

TP 1 MongoDB

Interagir avec les bases de données

Local Database

Connexion à la base de données locale

```
$mongosh
```

La sortie attendue est la suivante:

```
C:\Users\brice>mongosh
Current Mongosh Log ID: 636ec0f12c954d752ff67c6d
Connecting to:      mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+1.6.0
Using MongoDB:      6.0.2
Using Mongosh:       1.6.0

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

-----
  The server generated these startup warnings when booting
  2022-11-11T12:14:29.199+01:00: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
-----

-----
  Enable MongoDB's free cloud-based monitoring service, which will then receive and display
  metrics about your deployment (disk utilization, CPU, operation statistics, etc).

  The monitoring data will be available on a MongoDB website with a unique URL accessible to you
  and anyone you share the URL with. MongoDB may use this information to make product
  improvements and to suggest MongoDB products and deployment options to you.

  To enable free monitoring, run the following command: db.enableFreeMonitoring()
  To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
-----

test>
```

Afficher la version du serveur

```
test> db.version()
6.0.2
```

Les basiques de MongoDB

Bases de données

Lister les bases de données

```
test> show dbs
admin      80.00 KiB
config    108.00 KiB
local      76.00 KiB
```

Connexion à une base de données existante

```
test> use admin
switched to db admin
admin>
```

Lorsqu'on se connecte à une base de données, le texte affiché avant le curseur ">" change et porte désormais le nom de la base de données courante.

Connexion à une base de données inexistante

```
test> use myFirstDB
switched to db myFirstDB
myFirstDB>
```

Cette commande va créer une nouvelle base de données, si celle-ci n'existe pas encore. Si elle existe, elle va commencer à l'utiliser.

TAF: Créer une base de données relatives au management d'une école

Collections

Créer des collections

Pour effectuer des opérations dans la base de données courante, on utilise `db`.

```
esigManagement> db.createCollection("students")
{ ok: 1 }
```

Lister les collections

```
esigManagement> show collections
students
```

TAF: Créer une collection des enseignants de votre école et vérifier qu'elle a été créée.

Documents

```
esigManagement> db.students.insertOne({
  "firstName": "Encorvou",
  "lastName": "Ducobu",
  "email": "encorvou.ducobu@esigelec.com",
  "studentId": 20225454815
})
{
  acknowledged: true,
  insertedId: ObjectId("636ec2a25418039f85c97412")
}
```

Nous venons d'ajouter l'élève Ducobu à notre base de données students. Exécutez la commande suivante sans utiliser `db.createCollection`.

```
esigManagement> db.rooms.insertOne({"roomId": "B1215", "step": 3, "building": "B"})
```

TAF: Commenter le résultat de cette commande.

TAP: Une fois ce résultat commenté, supprimez la collection room en utilisant la méthode drop des collections.

TAF : Avec quels éléments d'une base relationnelle pourrait-on comparer une collection, un document ?

Nous pouvons en insérer plus et plusieurs d'un coup.

```

esigManagement> db.students.insertMany([
  {
    "firstName": "Son",
    "lastName": "Goku",
    "email": "son.goku@esigelec.com",
    "studentId": 20225454816
  },
  {
    "firstName": "Dora",
    "lastName": "exploratrice",
    "email": "dora.exploratrice@esigelec.com",
    "studentId": 20225454817
  }
])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("636ec2de5418039f85c97413"),
    '1': ObjectId("636ec2de5418039f85c97414")
  }
}

```

Nous venons d'ajouter les élèves Goku et Dora dans notre collection **students**.

TAF: Ajoutez des enseignants. Pour chaque enseignant, on doit être capable de savoir:

- L'ancienneté
- Les enseignements
- Le salaire
- Le département
- Temps-partiel/plein

TAF: Ajoutez des informations sur la localisation des étudiants, leur promo(année), et leur dominante.

Opérations de base sur les documents dans MongoDB

Compter des documents

La fonction count de la collection students permet de compter le nombre de documents.

```

esigManagement> db.students.countDocuments()
3

```

Lister les documents

Pour trouver des documents, exécutez la commande suivante

```

esigManagement> db.students.find()
[
  {
    _id: ObjectId("636ec2a25418039f85c97412"),
    firstName: 'Encorvou',
    lastName: 'Ducobu',
    email: 'encorvou.ducobu@esigelec.com',

```

```

    studentId: 20225454815
  },
  {
    _id: ObjectId("636ec2de5418039f85c97413"),
    firstName: 'Son',
    lastName: 'Goku',
    email: 'son.goku@esigelec.com',
    studentId: 20225454816
  },
  {
    _id: ObjectId("636ec2de5418039f85c97414"),
    firstName: 'Dora',
    lastName: 'exploratrice',
    email: 'dora.exploratrice@esigelec.com',
    studentId: 20225454817
  }
]

```

Elle va lister tous les documents contenus dans la collection **students**.

