

MongoDB – Le DML



MongoDB – Le DML

- Plan
 - Introduction
 - La récupération de données
 - L'extraction simple
 - L'agrégation
 - Exercice récapitulatif

Le DML - Introduction



- Le DML (Data Manipulation Language) est un langage permettant de manipuler les données contenues dans une base de données.
- Le DML permet la :
 - récupération des données
 - modification des données

Le DML – La récupération des données



- Dans MongoDB, il existe trois méthodes de récupération : l'extraction simple, l'agrégation et la « distinct ».
- L'extraction simple permet d'effectuer des requêtes simples de sélection et de projection des données.
 - La projection des données est une sélection des données à renvoyer.
- L'agrégation permet d'effectuer des actions sur les données avant de les renvoyer.
- La méthode « distinct » permet d'éviter les renvois de doublons.

L'extraction simple

Introduction

Les opérateurs de recherches

Les opérateurs conditionnels

Les opérateurs d'éléments

Les opérateurs d'évaluation

Les opérateurs sur tableau

Les opérateurs de projection

L'extraction simple – Introduction

- L'extraction simple est une méthode de la collection permettant de renvoyer un ou plusieurs document(s) selon un ou plusieurs critères et d'affichant que les attributs nécessaires.
- Structure générale d'une extraction:
`db.<collection>.find({<critères>},{<projections>})`
- Exemple d'extraction sans critère et sans projection:
 - requête: Afficher tous les clients.
 - traduction DML: `db.clients.find({}, {})`

L'extraction simple – Introduction

- Résultat:

```
{ "_id": "B332", "NOM": "MONTI", "ADRESSE": "23, a. Dumont", "LOCALITE": "Geneve", "CAT": "B2", "COMPTE": 0 }
{ "_id": "B112", "NOM": "HANSENNEA", "ADRESSE": "23, a. Dumont", "LOCALITE": "Poitiers", "CAT": "C1", "COMPTE": 1250 }
{ "_id": "F010", "NOM": "TOUSSAINT", "ADRESSE": "23, a. Dumont", "LOCALITE": "Poitiers", "CAT": "C1", "COMPTE": 0 }
{ "_id": "S127", "NOM": "VANDERKA", "ADRESSE": "23, a. Dumont", "LOCALITE": "Namur", "CAT": "C1", "COMPTE": -4580, "COMMANDES": [{"NCOM": 30182, "DETAIL": [{"produit_id": "PA60", "QCOM": 20}]}] }
{ "_id": "B062", "NOM": "GOFFIN", "ADRESSE": "23, a. Dumont", "LOCALITE": "Namur", "CAT": "B2", "COMPTE": -3200 }
{ "_id": "B512", "NOM": "GILLET", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "CAT": "B1", "COMPTE": -8700, "COMMANDES": [{"NCOM": 30188, "DETAIL": [{"produit_id": "PA60", "QCOM": 70}, {"produit_id": "PS222", "QCOM": 600}]}] }
{ "_id": "K111", "NOM": "VANBIST", "ADRESSE": "23, a. Dumont", "LOCALITE": "Lille", "CAT": "B1", "COMPTE": 720, "COMMANDES": [{"NCOM": 30178, "DETAIL": [{"produit_id": "CS464", "QCOM": 25}]}] }
{ "_id": "C123", "NOM": "MERCIER", "ADRESSE": "23, a. Dumont", "LOCALITE": "Namur", "CAT": "C1", "COMPTE": -2300 }
{ "_id": "C003", "NOM": "AVRON", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "CAT": "B1", "COMPTE": -1750 }
{ "_id": "K729", "NOM": "NEUMAN", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "COMPTE": 0 }
{ "_id": "C400", "NOM": "FERARD", "ADRESSE": "23, a. Dumont", "LOCALITE": "Poitiers", "CAT": "B2", "COMPTE": 350, "COMMANDES": [{"NCOM": 30179, "DETAIL": [{"produit_id": "CS262", "QCOM": 60}, {"produit_id": "PS222", "QCOM": 600}]}] }
{ "_id": "F011", "NOM": "PONCELET", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "CAT": "B2", "COMPTE": 0, "COMMANDES": [{"NCOM": 30185, "DETAIL": [{"produit_id": "PS222", "QCOM": 600}, {"produit_id": "PA60", "QCOM": 20}]}] }
{ "_id": "L422", "NOM": "FRANCK", "ADRESSE": "23, a. Dumont", "LOCALITE": "Namur", "CAT": "C1", "COMPTE": 0 }
{ "_id": "D063", "NOM": "MERCIER", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "COMPTE": -2250 }
{ "_id": "F400", "NOM": "JACOB", "ADRESSE": "23, a. Dumont", "LOCALITE": "Bruxelles", "CAT": "C2", "COMPTE": 0 }
{ "_id": "S712", "NOM": "GUILLAUME", "ADRESSE": "23, a. Dumont", "LOCALITE": "Paris", "CAT": "B1", "COMPTE": 0 }
{ "_id": "K112", "NOM": "EXO", "ADRESSE": "4/105, r. Charles Dubois", "LOCALITE": "Namur", "COMPTE": 125 }
{ "_id": "F401", "NOM": "1234", "ADRESSE": "23, a. Dumont", "LOCALITE": "Bruxelles", "CAT": "C2", "COMPTE": 2523 }
{ "_id": "B632", "NOM": "AROUNDI", "ADRESSE": "23, a. Dumont", "LOCALITE": "Geneve", "CAT": "B2", "COMPTE": 0.95 }
```

