LESSON

Querying in Relational and Non-Relational Databases

Google slide deck available here

Comparing data models

In this lesson, we will look very broadly at the data models in both SQL and in the MongoDB Query Language (MQL). We'll then describe at a high level how querying works with regard to these data models.

What is the Document Model?

A way to organize and store data as a set of field-value pairs

Similar to:

- Dictionaries in Python

- Maps in Java

- JSON Object in JavaScript

In order to understand query structure in MongoDB, you need to have a good understanding of the document model and its constructs.
A document is a way to organize and store data as a set of field-value pairs.
It represents information in a similar way that a dictionary in Python, a map in Java, and a Javascript Object Notation (JSON) object in JavaScript do.
In MongoDB, documents are actually stored as BSON (binary JSON) in the underlying storage engine. Documents in MongoDB are presented via the drivers or via the MongoDB Shell as JSON.

Contrasting Data Models

Tabular (Relational) Data Model
Related data is split across multiple records and tables
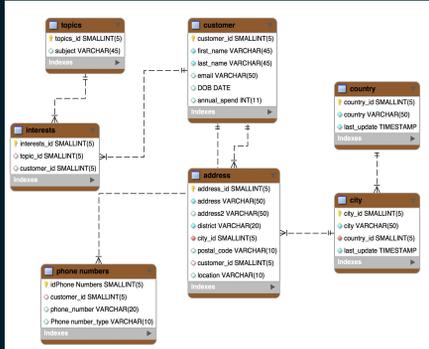
Document Data Model
Related data is contained in a single, rich document

```
"_id" : ObjectId("5ad88534e3632e1a35a58d00"),
"name" : {
  "first" : "John",
  "last" : "Doe" },
"address" : [
  { "location" : "work",
    "address" : {
      "street" : "16 Hatfields",
      "city" : "London",
      "postal_code" : "SE1 8DJ"},
    "geo" : { "type" : "Point", "coord" : [
      -0.109081, 51.5065752]}},
+ {...}
],
"dob" : ISODate("1977-04-01T05:00:00Z"),
"retirement_fund" : NumberDecimal("1292815.75")
}
```

The key difference between these data models is that the relational / tabular records are frequently split across records and tables whilst in documents the data is typically within a single document.

Document Model Constructs

1. Fields (Attributes)

2. Sub-documents (Objects)

3. Arrays
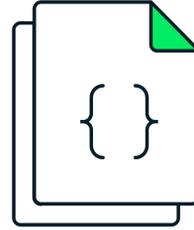
The document model representation is based on three constructs: fields (also referred to as attributes), sub-documents (also referred to as nested documents or objects), and arrays. We'll look at each of these in the coming slides in more detail.

Fields / Attributes

Cars				
_id	owner	make	model	year
007	Daniel	Ferrari	GTS	1982
Q08	Daniel	Fiat	500	2013

```
{
  "_id":    007,
  "owner":  "Daniel",
  "make":   "Ferrari",
  "model":  "GTS",
  "year":   1982
}
```

```
{
  "_id":     008,
  "owner":   "Daniel",
  "make":    "Fiat",
  "model":   "500",
  "submodel: "S",
  "year":    2013
}
```

Tabular (Relational) Data Model   Document Data Model

Here is an example for both models representing cars. Let's look at the various models and their fields/attributes.

In the Tabular Relational data model, a "cars" table would have a list of column names, and each car entry would have a set of values in the same order as the column names.

In MongoDB, using the document model, the column names correspond to the field names and values, whereas the relational database row names correspond to the values of the key-pairs.

The main difference is that column names are appearing in each document, allowing for greater flexibility in having documents with different shapes; in other words, having different fields.

Each document is self-describing without the need to refer to global metadata like a table definition.

Object/Sub-document: A One-to-One Relationship

Cars

_id	owner	make
007	Daniel	Ferrari
Q08	Daniel	Fiat

Engines

_id	car_id	power	consumption
234808	007	660	10
Q08	008	120	45

OR

Cars

_id	owner	make	power	consumption
007	Daniel	Ferrari	660	10
Q08	Daniel	Fiat	120	45

```
{
  "_id": 007,
  "owner": "Daniel",
  "make": "Ferrari",
  "engine": {
      "power": 660hp,
      "consumption": 10mpg
  }
  …
}
```

Tabular (Relational) Data Model
Header with column names
Row with values

Document Data Model
Each document explicitly list the
names of the fields and the values.

An object, or sub-document, in the document model allows for information to be grouped together or embedded as a one-to-one relationship between what would be two tables from the tabular relational model.

We can seen how the car can be treated in a relational database for this one-to-one relationship with either a car and a engine table or a single cars table. In the document data model, this would be within a document representing a car as an engine sub-document.

