАТОРНА РОБОТА №3
З дисципліни “Бази даних”
Тема: Засоби оптимізації роботи СУБД PostgreSQL**

**Виконав:
Мричко Богдан Тарасович
ФПМ, КП-83**

КИЇВ 2020

Мета: здобуття практичних навичок використання засобів оптимізації СУБД PostgreSQL.

Завдання

1. Перетворити модуль “Модель” з шаблону MVC лабораторної роботи No2 у вигляд об’єктно-реляційної проекції (ORM).
2. Створити та проаналізувати різні типи індексів у PostgreSQL.
3. Розробити тригер бази даних PostgreSQL.

Види індексів: BRIN, Hash

Умови для тригера: before update, delete

Завдання 1

Кожний клас об'єктів було переведено у відповідний клас моделі використаної ORM (sequelize). Приклади класів наведено нижче

Приклад моделі: employees

employees.ts

```
const { Sequelize, DataTypes, Model } = require('sequelize');
const sequelize = new Sequelize('sqlite::memory');

export default class Employees extends Model {}

Employees.init({
  // Model attributes are defined here
  employee_id: {
    type: DataTypes.NUMERIC,
    allowNull: false
  },
  name: {
    type: DataTypes.TEXT,
    allowNull: false
  },
  birth_day: {
    type: DataTypes.DATE,
    allowNull: false
  },
  contact_number: {
    type: DataTypes.NUMERIC,
    allowNull: false
  },
  home_address: {
    type: DataTypes.TEXT,
    allowNull: false
  },
  post: {
    type: DataTypes.TEXT,
    allowNull: false
  },
  department_id: {
    type: DataTypes.NUMERIC,
```

```

    allowNull: false
  },
  finished_cases: {
    type: DataTypes.NUMERIC
  }
}, {
  // Other model options go here
  sequelize, // We need to pass the connection instance
  modelName: 'employees' // We need to choose the model name
});

```

Приклад контроллера: carsController

carsController.ts

```

import Cars from '../models/cars';

export const onGet = async (): Promise<string> => {
  try {
    const cars = await Cars.get();
    return JSON.stringify(cars, null, 2);
  } catch (err) {
    console.log(err);
    return err.message;
  }
};

export const onDelete = async (id: number): Promise<string> => {
  try {
    await Cars.delete(id);
    return 'Successfully deleted';
  } catch (err) {
    return err.message;
  }
};

export const onUpdate = async (id: number, data: any):

```

```

Promise<string> => {
  try {

    await Cars.update(id, data);
    return 'Successfully update';
  } catch (err) {
    return err.message;
  }
};

```

Подібним чином була переписана логіка програми. Весь код є на [GitHub](#)

Завдання 2

Індекс Hash

Індексування даного типу доцільно використовувати для рідко змінних атомарних значень, оскільки за основу береться така структура даних як хеш таблиця.

Наведемо приклади запитів до індексування.

```

1 SELECT * FROM cars WHERE "licence_plate" = '1017b7ab0a'

```

Data Output Explain Messages Notifications

	car_id [PK] numeric	licence_plate text	vin_code text	department_id numeric	
1	7305	1017b7ab0a	de9ef7bc5a9...		2

✓ Successfully run. Total query runtime: 115 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 80 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 84 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 85 msec. 1 rows affected.

```
1 SELECT * FROM cars WHERE "licence_plate" = '1017b7ab0a' AND "vin_code" = 'de9ef7bc5a96bcd'
2
```

Data Output Explain Messages Notifications

	car_id [PK] numeric	licence_plate text	vin_code text	department_id numeric	
1	7305	1017b7ab0a	de9ef7bc5a9...		2

✓ Successfully run. Total query runtime: 105 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 86 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 102 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 80 msec. 1 rows affected.

✓ Successfully run. Total query runtime: 129 msec. 1 rows affected.

Команда для створення індексу

Query Editor Query History

```
1 CREATE INDEX car_plate_index ON cars USING hash("licence_plate");
2 CREATE INDEX car_vin_index ON cars USING hash("vin_code");
```

Data Output Explain Messages Notifications

CREATE INDEX

Query returned successfully in 246 msec.