L'extraction simple – Introduction

- Exemple d'extraction simple avec critère de sélection
 - Requête: Afficher tous les clients qui n'ont pas de catégorie.
 - Traduction DML: `db.clients.find({CAT: {$exists: false}}, {})`
 - Résultat:

```
{ "_id": "K729", "NOM": "NEUMAN", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "COMPTE": 0 }  
{ "_id": "D063", "NOM": "MERCIER", "ADRESSE": "23, a. Dumont", "LOCALITE": "Toulouse", "COMPTE": -2250 }  
{ "_id": "K112", "NOM": "EXO", "ADRESSE": "4/105, r. Charles Dubois", "LOCALITE": "Namur", "COMPTE": 125 }
```


L'extraction simple – Introduction


- Exemple de requêtes avec projection.
 - Afficher le NOM de tous les clients.
 - Traduction DML: `db.find({}, {NOM: true})`
 - Résultat:

```
{ "_id": "B332", "NOM": "MONTI" }
{ "_id": "B112", "NOM": "HANSENNEA" }
{ "_id": "F010", "NOM": "TOUSSAINT" }
{ "_id": "S127", "NOM": "VANDERKA" }
{ "_id": "B062", "NOM": "GOFFIN" }
{ "_id": "B512", "NOM": "GILLET" }
{ "_id": "K111", "NOM": "VANBIST" }
{ "_id": "C123", "NOM": "MERCIER" }
{ "_id": "C003", "NOM": "AVRON" }
{ "_id": "K729", "NOM": "NEUMAN" }
{ "_id": "C400", "NOM": "FERARD" }
{ "_id": "F011", "NOM": "PONCELET" }
{ "_id": "L422", "NOM": "FRANCK" }
{ "_id": "D063", "NOM": "MERCIER" }
{ "_id": "F400", "NOM": "JACOB" }
{ "_id": "S712", "NOM": "GUILLAUME" }
{ "_id": "K112", "NOM": "EXO" }
{ "_id": "F401", "NOM": 1234 }
{ "_id": "B632", "NOM": "AROUNDI" }
```

L'extraction simple – Introduction

- Exercices

1. Afficher le NOM et l'ADRESSE des clients habitant à Toulouse.
2. Afficher le NOM, le COMPTE et les COMMANDES des clients habitant à Poitiers.
3. Afficher tous les champs sauf le champ CAT pour tous les clients
4. Afficher le DETAIL des COMMANDES.



