SmartFit

Vladislav Mazur

Overview

- Intro
- DB Design
- Queries & Operations
- Patterns
- Wrap up

Background

Health & Wellness Application

Jack-Of-All-Traits

MyFitnessPal / BoostCamp / Google Sheets / Health / 18birdies / ETC…

Apple Health

Goal

One stop shop

Easy to insert

Quick Lookups

The why?

Personal Interest

Cluttered Phone

Non-Rigid Schema

Collections

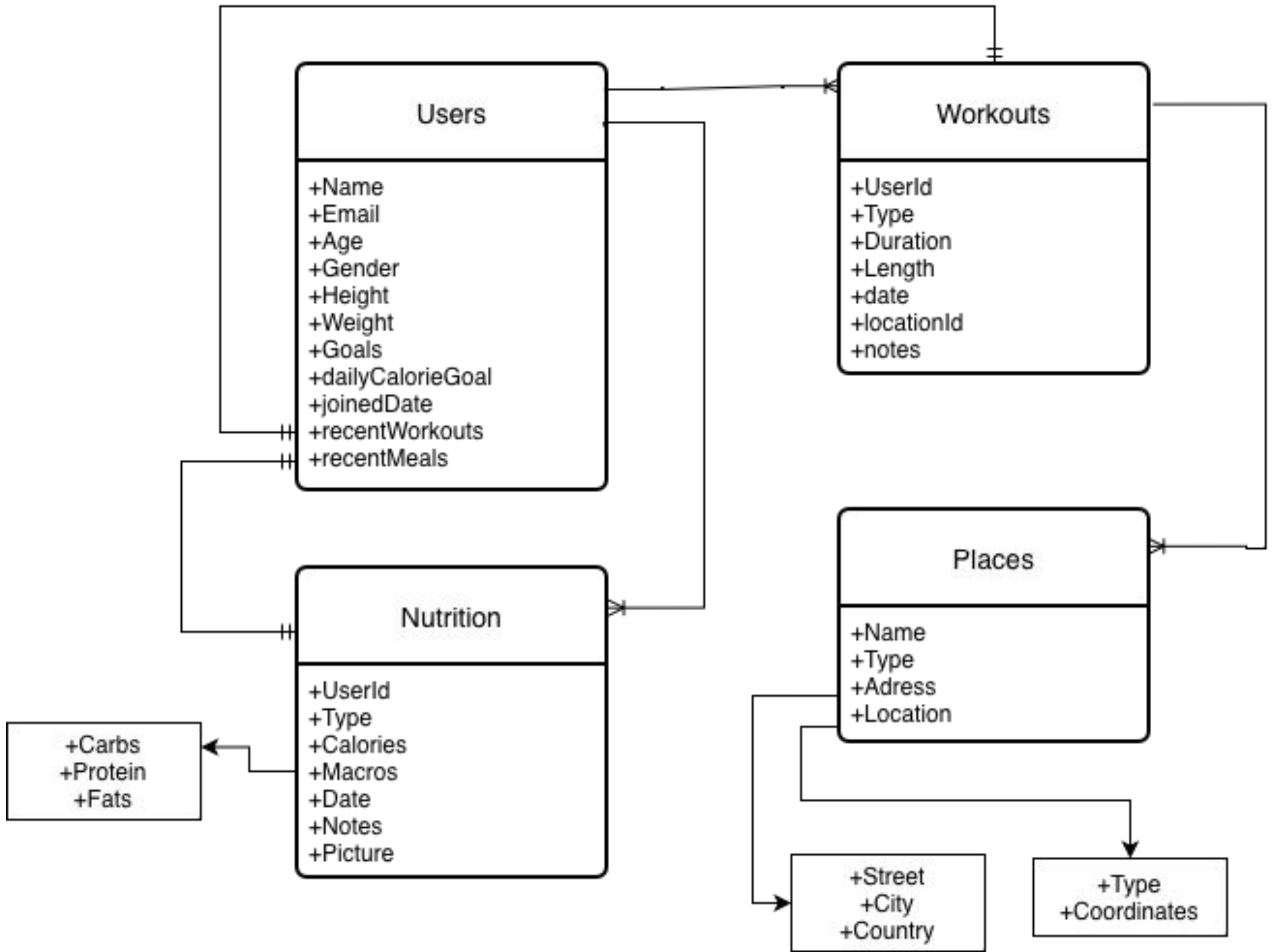
Users

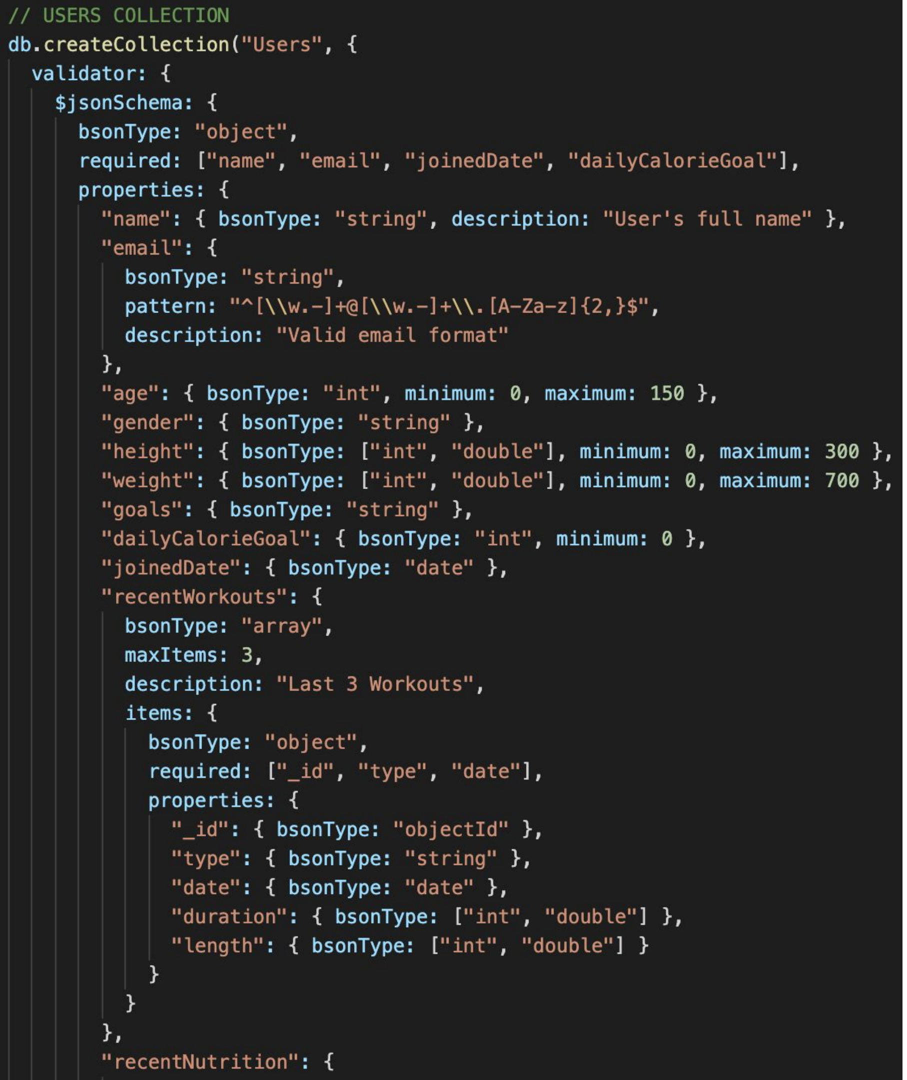