Запити після створення індексу

PoliceDepartments/postgres@BDTermOne

Query Editor Query History

```
1 SELECT * FROM cars WHERE "licence_plate" = '1017b7ab0a'
2
```

Data Output Explain Messages Notifications

	car_id [PK] numeric	licence_plate text	vin_code text	department_id numeric
1	7305	1017b7ab0a	de9ef7bc5a9...	2

✓ Successfully run. Total query runtime: 83 msec. 1 rows affected.
 ✓ Successfully run. Total query runtime: 84 msec. 1 rows affected.
 ✓ Successfully run. Total query runtime: 80 msec. 1 rows affected.

Query Editor Query History

```
1 SELECT * FROM cars WHERE "licence_plate" = '1017b7ab0a' AND "vin_code" = 'de9ef7bc5a96bcd'
2
```

Data Output Explain Messages Notifications

	car_id [PK] numeric	licence_plate text	vin_code text	department_id numeric
1	7305	1017b7ab0a	de9ef7bc5a9...	2

✓ Successfully run. Total query runtime: 99 msec. 1 rows affected.
 ✓ Successfully run. Total query runtime: 81 msec. 1 rows affected.
 ✓ Successfully run. Total query runtime: 100 msec. 1 rows affected.

Як бачимо час запитів майже не змінився, можливо для більш помітного результату необхідно перевірити на більшій кількості даних.

Індекс BRIN

Призначений для обробки дуже великих таблиць, у яких певні стовпці мають певну природну кореляцію з їх фізичним розташуванням у таблиці

Наведено приклади запитів до індексування.

Query Editor

Query History

1

```
SELECT * FROM cases WHERE commit_date >= '1995-01-01'
```

Data Output

Explain

Messages

Notifications

	policemen_id numeric	offender_personal_id numeric	case_id [PK] uuid	name text	law_article text	commit_date date	is_active boolean	finish_date text		
1	12345		11111	5b0b5191-73...	6318be...	100	1995-01-01	false	[null]	<div>✓ Successfully run. Total query runtime: 87 msec. 4793 rows affected.</div> <div>✓ Successfully run. Total query runtime: 72 msec. 4793 rows affected.</div> <div>✓ Successfully run. Total query runtime: 110 msec. 4793 rows affected.</div>
2	12345		11111	f426c04e-94...	31b48...	160	1995-01-02	false	[null]	
3	12345		11111	43d62579-d4...	dcef0f...	276	1995-01-03	false	[null]	
4	12345		11111	cd94fc5b-b4...	6a9956...	161	1995-01-04	false	[null]	
5	12345		11111	9e5d4148-60...	13621...	198	1995-01-05	false	[null]	
6	12345		11111	8aeb0429-ae...	1810d...	273	1995-01-06	false	[null]	
7	12345		11111	0273ef74-a3...	fef05d...	188	1995-01-07	false	[null]	
8	12345		11111	292a0c89-e2...	4f3b71...	438	1995-01-08	false	[null]	
9	12345		11111	395301f4-acf...	c0fbe7...	403	1995-01-09	false	[null]	
10	12345		11111	c2fd9b10-e1...	6e2172...	224	1995-01-10	false	[null]	
11	12345		11111	5f379ac9-f9b...	2ac316...	131	1995-01-11	false	[null]	
12	12345		11111	74f54838-09f...	3b4c0f...	351	1995-01-12	false	[null]	
13	12345		11111	9e447776-8d...	c5923f...	34	1995-01-13	false	[null]	
14	12345		11111	78e22534-2d...	49d98...	455	1995-01-14	false	[null]	
15	12345		11111	64887c4a-0a...	81873...	154	1995-01-15	false	[null]	
16	12345		11111	db32c094-3a...	ab61ad...	259	1995-01-16	false	[null]	
17	12345		11111	00a10666-4a...	f46b5d...	72	1995-01-17	false	[null]	

1

```
SELECT * FROM cases WHERE commit_date BETWEEN '1995-01-01' AND '2000-01-01'
```

