

Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Інститут комп'ютерних наук та інформаційних технологій
Кафедра «Системи штучного інтелекту»



Звіт до лабораторної роботи №13
З дисципліни «Організація Баз Даних»

Виконав:
студент групи КН-208
Дерев'янний Андрій

Прийняла:
Мельникова Н.І

Львів-2020

Тема: Аналіз та оптимізація запитів

Мета: Навчитися аналізувати роботу СУБД та оптимізувати виконання складних запитів на вибірку даних. Виконати аналіз складних запитів за допомогою директиви EXPLAIN, модифікувати найповільніші запити з метою їх пришвидшення

Завдання:

Проаналізувати роботу СУБД та оптимізувати виконання складних запитів на вибірку даних. Виконати аналіз складних запитів за допомогою директиви EXPLAIN, модифікувати найповільніші запити з метою їх пришвидшення

Виконання завдання:

Спочатку виведемо всі індекси, які входять в таблицю worker:

```
1      use db;
2
3 •    show index from worker;
```

Результат роботи запиту:

Result Grid											
Filter Rows:											
Export: Wrap Cell Content:											
	Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
▶	worker	0	PRIMARY	1	id	A	10	NULL	NULL		BTREE

Тепер відобразимо роботу індексів, використовуючи explain select:

```
8 •    explain select name, surname
9      from worker
10     group by name desc;
```

Результат роботи запиту:

Result Grid												
Filter Rows:												
Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	worker	NULL	ALL	NULL	NULL	NULL	NULL	10	100.00	Using temporary; Using filesort

Створимо індекс для оптимізації пошуку адреси офісу працівника:

```
12 •    explain select name, surname, contract_type, salary, position_name, department.address
13      from ((worker INNER JOIN contract) INNER JOIN position) INNER JOIN department
14      ON worker.id=contract.worker_id
15      AND contract.position_id=position.id
16      AND position.department_id=department.id
17      where department.address = "address"
18      and worker.surname = "surname";
```

Результат роботи запиту:

Result Grid												
Filter Rows:												
Export: Wrap Cell Content:												
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
▶	1	SIMPLE	department	NULL	ALL	PRIMARY	NULL	NULL	NULL	3	100.00	NULL
	1	SIMPLE	position	NULL	ref	PRIMARY, fk1	fk1	4	db.department.id	2	100.00	NULL
	1	SIMPLE	contract	NULL	ref	fk2, fk3	fk3	4	db.position.id	1	100.00	NULL
	1	SIMPLE	worker	NULL	eq_ref	PRIMARY	PRIMARY	4	db.contract.worker_id	1	100.00	NULL

Потрібно оптимізувати вивід даних з даблиці department, створивши індекс dept_index. Також створимо індекс worker_index:

```
20 • create index dept_index on department(address);
21 • create index worker_index on worker(surname);
```

Результат роботи запиту:

Result Grid Filter Rows: Export: Wrap Cell Content:											
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered
▶	1	SIMPLE	department	NULL	ref	PRIMARY,dept_index	dept_index	130	const	1	100.00
	1	SIMPLE	position	NULL	ref	PRIMARY,fk1	fk1	4	db.department.id	2	100.00
	1	SIMPLE	contract	NULL	ref	fk2,fk3	fk3	4	db.position.id	1	100.00
	1	SIMPLE	worker	NULL	eq_ref	PRIMARY	PRIMARY	4	db.contract.worker_id	1	100.00

Для того, щоб вивести у заданому порядку, використаємо straight_join

```
23 • explain select straight_join name, surname, contract_type, salary, position_name, department.address
24 from ((worker INNER JOIN contract) INNER JOIN position) INNER JOIN department
25 ON worker.id=contract.worker_id
26 AND contract.position_id=position.id
27 AND position.department_id=department.id
28 where department.address = "address"
29 and worker.surname = "surname";
```

Результат роботи запиту:

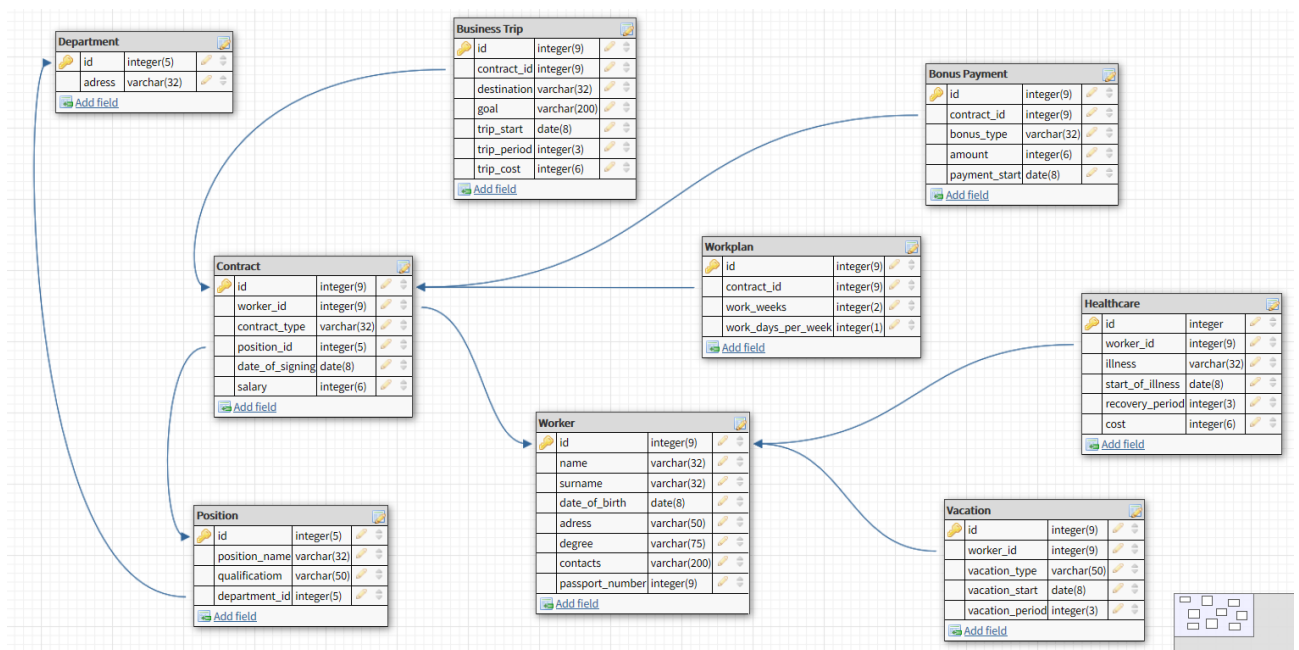
Result Grid Filter Rows: Export: Wrap Cell Content:											
	id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered
▶	1	SIMPLE	worker	NULL	ref	PRIMARY,worker_index	worker_index	130	const	1	
	1	SIMPLE	contract	NULL	ref	fk2,fk3	fk2	4	db.worker.id	1	
	1	SIMPLE	position	NULL	eq_ref	PRIMARY,fk1	PRIMARY	4	db.contract.position_id	1	
	1	SIMPLE	department	NULL	eq_ref	PRIMARY,dept_index	PRIMARY	4	db.position.department_id	1	

Висновок:

За час виконання лабораторної роботи я навчився аналізувати роботу СУБД та оптимізувати виконання складних запитів на вибірку даних, виконав аналіз складних запитів за допомогою директиви EXPLAIN, модифікував найповільніші запити з метою їх пришвидшення

Додаток:

Діаграма бази даних(відділ кадрів підприємства):



SQL-скрипт бази даних:

```
create database db;
```

```
use db;
```

```
CREATE TABLE `Department` (
    `id` INT(5) NOT NULL AUTO_INCREMENT,
    `adress` varchar(32) NOT NULL,
    PRIMARY KEY (`id`)
);
```

```
CREATE TABLE `Worker` (
    `id` INT(9) NOT NULL AUTO_INCREMENT,
    `name` varchar(32) NOT NULL,
    `surname` varchar(32) NOT NULL,
    `date_of_birth` DATE NOT NULL,
    `adress` varchar(50) NOT NULL,
    `degree` varchar(75) NOT NULL,
    `contacts` varchar(200) NOT NULL,
    `passport_number` INT(9) NOT NULL,
    PRIMARY KEY (`id`)
);
```