TAF: Que remarquez vous dans les documents affichés?

TAF: Affichez les enseignants enregistrés.

Trier les documents

Afficher les élèves par ordre d'enregistrement dans la base. Du plus récent au plus ancien.

```

esigManagement> db.students.find().sort({"_id":-1})
[
  {
    _id: ObjectId("636ec2de5418039f85c97414"),
    firstName: 'Dora',
    lastName: 'exploratrice',
    email: 'dora.exploratrice@esigelec.com',
    studentId: 20225454817
  },
  {
    _id: ObjectId("636ec2de5418039f85c97413"),
    firstName: 'Son',
    lastName: 'Goku',
    email: 'son.goku@esigelec.com',
    studentId: 20225454816
  },
  {
    _id: ObjectId("636ec2a25418039f85c97412"),
    firstName: 'Encorvou',
    lastName: 'Ducobu',
    email: 'encorvou.ducobu@esigelec.com',
    studentId: 20225454815
  }
]

```

Comment sont identifiés uniquement les documents?

TAF: Afficher la liste des étudiants par ordre alphabétique

TAF: Afficher la liste des étudiants par ordre d'ancienneté(du plus ancien au dernier arrivé)

```

esigManagement> db.students.find().sort({"firsrstName":1})
[
  {
    _id: ObjectId("636ec2de5418039f85c97414"),
    firsrstName: 'Dora',
    lastName: 'exploratrice',
    email: 'dora.exploratrice@esigelec.com',
    studentId: 20225454817
  },
  {
    _id: ObjectId("636ec2a25418039f85c97412"),
    firsrstName: 'Encorvou',
    lastName: 'Ducobu',
    email: 'encorvou.ducobu@esigelec.com',
    studentId: 20225454815
  },
  {
    _id: ObjectId("636ec2de5418039f85c97413"),
    firsrstName: 'Son',
    lastName: 'Goku',
    email: 'son.goku@esigelec.com',
    studentId: 20225454816
  }
]

```

Limitier les documents

Afficher le premier étudiant inscrit à l'école

```

esigManagement> db.students.find().limit(1)
[
  {
    _id: ObjectId("636ec2a25418039f85c97412"),
    firsrstName: 'Encorvou',
    lastName: 'Ducobu',
    email: 'encorvou.ducobu@esigelec.com',
    studentId: 20225454815
  }
]

```

TAF: Afficher le dernier étudiant inscrit à l'école

Se déconnecter du serveur

Il est possible d'ajouter des données dans des base MongoDB sans s'y être connecté au préalable. Pour le faire, Il faut quitter le serveur.

```
exit
```

Les importations de données

Nous allons le faire grâce à l'outil `mongoimport`.

Importer des fichiers .json et .csv

Importer des données depuis un fichier JSON

```
mongoimport --jsonArray --db dev --collection collection_name --file movies.json
```

TAF: Commentez le résultat de cette commande. A quel élément d'un modèle relationnel vous fait penser ce résultat?

TAF: Vérifier que les données sont bien importées au bon endroit.

Importer des données depuis un fichier CSV

```
mongoimport --type csv -d productDB -c products --headerline --drop products.csv
```

TAF: Vérifier que les données sont bien importées au bon endroit.

Exercice: Manipulation des données importées.

- Combien de produits avons nous en stock?
- Afficher la liste de tous ces produits.
- Afficher en priorité les produit avec le stock le plus faible.
- Quel est le produit le plus cher?

Bonus: Afficher uniquement le(s) produit(s) qui sont à commander aux fournisseurs(étant donné que le stock minimal recommandé est 5 par produit).

N.B: La méthode insertOne permet d'insérer un document. Lorsque le champ `_id` n'est pas mentionné, mongoDB va créer ce champ et lui affecter une valeur, dans ce cas de type « Object ID ». L'unicité est garantie.

Même s'il est possible de stocker dans un même champ des données de type différent, ce n'est pas une bonne pratique.

