Consultas en MongoDB

Bases de Datos No Relacionales

Instituto de Computación, FING, UdelaR – 2022

CC-BY Lorena Etcheverry lorenae@fing.edu.uy

Agenda

- Consultas básicas
  - filtrado y proyección
- Consultas de agregación
  - *Aggregation pipeline*

Nuestros datos de prueba

- Del tutorial de MongoDB

```
db.inventory.insertMany( [
  { item: "journal",
    status: "A",
    size: { h: 14, w: 21, uom: "cm" },
    instock: [ { warehouse: "A", qty: 5 } ] },
  { item: "notebook", status: "A",  size: { h: 8.5, w: 11, uom: "in" },
    instock: [ { warehouse: "C", qty: 5 } ] },
  { item: "paper", status: "D", size: { h: 8.5, w: 11, uom: "in" },
    instock: [ { warehouse: "A", qty: 60 } ] },
  { item: "planner", status: "D", size: { h: 22.85, w: 30, uom: "cm" },
    instock: [ { warehouse: "A", qty: 40 } ] },
  { item: "postcard", status: "A", size: { h: 10, w: 15.25, uom: "cm" },
    instock: [ { warehouse: "B", qty: 15 }, { warehouse: "C", qty: 35 } ] }
]);
```

Consultas básicas: *find*

***db.collection*.find(query, projection)**

- **query**: especifica condiciones o filtros
- **projection**: especifica los campos a proyectar

Devuelve los campos indicados de los documentos que satisfacen las condiciones.

Las secciones query y projection son opcionales

Query vacía devuelve todos los documentos (filas)

Projection vacía devuelve todos los campos (columnas)

Consultas sin condiciones

SQL	MongoDB
`SELECT *` `FROM inventory`	`db.inventory.find()`

`SELECT id,`
`item`
`FROM inventory`
	`db.inventory.find(`
	`{},`
	`{item:1})`

`SELECT item`
`FROM inventory`
	`db.inventory.find(`
	`{},`
	`{item:1,_id:0})`

Consultas con condiciones de filtrado

SQL	MongoDB

`SELECT item`
`FROM inventory`
`WHERE status=”A”`
	`db.inventory.find(`
	`  {status:”A”},`
	`  {item:1,_id:0})`

`SELECT item`
`FROM inventory`
`WHERE status <>”A”`
	`db.inventory.find(`
	`  {status:{$ne:”A”}},`
	`  {item:1,_id:0})`

`SELECT *`
`FROM inventory`
`WHERE status=“A” or`
`  item = “paper”`
	`db.inventory.find(`
	`  {$or:[{status:"A"},`
	`  {item:"paper"}]})`

Lista completa de operadores (de comparación, lógicos, etc)
https://docs.mongodb.com/manual/reference/operator/query

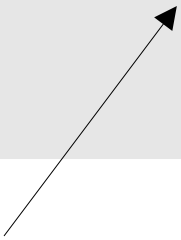
Algunos operadores

Condiciones de filtrado (ii)

¿cómo se imponen condiciones de filtrado sobre documentos anidados?

```
{
item: "journal",
status: "A",
size: { h: 14, w: 21, uom: "cm" },
instock: [ { warehouse: "A", qty: 5 } ]
}
```

Items con altura menor a 15	`db.inventory.find( {"size.h":{$lt: 15}} )`



Se usa *"dot notation"* para hacer referencia a los documentos anidados y sus atributos

Ordenación

cursor.sort(criterios)

- **criterios** :documento con parejas { field: value } con value es 1 (ascendente) o -1 (descendiente)

Devuelve la colección ordenada según los criterios.

SQL

```
SELECT *
FROM inventory
WHERE status="A"
ORDER BY item desc
```

MongoDB

```
db.inventory.find(
  {status:"A"}).sort({item:-1})
```

Consultas básicas: *count*

***db.collection*.count(query, options)**

- **query**: especifica condiciones o filtros
- **options**: varias opciones, ver documentación

Devuelve la cantidad de documentos que satisfacen las condiciones.

SQL	MongoDB

`SELECT count(*)`
`FROM inventory`
`WHERE status="A"`
	`db.inventory.count(`
	`  {status:"A"})`
	
	`db.inventory.find(`
	`  {status:"A"}).count()`

Consultas básicas: *distinct*

***db.collection*.distinct(field, query, options)**

- **field:** campo sobre el cual aplica el distinct
- **query**: especifica condiciones o filtros
- **options**: ver documentación

Devuelve los valores diferentes de cierto campo de los documentos que satisfacen las condiciones.

```
SELECT
DISTINCT("instock.qty")
FROM inventory
WHERE status="A"
```

```
db.inventory.distict(
"instock.qty",
{ status: "A" } )
```

Consultas de agregación

Hay dos mecanismos para hacer consultas de agregación en MongoDB:

- Consultas Map-Reduce (eliminadas a partir de MongoDB 5.0)
- Usar el *Aggregation Framework*

