Нереляционные базы данных

Лабораторная работа 6

Производительность и инструментарий

В этой лабораторной работе мы коснёмся некоторых вопросов производительности, а также рассмотрим инструментарий, доступный разработчикам MongoDB. Мы не станем сильно погружаться в эти темы, но рассмотрим наиболее важные аспекты каждой.

Индексы

В самом начале мы видели коллекцию system.indexes, которая содержит информацию о всех индексах в нашей базе данных. Индексы в MongoDB работают схожим образом с индексами в реляционных базах данных: они ускоряют выборку и сортировку данных. Индексы создаются с помощью ensureIndex:

```
db.unicorns.ensureIndex({name: 1});
```

И уничтожаются с помощью dropIndex:

```
db.unicorns.dropIndex({name: 1});
```

Уникальный индекс может быть создан, если во втором параметре установить unique вtrue:

```
db.unicorns.ensureIndex({name: 1}, {unique: true});
```

Можно создавать индексы над вложенными полями (опять же, используя точечную нотацию), либо над массивами. Также можно создавать составные индексы:

```
db.unicorns.ensureIndex({name: 1, vampires: -1});
```

Порядок вашего индекса (1 для восходящего и -1 для нисходящего) не играет роли в случае с простым индексом, однако он может быть существенен при сортировке или лимитировании с применением составных индексов.

На странице описания индексов можно найти дополнительную информацию.

Explain

Чтобы увидеть, используются ли индексы в ваших запросах, вызывайте у курсора методexplain:

```
db.unicorns.find().explain()
```

В результате мы видим информацию, что использовался BasicCursor (то есть не индексированный), сканирование происходило по 12 объектам, как много это времени заняло, применялся ли индекс, и если да, то какой, а также прочие полезные сведения.

Если мы изменим запрос так, чтобы он использовал индекс, мы увидим, что использовался курсор BtreeCursor, а также увидим индекс, использованный при выборке:

```
db.unicorns.find({name: 'Pilot'}).explain()
```

Запись без подтверждения

Мы уже упоминали, что запись данных в MongoDB происходит без подтверждения. Это может привести к приросту производительности, равно как и к риску потери данных в результате случайной ошибки. Возникает также побочный эффект, выражающийся в том, что когда обновление или вставка нарушают условие уникальности индекса, ошибки не происходит. Чтобы узнать о возникновении ошибки, после последней записи нужно вызывать db.getLastError(). Многие драйверы берут эту работу на себя, предоставляя возможность *безопасной* записи — часто для этого имеется специальный параметр.

К сожалению, консоль не умеет этого делать, и пронаблюдать это в консоли будет непросто.

Шардинг

MongoDB поддерживает авто-шардинг. Шардинг — это подход к масштабируемости, когда отдельные части данных хранятся на разных серверах. Примитивный пример — хранить данные пользователей, чьё имя начинается на буквы A-M на одном сервере, а остальных — на другом. Возможности шардинга MongoDB значительно превосходят данный простой пример. Рассмотрение шардинга выходит за пределы данной книги, однако вы должны знать, что он существует, и вы должны воспользоваться им, когда ваши задачи выйдут за рамки одного сервера.

Репликация

Репликация в MongoDB работает сходным образом с репликацией в реляционных базах данных. Записи посылаются на один сервер — ведущий

(master), который потом синхронизирует своё состояние с другими серверами — ведомыми *(slave)*. Вы можете разрешить или запретить чтение с ведомых серверов, в зависимости от того, допускается ли в вашей системе чтение несогласованных данных. Если ведущий сервер падает, один из ведомых может взять на себя роль ведущего. Репликация MongoDB также выходит за пределы данной книги.

Хотя репликация увеличивает производительность чтения, делая его распределённым, основная её цель — увеличение надёжности. Типичным подходом является сочетание репликации и шардинга. Например, каждый шард может состоять из ведущего и ведомого серверов. (Технически, вам также понадобится арбитр, чтобы разрешить конфликт, когда два ведомых сервера пытаются объявить себя ведущими. Но арбитр потребляет очень мало ресурсов и может быть использован для нескольких шардов сразу.)