Cloud Database

Load sample datasets

[Load sample Dataset](#)

```
mongosh mongodb+srv://{USERNAME}:{PASSWORD}@cluster0.rdy5gv.mongodb.net
```

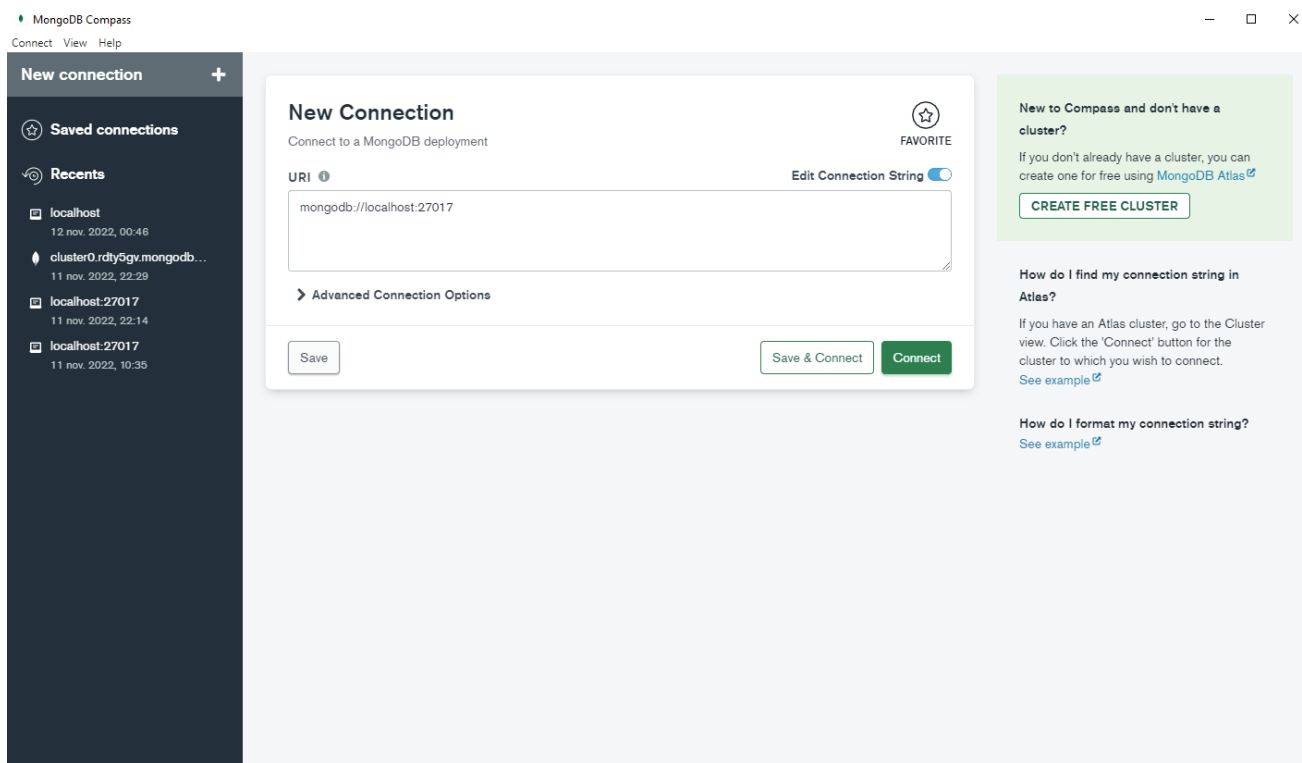
TAF: Vérifier que l'existence des données chargées. Il s'agit de 9 databases nommées suivant le pattern `sample_`

Exploration de données avec Mongo Compass

Généralités

Se connecter à sa base de données

Renseignez le lien de votre serveur



Explorer la base de données sample_analytics.

TAF: Décrire cette base de données en la comparant à une base de données relationnelle.

