МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №4

по дисциплине «Базы данных»

Тема: Нагрузочное тестирование БД

Студент гр. 1303 _____ Бутыло Е.А.

Преподаватель _____ Заславский М.М.

Санкт-Петербург
2023

Цель работы.

Заполнить большим количеством тестовых данных, измерить время выполнения запросов. Измерить влияние (или его отсутствие) индексов на скорость выполнения запросов.

Текст задания

Вариант 3

1. Написать скрипт, заполняющий БД большим количеством тестовых данных.

2. Измерить время выполнения запросов, написанных в ЛР3.

Проверить для числа записей:

100 записей в каждой табличке

1.000 записей

10.000 записей

100.000 записей

1.000.000 записей

Все запросы выполнять с фиксированным ограничением на вывод (LIMIT), т.к. запросы без LIMIT всегда будет выполняться O(n) от кол-ва записей.

Для измерения использовать фактическое (не процессорное и т.п.) время. Для node.js есть console.time и console.timeEnd.

3. Добавить в БД индексы (хотя бы 5 штук). Измерить влияние (или его отсутствие) индексов на скорость выполнения запросов.

Выполнение работы

Для выполнения работы был написан код на языке Python, для создания файлов типа json, в которых будут данные для заполнения БД.

Для генерации данных использовалась библиотека faker.

После получения данных, для их корректного использования были разработаны контроллеры, для каждого отношения, получающего информацию из файла.

Закончив заполнения базы данных замерили время, необходимое для обработки запросов из лабораторной работы №3:

1. 100 записей - 20ms
2. 1.000 записей - 20ms
3. 10.000 записей - 27ms
4. 100.000 записей - 160ms
5. 1.000.000 записей - 1s

Далее задали индексы для полей отношений Schedule и Teacher, так как они используются чаще всего.

```
indexes: [
    {
        fields: ['orderNumber'],
    },
    {
        fields: ['className'],
    },
    {
        fields: ['subjectName'],
    },
    {
        fields: ['teacherId'],
    },
    {
        fields: ['cabinetNumber'],
    },
    {
        fields: ['day'],
    },
]
```

Рисунок 1. – Задали индексы для полей отношения Schedule.

Замерили время после добавления индексов и получили результаты, которые говорят о том, что использование индексов уменьшает время необходимое на обработку запросов к БД.

Результаты:

1. 100 записей - 20ms
2. 1.000 записей - 20ms
3. 10.000 записей - 27ms
4. 100.000 записей - 35ms
5. 1.000.000 записей - 100ms

Из результатов также видно, что при малых количествах данных, разница по времени в обработке запросов на уровне погрешности. Когда же кол-во данных переходит за десять тысяч, разница является значительной, особенно это заметно при обработке 1.000.000 записей, время сократилось значительно.

Выводы.

Заполнить большим количеством тестовых данных, измерили время выполнения запросов. Индексы значительно уменьшают время обработки запросов.