Extraction simple: Les opérateurs de recherche

Les opérateurs \$eq,
\$gt, \$gte, \$lt, \$lte

L'opérateur \$in

Exercices



Extraction – Les opérateurs de recherche

L'opérateur $\$eq$, $\$gt(e)$, $\$lt(e)$,

- Ces opérateurs sont utilisés à des fins de comparaison.
- $\$eq$ signifie égale/équivalente.
- $\$gt$ et $\$gte$ signifient « plus grand » et « plus grand ou égal »
- $\$lt$ et $\$lte$ signifient « plus petit » et « plus petit ou égal »
- La structure générale de cet opérateur est:
 {<opérateur>: <valeur>}

Extraction – Les opérateurs de recherche

- Exemple de requête \$eq:
 - requête: Afficher les clients dont le nom est exactement « FERARD »
 - traduction DML: `db.clients.find({NOM: {$eq: 'FERARD'}})`
 - résultat: `{ "_id": "C400", "NOM": "FERARD", "ADRESSE": "23, a. Dumont", "LOCALITE": "Poitiers", "CAT": "B2", "COMPTE": 350 }`
- Exemple de requête \$gt(e), \$lt(e):
 - requête: Afficher le NOM des clients dont le COMPTE est strictement positif.
 - traduction DML: `db.clients.find({COMPTE: {$gt: 0}}, {NOM: true})`
 - résultat:

```
{ "_id": "B112", "NOM": "HANSENNEA" }  
{ "_id": "K111", "NOM": "VANBIST" }  
{ "_id": "C400", "NOM": "FERARD" }  
{ "_id": "K112", "NOM": "EXO" }  
{ "_id": "F401", "NOM": "1234" }  
{ "_id": "B632", "NOM": "AROUNDI" }
```

Extraction – Les opérateurs de recherche


L'opérateur \$in

- Cet opérateur est utilisé dans la comparaison
- Il signifie que le champ doit être égal au moins à un élément du tableau passé en valeur.
- Structure générale de cet opérateur:
`{ $in: [<expression1>, <expression2>, ...<expressionN>] }`
- Exemple de requête:
 - requête: Afficher les clients qui habitent à Bruxelles ou à Poitier.
 - traduction DML: `db.clients.find({LOCALITE: { $in: ['Bruxelles', 'Poitiers'] } }, {})`

Extraction – Les opérateurs de recherche

- Exercices

1. Afficher le NOM des clients dont le COMPTE est strictement négatif.
2. Afficher le NOM et la LOCALITE des clients dont le COMPTE est vide
3. Afficher la LOCALITE des clients dont celle-ci est plus grande que Bruxelles.
4. Afficher le NOM, la LOCALITE des clients dont la LOCALITE n'est pas 'Bruxelles' ou 'Poitiers'.



Extraction simple: Les opérateurs conditionnels

Les opérateurs \$and,
\$or

L'opérateur \$not

Exercices



Extraction – Les opérateurs conditionnels

L'opérateur \$and et \$or

- Ces opérateurs permettent d'ajouter des opérateurs logiques à la recherche.
- Structure générale de ces opérateurs:
 {<opérateur>: [<critère1>, <critère2>, ...]}
- Exemple de requêtes:
 - requête: Afficher les produits dont le prix est 185 ou si le NCOM est CS424.
 - traduction DML: `db.produits.find({$or: [{NCOM: 'CS424'}, {PRIX: 185}]}, {})`

Extraction – Les opérateurs conditionnels

L'opérateur \$not

- Cet opérateur permet la négation des opérateurs logiques.
- Structure de cet opérateur:
`{ $not: { <opérateur>: <expression> } }`
- Exemple de requêtes:
 - requête: Afficher le NOM des clients dont le COMPTE n'est pas strictement positif.
 - traduction: `db.clients.find({COMPTE: { $not: { $gt: 0 } } }, {})`

Extraction – Les opérateurs conditionnels


- Exercices

1. Afficher le NOM des clients qui sont dans la CAT 'B1' et dont le compte est strictement positif.
2. Afficher le NOM des clients qui ont un COMPTE strictement négatif habitant à 'Namur'.
3. Afficher le NOM des clients dont le compte est positif ou négatif et qui habitent à 'Namur' ou à 'Lille'.