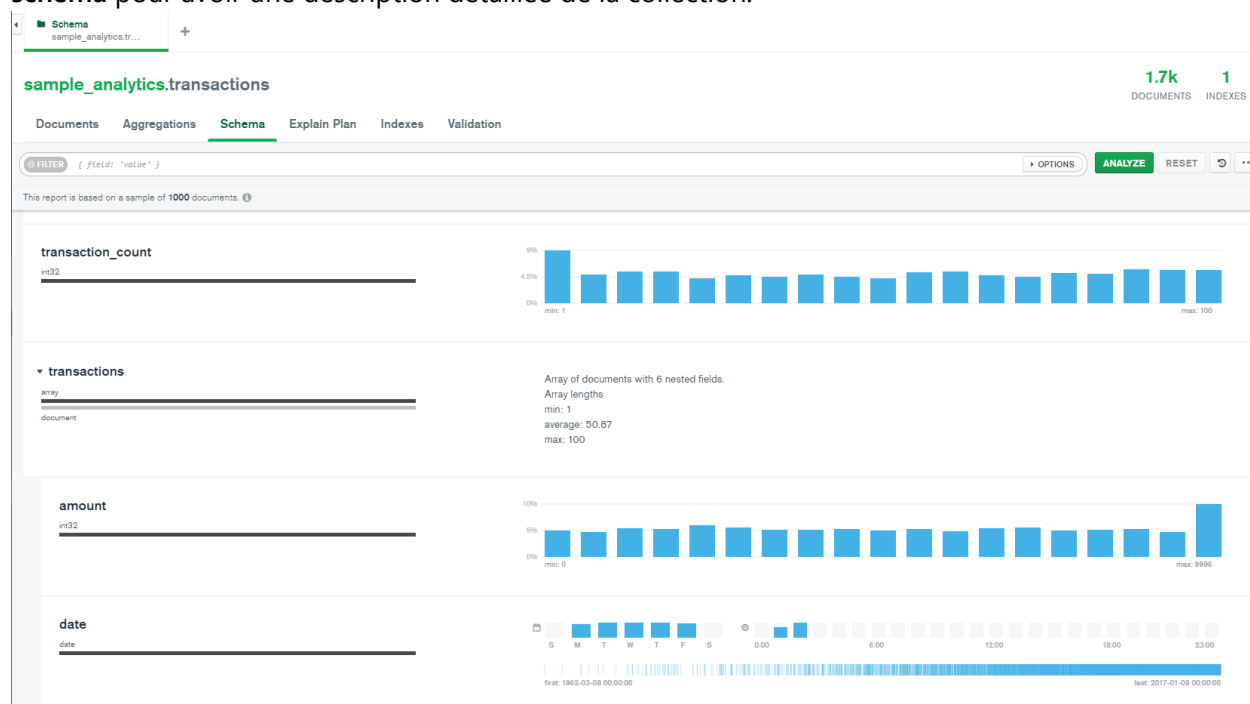
- Combien de tables y a-t-il?
- Y a-t-il des relations entre les tables? Si oui, lesquelles?
- La notion de normalisation est-elle respectée dans cette base de données? Expliquer.

TAF: Décrire les données d'un document de la collection `customers`.

- Quels sont les types de chaque champ?
- Ces types sont-ils tous utilisables dans une base de données type MySQL? Sinon, lesquels?

Analyse de schéma

Analyser le schéma de la collection `transactions` en allant vers l'onglet **Schema**. Cliquer sur **Analyze schema** pour avoir une description détaillée de la collection.



Les indicateurs et graphiques affichés sont collection sur un échantillon réduit des données de la collection.

TAF: Quel symbole intervient le plus dans les transactions?

TAF: Combien y-a til de codes de transactions? Lesquels?

TAF: A quelle heure y a il le plus de transcsactions?

Requêtes dans Mongo Compass

Pour effectuer les requêtes, retourner dans l'onglet **Documents**.

Cas d'usage:

- Je souhaite retrouver les transactions de l'utilisateur qui a ce pseudo(username): hmyers
- qui Pour retrouver les transactions liées au compte `tr`

Requêtes sur des champs simples

- Retrouver l'utilisateur et les informations disponibles sur lui/elle.

The screenshot shows the MongoDB Compass interface. The top bar indicates 500 documents and 1 index for the `sample_analytics.customers` collection. The `Documents` tab is active. A filter is applied: `{username: "hmyers"}`. The query results show a single document for a user named Dana Clarke, born in 1969, living in Wilkinsstad, PA. The document includes details about her accounts, tier, and benefits.

Il s'agit visiblement d'une femme née en Juin 1969, vivant dans le Wilkinsstad et possédant 3 comptes.

L'équivalent de cette requête sur Mongo Shell est:

```
sample_analytics>db.customer.find({username: "hmyers"})
```

- Retrouver les informations sur son premier compte 627629

The screenshot shows the MongoDB Compass interface. The top bar indicates 1.7k documents and 1 index for the `sample_analytics.accounts` collection. The `Documents` tab is active. A filter is applied: `{account_id: "627629"}`. The query results show a single document for an account with a limit of 10000 and a list of products: "Brokerage", "Derivatives", "InvestmentFund", "Commodity", and "InvestmentStock".

Ce compte a l'air d'être un compte dédié à des transactions financières.

L'équivalent de cette requête sur Mongo Shell est:

```
sample_analytics>db.accounts.find({account_id: 627629})
```

- Quels sont donc ces transactions? Combien y en a t il?

