Comment exécuter des tâches de réduction de carte MongoDB

Publié le : 13/07/2018 ( dernière mise à jour : 04/08/2021) par Thomas Zahn

Dans cet article, nous allons vous montrer le moyen le plus simple d'écrire, de déboguer et d'exécuter des tâches de réduction de carte MongoDB à l'aide de l'écran Map-Reduce de Studio 3T.

Vous n'avez pas Studio 3T sur votre machine ? Téléchargez-le ici , disponible pour Windows, Mac et Linux.

MongoDB Map-Reduce vs Pipeline d'agrégation

Map-Reduce de MongoDB est le cousin flexible du pipeline d'agrégation.

En général, cela fonctionne en faisant passer les données en deux étapes :

une étape de carte qui traite chaque document et émet un ou plusieurs objets pour chaque document d'entrée

une étape de réduction qui combine les objets émis à partir de la sortie de l'opération de carte

Le principal avantage par rapport au pipeline d'agrégation est que Map-Reduce peut utiliser du JavaScript arbitraire pour chaque étape, permettant des opérations autrement impossibles, mais au détriment des performances (temps d'exécution potentiellement plus élevés). Vous pouvez en savoir plus à ce sujet dans la documentation de référence de MongoDB .

MongoDB recommande le pipeline d'agrégation pour la plupart des options d'agrégation. Comme alternative à Map-Reduce, veuillez également consulter Aggregation Editor , le générateur de requêtes d'agrégation MongoDB de Studio 3T.

{

"\_id" : 592341,

"Mots clés" : [

"chats",

"chatons",

"voyager"

]

}

Un exemple de réduction de carte

Dans cet exemple, notre objectif est de regrouper les images par balise à l'exception de celles qui comportent la balise « travail ».

Pour y parvenir, nous devrons écrire un travail Map-Reduce qui :

Excluez toutes les images qui incluent la balise « travail ».

Demandez à la map() fonction d'émettre l'ID d'image pour chacune des balises en tant que clé.

Demandez à la reduce() fonction de combiner les identifiants d'image pour chaque balise.

Commençons par ouvrir le nouvel écran Map-Reduce de Studio 3T en sélectionnant l' option Open Map-Reduce dans le menu contextuel :

Ouvrir Map-Reduce dans MongoChef

Ouverture de Map-Reduce dans Studio 3T

Filtrage des données d'entrée

En cliquant sur l' onglet « Données d'entrée » puis sur le bouton de la barre d'outils « Aperçu de l'entrée » nous montre un aperçu des données de collecte. C'est ici que nous pouvons façonner les données introduites dans le travail Map-Reduce et omettre toute image étiquetée "travail". Ceci est réalisé par la requête suivante

{ "tags": { $ne: "work" } }

Nous pouvons inspecter les données qui seront introduites dans la fonction de carte en cliquant sur le bouton de la barre d'outils « Aperçu de la sortie ».

Exemple de données d'entrée

Exemple de données d'entrée

fonction map()

Pour la deuxième étape, nous passons à l' onglet « map() ».

Dans cet onglet, nous voulons spécifier la fonction responsable de l'émission d'une ou plusieurs paires de clés pour chaque document. La fonction suivante fait le travail :

fonction () {

for (index var dans this.tags) {

émet ( this.tags[index] , this.\_id );

}

}

Nous pouvons échantillonner la map() sortie de la fonction en cliquant sur le bouton d' aperçu , en vérifiant que cette fonction a réussi. La fonction de prévisualisation est extrêmement utile, en particulier avant de soumettre des travaux qui peuvent prendre des heures à s'exécuter. L' onglet " map() sample output " nous donne une ventilation détaillée du fonctionnement de notre map() fonction, montrant les paires clé/valeur émises ainsi que leur document d'origine \_id.

Aperçu de la sortie de la carte

Aperçu de la sortie de la carte

fonction réduire ()

L'implémentation par défaut de la reduce() fonction dans Studio 3T s'occupe du reste :

fonction (clé, valeurs) {

var ReducedValue = "" + valeurs ;

retourner la valeurréduite ;

}

Encore une fois, le bouton de la barre d'outils Aperçu de la sortie nous permettra de vérifier que notre fonction est réussie. Si nous écrivions une reduce() fonction plus complexe ou essayions de déboguer ce qui était introduit, nous pourrions échantillonner l'entrée en cliquant sur le bouton Aperçu de l'entrée . Cela nous donne quelques-unes des paires clé-valeur qui sont émises puis réduites.