Array: A One-to-Many

Cars		
_id	owner	make
007	Daniel	Ferrari
Q08	Daniel	Fiat

Wheels	
_id	car_id
234819	007
281928	007
392838	007
928038	007
950555	008

```
{
  "_id": 007,
  "owner": "Daniel",
  "make": "Ferrari",
  wheels: [
      { "partNo": 234819 },
      { "partNo": 281928 },
      { "partNo": 392838 },
      { "partNo": 928038 }
  ],
  ...
}
```

Tabular (Relational) Data Model

One-to-Many relationship
from a car to the its wheels

Document Data Model

One-to-Many wheels
expressed as an array

Arrays are typically used to model one-to-many relationships.

As for the array, it can contain values or objects. In other words, you can have an array of objects. The array models one-to-many relationships into a single document. These relationships would be represented by a parent-child table in traditional relational database models,

For example, in the tabular relational model, we have a parent "Cars" table and its child table "Wheels". In MongoDB, the "Wheels" data is simply embedded in the "Cars" collection. There is no need for a "Wheels" collection.

Quiz

Quiz

Which statements are accurate statements on the document model used in MongoDB? *Select all that apply.* More than one answer choice can be correct.

- A. It is identical to JSON
- B. It is similar to maps in Java and dictionaries in Python
- C. It allows us to model one-to-one and one-to-many relationships

Quiz

Which statements are accurate statements on the document model used in MongoDB? *Select all that apply.* More than one answer choice can be correct.

❌ A. It is identical to JSON

✅ B. It is similar to maps in Java and dictionaries in Python

✅ C. It allows us to model one-to-one and one-to-many relationships

INCORRECT: It is identical to JSON - This is incorrect. In examples, documents are often represented in JSON, but under the hood, MongoDB uses BSON which offers more data types and better performance.

CORRECT: It is similar to maps in Java and dictionaries in Python - This is correct. Conceptually the document model is very similar to maps in Java and dictionaries in Python. This provides a familiar data structure for programmers to use.

CORRECT: It allows us to model one-to-one and one-to-many relationships - This is correct. The document model provides modelling for one-to-one, one-to-many, and many-to-many relationships making it easy to model complex data in MongoDB.

Quiz

Which statements are accurate statements on the document model used in MongoDB? *Select all that apply.* More than one answer choice can be correct.

❌ A. It is identical to JSON

✅ B. It is similar to maps in Java and dictionaries in Python

✅ C. It allows us to model one-to-one and one-to-many relationships

This is incorrect. In examples, documents are often represented in JSON, but under the hood, MongoDB uses BSON which offers more data types and better performance.

INCORRECT: It is identical to JSON - This is incorrect. In examples, documents are often represented in JSON, but under the hood, MongoDB uses BSON which offers more data types and better performance.

Quiz

Which statements are accurate statements on the document model used in MongoDB? *Select all that apply.* More than one answer choice can be correct.

❌ A. It is identical to JSON

✅ B. It is similar to maps in Java and dictionaries in Python

✅ C. It allows us to model one-to-one and one-to-many relationships

This is correct. Conceptually the document model is very similar to maps in Java and dictionaries in Python. This provides a familiar data structure for programmers to use.

CORRECT: It is similar to maps in Java and dictionaries in Python - This is correct. Conceptually the document model is very similar to maps in Java and dictionaries in Python. This provides a familiar data structure for programmers to use.

Quiz

Which statements are accurate statements on the document model used in MongoDB? *Select all that apply.* More than one answer choice can be correct.

❌ A. It is identical to JSON

✅ B. It is similar to maps in Java and dictionaries in Python

✅ C. It allows us to model one-to-one and one-to-many relationships

This is correct. The document model provides modelling for one-to-one, one-to-many, and many-to-many relationships making it easy to model complex data in MongoDB.

CORRECT: It allows us to model one-to-one and one-to-many relationships - This is correct. The document model provides modelling for one-to-one, one-to-many, and many-to-many relationships making it easy to model complex data in MongoDB.

Querying

In this section, we'll broadly look at how querying works in relational databases and how it works in a document database, taking MongoDB as the specific non-relational database example.

Querying in MongoDB and in Relational DBs
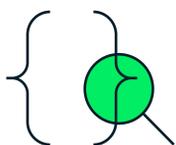
MongoDB Query Language
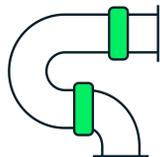
MongoDB Aggregation Framework
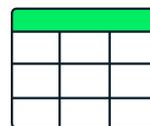
Structured Query Language

We will cover both of the MongoDB query approaches in more detail later but firstly let's introduce them as the counterpart to SQL for querying.

The MongoDB Query Language (MQL) is used to query single collections.
The MongoDB Aggregation Framework is used to create aggregations and manipulate data which can involve multiple collections.
Structured Query Language (SQL) performs both queries and aggregations.

SQL is a declarative language whilst MQL and the MongoDB Aggregation Framework are imperative languages. In SQL, you describe the desired result whilst in MongoDB query languages you tell it what to do.

MongoDB explicitly split single collection queries and CRUD operations on single collections to be the focus of MQL.
The Aggregation Framework was originally designed to replace earlier Map-Reduce functionality with MongoDB. The Aggregation Framework was built due to the clear requirement to be able to perform complex data processing tasks within the database itself.