Workouts

Nutrition

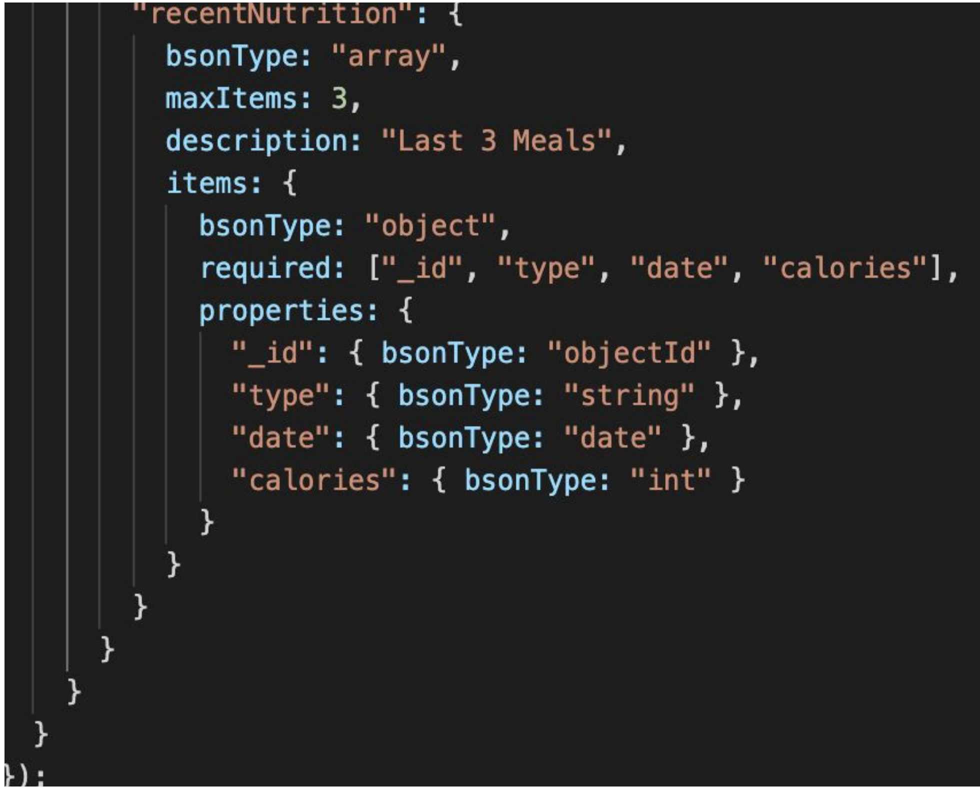
Places

CRD

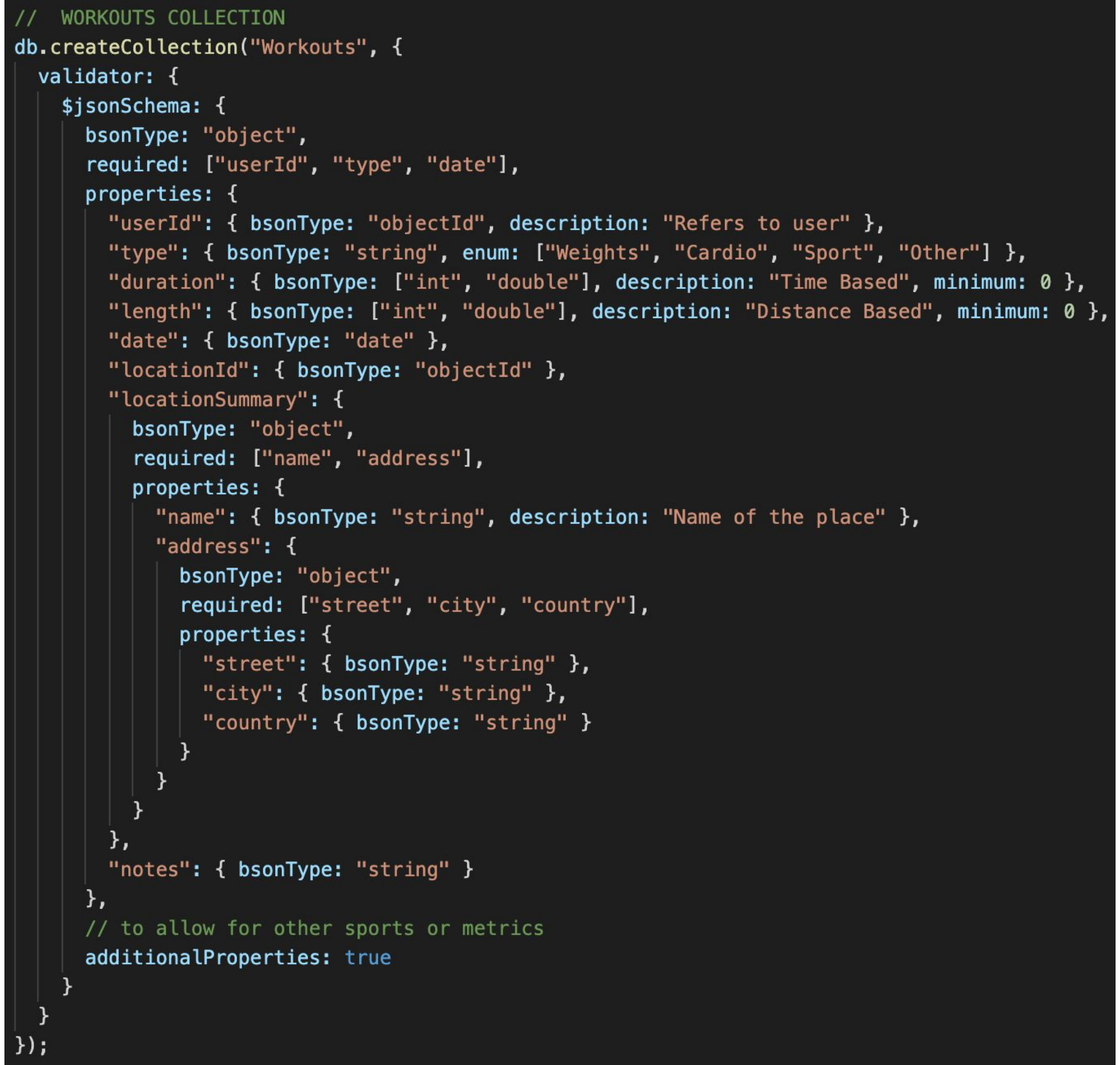

Users
+Name
+Email
+Age
+Gender
+Height
+Weight
+Goals
+dailyCalorieGoal
+joinedDate
+recentWorkouts
+recentMeals
Workouts
+UserId
+Type
+Duration
+Length
+date
+locationId
+notes
Nutrition
+UserId
+Type
+Calories
+Macros
+Date
+Notes
+Picture
+Carbs
+Protein
+Fats
Places
+Name
+Type
+Adress
+Location
+Street
+City
+Country
+Type
+Coordinates