Réduire l'entrée d'échantillon

Réduire l'entrée d'échantillon

fonction finalize()

MongoDB permet une étape finale à un travail Map-Reduce pour effectuer un traitement final avec l'utilisation d'une finalize() fonction. Utilisons ceci juste pour que la sortie soit plus facile à lire :

fonction (clé, valeurréduite) {

var finalValue = "tag '" + clé + "' a été trouvé dans les images : " + reduceValue;

renvoyer la valeur finale ;

}

Une inspection rapide de finalize()l'exemple de sortie de et nous sommes prêts à soumettre un travail qui traitera toutes les données.

Finaliser l'exemple de sortie

Finaliser l'exemple de sortie

Exécuter la tâche de réduction de carte

Maintenant que nous avons défini tous les paramètres du travail et que nous sommes sûrs que toutes nos fonctions s'exécutent comme prévu, nous pouvons soumettre le travail Map-Reduce pour qu'il s'exécute sur l'ensemble de la collection de données en cliquant sur le bouton « Exécuter » dans la barre d'outils.

Cette action ouvrira un nouvel onglet qui contiendra les résultats du travail lorsqu'il sera terminé :

Travail fini

Travail terminé !

Cliquer sur Show details fera apparaître une boîte de dialogue affichant les statistiques d'exécution ainsi qu'un résumé de configuration pour ce travail.

Statistiques d'emploi

Statistiques de tâche Map-Reduce terminées

Épilogue

Maintenant que le travail Map-Reduce est terminé, nous pouvons enregistrer tout ce travail sous forme de script. Le format est un code 100% JavaScript, ce qui permet d'exécuter le fichier enregistré dans IntelliShell ou même le shell mongo de base et produira des résultats identiques.

// \*\*\* 3T Software Labs, MongoChef : Job MapReduce \*\*\*\*

// Variable pour la base de données

var \_\_3t\_mongochef\_db = "examen" ;

// Variable pour la carte

var \_\_3t\_mongochef\_map = fonction () {

for (index var dans this.tags) {

émet ( this.tags[index] , this.\_id );

}

}

;

// Variable pour réduire

var \_\_3t\_mongochef\_reduce = fonction (clé, valeurs) {

var ReducedValue = "" + valeurs ;

retourner la valeurréduite ;

} ;

// Variable pour finaliser

var \_\_3t\_mongochef\_finalize = fonction (clé, ValeurRéduite) {

var finalValue = "tag '" + clé + "' a été trouvé dans les images : " + reduceValue;

renvoyer la valeur finale ;

}

;

db.runCommand({

mapReduce: "images",

carte : \_\_3t\_mongochef\_map,

réduire : \_\_3t\_mongochef\_reduce,

finaliser : \_\_3t\_mongochef\_finalize,

sortie : { "en ligne" : 1},

requête : { "tags": { $ne: "work" } },

sorte: { },

inputDB : "examen",

});

Avez-vous un script existant avec lequel vous avez déjà travaillé ? Pas de problème, Studio 3T le chargera dans l'écran Map-Reduce, cliquez simplement sur le bouton de la barre d'outils « Ouvrir le fichier Map-Reduce », sélectionnez le fichier et le tour est joué !

Une fois que vous avez terminé d'exécuter les tâches de réduction de carte MongoDB, pourquoi ne pas continuer sur votre lancée en apprenant à créer des requêtes d'agrégation MongoDB et en découvrant notre intégration de shell MongoDB, IntelliShell .

# **Comment exécuter des tâches de réduction de carte MongoDB**

**Publié le :** 13/07/2018 ( **dernière mise à jour :** 04/08/2021) par [Thomas Zahn](https://studio3t-com.translate.goog/whats-new/author/thomas-zahn/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc)

Dans cet article, nous allons vous montrer le moyen le plus simple d'écrire, de déboguer et d'exécuter des tâches de réduction de carte MongoDB à l'aide de l'écran Map-Reduce de Studio 3T.

Vous n'avez pas Studio 3T sur votre machine ? Téléchargez-le [ici](https://studio3t-com.translate.goog/download/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc) , disponible pour Windows, Mac et Linux.

## [**MongoDB Map-Reduce vs Pipeline d'agrégation**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#mongodb-map-reduce-vs-aggregation-pipeline)

Map-Reduce de MongoDB est le cousin flexible du pipeline d'agrégation.