Extraction simple: Les opérateurs d'élément

L'opérateur \$exists
L'opérateur \$type
Exercices



Extraction – Les opérateurs d'éléments

L'opérateur \$exists

- Cet opérateur est utilisé pour la recherche de champs présents dans le document ou non.
- Structure de cet opérateur:
{\$exists: <true/false>}
- Exemple de requête:
 - requête: Afficher le NOM des clients ayant passé au moins une COMMANDES.
 - traduction DML: `db.clients.find({COMMANDES: {$exists: true}}, {NOM: true})`
 - résultat:

Extraction – Les opérateurs d'éléments

L'opérateur \$ type


- Cet opérateur permet de spécifier le type de l'élément recherché.
- Structure de cet opérateur:
{\$type: '<type>'}
- Exemple de requête:
 - requête: Afficher le NOM du client dont le NOM est de type 'double'.
 - traduction DML: `db.clients.fnd({NOM: {$type: 'double'}},{NOM: true})`
 - résultat:

```
{"_id":"F401","NOM":1234}
```

Extraction – Les opérateurs d'éléments

- Exercices

1. Afficher tous les clients n'ayant pas passé de COMMANDES.
2. Afficher tous les clients n'ayant pas de COMMANDES, de CAT et dont le NOM est de type 'double'.
3. Afficher le NOM, l'ADRESSE et le COMPTE des clients n'ayant pas de CAT.



Extraction simple: Les opérateurs d'évaluation

L'opérateur \$mod

L'opérateur \$regex

L'opérateur \$text

L'opérateur \$where

Exercices



Extraction – Les opérateurs d'évaluation

L'opérateur \$mod

- Cet opérateur permet d'effectuer le modulo (le reste d'une division euclidienne).
- Structure générale de cet opérateur:
{\$mod: [<diviseur>, <reste>]}
- Exemple de requête:
 - requête: Afficher le NOM des clients dont le COMPTE est pair.
 - traduction DML: `db.clients.find({COMPTE: {$mod: [2,0]}},{NOM: true})`
 - résultat:

Extraction – Les opérateurs d'évaluation

L'opérateur \$regex

- Cet opérateur est extrêmement puissant, mais compliqué à mettre en place. Nous ne verrons ici que l'équivalent du « like » en SQL.
- Cet opérateur est utilisé afin de vérifier la correspondance entre la requête et le contenu de la valeur d'un champ.
- Structure de l'opérateur
`{ $regex: '.*' + <texte à trouver> + '.*', $options: '<options>' }`
- Pour plus d'informations sur le regex:
http://regexone.com/lesson/introduction_abcs

Extraction – Les opérateurs d'évaluation

- Exemple de requête avec l'opérateur \$regex.
 - Requête: Afficher le NOM des clients dont le NOM contient la lettre F.
 - Traduction DML:

```
db.clients.find({NOM: {$regex: '.*'+ 'F'+ '.*', $options: 'i'}},{NOM: true})
```
 - Résultat:

```
{ "_id": "B062", "NOM": "GOFFIN" }  
{ "_id": "C400", "NOM": "FERARD" }  
{ "_id": "L422", "NOM": "FRANCK" }
```

Extraction – Les opérateurs d'évaluation

L'opérateur \$text

- Cet opérateur ne fonctionne que sur des champs indexés.
- Il permet de trouver des chaînes de caractères incluses dans une valeur. (Équivalent du « like » SQL mais nécessite d'avoir le champ indexé).
- Structure de l'opérateur:
{\$text: <chaîne de caractère>}
- Exemple de requête:
 - requête: Afficher le NOM des clients dont le NOM contient 'F'.
 - traduction DML: `db.clients.find({NOM: {$text: 'F'}}, {NOM: true})`

Extraction – Les opérateurs d'évaluation


L'opérateur \$where

- Cet opérateur est utilisé pour évaluer une expression Javascript.
- Structure de cet opérateur
 - `{ $where: '<expression Javascript>' }`
- Exemple de requête:
 - requête: Afficher le NOM des clients ayant effectué plusieurs COMMANDES.
 - traduction DML:
`db.clients.find({COMMANDES: { $exists: true }, $where: 'this.COMMANDES.length > 1' }, {NOM: true})`
 - résultat:

Extraction – Les opérateurs d'évaluation

- Exercices

1. Afficher le NOM des clients ayant effectué exactement une COMMANDES.
2. Afficher le NOM des clients dont le COMPTE est pair et dont le NOM contient « er » dans un champ non indexé.
3. Afficher le NOM, la LOCALITE des clients n'ayant jamais commandé dont le COMPTE est positif et que le NOM n'est pas une chaine de caractère.



