**Лабораторная работа № 2**

**ПРОДОЛЖАЕМ РАБОТУ С MONGODB**

**Цель работы:** изучить основы работы с коллекциями в документоориентированной СУБД *MongoDB*, научиться проектировать триггеры.

**Теоретические сведения:**

MongoDB — документоориентированная система управления базами данных, не требующая описания схемы таблиц. Считается одним из классических примеров NoSQL-систем, использует JSON-подобные документы и схему базы данных. Написана на языке C++. Применяется в веб-разработке, в частности, в рамках JavaScript-ориентированного стека MEAN.

Система поддерживает ad-hoc-запросы: они могут возвращать конкретные поля документов и пользовательские JavaScript-функции. Поддерживается поиск по регулярным выражениям. Также можно настроить запрос на возвращение случайного набора результатов.

Имеется поддержка индексов.

Система может работать с набором реплик, то есть содержать две или более копии данных на различных узлах. Каждый экземпляр набора реплик может в любой момент выступать в роли основной или вспомогательной реплики. Все операции записи и чтения по умолчанию осуществляются с основной репликой. Вспомогательные реплики поддерживают в актуальном состоянии копии данных. В случае, когда основная реплика дает сбой, набор реплик проводит выбор, которая из реплик должна стать основной. Второстепенные реплики могут дополнительно являться источником для операций чтения.

Система масштабируется горизонтально, используя технику сегментирования (англ. sharding) объектов баз данных — распределение их частей по различным узлам кластера. Администратор выбирает ключ сегментирования, который определяет, по какому критерию данные будут разнесены по узлам (в зависимости от значений хэша ключа сегментирования). Благодаря тому, что каждый узел кластера может принимать запросы, обеспечивается балансировка нагрузки.

**Задание к лабораторной работе:**

1. Резервное копирование базы данных MongoDB;
2. Создать запрос к 2-ум таблицам;

**Выполнение работы:**

**Резервное копирование базы данных MongoDB**

Важным аргументом команды mongodump является –db; он указывает имя БД, бэкап которой нужно создать. Аргумент –out задаёт каталог для хранения дампа. Давайте попробуем создать резервную копию БД newdb и поместить её в каталог /var/backups/mongobackups. В идеале каждая резервная копия должна храниться в каталоге с текущей датой (к примеру, /var/backups/mongobackups/04-20-16 – копия за 20 апреля 2016). Создайте каталог:

sudo mkdir /var/backups/mongobackups

Теперь запустите команду для создания бэкапа:

sudo mongodump --db newdb --out /var/backups/mongobackups/`date +"%m-%d-%y"`

В случае успешного копирования команда вернёт:

2016-01-20T10:11:57.685-0500    writing newdb.restaurants to /var/backups/mongobackups/01-20-16/newdb/restaurants.bson  
2016-01-20T10:11:57.907-0500    writing newdb.restaurants metadata to /var/backups/mongobackups/01-20-16/newdb/restaurants.metadata.json  
2016-01-20T10:11:57.911-0500    done dumping newdb.restaurants (25359 documents)  
2016-01-20T10:11:57.911-0500    writing newdb.system.indexes to /var/backups/mongobackups/01-20-16/newdb/system.indexes.bson

Эта часть команды date +”%m-%d-%y”. Она автоматически устанавливает текущую дату. Это позволяет поместить копию в соответствующий каталог; эта функция особенно удобна при автоматическом резервном копировании.

Итак, у вас есть полная резервная копия БД newdb в каталоге /var/backups/mongobackups/01-20-16/newdb/. Эта копия позволяет полностью восстановить базу данных newdb, сохранив точность данных.

Как правило, резервные копии нужно создавать регулярно (например, на ежедневной основе); предпочтительно это делать во время низкой нагрузки на сервер. Для этого можно автоматизировать бэкап, поместив команду mongodump в cron. Откройте кронтаб:

sudo crontab -e

Команда sudo crontab откроет для редактирования кронтаб пользователя root. Рекомендуется использовать именно этот кронтаб, поскольку таблицы других пользователей могут работать некорректно (например, демон cron может потребовать пароль). К примеру, чтобы демон создавал копию каждый день в 03:03 AM, нужно добавить в кронтаб:

3 3 \* \* \* mongodump --out /var/backups/mongobackups/`date +"%m-%d-%y"`

В этой команде намеренно опущен аргумент –db, вследствие чего команда будет копировать все существующие базы данных.

