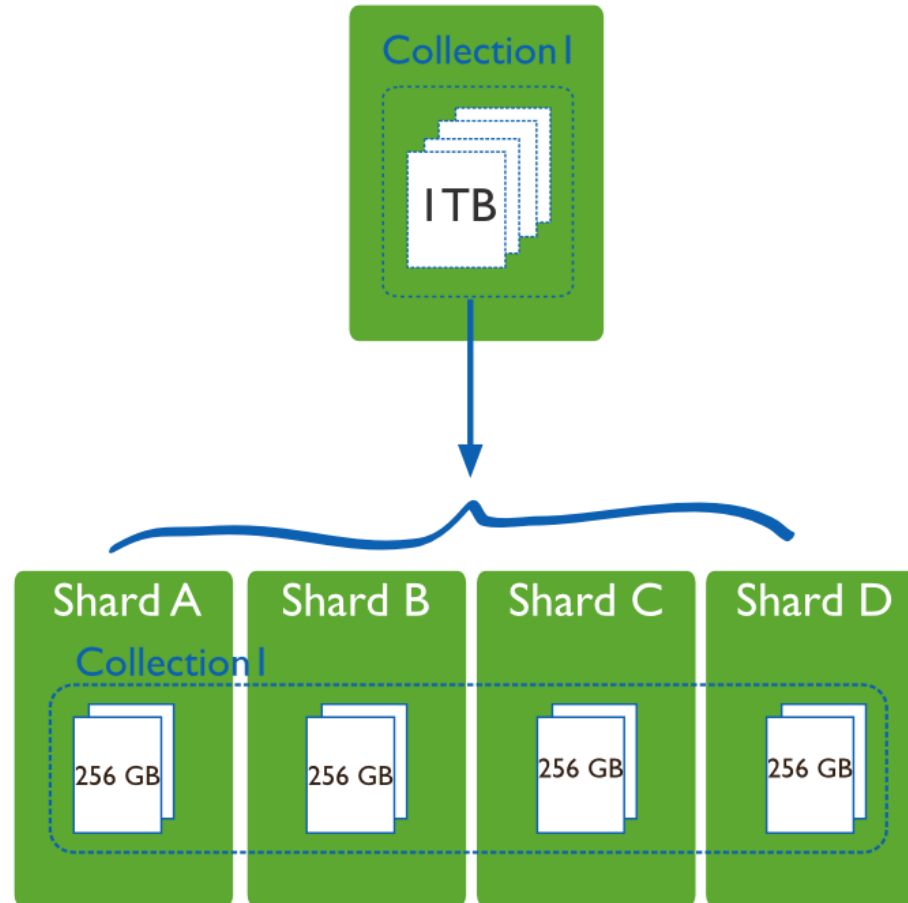


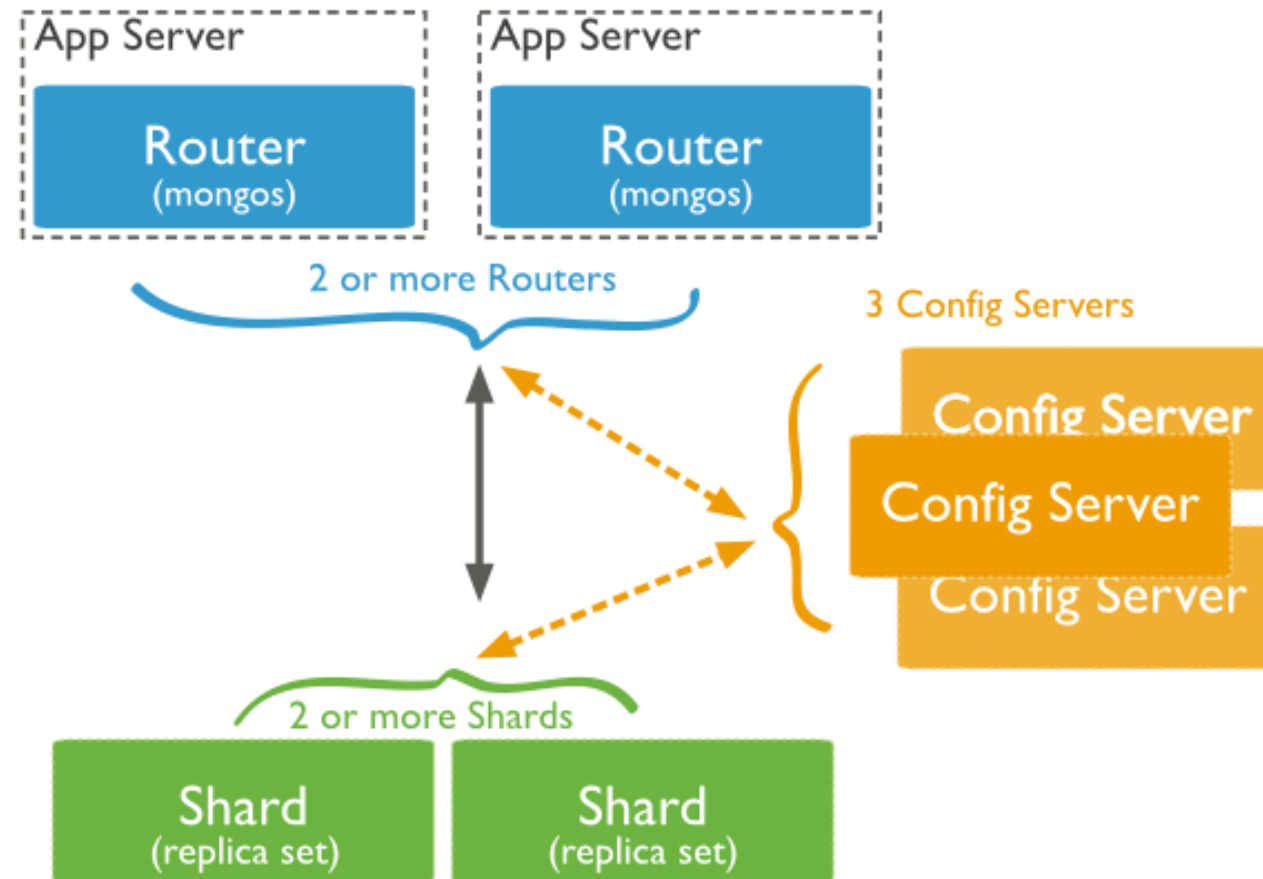
# MongoDB

Le Sharding et les opérations d'agrégation

# Qu'est ce que le sharding?



# Le Sharding par MongoDB



# Comment partager les données?

- On choisit une clé (un champ indexé) et MongoDB va partager les documents sur le cluster en fonction de cette clé
- Il existe trois modes de partage :
  - *Range based*
  - *Hash based*
  - *Tag based*

# Comment partager les données?

- *Range based* :
  - Serveur 1 : les clés allant de MIN\_VALUE à 100 (exclu)
  - Serveur 2 : les clés allant de 100 à 1000
  - ...
- *Hash based* :
  - On calcul un hash pour chaque donnée et on répartit l'ensemble des hash possible sur les serveurs
  - Serveur 1 : les hash de «00000000» à «0000FFFF»
  - Serveur 2 : les hash de «00010000» à «0FFFFFFF»

# Comment partager les données?

- *Tag based (custom sharding):*
  - On choisit une clé et on associe pour chaque sous-ensembles des valeurs de cette clé un tag puis on associe chaque tag à un serveur du sharding en particulier
  - Notre clé est une date on crée 4 tags pour les périodes suivantes : été, automne, hiver, printemps
    - Serveur 1 : été
    - Serveur 2 : automne
    - Serveur 2 : hiver
    - Serveur 2 : printemps

# Comment partager les données?

<i>Range based</i>	<i>Hash based</i>
Plus facile d'effectuer des requêtes sur des sous-ensembles distincts <u>Exemple</u> : je veux tous les documents entre les dates X et Y	Très difficile puisqu'il n'y a plus aucun de ces sous-ensembles
Risque de ne pas répartir équitablement les données sur le cluster <u>Exemple</u> : je n'ai que des _id allant de 1 à 1M et ça ne correspond qu'à mes deux premiers serveurs sur 100	Réparti équitablement les données pour tirer le meilleur parti du cluster

# Combien de *shards*?

- Combien de shards?
  1. Il faut que la somme de tous les espaces disques soit supérieure à l'espace requis
  2. Le « *working set* » doit tenir en RAM pour garantir une latence minimum
  3. La somme de tous les IOPS doit être supérieure au nombre requis d'IOPS



# Combien de *shards*?

- Exemple : une simple base de données
  - 5TB de données
  - 500 GB de working set
  - 300 000 IOPS
- Mes serveurs ont :
  - 2TB de SSD en RAID 10
  - 128GB de RAM