The screenshot shows the MongoDB Atlas web interface for the 'sample_analytics.transactions' collection. The 'Documents' tab is selected, and a filter is applied for 'account_id: 627629'. The interface displays 1.7k documents and 1 index. The first transaction is expanded, showing details like date, amount, transaction code, symbol, price, and total.

```
{
  "_id": ObjectId("5cabb01a2dd94ee58161cf5"),
  "account_id": 627629,
  "transaction_count": 90,
  "bucket_start_date": "1984-04-23T00:00:00.000+00:00",
  "bucket_end_date": "2017-01-08T00:00:00.000+00:00",
  "transactions": Array
    0: Object
      date: "2014-02-26T00:00:00.000+00:00"
      amount: 6939
      transaction_code: "sell"
      symbol: "znqa"
      price: "5.29144034631109949629035327234305441379547119140625"
      total: "56717.30456305271940475876136"
    1: Object
      date: "2009-08-06T00:00:00.000+00:00"
      amount: 7514
      transaction_code: "buy"
      symbol: "xduu"
      price: "32.2179019859054704966183635406196117401123046875"
      total: "242085.3155220937053115903836"
    2: Object
      date: "2014-10-13T00:00:00.000+00:00"
      amount: 8634
      transaction_code: "sell"
      symbol: "gooo"
      price: "538.1654192179040592234560769879138946533203125"
      total: "4646520.229527393647337046568"
    3: Object
      date: "2010-06-17T00:00:00.000+00:00"
      amount: 9159
      transaction_code: "sell"
      symbol: "aaPl"
      price: "84.90421145303967745318681509234011173248291018625"
      total: "264793.4612453506510193512243"
    4: Object
      date: "2015-07-16T00:00:00.000+00:00"
      amount: 5660
      transaction_code: "buy"
      symbol: "aaPl"
      price: "121.672388381830280399642162956297397613525390625"
      total: "688665.7182411594303061974642"
  }
```

Il semble y avoir 90 transactions dans ce compte. De type "buy" mais aussi de type "sell".

L'équivalent de cette requête sur Mongo Shell est:

```
sample_analytics>db.transactions.find({account_id: 627629})
```

Nous pourrions faire une analyse plus détaillée par la suite.

Les opérateurs de comparaison

Documentation : <https://docs.mongodb.com/manual/reference/operator/query-comparison/>

- J'aimerais savoir quels comptes effectuent autant de transaction que Dana avec une marge de +/- 10 transactions. Ce sont peut-être des utilisateurs avec un profil similaire. Pour l'instant, je ne veux que

les comptes avec 80,90 ou 100 transactions.

sample_analytics.transactions

1.7k1

DOCUMENTSINDEXES

DocumentsAggregationsSchemaExplain PlanIndexesValidation

1 FILTER

[transaction_count: {\$in: [80,90,100]}]

OPTIONS

FIND

RESET

...

ADD DATA

VIEW

0

Displaying documents 1 - 20 of 50

REFRESH

_id: ObjectId("5ca4bbca2dd94ee58161cbb")

account_id: 278866

transaction_count: 100

bucket_start_date: 1962-12-12T00:00:00.000+00:00

bucket_end_date: 2017-01-09T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161cfe")

account_id: 627629

transaction_count: 90

bucket_start_date: 1984-04-23T00:00:00.000+00:00

bucket_end_date: 2017-01-08T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161d36")

account_id: 630581

transaction_count: 80

bucket_start_date: 1986-03-07T00:00:00.000+00:00

bucket_end_date: 2017-01-09T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161d60")

account_id: 356033

transaction_count: 90

bucket_start_date: 1985-12-23T00:00:00.000+00:00

bucket_end_date: 2016-12-30T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161dad")

account_id: 531283

transaction_count: 90

bucket_start_date: 1963-03-15T00:00:00.000+00:00

bucket_end_date: 2017-01-08T00:00:00.000+00:00

transactions: Array

Pour retrouver ce nombre dans Mongo Shell:

```
sample_analytics>db.transactions.find({transaction_count: {$in: [80,90,100]}}).count()
50
```

50 comptes ont fait exactement ce nombre de transactions.

Les opérateurs logiques

Documentation : <https://docs.mongodb.com/manual/reference/operator/query-logical/>

- Cette estimation n'est pas vraiment fine, j'aimerais savoir concrètement combien de comptes font entre 80 et 100 transactions

sample_analytics.transactions

1.7k1

DOCUMENTSINDEXES

DocumentsAggregationsSchemaExplain PlanIndexesValidation

1 FILTER

[\$and: [[transaction_count:{\$gte:80}], [transaction_count:{\$lte:100}]]

OPTIONS

FIND

RESET

...