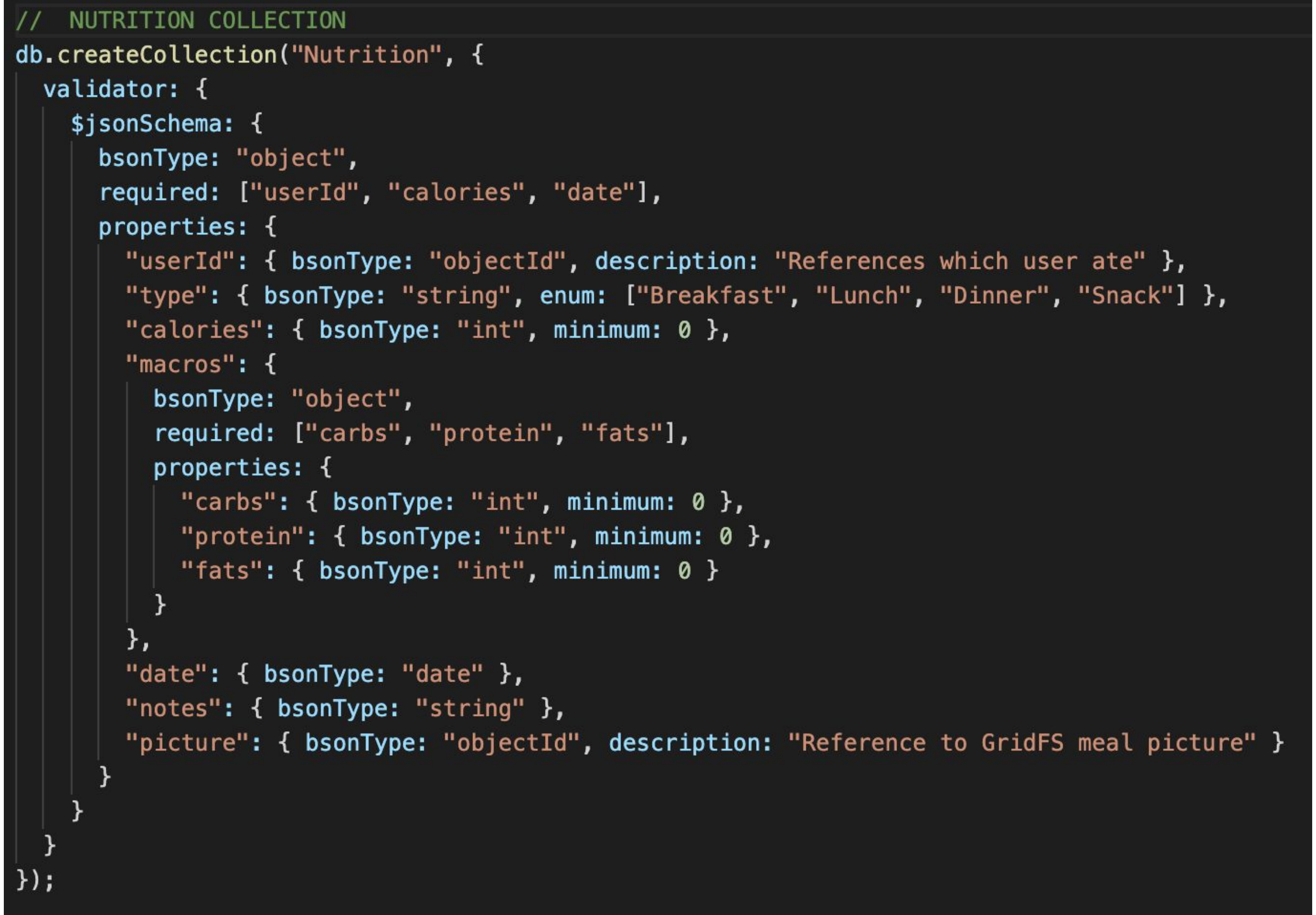
```
// USERS COLLECTION
db.createCollection("Users", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "email", "joinedDate", "dailyCalorieGoal"],
      properties: {
        "name": { bsonType: "string", description: "User's full name" },
        "email": {
          bsonType: "string",
          pattern: "^[\\w.-]+@[\\w.-]+\\.[A-Za-z]{2,}$",
          description: "Valid email format"
        },
        "age": { bsonType: "int", minimum: 0, maximum: 150 },
        "gender": { bsonType: "string" },
        "height": { bsonType: ["int", "double"], minimum: 0, maximum: 300 },
        "weight": { bsonType: ["int", "double"], minimum: 0, maximum: 700 },
        "goals": { bsonType: "string" },
        "dailyCalorieGoal": { bsonType: "int", minimum: 0 },
        "joinedDate": { bsonType: "date" },
        "recentWorkouts": {
          bsonType: "array",
          maxItems: 3,
          description: "Last 3 Workouts",
          items: {
            bsonType: "object",
            required: ["_id", "type", "date"],
            properties: {
              "_id": { bsonType: "objectId" },
              "type": { bsonType: "string" },
              "date": { bsonType: "date" },
              "duration": { bsonType: ["int", "double"] },
              "length": { bsonType: ["int", "double"] }
            }
          }
        },
        "recentNutrition": {
```



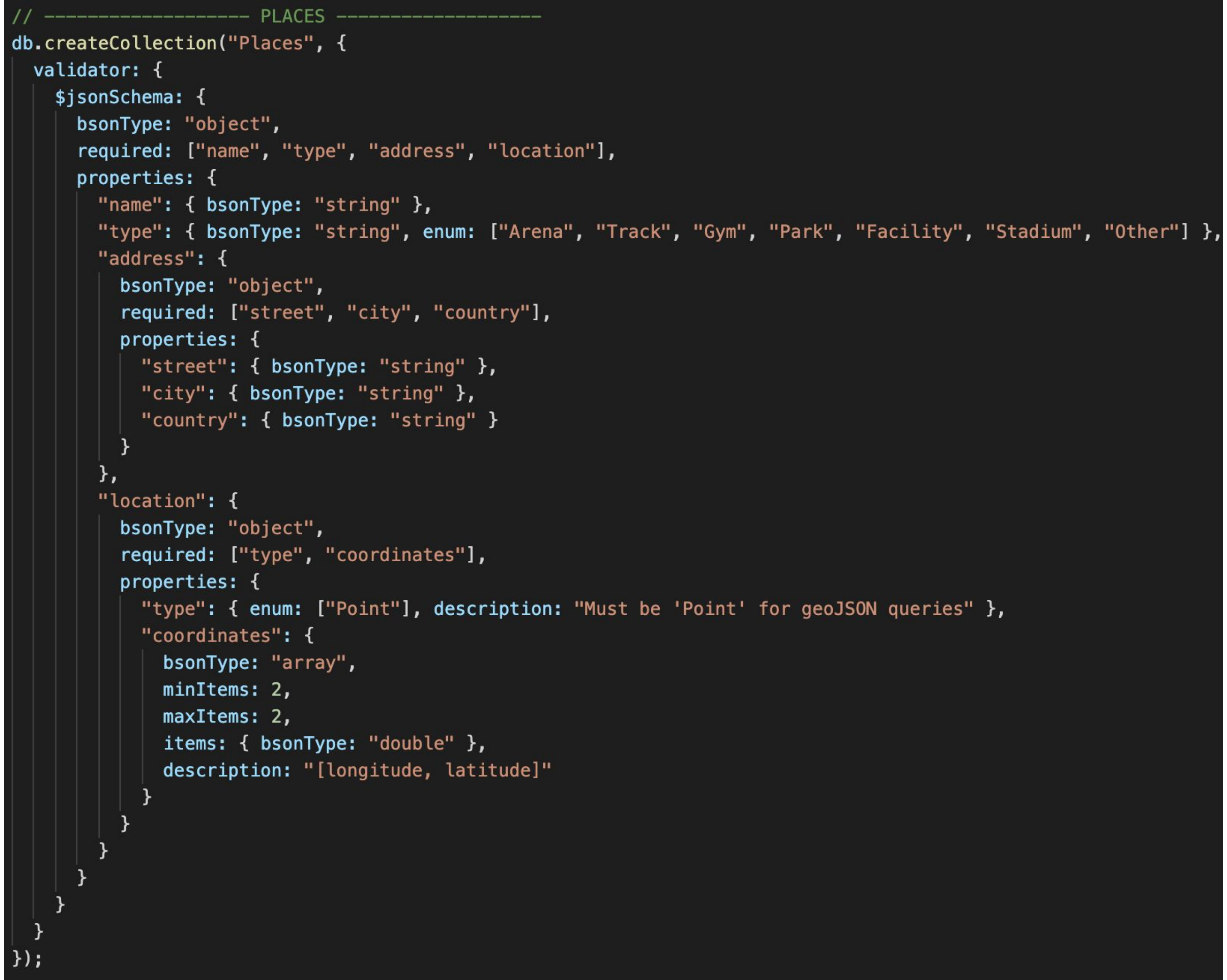
```
        "recentNutrition": {
          bsonType: "array",
          maxItems: 3,
          description: "Last 3 Meals",
          items: {
            bsonType: "object",
            required: ["_id", "type", "date", "calories"],
            properties: {
              "_id": { bsonType: "objectId" },
              "type": { bsonType: "string" },
              "date": { bsonType: "date" },
              "calories": { bsonType: "int" }
            }
          }
        }
      }
    }
  }
});
```



```
//  WORKOUTS COLLECTION
db.createCollection("Workouts", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["userId", "type", "date"],
      properties: {
        "userId": { bsonType: "objectId", description: "Refers to user" },
        "type": { bsonType: "string", enum: ["Weights", "Cardio", "Sport", "Other"] },
        "duration": { bsonType: ["int", "double"], description: "Time Based", minimum: 0 },
        "length": { bsonType: ["int", "double"], description: "Distance Based", minimum: 0 },
        "date": { bsonType: "date" },
        "locationId": { bsonType: "objectId" },
        "locationSummary": {
          bsonType: "object",
          required: ["name", "address"],