Data Output

Explain

Messages

Notifications

	policemen_id numeric	offender_personal_id numeric	case_id [PK] uuid	name text	law_article text	commit_date date	is_active boolean	finish_date text		
1	12345		11111	5b0b5191-73...	6318be...	100	1995-01-01	false	[null]	<div>✓ Successfully run. Total query runtime: 91 msec. 1827 rows affected.</div> <div>✓ Successfully run. Total query runtime: 70 msec. 1827 rows affected.</div> <div>✓ Successfully run. Total query runtime: 94 msec. 1827 rows affected.</div> <div>✓ Successfully run. Total query runtime: 82 msec. 1827 rows affected.</div>
2	12345		11111	f426c04e-94...	31b48...	160	1995-01-02	false	[null]	
3	12345		11111	43d62579-d4...	dcef0f...	276	1995-01-03	false	[null]	
4	12345		11111	cd94fc5b-b4...	6a9956...	161	1995-01-04	false	[null]	
5	12345		11111	9e5d4148-60...	13621...	198	1995-01-05	false	[null]	
6	12345		11111	8aeb0429-ae...	1810d...	273	1995-01-06	false	[null]	
7	12345		11111	0273ef74-a3...	fef05d...	188	1995-01-07	false	[null]	
8	12345		11111	292a0c89-e2...	4f3b71...	438	1995-01-08	false	[null]	
9	12345		11111	395301f4-acf...	c0fbe7...	403	1995-01-09	false	[null]	
10	12345		11111	c2fd9b10-e1...	6e2172...	224	1995-01-10	false	[null]	
11	12345		11111	5f379ac9-f9b...	2ac316...	131	1995-01-11	false	[null]	
12	12345		11111	74f54838-09f...	3b4c0f...	351	1995-01-12	false	[null]	
13	12345		11111	9e447776-8d...	c5923f...	34	1995-01-13	false	[null]	
14	12345		11111	78e22534-2d...	49d98...	455	1995-01-14	false	[null]	
15	12345		11111	64887c4a-0a...	81873...	154	1995-01-15	false	[null]	
16	12345		11111	db32c094-3a...	ab61ad...	259	1995-01-16	false	[null]	
17	12345		11111	00a10666-4a...	f46b5d...	72	1995-01-17	false	[null]	

Команда для створення індексу

Query Editor

Query History

1CREATE INDEX commit_date_index ON cases USING brin ("commit_date");

Data Output

Explain

Messages

Notifications

CREATE INDEX

Query returned successfully in 146 msec.

Запити після створення індексу

PoliceDepartments/postgres@BDTermOne

Query Editor

Query History

1SELECT * FROM cases WHERE commit_date >= '1995-01-01'

Data Output

Explain

Messages

Notifications

	policemen_id numeric	offender_personal_id numeric	case_id [PK] uuid	name text	law_article text	commit_date date	is_active boolean	finish_date text		
1	12345		11111	5b0b5191-73...	6318be...	100	1995-01-01	false	[null]	
2	12345		11111	f426c04e-94...	31b48...	160	1995-01-02	false	[null]	
3	12345		11111	43d62579-d4...	dcef0f...	276	1995-01-03	false	[null]	
4	12345		11111	cd94fc5b-b4...	6a9956...	161	1995-01-04	false	[null]	
5	12345		11111	9e5d4148-60...	13621...	198	1995-01-05	false	[null]	
6	12345		11111	8aeb0429-ae...	1810d...	273	1995-01-06	false	[null]	
7	12345		11111	0273ef74-a3...	fef05d...	188	1995-01-07	false	[null]	
8	12345		11111	292a0c89-e2...	4f3b71...	438	1995-01-08	false	[null]	
9	12345		11111	395301f4-acf...	c0fbe7...	403	1995-01-09	false	[null]	
10	12345		11111	c2fd9b10-e1...	6e2172...	224	1995-01-10	false	[null]	
11	12345		11111	5f379ac9-f9b...	2ac316...	131	1995-01-11	false	[null]	
12	12345		11111	74f54838-09f...	3b4c0f...	351	1995-01-12	false	[null]	✓ Successfully run. Total query runtime: 93 msec. 4793 rows affected.
13	12345		11111	9e447776-8d...	c5923f...	34	1995-01-13	false	[null]	
14	12345		11111	78e22534-2d...	49d98...	455	1995-01-14	false	[null]	✓ Successfully run. Total query runtime: 86 msec. 4793 rows affected.
15	12345		11111	64887c4a-0a...	81873...	154	1995-01-15	false	[null]	
16	12345		11111	db32c094-3a...	ab61ad...	259	1995-01-16	false	[null]	✓ Successfully run. Total query runtime: 97 msec. 4793 rows affected.
17	12345		11111	00a19666-dc...	f46b50...	73	1995-01-17	false	[null]	