Стоит отметить, что ежедневное резервное копирование относительно быстро потребляет дисковое пространство (в зависимости от объёма БД MongoDB). Потому рекомендуется регулярно удалять или сжимать старые копии. К примеру, чтобы удалить все копии старше 5 дней, можно использовать команду:

find /var/backups/mongobackups/ -mtime +5 -exec rm -rf {} \;

Эту команду можно также добавить в cron. Она должна запускаться незадолго до начала резервного копирования, например, в 03.01. Откройте кронтаб:

sudo crontab -e

Вставьте в таблицу следующую строку:

3 1 \* \* \* find /var/backups/mongobackups/ -mtime +7 -exec rm -rf {} \;

**Запрос с 2-м таблицам ([Aggregation](https://cloud.mongodb.com/v2/602f92a8a37906728a8186bc" \l "metrics/replicaSet/60547d65ffc6c96a654ffde7/explorer/sample_analytics/accounts/aggregation))**

**$lookup**- Выполняет левое внешнее соединение с необработанной коллекцией в той же базе данных для фильтрации документов из «объединенной» коллекции для обработки. К каждому входному документу этап $ lookup добавляет новое поле массива, элементы которого являются совпадающими документами из «объединенной» коллекции. Этап $lookup передает эти измененные документы на следующий этап.

Синтаксис запроса $lookup

{

$lookup:

{

from: <коллекция для присоединения>,

localField: <поле из входных документов>,

foreignField: <поле из документов коллекции "from">,

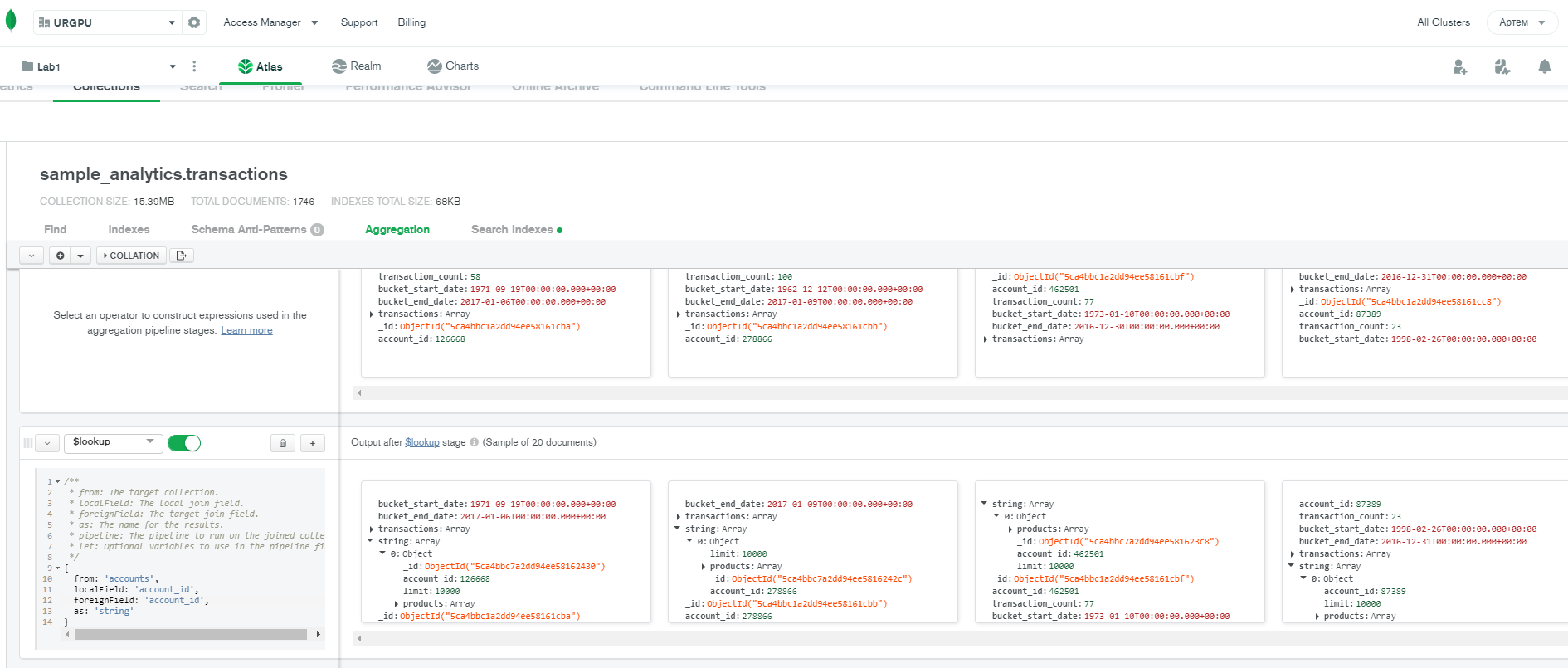
as: <поле массива вывода>

}

}

1) Создаем новую базу данных и 2 новых коллекции в ней

2) Во вкладке Aggregations выполним запрос ко второй коллекции и получим объединение (Рисунок 1).

  
Рисунок 1

***Вывод:***

Изучили резервное копирование базы данных MongoDB, научились создавать запрос к 2-ум таблицам в *MongoDB*.