          properties: {
            "name": { bsonType: "string", description: "Name of the place" },
            "address": {
              bsonType: "object",
              required: ["street", "city", "country"],
              properties: {
                "street": { bsonType: "string" },
                "city": { bsonType: "string" },
                "country": { bsonType: "string" }
              }
            }
          }
        },
        "notes": { bsonType: "string" }
      },
      // to allow for other sports or metrics
      additionalProperties: true
    }
  }
});
```



```
//  NUTRITION COLLECTION
db.createCollection("Nutrition", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["userId", "calories", "date"],
      properties: {
        "userId": { bsonType: "objectId", description: "References which user ate" },
        "type": { bsonType: "string", enum: ["Breakfast", "Lunch", "Dinner", "Snack"] },
        "calories": { bsonType: "int", minimum: 0 },
        "macros": {
          bsonType: "object",
          required: ["carbs", "protein", "fats"],
          properties: {
            "carbs": { bsonType: "int", minimum: 0 },
            "protein": { bsonType: "int", minimum: 0 },
            "fats": { bsonType: "int", minimum: 0 }
          }
        },
        "date": { bsonType: "date" },
        "notes": { bsonType: "string" },
        "picture": { bsonType: "objectId", description: "Reference to GridFS meal picture" }
      }
    }
  }
});
```



```
// -------------------- PLACES --------------------
db.createCollection("Places", {
  validator: {
    $jsonSchema: {
      bsonType: "object",
      required: ["name", "type", "address", "location"],
      properties: {
        "name": { bsonType: "string" },
        "type": { bsonType: "string", enum: ["Arena", "Track", "Gym", "Park", "Facility", "Stadium", "Other"] },
        "address": {
          bsonType: "object",
          required: ["street", "city", "country"],
          properties: {
            "street": { bsonType: "string" },
            "city": { bsonType: "string" },
            "country": { bsonType: "string" }
          }
        },
        "location": {
          bsonType: "object",
          required: ["type", "coordinates"],
          properties: {
            "type": { enum: ["Point"], description: "Must be 'Point' for geoJSON queries" },
            "coordinates": {
              bsonType: "array",
              minItems: 2,
              maxItems: 2,
              items: { bsonType: "double" },
              description: "[longitude, latitude]"
            }
          }
        }
      }
    }
  }
});
```

