

ЛАБОРАТОРНАЯ РАБОТА 5.1

ВВЕДЕНИЕ В СУБД MONGODB. УСТАНОВКА MONGODB. НАЧАЛО РАБОТЫ С БД

Выполнил: РЫБАЛКО ОЛЕГ К32392

Цель: овладеть практическими навыками установки СУБД MongoDB.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД MongoDB 4+, 6.0.6 (текущая)

Практическое задание:

1. Установка

Для установки `mongodb` мы воспользуемся Docker и конфигурацией `docker compose`. Для запуска базы данных необходимо выполнить команду *`docker compose up`*

`docker-compose.yml`:

`version: '3.9'`

`services:`

`mongodb:`

`image: mongo:latest`

`container_name: mongodb`

`restart: unless-stopped`

`environment:`

`MONGO_INITDB_ROOT_USERNAME: root`

`MONGO_INITDB_ROOT_PASSWORD: password`

`ports:`

`- 27017:27017`

`volumes:`

`- ./database-data:/data/db`

2. Проверка работоспособности

Для работы с базой данных будем использовать инструмент *mongosh*. После выполнения команды *mongosh* мы можем увидеть, что мы успешно подключились к базе данных. Также в логах нашей базы данных можно увидеть, что успешно было установлено подключение

```
> mongosh
Current Mongosh Log ID: 64775780865572fb0940b6d9
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.9.1
Using MongoDB:      6.0.6
Using Mongosh:       1.9.1

For mongosh info see: https://docs.mongodb.com/mongodb-shell/
```

Рисунок 1 - Вывод после вызова команды "mongosh"

```
mongod | {"t":{"date":"2023-05-31T14:24:12.402+00:00"},"s":"I", "c":"NETWORK", "id":22943, "ctx":"listener","msg":"Connection accepted","attr":{"remote":"172.20.0.1:57798","uid":"3448c956-4c3d-47d7-ab6b-adbf8f5c6922","connectionId":2,"connectionCount":2}}
mongod | {"t":{"date":"2023-05-31T14:24:12.404+00:00"},"s":"I", "c":"NETWORK", "id":51800, "ctx":"conn2","msg":"client metadata","attr":{"remote":"172.20.0.1:57798","client":"conn2","doc":{"application":{"name":"mongosh 1.9.1"},"driver":{"name":"nodejs|mongosh","version":"5.5.0|1.9.1"},"platform":"Node.js v16.20.0, LE","os":{"name":"darwin","architecture":"arm64","version":"22.3.0","type":"Darwin"}}}}
```

Рисунок 2 - логи базы данных

3. Выполнение методов

1. db.help()

```
Database Class:

getMongo                Returns the current database connection
getName                Returns the name of the DB
getCollectionNames      Returns an array containing the names of all collections in the current database.
getCollectionInfos      Returns an array of documents with collection information, i.e. collection name and options, for the current database.
runCommand              Runs an arbitrary command on the database.
adminCommand            Runs an arbitrary command against the admin database.
aggregate               Runs a specified admin/diagnostic pipeline which does not require an underlying collection.
getSiblingDB            Returns another database without modifying the db variable in the shell environment.
getCollection            Returns a collection or a view object that is functionally equivalent to using the db.<collectionName>.
dropDatabase            Removes the current database, deleting the associated data files.
createUser              Creates a new user for the database on which the method is run. db.createUser() returns a duplicate user error if the user already exists on the database.
updateUser              Updates the user's profile on the database on which you run the method. An update to a field completely replaces the previous field's values. This includes updates to the user's roles array.
changeUserPassword      Updates a user's password. Run the method in the database where the user is defined, i.e. the database you created the user.
logout                  Ends the current authentication session. This function has no effect if the current session is not authenticated.
dropUser                Removes the user from the current database.
dropAllUsers            Removes all users from the current database.
auth                   Allows a user to authenticate to the database from within the shell.
grantRolesToUser         Grants additional roles to a user.
revokeRolesFromUser      Removes a one or more roles from a user on the current database.
getUser                  Returns user information for a specified user. Run this method on the user's database. The user must exist on the database on which the method runs.
getUsers                 Returns information for all the users in the database.
createCollection         Create new collection
createEncryptedCollection Creates a new collection with a list of encrypted fields each with unique and auto-created data encryption keys (DEKs). This is a utility function that internally utilises ClientEncryption.createEncryptedCollection.
createView               Create new view
```

Рисунок 3 - вывод команды db.help()

2. db.stats()

Для выполнения данной команды нам необходимо подключиться к базе данных с логином и паролем

```
> mongosh "mongodb://root:password@127.0.0.1"
Current Mongosh Log ID: 647759dac95545bb73b59b90
Connecting to:      mongodb://<credentials>@127.0.0.1/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.9.1
Using MongoDB:      6.0.6
Using Mongosh:      1.9.1

For mongosh info see: https://docs.mongodb.com/mongosh-shell/
```

Рисунок 4 - подключение с паролем

```
test> db.stats()
{
  db: 'test',
  collections: 0,
  views: 0,
  objects: 0,
  avgObjSize: 0,
  dataSize: 0,
  storageSize: 0,
  indexes: 0,
  indexSize: 0,
  totalSize: 0,
  scaleFactor: 1,
  fsUsedSize: 0,
  fsTotalSize: 0,
  ok: 1
}
```

Рисунок 5 - вывод функции db.stats()

4. Создание новой БД

Для создания новой базы данных воспользуемся функцией use

```
test> use learn
switched to db learn
learn>
```

