MongoDB 3.6 - Nightclazz

Sommaire

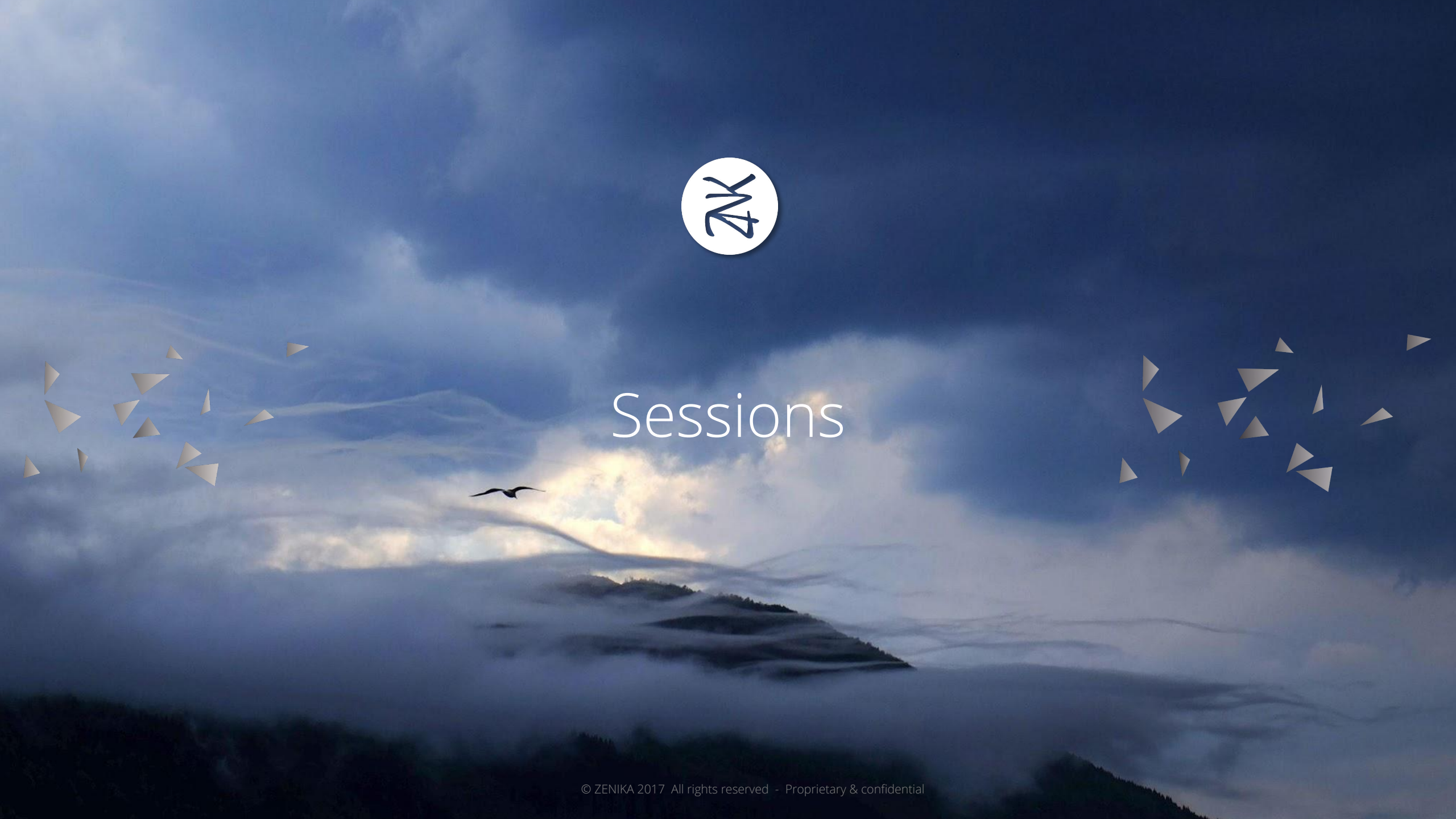
- Rappels sur MongoDB
- Sessions : Retryable Writes et Causal Consistency
- Wire compression protocol
- Validation de schéma
- Nouvelles étapes d’aggrégation
- Sécurité

- Travaux pratiques

Rappels sur MongoDB

Rappels sur MongoDB

- Base de données documents
- Schemaless
- Pas de relations
- Pas de transactions (pour l’instant !)
- CRUD basique
- Aggregation pour calculs complexes



