

Домашнее задание к занятию

«Индексы» Акимов Александр

Задание 1

Напишите запрос к учебной базе данных, который вернёт процентное отношение общего размера всех индексов к общему размеру всех таблиц.

ВЫБЕРИТЕ table_schema, sum(index_length / data_length) * 100 "Размер в %" ИЗ information_schema.ТАБЛИЦЫ ГРУППИРУЮТСЯ По table_schema, ИМЕЮЩЕЙ table_schema = "сакила";

Задание 2

Выполните explain analyze следующего запроса:

- перечислите узкие места; Избыточные запросы к таблицам, которые не нужны для получения этих данных, в оконной функции лишняя таблица, лишние условия.
- оптимизируйте запрос: внесите корректировки по использованию операторов, при необходимости добавьте индексы.

объясните, проанализируйте, выберите distinct concat(c.last_name, ' ', c.first_name), sum(p.amount) из payment p присоедините customer c к c.customer_id = p.customer_id, где p.payment_date >= '2005-07-30' и p.payment_date < date_add('2005-07-30', ИНТЕРВАЛ 1 ДЕНЬ), сгруппируйте по p.customer_id;

-> Ограничение: 200 строк (фактическое время = 7,38 .. 7,41 строк = 200 циклов = 1) -> Сортировка с удалением дубликатов: concat(c.last_name, ' ', c.first_name), sum(p.amount) (фактическое время = 7,38 .. 7,39 строк = 200 циклов = 1) -> Сканирование таблицы (фактическое время = 7,04 .. 7,11 строк = 391 цикл = 1) -> Агрегирование с использованием временной таблицы (фактическое время = 7,04 .. 7,04 строк = 391 цикл = 1) -> Внутреннее объединение вложенного цикла (стоимость = 507 строк = 634) (фактическое время = 0,0934 .. 6,23 строк = 634 цикла = 1) -> Сканирование диапазона индексов на p с использованием pay_date поверх ('2005-07-30 00:00:00' <= дата платежа < '2005-07-31 00:00:00'), с условием индексации: ((p.payment_date >= ВРЕМЕННАЯ МЕТКА '2005-07-30 00:00:00') и (p.payment_date < (('2005-07-30' + интервал 1 день)))) (стоимость = 286 строк = 634) (фактическое время = 0.072 .. 4.84 строк = 634 цикла = 1) -> Поиск по индексу в одной строке на c с

использованием PRIMARY (customer_id=p.customer_id) (стоимость = 0.25 строк = 1) (фактическое время = 0,00191..0,00195 рядов = 1 петля =634)

The screenshot shows the DBeaver SQL IDE interface. The main window displays an SQL query and its execution plan. The query is as follows:

```
explain analyze
select distinct concat(c.last_name, ' ', c.first_name), sum(p.amount)
from payment p
join customer c on c.customer_id = p.customer_id
where p.payment_date >= '2005-07-30' and p.payment_date < date_add('2005-07-30', INTERVAL 1 DAY)
group by p.customer_id;
```

The execution plan shows the following steps:

- Limit: 200 row(s) (actual time=7.38..7.41 rows=200 loops=1)
- Sort with duplicate removal: 'concat(c.last_name, ' ', c.first_name)', 'sum(p.amount)' (actual time=7.38..7.39 rows=200 loops=1)
- Table scan on <temporary> (actual time=7.04..7.11 rows=391 loops=1)
- Aggregate using temporary table (actual time=7.04..7.04 rows=391 loops=1)
- Nested loop inner join (cost=507 rows=634) (actual time=0.0934..6.23 rows=634 loops=1)
- Index range scan on p using pay_date over ('2005-07-30 00:00:00' <= payment_date < '2005-07-31 00:00:00'), with index condition: ((
- Single-row index lookup on c using PRIMARY (customer_id=p.customer_id) (cost=0.25 rows=1) (actual time=0.00191..0.00195 rows=1 loops=1)

The results pane shows the following data:

| concat(c.last_name, ' ', c.first_name) | sum(p.amount) |
|--|---------------|
| ADAMS KATHLEEN | 7.98 |
| ALEXANDER DIANA | 8.98 |
| ALLARD GORDON | 12.98 |
| ALLEN SHIRLEY | 0.99 |
| ALVAREZ CHARLENE | 16.98 |
| ANDERSON LISA | 16.98 |
| ARCE HARRY | 5.99 |
| ARCHULETA JORDAN | 5.98 |
| ARTIS CARL | 7.98 |
| AUSTIN ALMA | 5.99 |
| BAILEY MILDRED | 4.99 |
| BANDA EVERETT | 0.99 |
| BANKS JESSIE | 2.99 |

The status bar at the bottom indicates that 200 rows were retrieved in 10ms on 2023-07-25 at 10:27:20.