PoliceDepartments/postgres@BDTermOne										
Query Editor Query History										
1 SELECT * FROM cases WHERE commit_date BETWEEN '1995-01-01' AND '2000-01-01'										
Data Output Explain Messages Notifications										
	policemen_id numeric	offender_personal_id numeric	case_id [PK] uuid	name text	law_article text	commit_date date	is_active boolean	finish_date text		
1	12345	11111	5b0b5191-73...	6318be...	100	1995-01-01	false	[null]		
2	12345	11111	f426c04e-94...	31b48...	160	1995-01-02	false	[null]		
3	12345	11111	43d62579-d4...	dcef0f...	276	1995-01-03	false	[null]		
4	12345	11111	cd94fc5b-b4...	6a9956...	161	1995-01-04	false	[null]		
5	12345	11111	9e5d4148-60...	13621...	198	1995-01-05	false	[null]		
6	12345	11111	8aeb0429-ae...	1810d...	273	1995-01-06	false	[null]		
7	12345	11111	0273ef74-a3...	fef05d...	188	1995-01-07	false	[null]		
8	12345	11111	292a0c89-e2...	4f3b71...	438	1995-01-08	false	[null]		
9	12345	11111	395301f4-acf...	c0fbe7...	403	1995-01-09	false	[null]		
10	12345	11111	c2fd9b10-e1...	6e2172...	224	1995-01-10	false	[null]		
11	12345	11111	5f379ac9-f9b...	2ac316...	131	1995-01-11	false	[null]		
12	12345	11111	74f54838-09f...	3b4c0f...	351	1995-01-12	false	[null]		
13	12345	11111	9e447776-8d...	c5923f...	34	1995-01-13	false	[null]		
14	12345	11111	78e22534-2d...	49d98...	455	1995-01-14	false	[null]		
15	12345	11111	64887c4a-0a...	81873...	154	1995-01-15	false	[null]		
16	12345	11111	db32c094-3a...	ab61ad...	259	1995-01-16	false	[null]		
17	12345	11111	00a19666-dc...	f46b50...	23	1995-01-17	false	[null]		

Як бачимо різниця в швидкодії доволі незначна, однак в цьому випадку, я припускаю це пов'язано зі штучністю закономірності між датами (кожна нова дата це інкремент попередньої)

Завдання 3

Before delete trigger

```

CREATE OR REPLACE FUNCTION on_car_delete_function()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    DELETE FROM employee_car
    WHERE employee_car.car_id = OLD.car_id;
    RETURN NULL;
END;
$$;

CREATE TRIGGER on_car_delete
BEFORE DELETE
ON cars
EXECUTE PROCEDURE on_car_delete_function()

```

При видаленні автомобіля департаменту з бази даних відбувається видалення всіх прив'язок працівників до автомобілів

Таблиці сутностей до видалення

public.cars/PoliceDepartments/postgres@BDTermOne					
Query Editor Query History					
<pre> 1 SELECT * FROM public.cars 2 ORDER BY car_id ASC </pre>					
Data Output Explain Messages Notifications					
	car_id [PK] numeric	licence_plate text	vin_code text	department_id numeric	
1	1	newlicenceplate0	f23f3f23rffw...	1	
2	2	newLicenceplate1	dthstvw3g54...	1	
3	3	rth34tb4	g34qg434g4...	2	
4	4	qwd12e1	324g234g234	3	
5	5	fwef241f	gwrgregerg3...	3	
6	6	hnerhrheb	ehrtethrht	1	
7	7	ewfqweggrgv	efbegur4tga	2	
8	8	34g3gq34g3	gerg4gw4	3	
9	9	ngnyl7itleyj	we45wjqgrg3...	1	
10	10	4q3t3g34hqjy7	h3574w5tq4...	2	
11	11	hq45jgrtq35hq	qh5q5j5jfg34h	3	
12	12	q34h35j45yf	qg43jn4hgrv...	1	
13	13	qh5hq45hfebq5h	ye5nen67kru...	2	

public.employee_car/PoliceDepartments/postgres@BDTermOne				
Query Editor Query History				
<pre> 1 SELECT * FROM public.employee_car 2 ORDER BY id ASC </pre>				
Data Output Explain Messages Notifications				
	employee_id numeric	car_id numeric	id [PK] uuid	
1	12345		5	df77cbbe-45...
2	12345		6	df77cbbe-45...
3	23456	13		df77cbbe-45...