En général, cela fonctionne en faisant passer les données en deux étapes :

* une étape de **carte** qui traite chaque document et émet un ou plusieurs objets pour chaque document d'entrée
* une étape de **réduction** qui combine les objets émis à partir de la sortie de l'opération de carte

Le principal avantage par rapport au pipeline d'agrégation est que Map-Reduce peut utiliser du JavaScript arbitraire pour chaque étape, permettant des opérations autrement impossibles, mais au détriment des performances (temps d'exécution potentiellement plus élevés). Vous pouvez en savoir plus à ce sujet dans  [la documentation de référence de MongoDB](https://translate.google.com/website?sl=en&tl=fr&nui=1&prev=search&u=https://docs.mongodb.org/manual/core/map-reduce/) .

MongoDB recommande le pipeline d'agrégation pour la plupart des options d'agrégation. Comme alternative à Map-Reduce, veuillez également consulter [Aggregation Editor](https://studio3t-com.translate.goog/knowledge-base/articles/aggregation-editor/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc) , le générateur de requêtes d'agrégation MongoDB de Studio 3T.

{

"\_id" : 592341,

"Mots clés" : [

"chats",

"chatons",

"voyager"

]

}

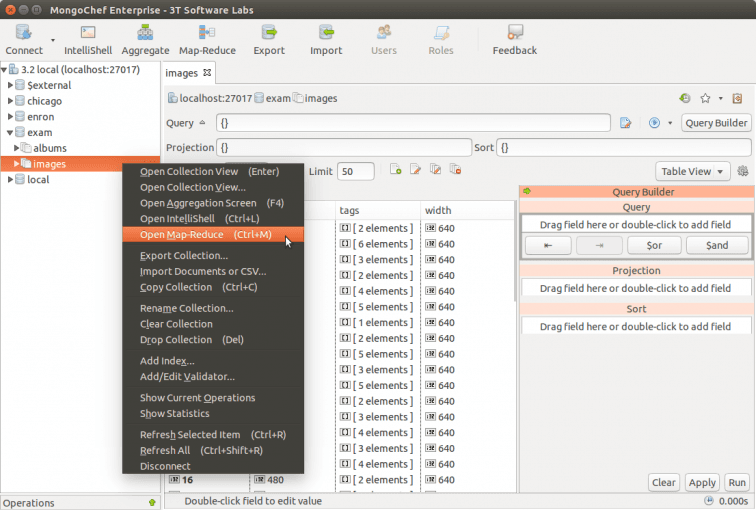
## [**Un exemple de réduction de carte**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#a-map-reduce-example)

Dans cet exemple, notre objectif est de regrouper les images par balise à l'exception de celles qui comportent la balise « travail ».

Pour y parvenir, nous devrons écrire un travail Map-Reduce qui :

1. Excluez toutes les images qui incluent la balise « travail ».
2. Demandez à la map() fonction d'émettre l'ID d'image pour chacune des balises en tant que clé.
3. Demandez à la reduce() fonction de combiner les identifiants d'image pour chaque balise.

Commençons par ouvrir le nouvel écran Map-Reduce de Studio 3T en sélectionnant l' option **Open Map-Reduce** dans le menu contextuel :

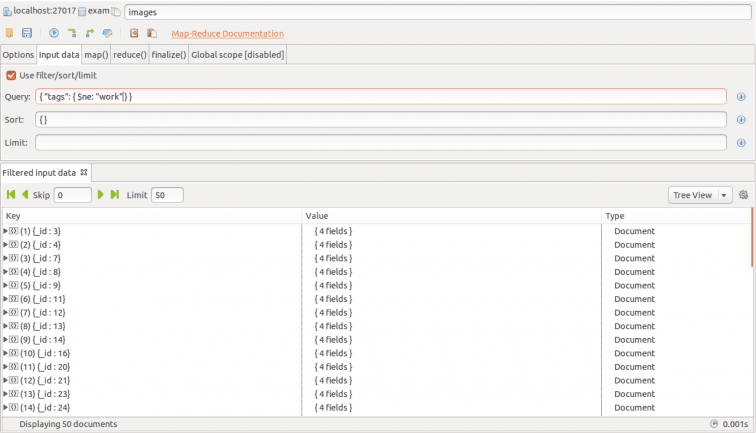
 Ouverture de Map-Reduce dans Studio 3T

### [**Filtrage des données d'entrée**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#filtering-the-input-data)

En cliquant sur l' onglet « **Données d'entrée** » puis sur le bouton de la barre d'outils « **Aperçu de l'entrée** » nous montre un aperçu des données de collecte. C'est ici que nous pouvons façonner les données introduites dans le travail Map-Reduce et omettre toute image étiquetée "travail". Ceci est réalisé par la requête suivante