ADD DATA

VIEW

0

Displaying documents 1 - 20 of 369

REFRESH

_id: ObjectId("5ca4bbca2dd94ee58161cb4")

account_id: 278603

transaction_count: 83

bucket_start_date: 1975-06-02T00:00:00.000+00:00

bucket_end_date: 2017-01-04T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161cbb")

account_id: 278866

transaction_count: 100

bucket_start_date: 1962-12-12T00:00:00.000+00:00

bucket_end_date: 2017-01-09T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161cbe")

account_id: 831097

transaction_count: 99

bucket_start_date: 1965-01-26T00:00:00.000+00:00

bucket_end_date: 2016-12-30T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161cc3")

account_id: 116505

transaction_count: 81

bucket_start_date: 1986-04-26T00:00:00.000+00:00

bucket_end_date: 2016-12-30T00:00:00.000+00:00

transactions: Array

_id: ObjectId("5ca4bbca2dd94ee58161cce")

account_id: 387979

transaction_count: 95

bucket_start_date: 1973-08-24T00:00:00.000+00:00

bucket_end_date: 2016-12-21T00:00:00.000+00:00

transactions: Array

Pour retrouver ce nombre dans Mongo Shell:

```
sample_analytics>db.transactions.find(
  {$and: [
    {transaction_count:{$gte:80}},
    {transaction_count:{$lte:100}}]}]
).count()
369
```

J'en ai 369 au total.

Tri des résultats

- Ma décision est prise, je souhaite entrer en contact avec les utilisateurs de ces comptes. Ce pendant il y en a 369. Je vais donc prioriser mes envois de mails.

Pour cela, j'ai besoin de contacter en premier les comptes avec les plus de transactions et parmi ces derniers, je vais démarrer pas les plus anciens comptes.

sample_analytics.transactions 1.7k 1
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {\$and: [{transaction_count:{\$gte:80}}, {transaction_count:{\$lte:100}}]} **OPTIONS** **FIND** **RESET** **...**

PROJECT { field: 0 }

SORT {transaction_count:-1, bucket_start_date:1} **MAX TIME MS** 60000

COLLATION { locale: 'simple' } **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **...**

Displaying documents 1 - 20 of 369 **REFRESH**

```
{
  "_id": ObjectId("5ca4bb1a2dd94ee58162361"),
  "account_id": 880595,
  "transaction_count": 100,
  "bucket_start_date": 1962-06-18T00:00:00.000+00:00,
  "bucket_end_date": 2016-12-17T00:00:00.000+00:00,
  "transactions": Array
}
```

```
{
  "_id": ObjectId("5ca4bb1a2dd94ee58162054"),
  "account_id": 408145,
  "transaction_count": 100,
  "bucket_start_date": 1962-06-28T00:00:00.000+00:00,
  "bucket_end_date": 2017-01-02T00:00:00.000+00:00,
  "transactions": Array
}
```

```
{
  "_id": ObjectId("5ca4bb1a2dd94ee58161cbb"),
  "account_id": 278866,
  "transaction_count": 100,
  "bucket_start_date": 1962-12-12T00:00:00.000+00:00,
  "bucket_end_date": 2017-01-09T00:00:00.000+00:00,
  "transactions": Array
}
```

```
{
  "_id": ObjectId("5ca4bb1a2dd94ee581620a7"),
  "account_id": 134434,
  "transaction_count": 100,
  "bucket_start_date": 1970-08-23T00:00:00.000+00:00,
  "bucket_end_date": 2016-12-29T00:00:00.000+00:00,
  "transactions": Array
}
```

Pour retrouver cette liste via Mongo Shell, exécuter la commande suivante:

```
sample_analytics>db.transactions.find(
  {$and: [
    {transaction_count:{$gte:80}},
    {transaction_count:{$lte:100}}]}]
).sort({
  transaction_count:-1,
  bucket_start_date:1})
```

Recherche dans des documents imbriqués

- Un autre paramètre déterminant dans ma décision est l'indice qu'achètent ou vendent mes comptes cibles. Pour optimiser mes chances de réussite, je choisis de contacter des personnes qui ont investi dans des boîtes au moins une boîte tech notamment **Google**.

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER `{ $and: [{ transaction_count: { $gte: 80 } }, { transaction_count: { $lte: 100 } }, { "transactions.symbol": "goog" }] }` **OPTIONS** **FIND** **RESET**

PROJECT `{ field: 0 }`

SORT `{ transaction_count: -1, bucket_start_date: 1 }` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **REFRESH** Displaying documents 1 - 20 of 134

```

{
  "_id": ObjectId("5ca4bb1a2dd94ee58161cbb"),
  "account_id": 278966,
  "transaction_count": 100,
  "bucket_start_date": 1962-12-12T00:00:00.000+00:00,
  "bucket_end_date": 2017-01-09T00:00:00.000+00:00,
  "transactions": Array
}

{
  "_id": ObjectId("5ca4bb1a2dd94ee58162097"),
  "account_id": 845284,
  "transaction_count": 100,
  "bucket_start_date": 1972-02-19T00:00:00.000+00:00,
  "bucket_end_date": 2017-01-07T00:00:00.000+00:00,
  "transactions": Array
}

{
  "_id": ObjectId("5ca4bb1a2dd94ee581620b6"),
  "account_id": 77397,
  "transaction_count": 100,
  "bucket_start_date": 1982-12-10T00:00:00.000+00:00,
  "bucket_end_date": 2016-12-30T00:00:00.000+00:00,
  "transactions": Array
}

{
  "_id": ObjectId("5ca4bb1a2dd94ee5816219c"),
  "account_id": 972116,
  "transaction_count": 100,
  "bucket_start_date": 1989-01-18T00:00:00.000+00:00,
  "bucket_end_date": 2017-01-08T00:00:00.000+00:00,
  "transactions": Array
}