Embedding

Recent Workout + Meals - User

Macros - Nutrition

Address - Places

Referencing

User Collection from Workouts

User Collection from Nutrition

Places Collection from Workouts

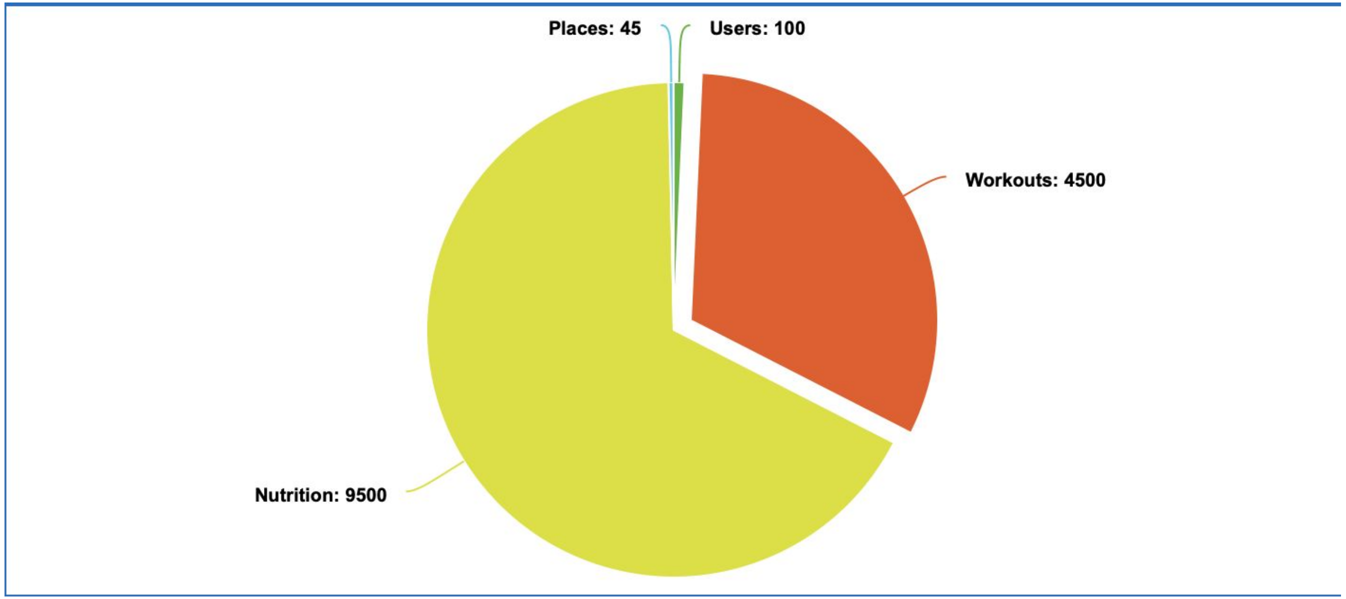
Data Set

Users - 100

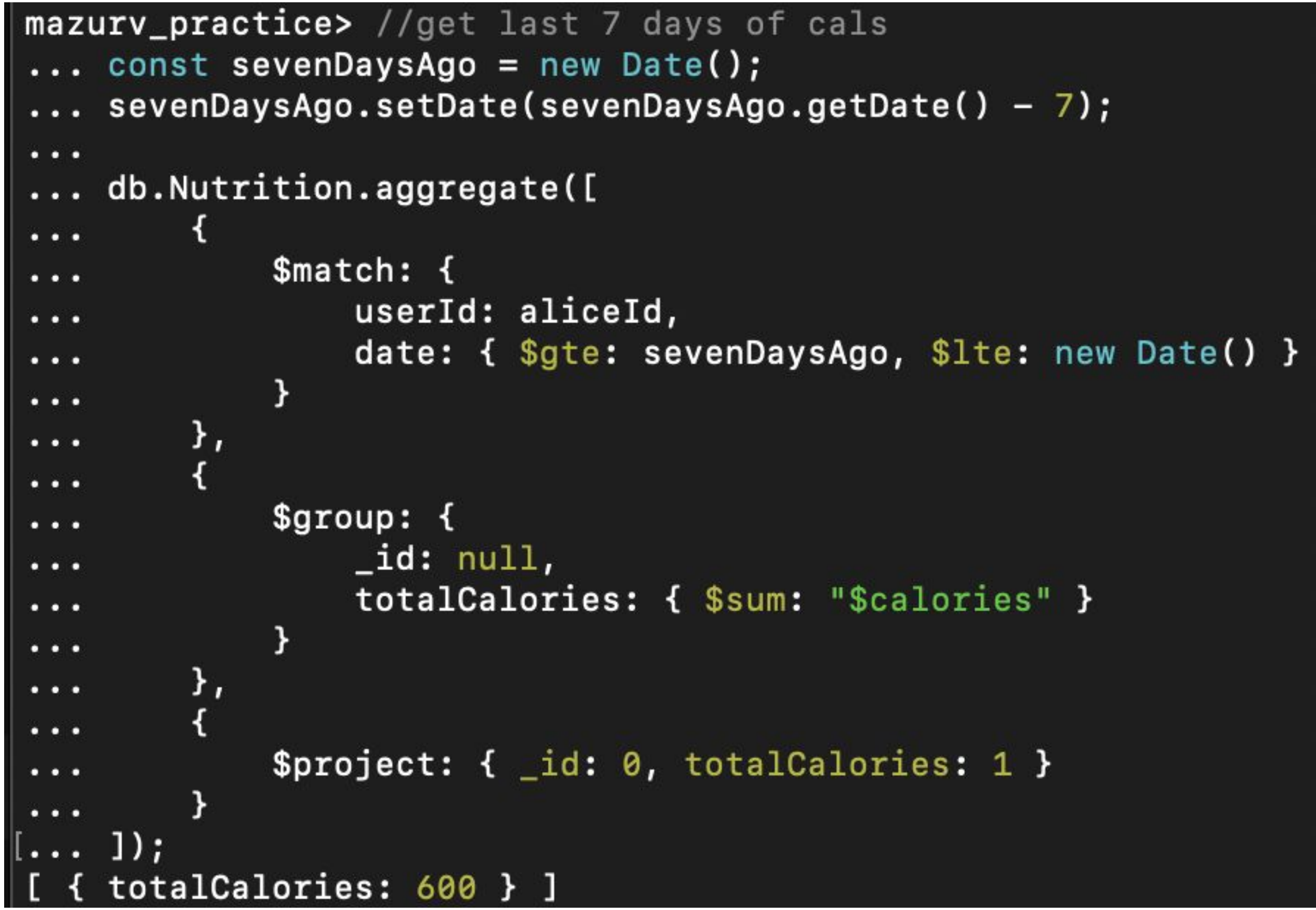
Workouts - 4.5k

Nutrition - 9.5k

Places - 45



Aggregation



```
mazurv_practice> //get last 7 days of cals
... const sevenDaysAgo = new Date();
... sevenDaysAgo.setDate(sevenDaysAgo.getDate() - 7);
...
... db.Nutrition.aggregate([
...     {
...         $match: {
...             userId: aliceId,
...             date: { $gte: sevenDaysAgo, $lte: new Date() }
...         }
...     },
...     {
...         $group: {
...             _id: null,
...             totalCalories: { $sum: "$calories" }
...         }
...     },
...     {
...         $project: { _id: 0, totalCalories: 1 }
...     }
[... ]);
[ { totalCalories: 600 } ]
```

Indexes

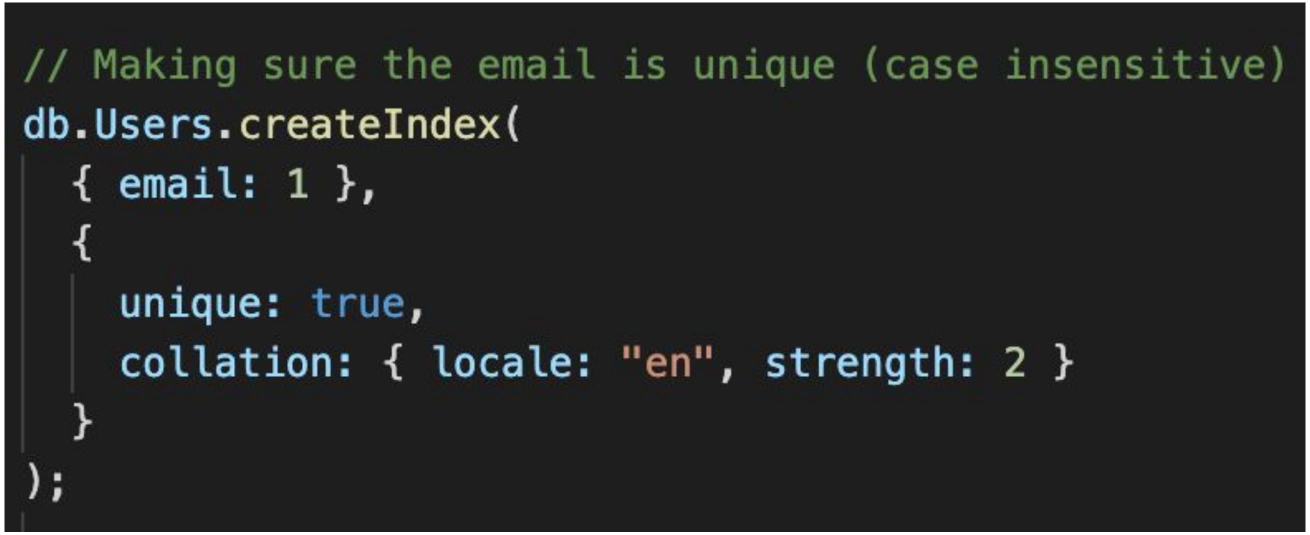
Most used queries:

- Fetch last __ workouts / meals
- Add workout / meal
- Increment / Decrement Weight