# Combien de *shards*?

- Combien de serveurs me faut-il?
  - En disque :  $5 / 2 = 2.5$  soit 3 serveurs
  - En RAM :  $500 / 128 = 3.9$  soit 4 serveurs
  - En IOPS... Mieux vaut tester car incalculable

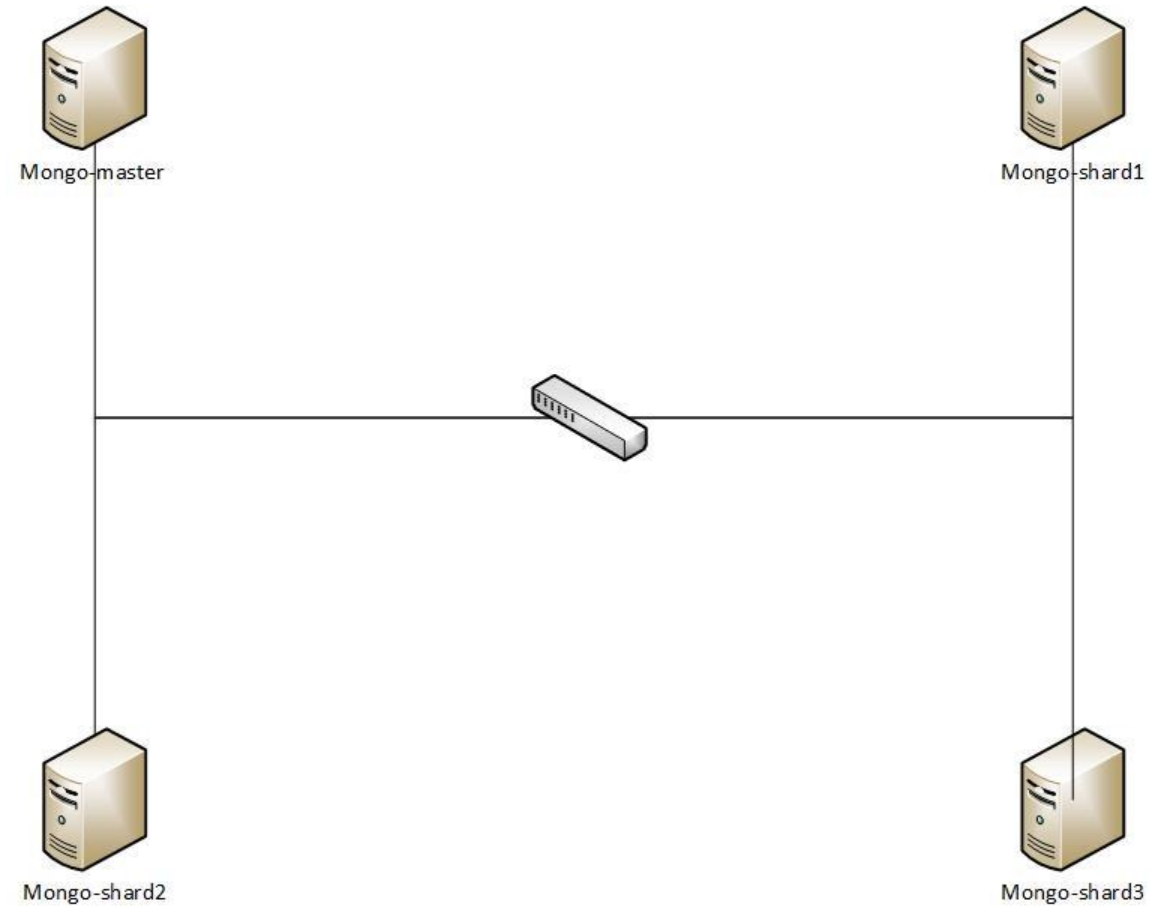
=> Il nous faut donc 4 serveurs

Trêve de bavardages... Let's try

# Le cas du NOAA

- Nous sommes le NOAA (National Oceanic and Atmospheric Administration)
- On possède des centaines de stations météorologiques dans les océans autour des USA
- Chaque station envoie ses données à un centre
- Il faut sauvegarder toutes les données (ils ont un historique de 200 ans)

# Notre architecture



# Notre base de données

- 102 553 enregistrements océanographiques
- 386 stations océanographiques
- MongoDB version : 3.2 RC3
- Moteur de stockage : Wired Tiger

# Notre architecture

- Serveurs : 6GB de RAM et 20GB de disque
- Mongo-master :
  - 1 serveur de configuration
  - 1 serveur mongos
  - 1 serveur DNS
- Mongo-shard1 :
  - 1 serveur mongos
  - 1 serveur de configuration
  - 1 serveur mongod (shard)