Рисунок 6 - вывод функции use

5. Получение списка доступных БД

Для получения списка доступных баз данных воспользуемся командой админа *listDatabases*. Вместо единицы в качестве значения можно использовать любой поддерживаем тип данных в mongo, так как это значение не влияет на результат выполнения команды.

```
learn> db.adminCommand(
...   {
...     listDatabases: 1
...   }
... )
{
  databases: [
    { name: 'admin', sizeOnDisk: Long("102400"), empty: false },
    { name: 'config', sizeOnDisk: Long("110592"), empty: false },
    { name: 'local', sizeOnDisk: Long("73728"), empty: false }
  ],
  totalSize: Long("286720"),
  totalSizeMb: Long("0"),
  ok: 1
}
```

Рисунок 7 - вывод команды для списка бд

6. Создание новой коллекции и вставка данных

Воспользуемся функцией *db.createCollection(<name>)* для создания коллекции под названием “unicorns”. Затем вставим в нее новый документ *{name: 'Aurora', gender: 'f', weight: 450}*, для этого воспользуемся функцией *insert* у коллекции *unicorns*

```
learn> db.createCollection("unicorns")
{ ok: 1 }
learn> db.unicorns.insert({name: 'Aurora', gender: 'f', weight: 450})
DeprecationWarning: Collection.insert() is deprecated. Use insertOne, insertMany, or bulkWrite.
{
  acknowledged: true,
  insertedIds: { '0': ObjectId("64775cbdc95545bb73b59b91") }
}
```

Рисунок 8 - создание коллекции и вставка документа

7. Просмотр списка текущих коллекций

Для просмотра списка текущих коллекций мы можем воспользоваться методом `db.runCommand({ listCollections: 1 })`

```
learn> db.runCommand({ listCollections: 1 })
{
  cursor: {
    id: Long("0"),
    ns: 'learn.$cmd.listCollections',
    firstBatch: [
      {
        name: 'unicorns',
        type: 'collection',
        options: {},
        info: {
          readOnly: false,
          uuid: new UUID("8af95163-1164-400f-83e1-62005df1b6dd")
        },
        idIndex: { v: 2, key: { _id: 1 }, name: '_id_' }
      }
    ]
  },
  ok: 1
}
learn>
```

Рисунок 9 - список текущих коллекций

8. Переименование коллекции

Для переименования коллекции `unicorns` воспользуемся методом `db.collection_name.renameCollection(<newName>)`

```
learn> db.unicorns.renameCollection("unicorns_new")
{ ok: 1 }
learn>
```

Рисунок 10 - переименование коллекции `unicorns`

9. Статистика коллекции

Для просмотра статистики коллекции `unicorns_new` воспользуемся функцией `db.unicorns_new.stats()`

```
learn> db.unicorns_new.stats()
{
  ok: 1,
  capped: false,
  wiredTiger: {
    metadata: { formatVersion: 1 },
    creationString: 'access_pattern_hint=none,allocation_size=4KB,app_metadata=(formatVersion=1),assert=(commit_timestamp=none,durable_timestamp=none,read_timestamp=none,write_timestamp=off),block_allocation=best,block_compressor=snappy,cache_resident=false,checksum=on,colgroups=,collator=,columns=,dictionary=0,encryption=(keyid=,name=),exclusive=false,extractor=,format=btree,huffman_key=,huffman_value=,ignore_in_memory_cache_size=false,immutable=false,import=(compare_timestamp=oldest_timestamp,enabled=false,file_metadata=,metadata_file=,repair=false),internal_item_max=0,internal_key_max=0,internal_key_truncate=true,internal_page_max=4KB,key_format=q,key_gap=10,leaf_item_max=0,leaf_key_max=0,leaf_page_max=32KB,leaf_value_max=64MB,log=(enabled=true),lsm=(auto_throttle=true,bloom=true,bloom_bit_count=16,bloom_config=,bloom_hash_count=8,bloom_oldest=false,chunk_count_limit=0,chunk_max=5GB,chunk_size=10MB,merge_custom=(prefix=,start_generation=0,suffix=),merge_max=15,merge_min=0),memory_page_image_max=0,memory_page_max=10m,os_cache_dirty_max=0,os_cache_max=0,prefix_compression=false,prefix_compression_min=4,readonly=false,source=,split_deepen_min_child=0,split_deepen_per_child=0,split_pct=90,tiered_object=false,tiered_storage=(auth_token=,bucket=,bucket_prefix=,cache_directory=,local_retention=300,name=,object_target_size=0),type=file,value_format=u,verbose=[],write_timestamp_usage=none',
    type: 'file',
    uri: 'statistics:table:collection-0--1043455439032787910',
    LSM: {
      'bloom filter false positives': 0,
      'bloom filter hits': 0,
      'bloom filter misses': 0,
      'bloom filter pages evicted from cache': 0,
      'bloom filter pages read into cache': 0,
      'bloom filters in the LSM tree': 0,
      'chunks in the LSM tree': 0,
      'highest merge generation in the LSM tree': 0,
      'queries that could have benefited from a Bloom filter that did not exist': 0,
      'sleep for LSM checkpoint throttle': 0,
      'sleep for LSM merge throttle': 0,
      'total size of bloom filters': 0
    }
  },
  autocommit: {
    'retries for readonly operations': 0,
  }
}
```

Рисунок 11 - статистика коллекции unicorns_new

10. Удаление коллекции и БД

Для удаления коллекции воспользуемся методом `db.unicorns_new.drop()`.

Для удаления базы данных воспользуемся методом `db.dropDatabase()`

```
learn> db.unicorns_new.drop()
true
learn> db.dropDatabase()
{ ok: 1, dropped: 'learn' }
learn>
```

Рисунок 12 - результаты удаления

11. Вывод

В ходе выполнения данной лабораторной работы была установлена база данных mongodb, инструмент для работы с базой данных mongosh и были выполнены различные функции в базе. В итоге удалось овладеть навыками установки mongodb