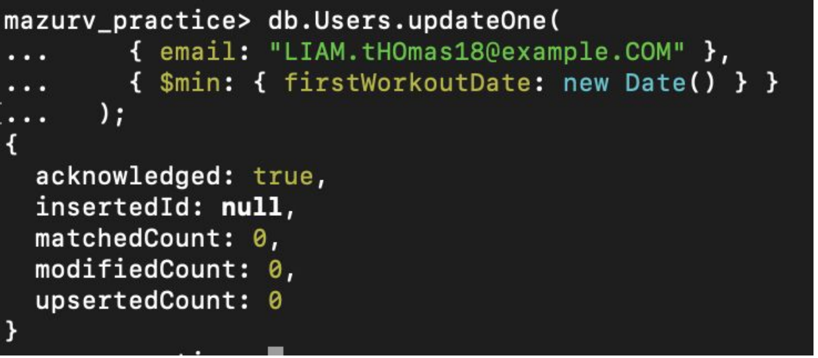
Unique

Email

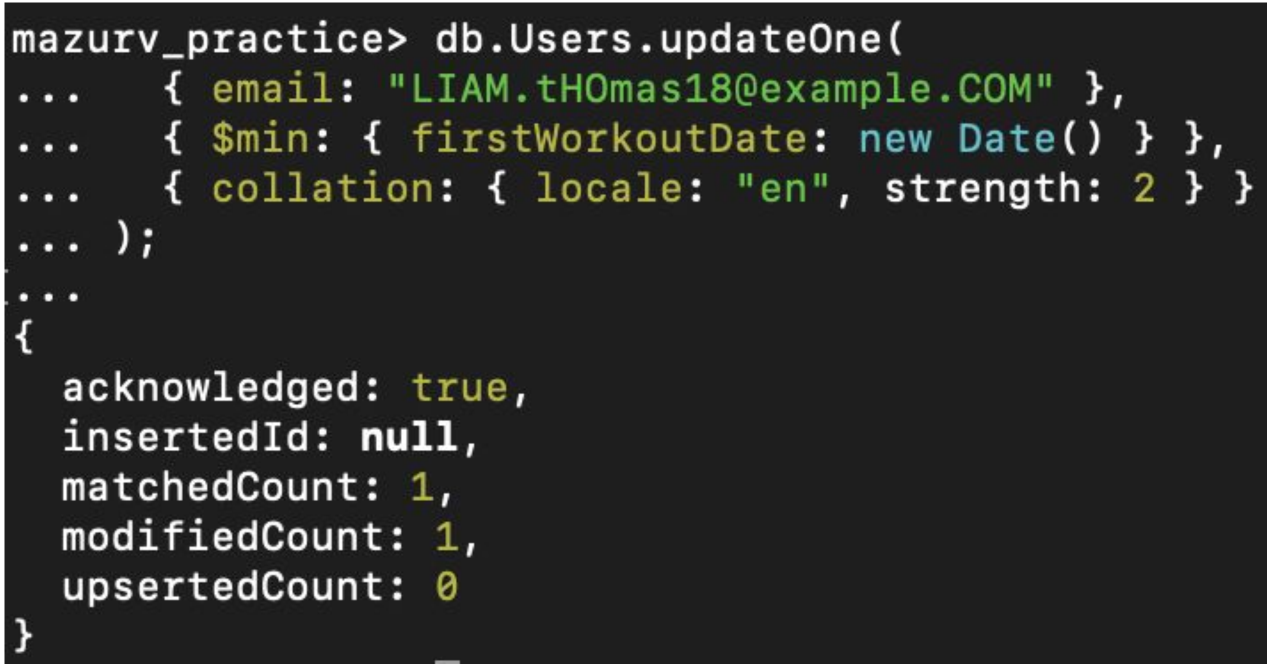
Collation Index /= Collation Find



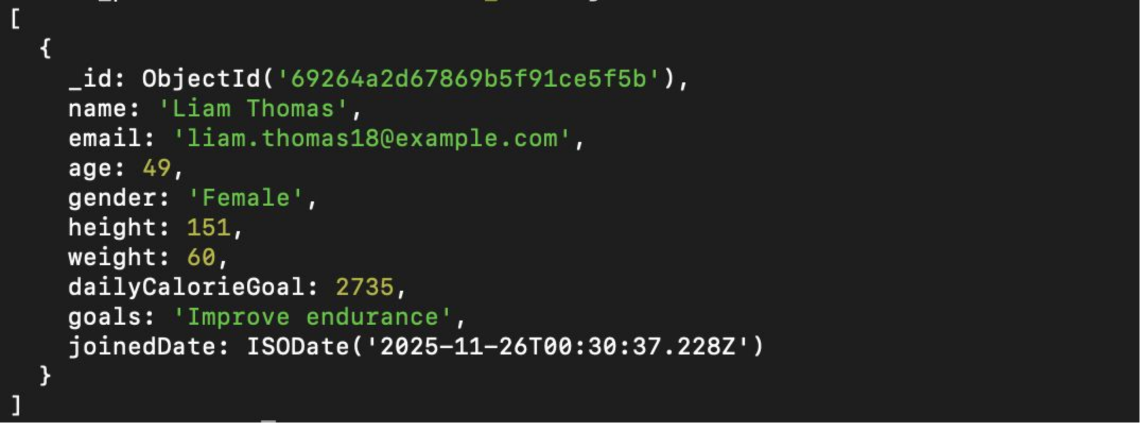
```
// Making sure the email is unique (case insensitive)
db.Users.createIndex(
  { email: 1 },
  {
    unique: true,
    collation: { locale: "en", strength: 2 }
  }
);
```



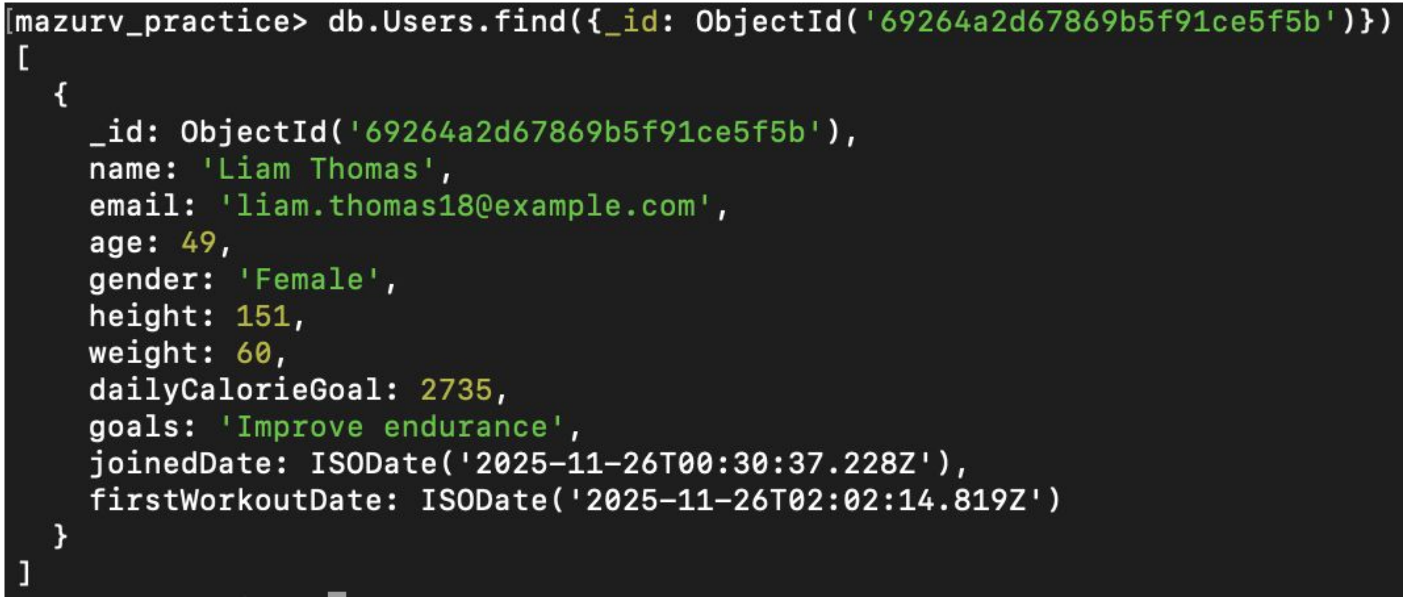
```
mazurv_practice> db.Users.updateOne(
...     { email: "LIAM.tHOmas18@example.COM" },
...     { $min: { firstWorkoutDate: new Date() } }
...   );
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 0,
  modifiedCount: 0,
  upsertedCount: 0
}
```



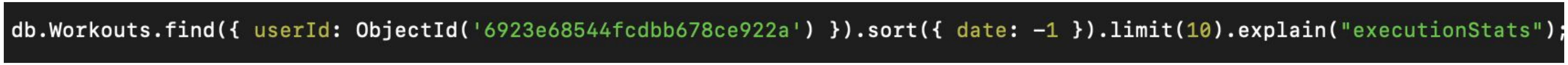
```
mazurv_practice> db.Users.updateOne(
...   { email: "LIAM.tHOmas18@example.COM" },
...   { $min: { firstWorkoutDate: new Date() } },
...   { collation: { locale: "en", strength: 2 } }
... );
...
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```



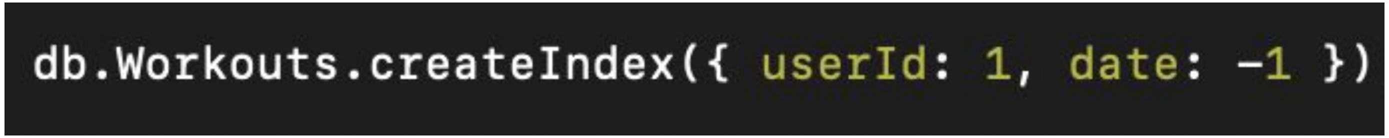
```
[
  {
    _id: ObjectId('69264a2d67869b5f91ce5f5b'),
    name: 'Liam Thomas',
    email: 'liam.thomas18@example.com',
    age: 49,
    gender: 'Female',
    height: 151,
    weight: 60,
    dailyCalorieGoal: 2735,
    goals: 'Improve endurance',
    joinedDate: ISODate('2025-11-26T00:30:37.228Z')
  }
]
```