{ "tags": { $ne: "work" } }

Nous pouvons inspecter les données qui seront introduites dans la fonction de carte en cliquant sur le bouton de la barre d'outils « **Aperçu de la sortie** ».

 Exemple de données d'entrée

### [**fonction map()**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#map-function)

Pour la deuxième étape, nous passons à l' onglet « **map()** ».

Dans cet onglet, nous voulons spécifier la fonction responsable de l'émission d'une ou plusieurs paires de clés pour chaque document. La fonction suivante fait le travail :

fonction () {

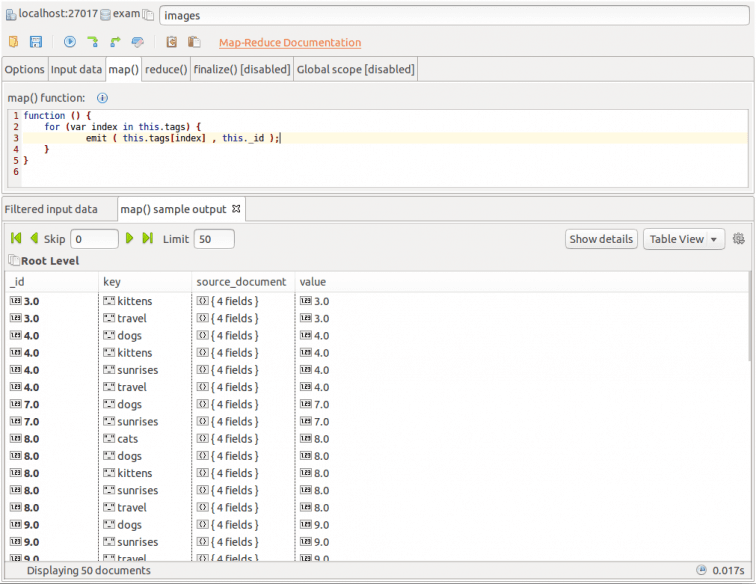
for (index var dans this.tags) {

émet ( this.tags[index] , this.\_id );

}

}

Nous pouvons échantillonner la map() sortie de la fonction en cliquant sur le bouton d' **aperçu** , en vérifiant que cette fonction a réussi. La fonction de prévisualisation est extrêmement utile, en particulier avant de soumettre des travaux qui peuvent prendre des heures à s'exécuter. L' onglet " **map() sample output** " nous donne une ventilation détaillée du fonctionnement de notre map() fonction, montrant les paires clé/valeur émises ainsi que leur document d'origine \_id.

 Aperçu de la sortie de la carte

### [**fonction réduire ()**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#reduce-function)

L'implémentation par défaut de la reduce() fonction dans Studio 3T s'occupe du reste :

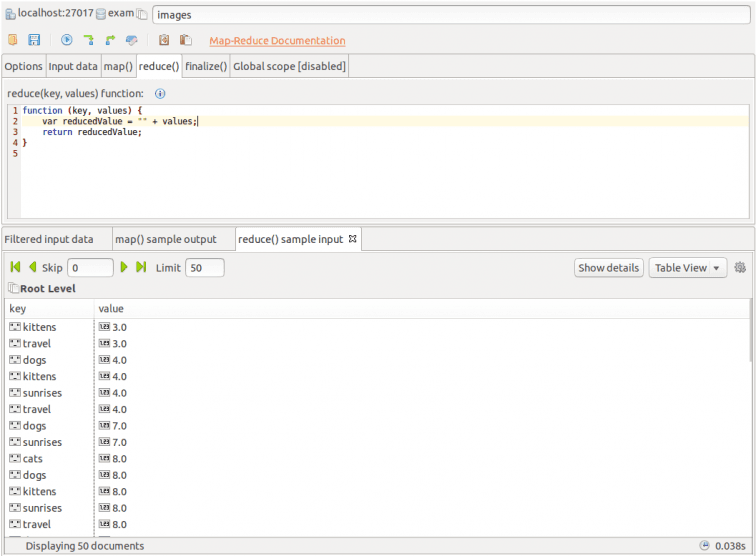
fonction (clé, valeurs) {

var ReducedValue = "" + valeurs ;

retourner la valeurréduite ;

