Лабораторная работа 15

ПСКП

ПОИТ-3

**Задание 01**

1. Зарегистрируйтесь в облаке [https://www.mongodb.com](https://www.mongodb.com/)
2. Создайте базу данных(БД) с именем **BSTU**
3. Создайте в БД коллекции: **faculty**, **pulpit**.
4. **faculty**

****

1. **pulpit**

****

**Задание 02**

1. Разработайте приложение **15-01**, представляющее собой HTTP-сервер, прослушивающий порт ***3000***. Сервер должен обрабатывать запросы, описанные в следующих таблицах.

|  |  |  |
| --- | --- | --- |
| Метод  запроса | URI | Назначение |
| GET | api/faculties | получить список всех факультетов в json-формате |
| GET | api /pulpits | получить список всех кафедр в json-формате |
| POST | api/faculties | добавить новый факультет, данные в json-формате |
| POST | api/pulpits | добавить новую кафедру, данные в json-формате |
| PUT | api/faculties | корректировать информацию о факультете, данные в json-формате |
| PUT | api/pulpits | корректировать информацию о кафедре, данные в json-формате |
| DELETE | api/faculties/xyz | xyz - код факультета, удалить факультет с кодом xyz |
| DELETE | api/pulpits/xyz | xyz - код кафедры, удалить кафедру с кодом xyz |

|  |  |
| --- | --- |
| Метод  запроса | Назначение |
| GET | найденные в БД данные или сообщение об ошибке в json-формате |
| POST | принимает данные в json-формате,  возвращает добавленные в БД данные или сообщение об ошибке в json-формате |
| PUT | принимает данные в json-формате,  возвращает измененные в БД данные или сообщение об ошибке в json-формате |
| DELETE | возвращает удаленные в БД данные или сообщение об ошибке в json-формате |

1. Для проверки работоспособности приложения, используйте **POSTMAN**.

**Задание 04** Ответьте на следующие вопросы

1. Опишите структуру БД, поддерживаемую СУБД MangoDB.

Если в реляционных бд содержимое составляют таблицы, то в mongodb база данных состоит из **коллекций**.



В отличие от реляционных баз данных MongoDB не использует табличное устройство с четко заданным количеством столбцов и типов данных. MongoDB является документо-ориентированной системой, в которой центральным понятием является **документ**.

1. Перечислите все функции API СУБД MangoDB с помощью которых можно извлечь данные из БД.

Наиболее простой способом получения содержимого БД представляет использование функции find. Действие этой функции во многом аналогично обычному запросу SELECT \* FROM Table, который извлекает все строки. Например, чтобы извлечь все документы из коллекции users, созданной в прошлой теме, мы можем использовать команду db.users.find().

Выведем все документы, в которых name=Tom:

db.users.find({name: "Tom"})

1. Перечислите все функции API СУБД MangoDB с помощью которых можно добавить данные в БД.

Для добавления в коллекцию могут использоваться три ее метода:

* **insertOne()**: добавляет один документ
* **insertMany()**: добавляет несколько документов
* **insert()**: может добавлять как один, так и несколько документов

db.users.insertOne({"name": "Tom", "age": 28, languages: ["english", "spanish"]})

db.users.insert({"name": "Tom", "age": 28, languages: ["english", "spanish"]})

1. Перечислите все функции API СУБД MangoDB с помощью которых можно удалить данные в БД.

Для удаления документов в MongoDB предусмотрен метод remove:

db.users.remove({name : "Tom"})

Для удаления коллекций используется функция **drop**:

db.users.drop()

Чтобы удалить всю базу данных, надо воспользоваться функцией **dropDatabase()**:

db.dropDatabase()

1. Перечислите все функции API СУБД MangoDB с помощью которых можно изменить данные в БД.

Наиболее простым для использования является метод save. В качестве параметра этот метод принимает документ.

db.users.save({name: "Eugene", age : 29, languages: ["english", "german", "spanish"]})

Более детальную настройку при обновлении предлагает функция **update**. Она принимает три параметра:

* query: принимает запрос на выборку документа, который надо обновить
* objNew: представляет документ с новой информацией, который заместит старый при обновлении
* options: определяет дополнительные параметры при обновлении документов. Может принимать два аргумента: upsert и multi.

Если параметр upsert имеет значение true, что mongodb будет обновлять документ, если он найден, и создавать новый, если такого документа нет. Если же он имеет значение false, то mongodb не будет создавать новый документ, если запрос на выборку не найдет ни одного документа.

Параметр multi указывает, должен ли обновляться первый элемент в выборке (используется по умолчанию, если данный параметр не указан) или же должны обновляться все документы в выборке.

db.users.update({name : "Tom"}, {name: "Tom", age : 25}, {upsert: true})

Теперь документ, найденный запросом {name : "Tom"}, будет перезаписан документом {"name": "Tom", "age" :"25"}.

Часто не требуется обновлять весь документ, а только значение одного из его ключей. Для этого применяется оператор **$set**. Если документ не содержит обновляемое поле, то оно создается.

db.users.update({name : "Tom", age: 29}, {$set: {age : 30}})

Если обновляемого поля в документе нет, до оно добавляется

Метод **updateOne** похож на метод update за тем исключением, что он обновляет только один документ

Если необходимо обновить все документы, соответствующие некоторому критерию, то применяется метод **updateMany()**