Лабораторная работа 17

ПСКП

ПОИТ-3

**Задание 01**

1. Ознакомьтесь с литературой [*https://diskstation.belstu.by:5001*](https://diskstation.belstu.by:5001) *Для\_студентов\_ФИТ\_БГТУ\ЛИТЕРАТУРА\REDIS*

[*http://onreader.mdl.ru/Redis4xCookbook/content/index.html*](http://onreader.mdl.ru/Redis4xCookbook/content/index.html)

1. Установите СУБД Redis на локальном компьютере или воспользуйтесь облачным хранилищем (например, [*https://redislabs.com*](https://redislabs.com) )
2. Установите npm-пакет redis.
3. Разработайте приложение 17-01.js, тестирующее соединение с сервером базы данных Redis.

**Задание 02**

1. Разработайте приложение 17-02.js, позволяющее исследовать скорость выполнения 10000 запросов **set,** 10000 запросов **get** и10000 запросов **del**.
2. Заполните таблицу

|  |  |  |
| --- | --- | --- |
| № | Операция | Выполнение 10000 операций  (мс) |
| 1 | *set(n, ‘set****n****’)*  **n** = 1…10000 |  |
| 2 | *get(n)*  **n** = 1…10000 |  |
| 3 | *del(n)*  **n** = 1…10000 |  |

**Задание 03**

1. Разработайте приложение 17-03.js, позволяющее исследовать скорость выполнения 10000 запросов **incr** и 10000 запросов **decr**.
2. Заполните таблицу

|  |  |  |
| --- | --- | --- |
| № | Операция  set(‘incr’,0) | Выполнение  10000 операций  (мс) |
| 1 | *incr(‘incr’)* |  |
| 2 | *decr((‘incr’)* |  |

**Задание 04**

1. Разработайте приложение 17-04.js, позволяющее исследовать скорость выполнения 10000 запросов **hset** и 10000 запросов **hget**.
2. Заполните таблицу

|  |  |  |
| --- | --- | --- |
| № | Операция  set(‘incr’,0) | Выполнение  10000 операций  (мс) |
| 1 | *hset(****n****, ‘{id:n,val:”val-****n****”}’),*  **n** = 1…10000 |  |
| 2 | *hget(n),*  **n** = 1…10000 |  |

**Задание 05**

1. Разработайте приложение 17-05.js, демонстрирующее механизм **publish/subscribe.**

**Задание 06** Ответьте на следующие вопросы

1. Поясните понятие **in-memory Database**.

База данных в памяти (англ. In-memory database, IMDB) — это система управления базами данных, которая при хранении компьютерных данных в основном полагается на оперативную память. Она выделяется на фоне СУБД, в которых используется механизм дискового хранения. Базы данных в памяти быстрее, чем базы данных, оптимизированные для использования дисковых накопителей, поскольку доступ к диску медленнее, чем доступ к памяти, а внутренние алгоритмы оптимизации проще и выполняют меньше инструкций ЦП. Доступ к данным из памяти исключает время на поиск при их запросе, что обеспечивает более быструю и предсказуемую работу, чем при использовании дискового накопителя.

1. Поясните понятие **хэш-таблица**.

Хеш-таблица (hash table) — это специальная структура данных для хранения пар ключей и их значений. По сути это ассоциативный массив, в котором ключ представлен в виде хеш-функции.

1. Каким образом обеспечивается персистентность данных в СУБД Redis?

Redis хранит данные в оперативной памяти, но тем не менее может оставаться персистентным. Персистентность означает сохранение состояния после завершения процесса (например, после отключения питания компьютера). В Redis персистентность обеспечивается сохранением данных на диск.

Первый способ — использовать RDB-снапшоты. Вы настраиваете периодичность, с которой происходит сохранение данных на диск. Когда такой момент наступает, Redis делает форк, то есть создает дочерний процесс. Этот процесс начинает записывать состояние базы данных во временный файл в формате RDB. Как только запись завершается, временный файл заменяет старую версию снапшота.

Второй подход — дозапись в конец файла. Redis будет записывать все изменения, происходящие с базой данных в Append Only File. Для включения этой функции необходимо указать в конфиге appendonly yes. Кроме того можно настроить частоту записи: каждую операцию (appendfsync always) или каждую секунду (appendfsync everysec). Первый вариант крайне медленный, но в случая отказа даже самые свежие изменения будут сохранены на диск.

1. Поясните назначение команд СУБД Redis: **set**, **get**, **del, getset**.

Для привязки строкового значения к некоторому ключу воспользуйтесь командой **SET**.

Выборку имеющегося строкового значения можно сделать просто воспользовавшись командой **GET**:

удалять ключи из Redis при помощи соответствиующей команды **DEL** getset устанавливает новое значение и возвращает старое

1. Поясните назначение команд СУБД Redis: **incr**, **decr**.

**Redis DECR** command is used to **decrement** the integer value of a key by one. If the key does not exist, it is set to 0 before

1. Поясните назначение команд СУБД Redis: **mset, mget**.

Мы можем выполнять установку и выборку множества ключей за раз применяя **MSET** и **MGET**. Основное преимущество операции **MSET** состоит в том, что эта операция целиком атомарная, что означает, что все значения ключей устанавливаются за один проход от клиента к серверу. Таким образом мы можем избегать излишних накладных расходов в сети применяя команду **MSET** вместо исполнения множества команд **SET**.

1. Поясните назначение команд СУБД Redis: **hset, hget.**

**HSET**, которая устанавливает необходимое значение для какого- то отдельного поля

Когда вы используете hset, вы сохраняете данные в хеше (hash)

1. Поясните назначение команд СУБД Redis: **hmset,hmget.**

 HMSET для хешей, для установки множественных пар внутри объекта значения хеша

1. Поясните назначение команды СУБД Redis: **exists**.

команда EXISTS выполняет проверку на наличие существующего ключа