}

Encore une fois, le bouton de la barre d'outils **Aperçu de la sortie** nous permettra de vérifier que notre fonction est réussie. Si nous écrivions une reduce() fonction plus complexe ou essayions de déboguer ce qui était introduit, nous pourrions échantillonner l'entrée en cliquant sur le bouton **Aperçu de l'entrée** . Cela nous donne quelques-unes des paires clé-valeur qui sont émises puis réduites.

 Réduire l'entrée d'échantillon

### [**fonction finalize()**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#finalize-function)

MongoDB permet une étape finale à un travail Map-Reduce pour effectuer un traitement final avec l'utilisation d'une finalize() fonction. Utilisons ceci juste pour que la sortie soit plus facile à lire :

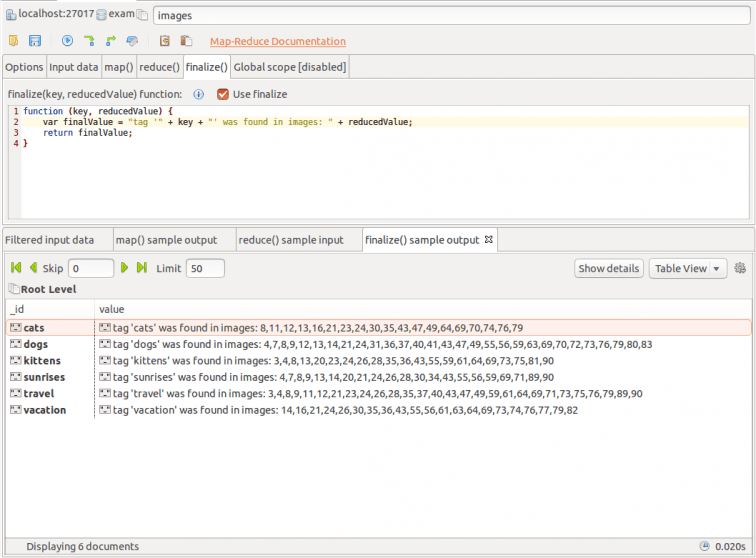
fonction (clé, valeurréduite) {

var finalValue = "tag '" + clé + "' a été trouvé dans les images : " + reduceValue;

renvoyer la valeur finale ;

}

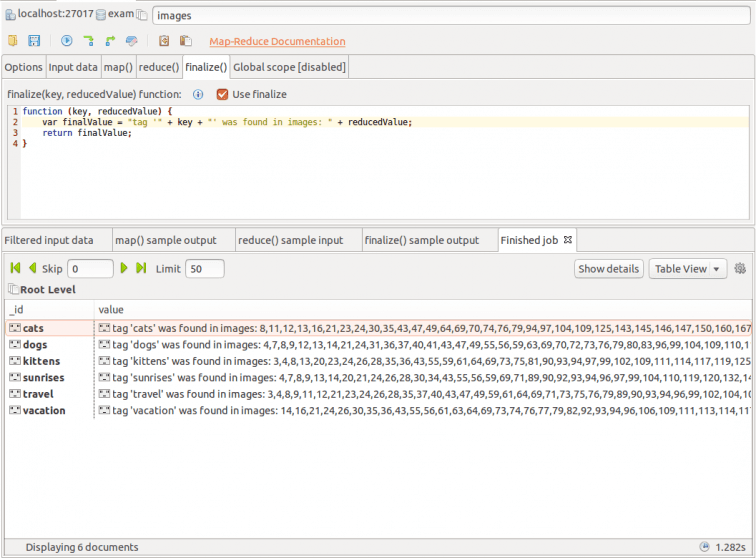
Une inspection rapide de finalize()l'exemple de sortie de et nous sommes prêts à soumettre un travail qui traitera toutes les données.