# Notre architecture

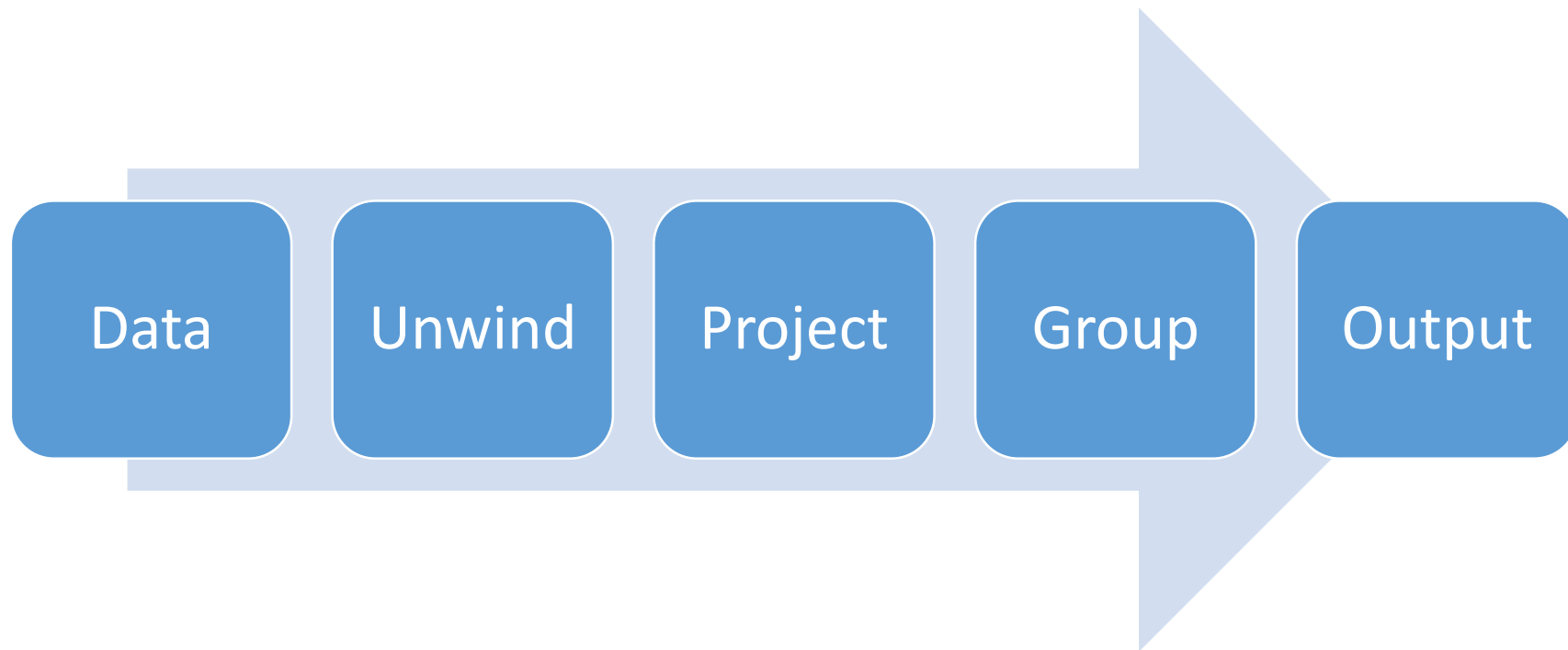
- Mongo-shard2 :
  - 1 serveur de configuration
  - 1 serveur mongod (shard)
- Mongo-shard3 :
  - 1 serveur mongod (shard)
- Total
  - 3 shard
  - 2 serveurs mongos
  - 3 serveurs de configuration



DEMO

# Agrégation

- Le flux de données passe à travers une suite d'opérations définies par un tableau d'opérations



# L'avenir

- On va vers des bases de données de plus en plus spécialisées et de plus en plus réparties pour des raisons de rapidité et/ou de sécurité
  - RethinkDB : base de données « Document store » spécialisée dans les données temps réel
  - InfluxDB : base de données spécialisée dans les données de type « Time series »
  - Prometheus : base de données spécialisée dans les données de type « Time series » et plus spécifiquement le monitoring
  - Neo4J : base de données spécialisée dans les données de type graphe (surtout social)
  - Tidb : « MySQL like » répartie basé sur des Key Value store