```

Pour retrouver cette liste via Mongo Shell, exécuter la commande suivante:

```

sample_analytics>db.transactions.find(
    { $and: [
        { transaction_count: { $gte: 80 } },
        { transaction_count: { $lte: 100 } },
        { "transactions.symbol": "goog" } ] }
    ).sort({
        transaction_count: -1,
        bucket_start_date: 1 }).count()
134

```

En appliquant un count à la fin de la requête, l'on remarque que ma liste de compte est réduite à 134.

Recherche de documents ayant des champs non renseignés

Rechercher géospatiale

Recherche sur des tableaux (Array)

- Je souhaite exclure des comptes que je vais contacter les portefeuilles:

- Non diversifiés: La requête à taper dans Compass est: `{products: {$not : {$size: 1}}}`

sample_analytics.accounts 1.7k 1 DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {products: {\$not : {\$size: 1}}}

PROJECT { field: 0 }

SORT { field: -1 } or [{"field", -1}]

COLLATION { locale: 'simple' }

MAX TIME MS 60000 SKIP 0 LIMIT 0

ADD DATA VIEW

Displaying documents 1 - 20 of 1684

```

_id: ObjectId('5ca4bbc7a2dd94ee5816239d')
account_id: 371138
limit: 9000
products: Array
  0: "Derivatives"
  1: "InvestmentStock"

_id: ObjectId('5ca4bbc7a2dd94ee5816239d')
account_id: 557378
limit: 10000
products: Array
  0: "InvestmentStock"
  1: "Commodity"
  2: "Brokerage"
  3: "CurrencyService"

_id: ObjectId('5ca4bbc7a2dd94ee5816239e')
account_id: 198100
limit: 10000
products: Array
  0: "Derivatives"
  1: "CurrencyService"
  2: "InvestmentStock"

```

- Comportant des produits dérivés: La requête à taper dans Compass est: `{products: {$ne: "Derivatives"}}`

sample_analytics.accounts 1.7k 1 DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {products: {\$ne: "Derivatives"}}

PROJECT { field: 0 }

SORT { field: -1 } or [{"field", -1}]

COLLATION { locale: 'simple' }

MAX TIME MS 60000 SKIP 0 LIMIT 0

ADD DATA VIEW

Displaying documents 1 - 20 of 1040

```

_id: ObjectId('5ca4bbc7a2dd94ee5816239d')
account_id: 557378
limit: 10000
products: Array
  0: "InvestmentStock"
  1: "Commodity"
  2: "Brokerage"
  3: "CurrencyService"

_id: ObjectId('5ca4bbc7a2dd94ee5816239f')
account_id: 674364
limit: 10000
products: Array
  0: "InvestmentStock"

_id: ObjectId('5ca4bbc7a2dd94ee58162390')
account_id: 278603
limit: 10000
products: Array
  0: "Commodity"
  1: "InvestmentStock"

_id: ObjectId('5ca4bbc7a2dd94ee58162392')
account_id: 794875
limit: 9000
products: Array
  0: "InvestmentFund"
  1: "InvestmentStock"

```

Pour exclure ces 2 catégories, nous avons donc: `{$and: [{products: {$not : {$size: 1}}},{products: {$ne: "Derivatives"}}]}`

sample_analytics.accounts

1.7k 1
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER `{ $and: [{ products: { $not: { $size: 1 } } }, { products: { $ne: "Derivatives" } }] }` **OPTIONS** **FIND** **RESET** **...**

PROJECT `{ field: 0 }`

SORT `{ field: -1 } or [['field', -1]]` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 8

ADD DATA **VIEW** **...**

Displaying documents 1 - 20 of 978 **REFRESH**

```
{
  "_id": ObjectId("50a4bbc7a2dd94ee5816238d"),
  "account_id": 557378,
  "limit": 10000,
  "products": Array
    0: "InvestmentStock"
    1: "Commodity"
    2: "Brokerage"
    3: "CurrencyService"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162390"),
  "account_id": 278603,
  "limit": 10000,
  "products": Array
    0: "Commodity"
    1: "InvestmentStock"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162392"),
  "account_id": 794875,
  "limit": 9000,
  "products": Array
    0: "InvestmentFund"
    1: "InvestmentStock"
}
```

