МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"
ІНСТИТУТ ПІСЛЯДИПЛОМНОЇ ОСВІТИ
КАФЕДРА СИСТЕМ ШТУЧНОГО ІНТЕЛЕКТУ

ЗВІТ ДО ЛАБОРАТОРНОЇ РОБОТИ №13

Підготувала:

Студентка групи КН-209

Кульчицька Олена

Викладач:

Мельникова Н.І.

2020

Лабораторна робота №13
на тему:
“Аналіз та оптимізація запитів”

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

Короткі теоретичні відомості.

Для аналізу виконання запитів в MySQL існує декілька спеціальних директив. Основна з них – EXPLAIN.

Директива EXPLAIN дозволяє визначити поля таблиці, для яких варто створити додаткові індекси, щоб пришвидшити вибірку даних. Індекс – це механізм, який підвищує швидкість пошуку та доступу до записів за індексованими полями. Загалом, варто створювати індекси для тих полів, за якими відбувається з’єднання таблиць, перевірка умови чи пошук.

За допомогою директиви EXPLAIN також можна визначити послідовність, в якій відбувається з’єднання таблиць при вибірці даних. Якщо оптимізатор вибирає не найкращу послідовність з’єднання таблиць, потрібно використати опцію STRAIGHT_JOIN директиви SELECT. Тоді з’єднання таблиць буде відбуватись в тому порядку, в якому перераховані таблиці у запиті. Також, за допомогою опцій FORCE INDEX, USE INDEX та IGNORE INDEX можна керувати використанням індексів у випадку їх неправильного вибору оптимізатором, тобто, якщо вони не підвищують ефективність вибірки рядків.

Опис директив.

SELECT BENCHMARK(*кількість_циклів*, *вираз*)

Виконує вираз вказану кількість разів, і повертає загальний час виконання.

EXPLAIN SELECT …

Використовується разом із запитом SELECT. Виводить інформацію про план обробки і виконання запиту, включно з інформацією про те, як і в якому порядку з’єднувались таблиці. EXPLAIN EXTENDED виводить розширену інформацію.

Результати директиви виводяться у вигляді рядків з такими полями:

id – порядковий номер директиви SELECT у запиті;

select_type – тип вибірки (simple, primary, union, subquery, derived, uncachable subquery тощо);

table – назва таблиці, для якої виводиться інформація;

type – тип з’єднання (system, const, eq_ref, ref, fulltext, range тощо); possible_keys – індекси, які наявні у таблиці, і можуть бути використані; key – назва індексу, який було обрано для виконання запиту;

key_len – довжина індекса, який був використаний при виконанні запиту;

ref – вказує, які рядки чи константи порівнюються зі значенням індекса при відборі;

rows – (прогнозована) кількість рядків, потрібних для виконання запиту;

Extra – додаткові дані про хід виконання запиту.

ANALYZE TABLE
Оновлює статистичну інформацію про таблицю (наприклад, поточний розмір ключових полів). Ця інформація впливає на роботу оптимізатора запитів, і може вплинути на вибір індексів при виконанні запитів.
SHOW INDEX FROM *ім'я_таблиці*
Виводить інформацію про індекси таблиці.
CREATE [UNIQUE | FULLTEXT] INDEX *назва*
ON *ім'я_таблиці* (*перелік_полів*)
Створює індекс для одного або декількох полів таблиці. Одне поле може входити до кількох індексів. Якщо індекс оголошено як UNIQUE, то значення відповідних полів таблиці повинні бути унікальними. Таблиці MyISAM підтримують створення повнотекстових індексів (FULLTEXT) для полів типу TEXT, CHAR, VARCHAR.

Хід роботи.

```
SHOW INDEX FROM confectionary.order;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
order	0	PRIMARY	1	id	A	7	NULL	NULL		BTREE		
order	1	customer_idx	1	customer_id	A	4	NULL	NULL		BTREE		
order	1	staff_idx	1	staff_id	A	1	NULL	NULL		BTREE		

```
SHOW INDEX FROM confectionary.staff;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
staff	0	PRIMARY	1	id	A	2	NULL	NULL		BTREE		

```
SHOW INDEX FROM confectionary.customer;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
customer	0	PRIMARY	1	id	A	9	NULL	NULL		BTREE		

Завдання 1

```
EXPLAIN  SELECT first_name, second_name FROM confectionary.customer
WHERE first_name LIKE 'O%' AND second_name LIKE 'L%'
ORDER BY first_name ASC;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	customer	NULL	ALL	NULL	NULL	NULL	NULL	9	11.11	Using where; Using filesort

```
SHOW INDEX FROM confectionary.customer;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
customer	0	PRIMARY	1	id	A	9	NULL	NULL		BTREE		