```
mazurv_practice> db.Users.find({_id: ObjectId('69264a2d67869b5f91ce5f5b')})
[
  {
    _id: ObjectId('69264a2d67869b5f91ce5f5b'),
    name: 'Liam Thomas',
    email: 'liam.thomas18@example.com',
    age: 49,
    gender: 'Female',
    height: 151,
    weight: 60,
    dailyCalorieGoal: 2735,
    goals: 'Improve endurance',
    joinedDate: ISODate('2025-11-26T00:30:37.228Z'),
    firstWorkoutDate: ISODate('2025-11-26T02:02:14.819Z')
  }
]
```



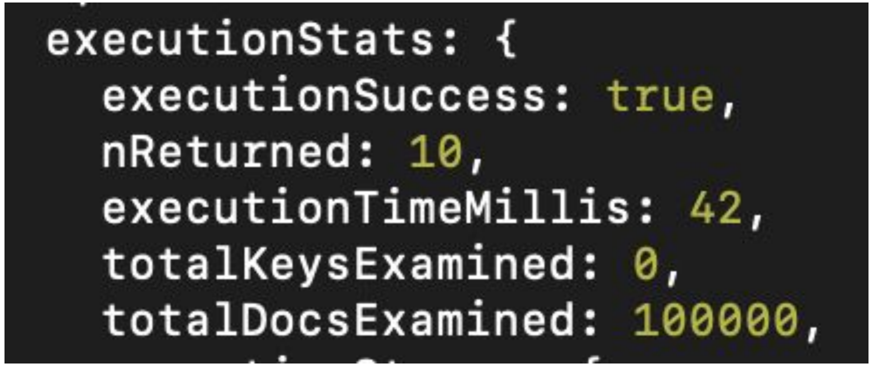
```
db.Workouts.find({ userId: ObjectId('6923e68544fcdbb678ce922a') }).sort({ date: -1 }).limit(10).explain("executionStats");
```



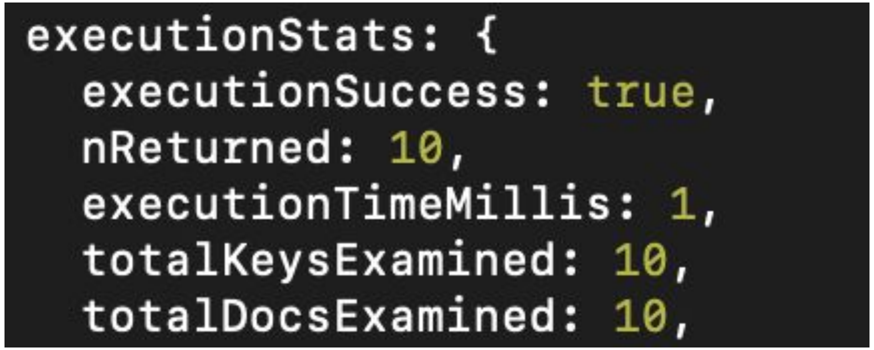
```
db.Workouts.createIndex({ userId: 1, date: -1 })
```

w/o

w/



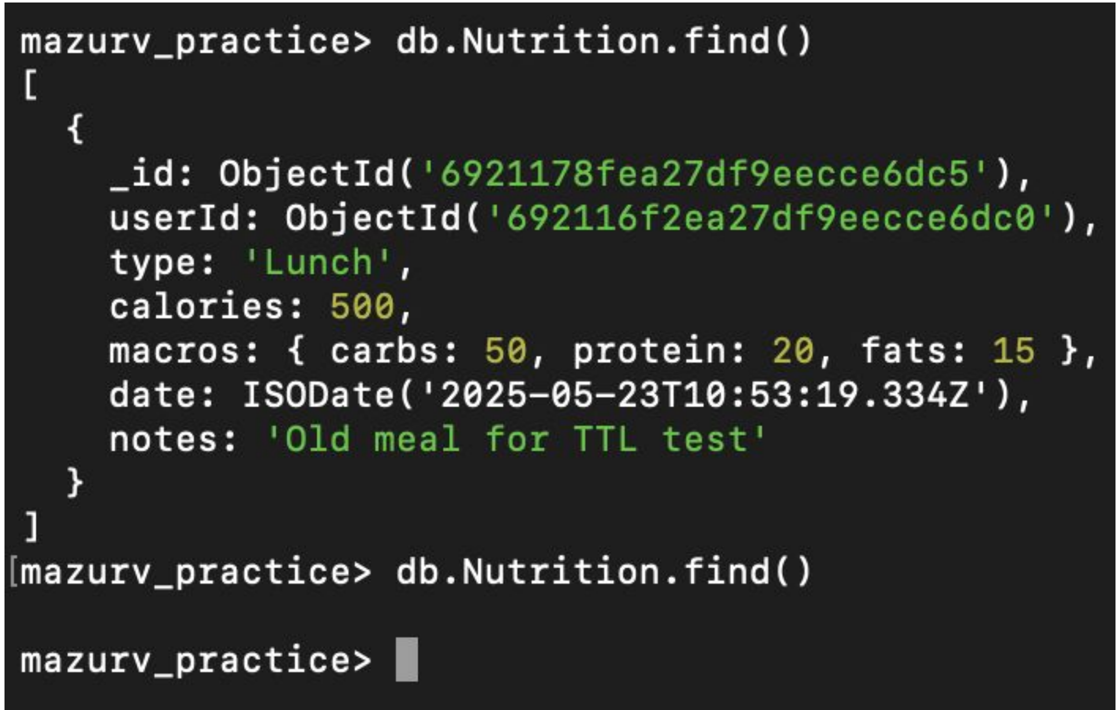
```
executionStats: {
  executionSuccess: true,
  nReturned: 10,
  executionTimeMillis: 42,
  totalKeysExamined: 0,
  totalDocsExamined: 100000,
```



```
executionStats: {
  executionSuccess: true,
  nReturned: 10,
  executionTimeMillis: 1,
  totalKeysExamined: 10,
  totalDocsExamined: 10,
```

TTL

```
// TTL for nutrition
db.Nutrition.createIndex({ date: 1 }, { expireAfterSeconds: 15552000 }) // ~6 months
```

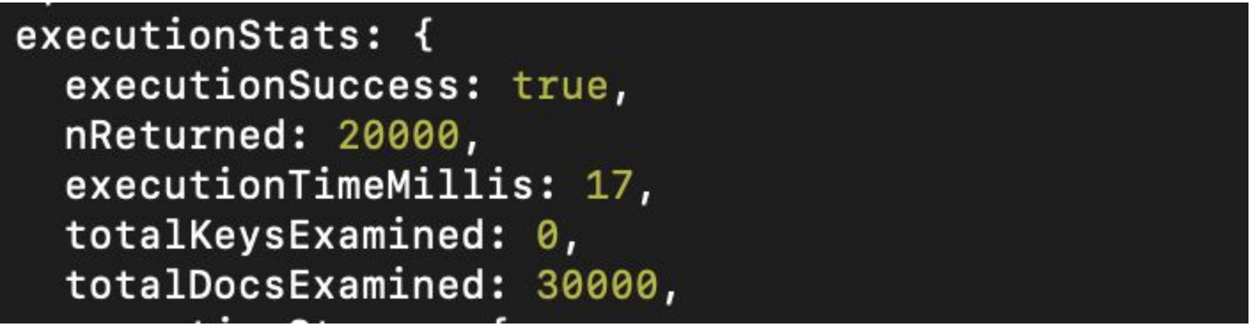


