

# Data Aggregation

## In mongoDB

# Aggregation Types

- Single Purpose Aggregation Operation
- Map-Reduce
- Aggregation Framework

# Single Purpose Aggregation Operation

- `db.collection.count()`
- `db.collection.group()`
- `db.collection.distinct()`
- ...

**Fast but inflexible**

# Map Reduce

- Write Map & Reduce functions in javascript and evaluate them in a mongod instance
- Unable to do this in sharded mongo configuration
- Might be slow if the collection is huge

**Slow & Flexible**

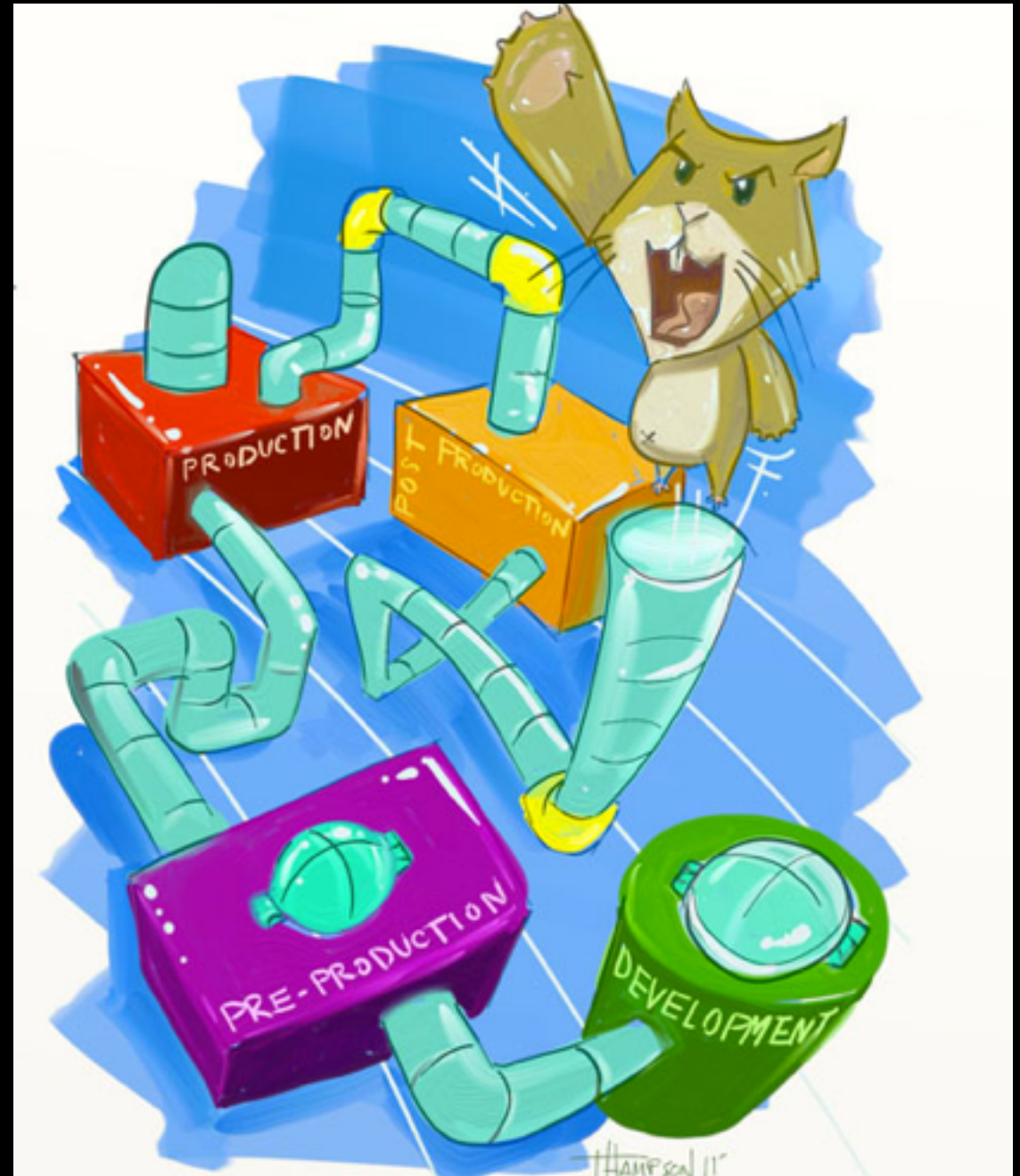
# Aggregation Framework

- Write operating instructions and assemble them to a pipeline
- C++ Implementation
- Compatible with Sharded Mongo

**Fast & not so Flexible**

# Pipeline

- Imagine data is those goods that streaming on the pipeline
- Each step is a operator



# Data schema

- db.brands

```
{  
  _id: ObjectId('xxxx'),  
  products: [pid, pid, pid...]  
}
```

- db.products

```
{  
  _id: ObjectId('xxxx'),  
  features: [  
    {  
      feature: fid,  
      value: value  
    },  
    {f,v}, {f,v}  
  ]  
}
```

1. Find all products : \$match
2. Only keep the needed fields : \$project
3. Unpack the features array : \$unwind
4. Filter out none-numeric values : \$match
5. Calculate average of grouped values : \$group
6. Only returned the fields we want : \$project