```
CREATE INDEX firstNameINDX ON confectionary.customer (first_name);
CREATE INDEX secondNameINDX ON confectionary.customer (second_name);
SHOW INDEX FROM confectionary.customer;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
customer	0	PRIMARY	1	id	A	9	NULL	NULL		BTREE		
customer	1	firstNameINDX	1	first_name	A	8	NULL	NULL		BTREE		
customer	1	secondNameINDX	1	second_name	A	9	NULL	NULL		BTREE		

```
EXPLAIN  SELECT first_name, second_name FROM confectionary.customer
WHERE first_name LIKE 'O%' AND second_name LIKE 'L%'
ORDER BY first_name ASC;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	customer	NULL	range	firstNameINDX,secondNameINDX	firstNameINDX	137	NULL	2	22.22	Using index condition; Using where

Завдання 2

```
EXPlAIN  SELECT customer.first_name as customer, staff.first_name as staff
FROM (confectionary.customer INNER JOIN confectionary.order )
INNER JOIN confectionary.staff
ON order.staff_id=staff.id
AND order.customer_id=customer.id
ORDER BY customer.first_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	order	NULL	ALL	customer_idx,staff_idx	NULL	NULL	NULL	7	100.00	Using temporary; Using filesort
1	SIMPLE	staff	NULL	ALL	PRIMARY	NULL	NULL	NULL	2	50.00	Using where; Using join buffer (Block
1	SIMPLE	customer	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.order.customer_id	1	100.00	NULL

```
CREATE INDEX firstNameStINDX ON confectionary.staff (first_name);
SHOW INDEX FROM confectionary.staff;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
staff	0	PRIMARY	1	id	A	2	NULL	NULL		BTREE		
staff	1	firstNameStINDX	1	first_name	A	2	NULL	NULL		BTREE		

```
EXPlAIN  SELECT customer.first_name as customer, staff.first_name as staff
FROM (confectionary.customer INNER JOIN confectionary.order )
INNER JOIN confectionary.staff
ON order.staff_id=staff.id
AND order.customer_id=customer.id
ORDER BY customer.first_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	order	NULL	ALL	customer_idx,staff_idx	NULL	NULL	NULL	7	100.00	Using temporary; Using filesort
1	SIMPLE	staff	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.order.staff_id	1	100.00	NULL
1	SIMPLE	customer	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.order.customer_id	1	100.00	NULL

```
CREATE INDEX orderStaffINDX ON confectionary.order (staff_id,customer_id);
SHOW INDEX FROM confectionary.order;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment
order	0	PRIMARY	1	id	A	7	NULL	NULL		BTREE		
order	1	customer_idx	1	customer_id	A	4	NULL	NULL		BTREE		
order	1	staff_idx	1	staff_id	A	1	NULL	NULL		BTREE		
order	1	orderStaffINDX	1	staff_id	A	1	NULL	NULL		BTREE		
order	1	orderStaffINDX	2	customer_id	A	5	NULL	NULL		BTREE		

```
EXPlAIN  SELECT customer.first_name as customer, staff.first_name as staff
FROM (confectionary.customer INNER JOIN confectionary.order )
INNER JOIN confectionary.staff
ON order.staff_id=staff.id
AND order.customer_id=customer.id
ORDER BY customer.first_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	order	NULL	index	customer_idx,staff_idx,orderStaffINDX	orderStaffINDX	8	NULL	7	100.00	Using index; Using temp
1	SIMPLE	staff	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.order.staff_id	1	100.00	NULL
1	SIMPLE	customer	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.order.customer_id	1	100.00	NULL

Завдання 3

```
EXPLAIN SELECT dish.id, dish.name AS dish_name, COUNT(ingredient.id) AS amount
FROM (ingredient_dish INNER JOIN(provider INNER JOIN ingredient))
INNER JOIN dish
ON provider.id=ingredient.provider_id
AND ingredient.id=ingredient_dish.ingredient_id
AND ingredient_dish.id_dish=dish.id
WHERE ingredient.price_for_unit BETWEEN 9 AND 40
GROUP BY dish_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	ingredient	NULL	ALL	PRIMARY,provider_idx	NULL	NULL	NULL	9	11.11	Using where; Using temp
1	SIMPLE	provider	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.ingredient.provider_id	1	100.00	Using index
1	SIMPLE	ingredient_dish	NULL	ref	ingredient_idx,dish_idx	ingredient_idx	4	confectionary.ingredient.id	1	100.00	NULL
1	SIMPLE	dish	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.ingredient_dish.id_dish	1	100.00	NULL