Sessions

Sessions

- Stockées dans la collection system.sessions de chaque DB
- Basées sur un ID unique partagé par tous les noeuds de mongo (fourni par WiredTiger)
- Rien de spécial à faire, c'est le driver qui s'occupe de tout
- Nécessaire de mettre à jour les drivers en plus du serveur (et que le driver supporte bien les sessions)

- Intérêt ? Voir la suite ;-)

Retryable Writes

Retryable writes

- Avant : fonctionnement de try/catch dans le code, pour compenser les potentielles écritures ratées
- Problème, lancer une seconde fois une opération non idempotente peut provoquer des problèmes (=> $inc)
- Grâce aux sessions, le serveur note toutes les opérations et leur sortie en base.
- Ainsi, le driver peut renvoyer une opération plusieurs fois, le serveur s'occupe de savoir si l'opération a déjà eu lieu
- Limitations : Quelques opérations d'écritures multiples ne sont pas gérées (ex : une requête qui touche plusieurs documents)

Causal Consistency

Causal consistency

- Avant, en écrivant sur le primary et en lisant sur un secondary, on n'était pas sûr de récupérer la dernière valeur écrite, un secondary pouvait être en retard
- Le causal consistency, s'appuyant sur les sessions, permet de contourner ce problème
- Ainsi, au sein de la même session, si on écrit sur le Primary et lit sur un secondary, on est sûrs que si la lecture vient après l' écriture, le résultat sera bien la valeur écrite

Wire Compression Protocol

Wire Compression Protocol

- Disponible depuis 3.4, mais pas activé par défaut. C'est le cas depuis 3.6
- Permet de compresser les données sur le réseau (et donc alléger la conso de bande passante :))
- Nécessite que la source et la destination supportent la fonction
- Outils pas compatibles : mongodump, mongorestore, etc
- Principalement pour driver <> serveurs

Validation de Schéma avec $jsonSchema

Validation de Schéma avec $jsonSchema

- Deux fonctionnements :
  - Création de collection, sous le document “validator”, document $jsonSchema
  - Lors d’un find, permet de filtrer en utilisant directement l’opérateur $jsonSchema

- Possibilité de mettre des champs requis, leur type, des ranges de valeur

Aggrégation de tableaux et autres opérateurs

Aggrégation de tableaux

- **$arrayToObject** : Transformer un tableau en objet json. En entrée :
  - un tableau de tableau : `[[ “key”, “value” ], [ “key2”, “value2” ]]`
  - OU un tableau de documents : `[{ “k” : “key”, “v” : “value” }, { “k” : “key2”, “v” : “value2” }]`

Et rend : `{ “key” : “value”, “key2” : “value2” }`

- **$objectToArray** : Transforme un objet vers un tableau de forme `[{ “k” : “key”, “v” : “value” }, { “k” : “key2”, “v” : “value2” }]`
- **$mergeObjects** : Accumulateur qui mixe plusieurs documents en un seul. Si la même clé est présente plusieurs fois, c’est la dernière qui gagne. S’utilise soit via un $replaceRoot, soit via un $group

Autres étapes d'aggrégation

- **$dateFromString** : Construit une date à partir d'une chaine. Complément de dateToString qui existait déjà
- **$dateFromParts** : Construit une date à partir d'une date explosée (year, month, day, etc). Possibilité de fournir la timezone (et donc de ne pas être limité à UTC)
- **$dateToParts** : Explose une date en plusieurs parties : year, month, day, etc. . Possibilité de fournir la timezone.
- Possibilité d'utiliser **$hint** pour l'aggrégation, pour lui proposer un index plutôt que laisser mongo faire

Autres opérateurs

- **$[]** : Met à jour tous les éléments d'un tableau avec une opération
- **arrayFilters** : Permet de mettre à jour plusieurs éléments spécifiques d'un tableau. Par exemple, mettre à jour tous les éléments > 30 d'un tableau
- **$position** accepte maintenant une position négative pour compter à partir de la fin du tableau.
- **$$REMOVE** permet, dans une condition, d'afficher ou non un champs. C'est une projection conditionnelle

Sécurité

Sécurité

- mongod et mongos écoutent par défaut sur localhost uniquement. Ce qui n'était pas le cas de tous les binaires mongodb auparavant.

- Ajout de la possibilité de contrôler l'accès d'un utilisateur suivant son IP d'accès

Travaux pratiques

Récupération du Github

- https://github.com/Ptijohn/nightclazzZenika

- Nécessite Docker
- Suivre le README pour initialiser le projet
- Se reporter au dossier windows pour OSX et Windows

Sujets du TP

- Sessions : Retryable Writes et Causal Consistency
- Aggrégation de tableaux