```
mazurv_practice> db.Nutrition.find()
[
  {
    _id: ObjectId('6921178fea27df9eecce6dc5'),
    userId: ObjectId('692116f2ea27df9eecce6dc0'),
    type: 'Lunch',
    calories: 500,
    macros: { carbs: 50, protein: 20, fats: 15 },
    date: ISODate('2025-05-23T10:53:19.334Z'),
    notes: 'Old meal for TTL test'
  }
]
mazurv_practice> db.Nutrition.find()

mazurv_practice>
```

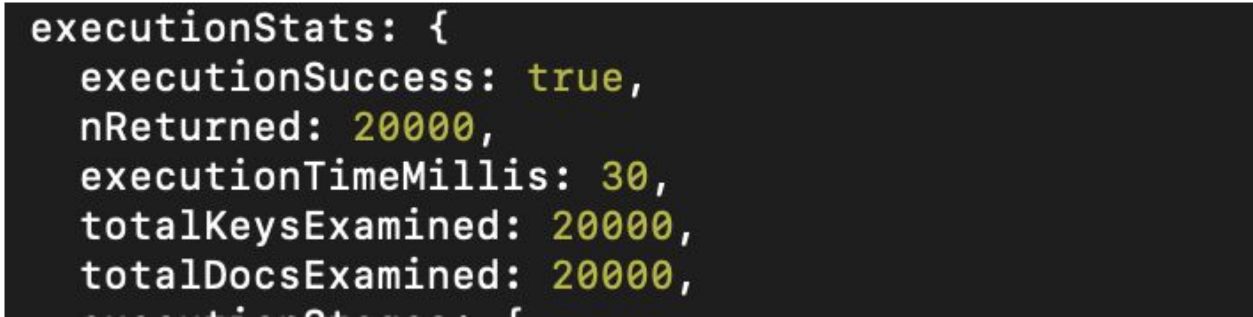
Text

w/o



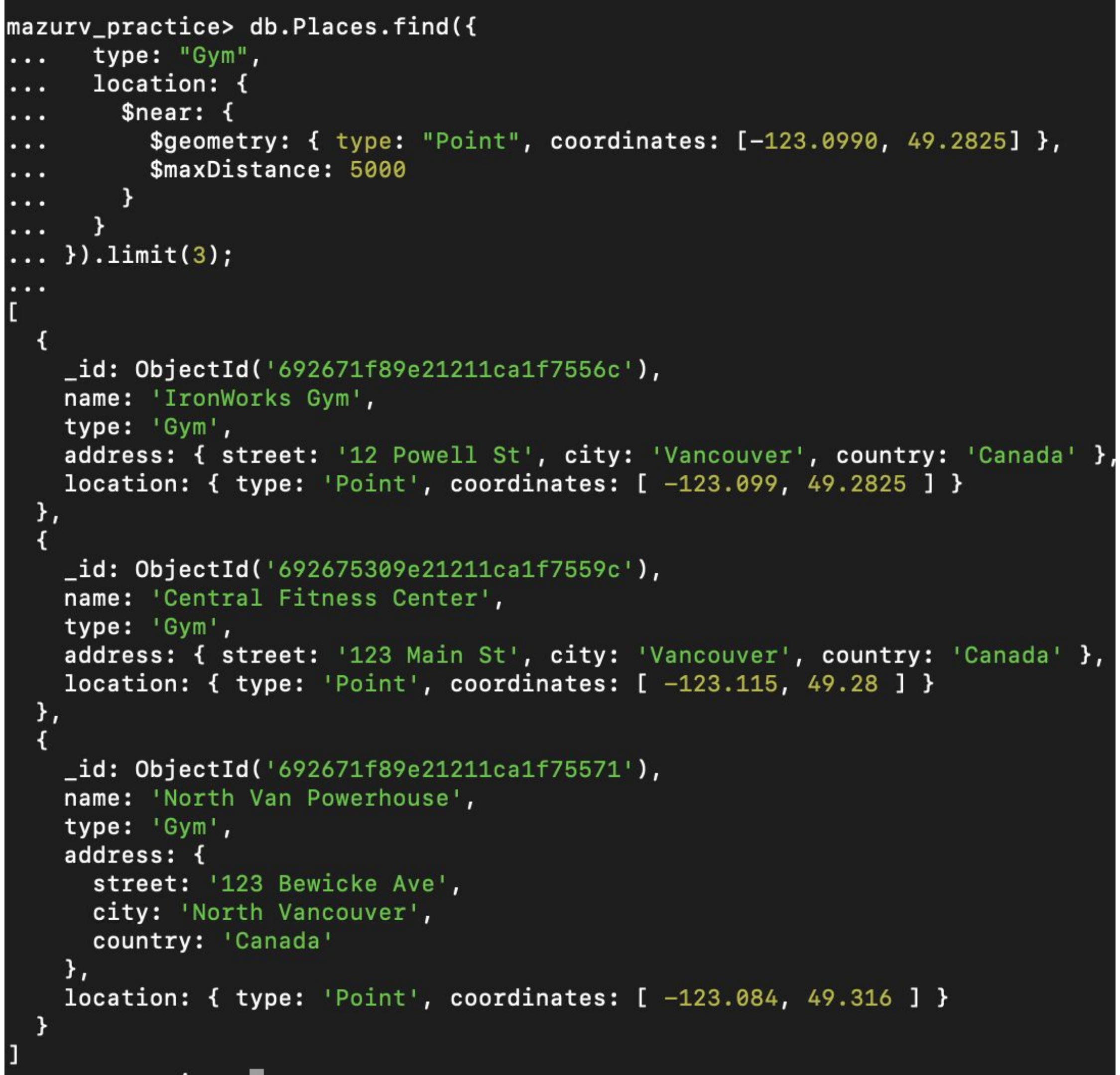
```
executionStats: {
  executionSuccess: true,
  nReturned: 20000,
  executionTimeMillis: 17,
  totalKeysExamined: 0,
  totalDocsExamined: 30000,
```

w/



```
executionStats: {
  executionSuccess: true,
  nReturned: 20000,
  executionTimeMillis: 30,
  totalKeysExamined: 20000,
  totalDocsExamined: 20000,
```

Geo Spatial



```
mazurv_practice> db.Places.find({
...   type: "Gym",
...   location: {
...     $near: {
...       $geometry: { type: "Point", coordinates: [-123.0990, 49.2825] },
...       $maxDistance: 5000
...     }
...   }
... }).limit(3);
...
[
  {
    _id: ObjectId('692671f89e21211ca1f7556c'),
    name: 'IronWorks Gym',
    type: 'Gym',
    address: { street: '12 Powell St', city: 'Vancouver', country: 'Canada' },
    location: { type: 'Point', coordinates: [ -123.099, 49.2825 ] }
  },
  {
    _id: ObjectId('692675309e21211ca1f7559c'),
    name: 'Central Fitness Center',
    type: 'Gym',
    address: { street: '123 Main St', city: 'Vancouver', country: 'Canada' },
    location: { type: 'Point', coordinates: [ -123.115, 49.28 ] }
  },
  {
    _id: ObjectId('692671f89e21211ca1f75571'),
    name: 'North Van Powerhouse',
    type: 'Gym',
    address: {
      street: '123 Bewicke Ave',
      city: 'North Vancouver',
      country: 'Canada'
    },
    location: { type: 'Point', coordinates: [ -123.084, 49.316 ] }
  }
]
```

Patterns

Polymorphous

Extended Reference - Workouts (placeId / Address)

Subset - User Collection

Anti-Patterns

Over-Embedding Large Documents

Missing Indexes on Frequent Queries

Lessons Learned

Start w/ Queries

Index Based on Frequency of Use

Data is King

Challenges

Shallow Data

Schema Changing

Getting Rid of SQL thinking

Next Steps

Aggregation To Fill Recent Array

GridFS - Food Photos

Streamline userId + placeId reference

Clean up .js files

Questions?