ПРИЛОЖЕНИЕ А

Pull request: https://github.com/moevm/sql-2023-1303/pull/51

ПРИЛОЖЕНИЕ Б

Файл lab2.sql:

```
CREATE OR REPLACE FUNCTION random_between(low INT, high INT)
    RETURNS INT AS
$$
BEGIN
    RETURN floor(random() * (high - low + 1) + low);
END;
$$ language 'plpgsql' STRICT;


CREATE TABLE class
(
    class_name TEXT PRIMARY KEY
);

CREATE TABLE subject
(
    subject_name TEXT PRIMARY KEY
);

CREATE TABLE cabinet
(
    cabinet_number SMALLINT PRIMARY KEY
);

CREATE TABLE teacher
(
    teacher_id  SMALLSERIAL PRIMARY KEY,
    second_name TEXT NOT NULL,
    first_name  TEXT NOT NULL,
    patronymic  TEXT NOT NULL
);

CREATE TABLE student
(
    student_id  SMALLSERIAL PRIMARY KEY,
    class_name  TEXT,
    second_name TEXT NOT NULL,
    first_name  TEXT NOT NULL,
    patronymic  TEXT NOT NULL,
    FOREIGN KEY (class_name) REFERENCES class (class_name) ON DELETE SET
NULL
);

CREATE TABLE grade
(
    student_id   SMALLINT NOT NULL,
    subject_name TEXT     NOT NULL,
    value        SMALLINT,
    PRIMARY KEY (student_id, subject_name),
    FOREIGN KEY (student_id) REFERENCES student (student_id) ON DELETE
CASCADE,
    FOREIGN KEY (subject_name) REFERENCES subject (subject_name) ON
DELETE CASCADE
);
```

```
CREATE TABLE teacher_cabinet
(
    teacher_id     SMALLINT PRIMARY KEY,
    cabinet_number SMALLINT NOT NULL,
    FOREIGN KEY (teacher_id) REFERENCES teacher (teacher_id) ON DELETE
CASCADE,
    FOREIGN KEY (cabinet_number) REFERENCES cabinet (cabinet_number) ON
DELETE CASCADE
);

CREATE TABLE school_schedule
(
    subject_schedule_id SERIAL PRIMARY KEY,
    order_number        SMALLINT NOT NULL,
    day                 TEXT     NOT NULL,
    class_name          TEXT     NOT NULL,
    subject_name        TEXT     NOT NULL,
    teacher_id          SMALLINT,
    cabinet_number      SMALLINT,
    FOREIGN KEY (class_name) REFERENCES class (class_name) ON DELETE
CASCADE,
    FOREIGN KEY (subject_name) REFERENCES subject (subject_name) ON
DELETE CASCADE,
    FOREIGN KEY (teacher_id) REFERENCES teacher (teacher_id) ON DELETE
SET NULL,
    FOREIGN KEY (cabinet_number) REFERENCES cabinet (cabinet_number) ON
DELETE SET NULL
);


INSERT INTO cabinet
    (cabinet_number)
VALUES (11),
       (12),
       (13), -- free
       (14),
       (15),
       (16),
       (17);

INSERT INTO teacher
    (second_name, first_name, patronymic)
VALUES ('Иванов', 'Иван', 'Иванович'),
       ('Забудь', 'Светлана', 'Ивановна'),
       ('Макаревич', 'Андрей', 'Игоревич'),
       ('Семашко', 'Зинаида', 'Климовна'), -- free
       ('Попова', 'Ульяна', 'Владиславовна'),
       ('Вечёрко', 'Анастасия', 'Дмитриевна'),
       ('Ожог', 'Николай', 'Витальевич');


INSERT INTO teacher_cabinet
    (teacher_id, cabinet_number)
VALUES (1, 11),
       (2, 12),
       (3, 14),
```

```
       (5, 15),
       (6, 16),
       (7, 17);

INSERT INTO class
    (class_name)
VALUES ('4В'),
       ('7А'),
       ('7Б'),
       ('9Б'),
       ('11А');

INSERT INTO student
    (class_name, second_name, first_name, patronymic)
VALUES ('4В', 'Степанов', 'Василий', 'Анатольевич'),
       ('4В', 'Авдеев', 'Георгий', 'Фёдорович'),
       ('7А', 'Виноградов', 'Анатолий', 'Алексеевич'),
       ('7А', 'Кузнецов', 'Илья', 'Михайлович'),
       ('7Б', 'Смирнов', 'Николай', 'Денисовчи'),
       ('7Б', 'Назарова', 'Мария', 'Михайловна'),
       ('9Б', 'Михеева', 'Елизавета', 'Алексеевна'),
       ('9Б', 'Ковалева', 'Анастасия', 'Сегреевна'),
       ('11А', 'Анисимов', 'Ярослав', 'Юрьевич'),
       ('11А', 'Нечаева', 'Светлана', 'Генадьевна');


INSERT INTO subject
    (subject_name)
VALUES ('Математика'),
       ('Физика'),
       ('Русский язык'),
       ('География'),
       ('Химия'),
       ('Биология'),
       ('Физкультура');


INSERT INTO grade
    (student_id, subject_name)
SELECT grade_temp.student_id, grade_temp.subject_name
FROM (student
    CROSS JOIN subject) AS grade_temp;

UPDATE grade
SET value = random_between(2, 10)
WHERE value IS NULL;


INSERT INTO school_schedule
(order_number, day, class_name, subject_name, teacher_id, cabinet_number)
VALUES (1, 'Понедельник', '4В', 'Математика', 1, 17),
       (1, 'Вторник', '4В', 'Физика', 2, 12),
       (1, 'Среда', '4В', 'Математика', 1, 17),
       (2, 'Среда', '4В', 'Физика', 1, 12),
       (3, 'Среда', '4В', 'Физкультура', 5, 16),
       (4, 'Среда', '4В', 'География', 3, 14),
       (5, 'Среда', '4В', 'Химия', 4, 12),
```

```
        (1, 'Четверг', '4В', 'Русский язык', 7, 11),
        (1, 'Пятница', '4В', 'Математика', 1, 17),
        (1, 'Среда', '9Б', 'Математика', 1, 13);


-- Какой предмет будет в заданном классе, в заданный день недели на
заданном уроке?
SELECT subject_name AS subject
FROM school_schedule
WHERE class_name = '4B'
  AND day = 'Среда'
  AND order_number = 2;

-- Кто из учителей преподает в заданном классе?
SELECT DISTINCT concat_ws(' ', second_name, first_name, patronymic) AS
teacher
FROM teacher
         INNER JOIN school_schedule ON teacher.teacher_id =
school_schedule.teacher_id
WHERE class_name = '4B';

-- В каком кабинете будет 5-й урок в среду у некоторого класса?
SELECT cabinet_number AS cabinet
FROM school_schedule
WHERE order_number = 5
  AND day = 'Среда'
  AND class_name = '4B';

-- В каких классах преподает заданный предмет заданный учитель?
SELECT DISTINCT class_name
FROM school_schedule
         INNER JOIN teacher ON teacher.teacher_id =
school_schedule.teacher_id
WHERE subject_name = 'Математика'
  AND (second_name = 'Иванов'
    AND first_name = 'Иван'
    AND patronymic = 'Иванович');


-- Расписание на заданный день недели для указанного класса?
SELECT order_number, subject_name AS subject, concat_ws(' ', second_name,
first_name, patronymic) AS teacher
FROM school_schedule
         INNER JOIN teacher ON teacher.teacher_id =
school_schedule.teacher_id
WHERE day = 'Среда'
  AND class_name = '4B'
ORDER BY order_number;

-- Сколько учеников в указанном классе?
SELECT COUNT(*) AS count_of_students
FROM student
WHERE class_name = '4B';
```