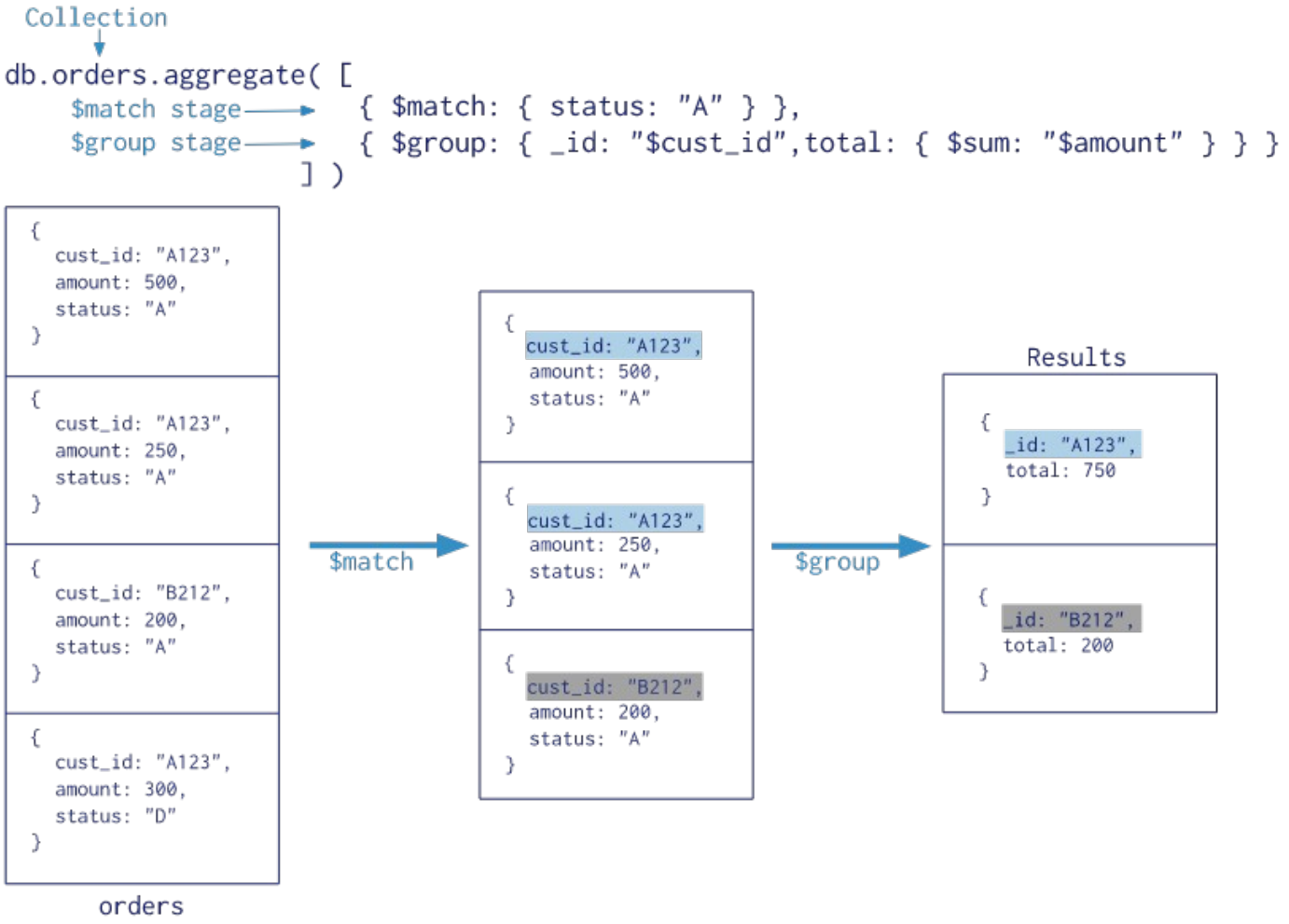
Aggregation framework

La idea principal es la de
pipeline de ejecución.

```
ls -l | grep -i mongo
```

Secuencia de etapas (*stages*) de:
filtrado,
transformación,
agrupación,
ordenamiento,
y proyección.