Виконаємо команду видалення

Query Editor Query History				
<pre> 1 DELETE FROM cars WHERE car_id=13 </pre>				
Data Output Explain Messages Notifications				
DELETE 1				
Query returned successfully in 68 msec.				

Таблиці сутностей після видалення

Query Editor		Query History			
1 SELECT * FROM public.cars					
2 ORDER BY car_id ASC					
Data Output		Explain	Messages	Notifications	
	car_id [PK] numeric		licence_plate text	vin_code text	department_id numeric
1		1	newlicenceplate0	f23f3f23rffw...	1
2		2	newLicenceplate1	dthstvw3g54...	1
3		3	rth34tb4	g34qg434g4...	2
4		4	qwd12e1	324g234g234	3
5		5	fwef241f	gwrgregerg3...	3
6		6	hnerhrheb	ehrtethrbt	1
7		7	ewfqweggrgv	efbegur4tga	2
8		8	34g3gq34g3	gerg4gw4	3
9		9	ngnyl7itleyj	we45wjqgrg3...	1
10		10	4q3t3g34hqjy7	h3574w5tq4...	2
11		11	hq45jgrtq35hq	qh5q5j5jfg34h	3
12		12	q34h35j45yf	qg43jn4hgrv...	1

Query Editor		Query History			
1 SELECT * FROM public.employee_car					
2 ORDER BY id ASC					
Data Output		Explain	Messages	Notifications	
	employee_id numeric		car_id numeric	id [PK] uuid	
1	12345			5 df77cbbe-45...	
2	12345			6 df77cbbe-45...	

Before insert trigger

```

CREATE OR REPLACE FUNCTION on_case_insert_function()
RETURNS TRIGGER
LANGUAGE plpgsql
AS $$
BEGIN
    IF LENGTH(NEW.name) < 2 THEN
        NEW.name = 'default case name';
    END IF;
    RETURN NEW;
END;
$$;

CREATE TRIGGER on_case_insert
BEFORE DELETE
ON cases
EXECUTE PROCEDURE on_case_insert_function()

```

При створенні нової справи, якщо довжина назви справи менша двох символів, її назва замінюється на назву по замовчуванні.

Таблиця сутностей до внесення змін (для демонстрації всі дані видалені)





[Query Editor](#) [Query History](#)

```

1 SELECT * FROM public.cases
2 ORDER BY case_id ASC

```

[Data Output](#) [Explain](#) [Messages](#) [Notifications](#)

								
policemen_id	offender_perso	case_id	name	law_article	commit_date	is_active	finish_date	
numeric	numeric	[PK] uuid	text	text	date	boolean	text	

Виконання функції по створенню запису (з даними які викличуть спрацювання умови в тригері)

Query Editor Query History

```
1 INSERT INTO cases VALUES (  
2     12345,  
3     11111,  
4     uuid_in(md5(random()::text || clock_timestamp()::text)::cstring),  
5     'q',  
6     (random() * (500 - 1) + 1)::integer,  
7     date '1980-09-28',  
8     false  
9 );
```

Таблиця сутностей після внесення змін

 public.cases/PoliceDepartments/postgres@BDTermOne

Query Editor Query History

```
1 SELECT * FROM public.cases  
2 ORDER BY case_id ASC
```

Data Output Explain Messages Notifications

	 policemen_id numeric	 offender_perso numeric	 case_id [PK] uuid	 name text	 law_article text	 commit_date date	 is_active boolean	 finish_date text
1	12345	11111	43c1bf59-d32c-ed7...	default case ...	148	1980-09-28	false	[null]

Висновок

Я здобув практичні навички використання засобів оптимізації СУБД PostgreSQL таких як індексування та створення тригерів, що реагують на деякі події, заміщуючи собою програмну реалізацію у додатках, що пришвидшує роботу та зменшує кількість запитів.