 Finaliser l'exemple de sortie

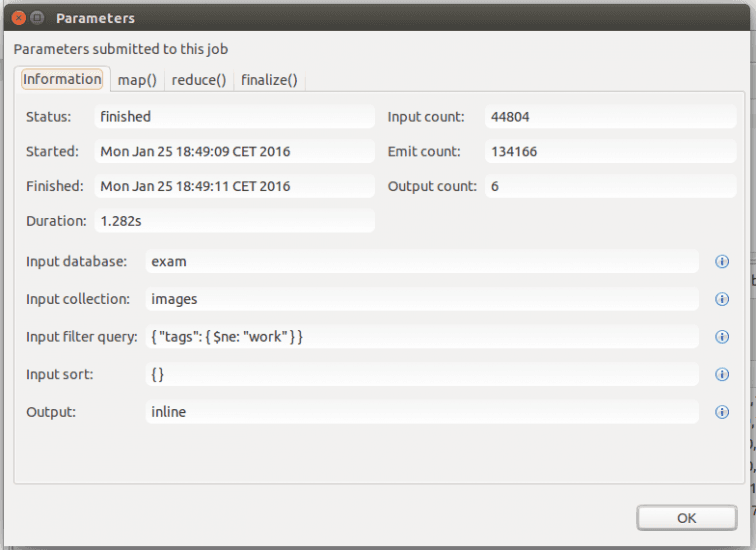
### [**Exécuter la tâche de réduction de carte**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#running-the-map-reduce-job)

Maintenant que nous avons défini tous les paramètres du travail et que nous sommes sûrs que toutes nos fonctions s'exécutent comme prévu, nous pouvons soumettre le travail Map-Reduce pour qu'il s'exécute sur l'ensemble de la collection de données en cliquant sur le bouton « **Exécuter** » dans la barre d'outils.

Cette action ouvrira un nouvel onglet qui contiendra les résultats du travail lorsqu'il sera terminé :

 Travail terminé !

Cliquer sur **Show details** fera apparaître une boîte de dialogue affichant les statistiques d'exécution ainsi qu'un résumé de configuration pour ce travail.

 Statistiques de tâche Map-Reduce terminées

### [**Épilogue**](https://studio3t-com.translate.goog/knowledge-base/articles/run-mongodb-map-reduce-jobs/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc#epilogue)

Maintenant que le travail Map-Reduce est terminé, nous pouvons enregistrer tout ce travail sous forme de script. Le format est un code 100% JavaScript, ce qui permet d'exécuter le fichier enregistré dans IntelliShell ou même le shell mongo de base et produira des résultats identiques.

// \*\*\* 3T Software Labs, MongoChef : Job MapReduce \*\*\*\*

// Variable pour la base de données

var \_\_3t\_mongochef\_db = "examen" ;

// Variable pour la carte

var \_\_3t\_mongochef\_map = fonction () {

for (index var dans this.tags) {

émet ( this.tags[index] , this.\_id );

}

}

;

// Variable pour réduire

var \_\_3t\_mongochef\_reduce = fonction (clé, valeurs) {

var ReducedValue = "" + valeurs ;

retourner la valeurréduite ;

} ;

// Variable pour finaliser

var \_\_3t\_mongochef\_finalize = fonction (clé, ValeurRéduite) {

var finalValue = "tag '" + clé + "' a été trouvé dans les images : " + reduceValue;

renvoyer la valeur finale ;

}

;

db.runCommand({

mapReduce: "images",

carte : \_\_3t\_mongochef\_map,

réduire : \_\_3t\_mongochef\_reduce,

finaliser : \_\_3t\_mongochef\_finalize,

sortie : { "en ligne" : 1},

requête : { "tags": { $ne: "work" } },

sorte: { },

inputDB : "examen",

});

Avez-vous un script existant avec lequel vous avez déjà travaillé ? Pas de problème, Studio 3T le chargera dans l'écran Map-Reduce, cliquez simplement sur le bouton de la barre d'outils « Ouvrir le fichier **Map-Reduce** », sélectionnez le fichier et le tour est joué !

Une fois que vous avez terminé d'exécuter les tâches de réduction de carte MongoDB, pourquoi ne pas continuer sur votre lancée en apprenant  [à créer des requêtes d'agrégation MongoDB](https://studio3t-com.translate.goog/knowledge-base/articles/aggregation-editor/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc) et en découvrant notre intégration de shell MongoDB,  [IntelliShell](https://studio3t-com.translate.goog/knowledge-base/articles/mongo-shell-intellishell/?_x_tr_sl=en&_x_tr_tl=fr&_x_tr_hl=fr&_x_tr_pto=nui,sc) .