Extraction simple: Les opérateurs sur tableau

L'opérateur \$all

L'opérateur
\$elemMatch

L'opérateur \$size

Exercices



Extraction – Les opérateurs sur tableau

L'opérateur \$all

- Cet opérateur est utilisé pour sélectionner les documents contenant tous les éléments demandés.
- Structure de cet opérateur:
`{ $all: [<valeur1>, <valeur2>, ...] }`
- Exemple de requêtes:
 - requête: Afficher les expériences dont le champ « A » comporte tous les éléments « red » et « blue »
 - traduction DML: `db.experiments.find({A: { $all: ['red', 'blue'] }}, {})`

Extraction – Les opérateurs sur tableau

L'opérateur \$elemMatch

- Cet opérateur est utilisé pour sélectionner les documents contenant un ou plusieurs éléments demandés
- Structure de cet opérateur:
`{ $elemMatch: { <expression1>, <expression2>, ... } }`
- Exemple de requêtes:
 - requête: Afficher les expériences dont le champ « A » comporte tous les éléments « red » et « blue »
 - traduction DML: `db.experiments.find({A: { $elemMatch: [{ $gt: 'blue' }, { $eq: 'red' }] } }, {})`

Extraction – Les opérateurs sur tableau

L'opérateur \$size

- Cet opérateur est utilisé pour sélectionner les documents selon la taille du tableau d'un champ
- Structure de cet opérateur:
{\$size: <nombre>}
- Exemple de requêtes:
 - requête: Afficher les expériences dont le champ « A » comporte deux éléments
 - traduction DML: `db.experiments.find({A: {$size: 2}},{})`

Extraction – Les opérateurs sur tableau

- Exercices

1. Afficher l'id des expériences dont le champ « A » contient deux éléments et dont le contenu de « B » est « red » et « blue ».
2. Afficher le champ « A » des expériences dont le champ « B » ne contient pas de composant.
3. Afficher les clients ayant passé une COMMANDES



Extraction simple: Les opérateurs de projection

L'opérateur \$

L'opérateur
\$elemMatch

L'opérateur \$slice



Extraction – Les opérateurs de projection

L'opérateur \$ (projection)

- Cet opérateur est utilisé pour limiter les éléments retournés d'un tableau à afficher au premier élément.
- Structure:
`{<tableau>.$: true}`
- Exemple de requête:
 - requête: Afficher la première COMMANDES effectuée par les clients.
 - traduction DML: `db.clients.find({COMMANDES: {$exists: true}}, {"COMMANDES.$": true})`

Extraction – Les opérateurs de projection

L'opérateur \$elemMatch (projection)

- Cet opérateur permet l'affichage conditionné.
- Structure de cet opérateur:
{\$elemMatch: {<expression>}}
- Exemple de requête:
 - requête: Afficher tous les _id des clients, mais afficher en plus la commande dont le NCOM est 30182.
 - traduction DML: `db.clients.find({}, {COMMANDES: {$elemMatch: {NCOM: 30182}}})`

Extraction – Les opérateurs de projection

L'opérateur \$slice

- Cet opérateur permet l'affichage d'un ensemble d'éléments d'un tableau.
- Structure de cet opérateur:
{\$slice: [<index_start>, <nombre_element>]}
L'index de départ est optionnel.
- Exemple de requête:
 - requête: Afficher le deuxième élément du tableau de COMMANDES des clients.
 - traduction DML: `db.client.find({COMMANDES: {$exists: true}}, {COMMANDES: {$slice: [1,1]}})`

Extraction – Les opérateurs de projection

- Exercice

1. Afficher le NCOM de la première COMMANDES effectuée par les clients.
2. Afficher le premier PRODUIT du DETAIL d'une commandes des clients.
3. Afficher la QCOM de tous les produits de la deuxième COMMANDES des clients.

Extraction – Exercices globales

1. Afficher le NOM, le COMPTE et l'ADRESSE du client dont le NOM possède un « n » et dont le COMPTE n'est pas égale null/0 de toutes les manières possibles.