Je souhaite avoir les comptes qui ont le produit **Brokerage** en premier : `{"products.0": "Brokerage"}`

sample_analytics.accounts

1.7k 1
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER `{ "products.0": "Brokerage" }` **OPTIONS** **FIND** **RESET** **...**

PROJECT `{ field: 0 }`

SORT `{ field: -1 } or [['field', -1]]` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **...**

Displaying documents 1 - 20 of 305 **REFRESH**

```
{
  "_id": ObjectId("50a4bbc7a2dd94ee58162394"),
  "account_id": 457188,
  "limit": 10000,
  "products": Array
    0: "Brokerage"
    1: "CurrencyService"
    2: "InvestmentStock"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162395"),
  "account_id": 910579,
  "limit": 10000,
  "products": Array
    0: "Brokerage"
    1: "InvestmentStock"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162397"),
  "account_id": 668949,
  "limit": 10000,
  "products": Array
    0: "Brokerage"
    1: "CurrencyService"
    2: "InvestmentStock"
    3: "Derivatives"
}
```

J'aimerais aussi savoir combien de comptes ont le produit : **investmentFund**: `{"products": "InvestmentFund"}`

sample_analytics.accounts

1.7k 1
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER `{ "products": "InvestmentFund" }` **OPTIONS** **FIND** **RESET** **...**

PROJECT `{ field: 0 }`

SORT `{ field: -1 } or [['field', -1]]` **MAX TIME MS** 60000

COLLATION `{ locale: 'simple' }` **SKIP** 0 **LIMIT** 0

ADD DATA **VIEW** **...**

Displaying documents 1 - 20 of 728 **REFRESH**

```
{
  "_id": ObjectId("50a4bbc7a2dd94ee58162391"),
  "account_id": 383777,
  "limit": 10000,
  "products": Array
    0: "CurrencyService"
    1: "Derivatives"
    2: "InvestmentFund"
    3: "Commodity"
    4: "InvestmentStock"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162392"),
  "account_id": 794875,
  "limit": 9000,
  "products": Array
    0: "InvestmentFund"
    1: "InvestmentStock"
}

{
  "_id": ObjectId("50a4bbc7a2dd94ee58162396"),
  "account_id": 260499,
  "limit": 10000,
  "products": Array
    0: "InvestmentFund"
    1: "Derivatives"
    2: "InvestmentStock"
}
```

Projections

Les projections permettent de limiter les champs à afficher.

Lorsque je cherche les comptes qui ont le produit **investmentFund**, l'information finale qui m'intéresse l'identifiant unique de chaque compte afin de les retrouver dans la collections des clients.

Le champ `_id` est inclus par défaut dans le résultat. Dans la requête de projection `{_id:0, account_id:1}` ci-dessous, le champ `account_id` est retenu, le champ `_id` est exclu.

sample_analytics.accounts 1.7k 1
DOCUMENTS INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER {"products": "InvestmentFund"} **PROJECT** {_id:0, account_id:1} **MAX TIME MS** 60000 **SKIP** 0 **LIMIT** 0

VIEW

Displaying documents 1 - 20 of 728

account_id: 383777
account_id: 794875
account_id: 260499
account_id: 299072
account_id: 977982

Dans Mongo Shell, l'équivalent de cette requête est:

```
sample_analytics>db.accounts.find(
  {"products": "InvestmentFund"},
  {_id:0, account_id:1}
)
[
  { account_id: 383777 }, { account_id: 794875 },
  { account_id: 260499 }, { account_id: 299072 },
  { account_id: 977982 }, { account_id: 212024 },
  { account_id: 433811 }, { account_id: 403363 },
  { account_id: 276528 }, { account_id: 383701 },
  { account_id: 463155 }, { account_id: 895735 },
  { account_id: 984021 }, { account_id: 503933 },
  { account_id: 475387 }, { account_id: 775690 },
  { account_id: 136137 }, { account_id: 522933 },
  { account_id: 785786 }, { account_id: 462501 }
]
```

Update

Par appel téléphonique, la cliente portant le nom "Katherine David" m'a indiqué un changement d'adresse e-mail..

Sa nouvelle adresse est `katherine.david@gmail.com`.

```
sample_analytics>db.customers.updateOne(
  { name: "Katherine David"},
  { $set: { email: "katherine.david@gmail.com"} }
)
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

Delete

Le service juridique m'informe de la volonté d'un client de supprimer ses données client. Ils'agit de Brad Cardenas.

```
sample_analytics>db.customers.deleteOne({"name": "Brad Cardenas"})  
{ acknowledged: true, deletedCount: 1 }
```