

# MongoDB Query Language and the Aggregation Pipeline

---

# Discussion Format

---

## Learning Objective:

Identify the differences between the MongoDB Query Language and the aggregation pipeline.

## Approach:

- Frame the differences (and similarities) in terms of business needs and technical details.
- Juxtapose that to three user profiles:
  - analyst
  - developer
  - data scientist

## Scope:

- Business Needs
- General Comparison
- Technical Differences
- Activities

## Business Need: MQL || Aggregation Pipeline

---

- Perform operations on grouped data to return a single result.
  - aggregation pipeline
- Query and sort documents for some business process.
  - MQL or aggregation pipeline
- Present data changes over time.
  - aggregation pipeline
- Reveal patterns in data.
  - aggregation pipeline

# General Comparison

---

## MongoDB Query Language (MQL)

- Focused on business process problem-solving
- Simple queries
- Transactional
- Used for CRUD operations

## Aggregation Pipeline

- Built to combine and unwind business data
- Complex queries
- Transformational
- Operates at the data science level
- Declarative style



Analyst

Developer

Data  
Scientist

# Technical Differences

---

## MQL

- Query operators
  - `db.collection.find()`
- Application of query criteria
  - Comparison (`$gt`, `$lt`)
  - Logical(`$and`, `$or`)
- Cursor modifier
  - `.limit(5)`
- Projections
  - `{_id: 0}`

## Aggregation Pipeline

- Built with stages
  - Pipeline operator(s) as a stage
  - `$match` | `$group` | `$limit`
    - `$match` filters for matching docs
- Stages
  - Operate on and process input documents
- Input to Result
  - `{}, {}`  $\Rightarrow$  `{}, {}`
- `$set` and `$unset`



Analyst

Developer

Data  
Scientist

## Code Comparison

---

MQL:

```
db.users.find(  
  {  
    role:"developer"  
  }  
)
```

Action: find and  
return

```
[  
  {  
    "firstName": "Maria",  
    "lastName": "DelCampo",  
    "role": "developer",  
    "coursesCompleted" : 4,  
  },  
  {  
    "firstName": "Ger",  
    "lastName": "Sletteemon",  
    "role": "business analyst",  
    "coursesCompleted" : 1,  
  },  
)
```

Aggregation Pipeline:

```
db.users.aggregate( [  
  { $match: { role:  
    "developer" } }  
)
```

Action: filter and output

# Engage: MongoDB Atlas

---



1. Recording of an instructor illustrating the following actions (documentation of steps to be included)
2. Under the “Find” tab, in the “options” dropdown enter some MQL queries:
  - a. Filter: { cuisine: “Irish” }
  - b. Project: {restaurant\_id: 0, name: 0}
3. Under the “Aggregation” tab, find the “Select” dropdown and choose “\$match”:
  - a. Enter: { cuisine: “Irish” }
  - b. Choose: “Add Stage”
    - i. Select \$limit
    - ii. Enter: 5

## Engage: NodeJS Project

---



1. Recording of an instructor building a simple project (documentation of steps to be included):
  - a. Using MQL
    - i. Pre-built find query
    - ii. User adds additional methods
  - b. Using Aggregation Pipeline
    - i. Pre-built \$match stage
    - ii. User builds additional stages with \$group and \$limit



## Close

---

Identification of the differences between the MongoDB Query Language and the aggregation pipeline.

Methodology:

- Business Needs
- General Comparison
- Technical Differences
- Activities