While SQL is most often used and associated with relational databases, you can still use it within MongoDB using a connector. It is not a native language for MongoDB but connectors can take a SQL query and translate it to a MQL query.

Simple syntax

Designed to query documents

Only queries a single collection
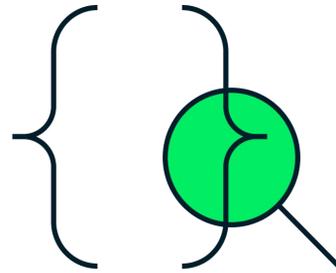
MongoDB Query Language (MQL)

MQL uses a simple syntax which is designed to query documents within a single collection. It is not designed for aggregations or complex manipulation of documents. The structure of the syntax in MQL specifically limits it to a single collection. We'll explore MQL in more depth later in this course.

If you need to construct a query across multiple collections or indeed across databases then you must instead use the MongoDB Aggregation Framework which we'll introduce now.

Designed for aggregations

Complex operations broken into stages

Operators called within stages

Functionality within the DB
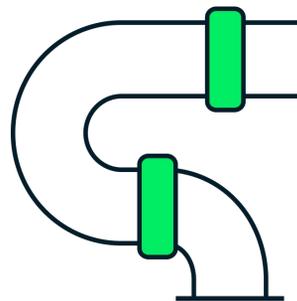
MongoDB Aggregation Framework

MongoDB's Aggregation Framework was designed to make it easier to create aggregations by breaking complex operations into smaller stages. It provides a set of common operators for data manipulation/transformation which can be used within the stages.

Aggregation pipelines are used when you have to run complex queries and a simple MQL query will not suffice. Typical cases include where you need to restructure and change the data, where the data is in multiple collections, or for reporting and analytics.

Complex aggregations are broken down into stages, each stage typically uses one operator to manipulate the data. A big advantage of this approach is that errors or issues can easily be tracked back to the problematic stage or stages. It greatly simplifies debugging aggregations.

This functionality is an integral part of MongoDB.

Domain specific language

Designed to query records

De-facto standard in many relational databases

Structured Query Language

SQL was the first query language to allow for many records to be queried using a single command. Secondly, it removed the need to explicitly define how the DB should locate the record (whether with or without an index).

It is a domain specific language for data querying and data management.

It was designed for tabular records and relational databases.

It is the standard for querying in many database systems.

Quiz

Quiz

When querying in SQL, you query a _____?

When querying in MongoDB, you query a _____?

What are the two languages for querying in MongoDB?

In this quiz, you should fill in the blank for the first two questions:
1. When querying in SQL, you query a …….?
2. When querying in MongoDB, you query a ……?

In the final question, you should list the two languages you can use to query in MongoDB

Quiz

When querying in SQL, you query a _____?

When querying in MongoDB, you query a _____?

What are the two languages for querying in MongoDB?

Fill in the blank - When querying in SQL, you query a _____?

Quiz

When querying in SQL, you query a **table**.

When querying in MongoDB, you query a _____?

What are the two languages for querying in MongoDB?

Records are stored in tables in relational databases, when you query a relational database the query can involve one or more tables.

Quiz

When querying in SQL, you query a table.

When querying in MongoDB, you query a _____?

What are the two languages for querying in MongoDB?

Fill in the blank - When querying in MongoDB, you query a _____?

Quiz

When querying in SQL, you query a table.

When querying in MongoDB, you query a **collection**.

What are the two languages for querying in MongoDB?

In MongoDB documents are stored in a collection, when you query in MongoDB you query one or more collections.

Quiz

When querying in SQL, you query a table.

When querying in MongoDB, you query a collection.

What are the two languages for querying in MongoDB?

What are the two languages for querying in MongoDB?

Quiz

When querying in SQL, you query a table.

When querying in MongoDB, you query a collection.

What are the two languages for querying in MongoDB?

MongoDB Query Language (MQL)

MongoDB Aggregation Framework

The two languages used for querying in MongoDB are the MongoDB Query Language (MQL) and the MongoDB Aggregation Framework, we covered them very briefly in this lesson but that's a far more to cover on each topic.

Continue Learning!

Github Student Developer Pack

MongoDB University has free self-paced courses and labs ranging from beginner to advanced levels.
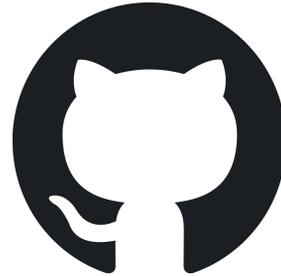
Sign up for the MongoDB Student Pack to receive $50 in Atlas credits and free certification!

This concludes the material for this lesson. However, there are many more ways to learn about MongoDB and non-relational databases, and they are all free! Check out MongoDB's University page to find free courses that go into more depth about everything MongoDB and non-relational. For students and educators alike, MongoDB for Academia is here to offer support in many forms. Check out our educator resources and join the Educator Community. Students can receive $50 in Atlas credits and free certification through the Github Student Developer Pack.