```
EXPLAIN SELECT STRAIGHT_JOIN dish.id, dish.name AS dish_name, COUNT(ingredient.id) AS amount
FROM (ingredient_dish INNER JOIN(provider INNER JOIN ingredient))
INNER JOIN dish
ON provider.id=ingredient.provider_id
AND ingredient.id=ingredient_dish.ingredient_id
AND ingredient_dish.id_dish=dish.id
WHERE ingredient.price_for_unit BETWEEN 9 AND 40
GROUP BY dish_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	ingredient_dish	NULL	ALL	ingredient_idx,dish_idx	NULL	NULL	NULL	4	100.00	Using temporary; Using filesor
1	SIMPLE	provider	NULL	index	PRIMARY	PRIMARY	4	NULL	2	100.00	Using index; Using join buffer
1	SIMPLE	ingredient	NULL	ALL	PRIMARY,provider_idx	NULL	NULL	NULL	9	11.11	Using where; Using join buffe
1	SIMPLE	dish	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.ingredient_dish.id_dish	1	100.00	NULL

```
CREATE INDEX dish_indx ON dish(name);
SHOW INDEX FROM confectionary.dish;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
dish	0	PRIMARY	1	id	A	5	NULL	NULL		BTREE
dish	1	dish_indx	1	name	A	5	NULL	NULL		BTREE

```
CREATE INDEX ing_diss_indx ON ingredient_dish(ingredient_id,id_dish);
SHOW INDEX FROM confectionary.ingredient_dish;
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
ingredient_dish	0	PRIMARY	1	id	A	4	NULL	NULL		BTREE
ingredient_dish	1	ingredient_idx	1	ingredient_id	A	4	NULL	NULL		BTREE
ingredient_dish	1	dish_idx	1	id_dish	A	2	NULL	NULL		BTREE
ingredient_dish	1	ing_diss_indx	1	ingredient_id	A	4	NULL	NULL		BTREE
ingredient_dish	1	ing_diss_indx	2	id_dish	A	4	NULL	NULL		BTREE

```
CREATE INDEX price_indx ON ingredient(price_for_unit);
SHOW INDEX FROM confectionary.ingredient
```

Результат запиту:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
ingredient	0	PRIMARY	1	id	A	9	NULL	NULL		BTREE
ingredient	1	provider_idx	1	provider_id	A	1	NULL	NULL		BTREE
ingredient	1	price_indx	1	price_for_unit	A	6	NULL	NULL		BTREE

```
EXPLAIN SELECT dish.id, dish.name AS dish_name, COUNT(ingredient.id) AS amount
FROM (ingredient_dish INNER JOIN(provider INNER JOIN ingredient))
INNER JOIN dish
ON provider.id=ingredient.provider_id
AND ingredient.id=ingredient_dish.ingredient_id
AND ingredient_dish.id_dish=dish.id
WHERE ingredient.price_for_unit BETWEEN 9 AND 40
GROUP BY dish_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	ingredient_dish	NULL	index	ingredient_idx,dish_idx,ing_diss_indx	ing_diss_indx	8	NULL	4	100.00	Using inde
1	SIMPLE	dish	NULL	eq_ref	PRIMARY,dish_indx	PRIMARY	4	confectionary.ingredient_dish.id_dish	1	100.00	NULL
1	SIMPLE	ingredient	NULL	eq_ref	PRIMARY,provider_idx,price_indx	PRIMARY	4	confectionary.ingredient_dish.ingredient_id	1	66.67	Using whe
1	SIMPLE	provider	NULL	eq_ref	PRIMARY	PRIMARY	4	confectionary.ingredient.provider_id	1	100.00	Using inde

```
EXPLAIN SELECT STRAIGHT_JOIN dish.id, dish.name AS dish_name, COUNT(ingredient.id) AS amount
FROM (ingredient_dish INNER JOIN(provider INNER JOIN ingredient))
INNER JOIN dish
ON provider.id=ingredient.provider_id
AND ingredient.id=ingredient_dish.ingredient_id
AND ingredient_dish.id_dish=dish.id
WHERE ingredient.price_for_unit BETWEEN 9 AND 40
GROUP BY dish_name;
```

Результат запиту:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	ingredient_dish	NULL	index	ingredient_idx,dish_idx,ing_diss_indx	ing_diss_indx	8	NULL	4	100.00	Using inde
1	SIMPLE	provider	NULL	index	PRIMARY	PRIMARY	4	NULL	2	100.00	Using inde
1	SIMPLE	ingredient	NULL	eq_ref	PRIMARY,provider_idx,price_indx	PRIMARY	4	confectionary.ingredient_dish.ingredient_id	1	66.67	Using whe
1	SIMPLE	dish	NULL	eq_ref	PRIMARY,dish_indx	PRIMARY	4	confectionary.ingredient_dish.id_dish	1	100.00	NULL

Висновок. На даній лабораторній роботі я навчився аналізувати і оптимізувати виконання запитів. Для аналізу запитів було використано директиву EXPLAIN, а для оптимізації – модифікація порядку з’єднання таблиць і створення додаткових індексів CREATE INDEX та EXPLAIN SELECT STRAIGHT_JOIN.