MONGO DB

Persistent Data for NodeJS

https://docs.mongodb.org/manual/

First, a glance of what you know

- אם נרצה ליצור טבלה בSQL עבור משתמשים בשרת שלנו, איך היינו עושים את זה?

```
CREATE TABLE Users
(
        UserID int,
        LastName varchar(255),
        FirstName varchar(255),
        Email varchar(255),
        Password varchar(255)
);
```

וזה עוד החלק הקל בSQL.

השאלה המתבקשת היא?

אין איזו דרך יותר קלה ותכנותית לעבוד עם מסדי נתונים

noSQL, none relational Db

- כיום בפלטפורמות רבות, נהוג להשתמש במסדי נתונים שאינם ראציונלים ואינם SQLים
- הסיבות הן פשוטות:
  - *רובנו לא משתמשים במסד נתונים כל כך גדול שאנחנו צריכים את היכולות של* SQL
  - *רובנו לא באמת עושים חתכים מיוחדים בנתונים ולא מערבבים בינהם הרבה*
  - *כולנו רוצים דרך נוחה וקלה יותר לבנות את מסד הנתונים*
  - *נעדיף שזה ירגיש כמו תכנות, ויהיה חלק מהשפה, עם אותו הסינטקס ואותה ההתנהגות*
- ולכן, כולנו משתמשים במסדי נתונים שאינם SQL לרוב, כמו MongoDb

שיטת עבודה

- Install mongo
- How To Use Mongo?
  - *You can use the shell*
  - *You can use in the code*
  - *You can use with additional graphical apps*
- How to do it right?
  - *Test your idea in the shell on local Db*
    - Works? Right it in the code!
  - *Have a bug in the code?*
    - Test the code in the shell to see why the bugs keep coming

Now and Then..

- כאשר עובדים עם SQL, אנחנו מוגבלים לפי טבלה.
  - *לכל טבלה יש מבנה משלה*
  - *כל נתון שנכניס חייב לעמוד בתנאי*
- לכן המון פעמים אנחנו נתקלים במצבים בהם אנחנו יוצרים יותר מטבלה אחת על מנת לייצג אוסף נתונים אחד

Documents

- במקום להציג נתון כשורה בטבלה, יהיה לנו נוח יותר לעבוד עם אובייקטים
- מסמכים הם אובייקטים במונגו, המייצגים נתון.
- מסמכים נותנים לנו יותר חופש עבודה ונוחות בתכנות
- מסמך יכול להיות קיים לפני שיש בכלל טבלה לאחסן אותו בתוכה

Collections

אם כבר הפכנו את הנתון בטבלה לנוח, למה שנמשיך לעבוד עם טבלאות?! -- כל מתכנת עם ראש

- במקום טבלאות, אנחנו עובדים עם אוספים.
- אוסף מייצג בשבילנו מאגר נתונים עם אופי אחיד. כמו רשימת משתמשים
- אוסף יכול להכיל מסמכים
- אין שום הגבלה או התנייה למבנה המסמכים, אוסף יכול הלכי מסמכים עם מבנה שונה
- את פעולות החיפוש, סינון, וכל השאר נבצע על אוסף

SQL vs Mongo

פעולה	SQL	Mongo
שמירת נתון	שורה בטבלה	אובייקט
שמירת אוסף נתונים	טבלה	אוסף אובייקטים
חיפוש נתון באוסף	ביצוע פעולה מתמטית על הטבלה	Find()
יצירת נתון חדש	קודם צריך טבלה	יוצרים אובייקט
הגבלות באוסף נתונים	על כל הנתונים בטבלה להיות באותו מבנה	כל אובייקט באוסף יכול להיות עם המבנה שמתאים לו

Operations in Mongo

Operation	Code
Create Collection (table in SQL) “users”	db.createCollection(“users”,<options>)
Create data	Well, just create an object
Add data to collection	db.users.insert(<data>)
Find data in collection	db.users.find(<data in object>)
Get all data from a collection	db.users.find()
data object type	BSON
What a data can have inside	Numbers, strings, dates, lists, object, and much more

Operations in Mongo

Operation	Code
Update data in collection	db.users.update(<query param>, {<update param>:<data>}, <options>)
Update parameters	https://docs.mongodb.org/manual/reference/operator/update/#id1
Remove Document from a collection	db.users.remove({name:value})

Update method API: https://docs.mongodb.org/manual/reference/method/db.collection.update/

Update Parameters

Syntax	Operation
$set:{name:value}	Set name to be value if found
$inc:{name:step}	Increment name by step
$unset:{name:value}	Remove the name (value doesn't matter)
$rename:{name:value}	Change the key-name from name to value
$set:{name.**$**:value}	When value is as list, Will change the key matches to the query parameter **inside** value
$set:{name.**innerNamw**:value}	When value is an object, Will change the inner key **inside** value
$pop:{name:value}	Remove the value from the name list
$push:{name:value}	Add the value to the name list

More on update parameters: https://docs.mongodb.org/manual/reference/operator/update/#id1

Update options

Parameter	What it does
Upsert [Boolean]	will create a **new** document if update didn’t find anyone itself
Multi [Boolean]	Will update all the found docs, if false only the first found will be updated

Comparison Operators

- Syntax
  - *{name: {operatr: val, ...*
- Examples:
  - *$gt – greater than*
  - *$lt – less than*
  - *$eq – equals to*
  - *....*

https://docs.mongodb.org/manual/reference/operator/query/