Статистика

Статистику базы данных можно получить с помощью вызова db.stats(). В основном информация касается размера вашей базы данных. Также можно получить статистику коллекции, например unicorns, с помощью вызова db.unicorns.stats(). Большая часть получаемой информации, опять же, касается размеров коллекции.

Веб-интерфейс

Когда mongod запускается, в консоли появляется, среди прочих, строчка со ссылкой на административный веб-интерфейс. Вы можете получить к нему доступ, зайдя в браузере на http://localhost:28017/. Чтобы получить от него максимальную отдачу, можете добавить rest=true в конфигурационный файл и перезапустить процесс mongod. Веб-интерфейс даёт много интересной информации о текущем состоянии сервера.

Профайлер

Профайлер MongoDB можно включить с помощью следующего вызова:

```
db.setProfilingLevel(2);
```

Со включённым профайлером можно запустить команду:

```
db.unicorns.find({weight: {$gt: 600}});
```

И обратиться к профайлеру:

db.system.profile.find()

В результате мы увидим, что и когда запускалось, как много документов сканировалось, как много данных было возвращено.

Можно выключить профайлер, повторно вызвав setProfileLevel, только передав 0 в качестве аргумента. Можно также передать 1для профилирования запросов, выполняющихся дольше 100 миллисекунд. Также, можно вторым параметром передать время в миллисекундах:

```
//профилировать всё, что занимает более 1 секунды
db.setProfilingLevel(1, 1000);
```

Резервное копирование и восстановление

В папке bin MongoDB есть утилита mongodump. После выполнения mongodump произойдёт подключение к localhost и резервное копирование всех баз данных в подпапку dump. Можно набрать mongodump --help и увидеть дополнительные опции. Распространённые опции: --db DBNAME для резервного копирования только указанной базы данных и--collection COLLECTIONAME для резервного копирования только указанной коллекции. После этого можно использовать mongorestore, расположенный в той же папке bin, чтобы восстановить базу данных из предварительно сделанной резервной копии. Здесь также можно указать --db и --collection, чтобы восстановить только указанные базу данных и коллекцию.

Например, чтобы сделать резервную копию базы данных learn в папку backup, мы должны выполнить (разумеется не в консоли самой MongoDB, а просто в консоли операционной системы):

mongodump --db learn --out backup

Чтобы восстановить только коллекцию unicorns мы должны сделать следующее:

mongorestore --collection unicorns backup/learn/unicorns.bson

Также, стоит упомянуть, что есть две утилиты mongoexport и mongoimport, предназначенные для экспорта и импорта данных в виде JSON и CSV. Например, можно получить результат в виде JSON следующим образом:

mongoexport --db learn -collection unicorns

И CSV:

mongoexport --db learn -collection unicorns --csv -fields name,weight,vampires

Имейте в виду, что mongoexport и mongoimport не могут полностью отражать ваши данные. Только mongodump и mongorestore должны использоваться для настоящего резервного копирования.

Выводы к лабораторной работе

В этой лабораторной работе мы рассмотрели различные команды, инструменты и нюансы производительности MongoDB. Мы коснулись не всех тем, однако рассмотрели наиболее распространённые. Индексирование в MongoDB похоже на индексирование в реляционных базах данных, то же касается большинства инструментария. Однако в MongoDB пользоваться всем намного проще.

Требования к сдаче лабораторной работы

1. Отчет к лабораторной работе, содержащий пошаговое исполнение заданий и промежуточные результаты

Задания к лабораторной работе

1. Продемонстрируйте различные комбинации индексов в MongoDB на примере сортировки. Продемонстрируйте работу уникального индекса.
2. Продемонстрируйте работу explain для индексируемых и неиндексируемых коллекций.
3. Продемонстрируйте сбор статистики для базы данных и коллекции.
4. Изучите веб-интерфейс MongoDB, сделайте вывод о возможностях его применения.
5. Продемонстируйте работу профайлера для коллекций из задания 2.
6. Сделайте резервную копию одной из коллекций. Удалите ее и восстановите из резервной копии.
7. Продемонстрируйте экспорт и импорт коллекций в JSON и CSV.