);

```
CREATE TABLE `Vacation` (  
    `id` INT(9) NOT NULL AUTO_INCREMENT,  
    `worker_id` INT(9) NOT NULL,  
    `vacation_type` varchar(50),  
    `vacation_start` DATE,  
    `vacation_period` INT(3),  
    PRIMARY KEY (`id`),  
  
    CONSTRAINT fk0 FOREIGN KEY (worker_id)  
        REFERENCES Worker(id)  
        ON DELETE CASCADE ON UPDATE CASCADE  
);
```

```
CREATE TABLE `Position` (  
    `id` INT(5) NOT NULL AUTO_INCREMENT,  
    `position_name` varchar(32) NOT NULL,  
    `qualification` varchar(50) NOT NULL,  
    `department_id` INT(5) NOT NULL,  
    PRIMARY KEY (`id`),  
  
    CONSTRAINT fk1 FOREIGN KEY (department_id)  
        REFERENCES Department(id)  
        ON DELETE CASCADE ON UPDATE CASCADE  
);
```

```
CREATE TABLE `Contract` (  
    `id` INT(9) NOT NULL AUTO_INCREMENT,  
    `worker_id` INT(9) NOT NULL,  
    `contract_type` varchar(32) NOT NULL,  
    `position_id` INT(5) NOT NULL,  
    `date_of_signing` DATE NOT NULL,  
    `salary` INT(6) NOT NULL,  
    PRIMARY KEY (`id`),
```

```
CONSTRAINT fk2 FOREIGN KEY (worker_id)
REFERENCES Worker(id)
ON DELETE CASCADE ON UPDATE CASCADE,
```

```
CONSTRAINT fk3 FOREIGN KEY (position_id)
REFERENCES `Position`(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Workplan` (
  `id` INT(9) NOT NULL AUTO_INCREMENT,
  `contract_id` INT(9) NOT NULL,
  `work_weeks` INT(2),
  `work_days_per_week` INT(1),
  PRIMARY KEY (`id`),
```

```
CONSTRAINT fk4 FOREIGN KEY (contract_id)
REFERENCES Contract(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Bonus_Payment` (
  `id` INT(9) NOT NULL AUTO_INCREMENT,
  `contract_id` INT(9) NOT NULL,
  `bonus_type` varchar(32),
  `amount` INT(6),
  `payment_start` DATE,
  PRIMARY KEY (`id`),
```

```
CONSTRAINT fk5 FOREIGN KEY (contract_id)
REFERENCES Contract(id)
ON DELETE CASCADE ON UPDATE CASCADE
```

```
);
```

```
CREATE TABLE `Healthcare` (
  `id` INT NOT NULL AUTO_INCREMENT,
```

```
`worker_id` INT(9) NOT NULL,  
`illness` varchar(32),  
`start_of_illness` DATE,  
`recovery_period` INT(3),  
`cost` INT(6),  
PRIMARY KEY (`id`),
```

```
CONSTRAINT fk6 FOREIGN KEY (worker_id)  
REFERENCES Worker(id)  
ON DELETE CASCADE ON UPDATE CASCADE  
);
```

```
CREATE TABLE `Business_Trip` (  
  `id` INT(9) NOT NULL AUTO_INCREMENT,  
  `contract_id` INT(9) NOT NULL,  
  `destination` varchar(32),  
  `goal` varchar(200),  
  `trip_start` DATE,  
  `trip_period` INT(3),  
  `trip_cost` INT(6),  
  PRIMARY KEY (`id`),  
  
  CONSTRAINT fk7 FOREIGN KEY (contract_id)  
  REFERENCES Contract(id)  
  ON DELETE CASCADE ON UPDATE CASCADE  
);
```