Un ejemplo


Collection
db.orders.aggregate([
$match stage
{ $match: { status: "A" } },
$group stage
{ $group: { _id: "$cust_id",total: { $sum: "$amount" } } }
])
{ cust_id: "A123", amount: 500, status: "A" }
{ cust_id: "A123", amount: 250, status: "A" }
{ cust_id: "B212", amount: 200, status: "A" }
{ cust_id: "A123", amount: 300, status: "D" }
orders
$match
{ cust_id: "A123", amount: 500, status: "A" }
{ cust_id: "A123", amount: 250, status: "A" }
{ cust_id: "B212", amount: 200, status: "A" }
$group
Results
{ _id: "A123", total: 750 }
{ _id: "B212", total: 200 }

- Las slides a continuación son parte de la presentación Aggregation Framework (Rick Houlihan, MongoDB) realizada en el contexto de la conferencia MongoDB World 2014

MongoDB World

Aggregation Framework

Rick Houlihan

Senior Solutions Architect, MongoDB

mongoDB

Pipeline Operators

- $match
  - Filter documents
- $project
  - Reshape documents
- $group
  - Summarize documents
- $unwind
  - Expand documents
- $sort
  - Order documents
- $limit / $skip
  - Paginate documents
- $redact
  - Restrict documents
- $geoNear
  - Proximity sort documents
- $let, $map
  - Subexpression variables

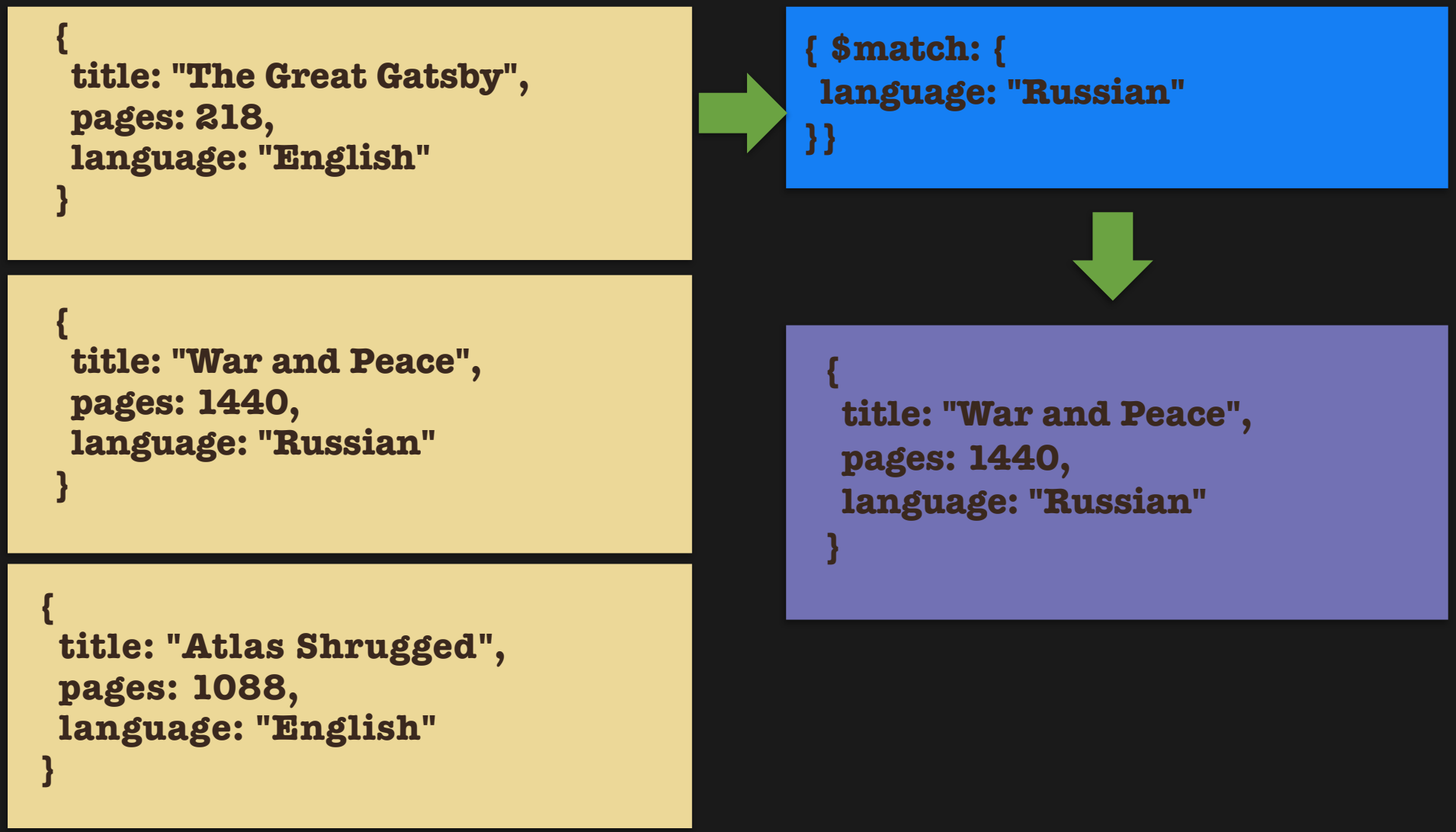
Our Example Data

```
{
 _id: 375,
 title: "The Great Gatsby",
 ISBN: "9781857150193",
 available: true,
 pages: 218,
 chapters: 9,
 subjects: [
  "Long Island",
  "New York",
  "1920s"
 ],
 language: "English"
}
```

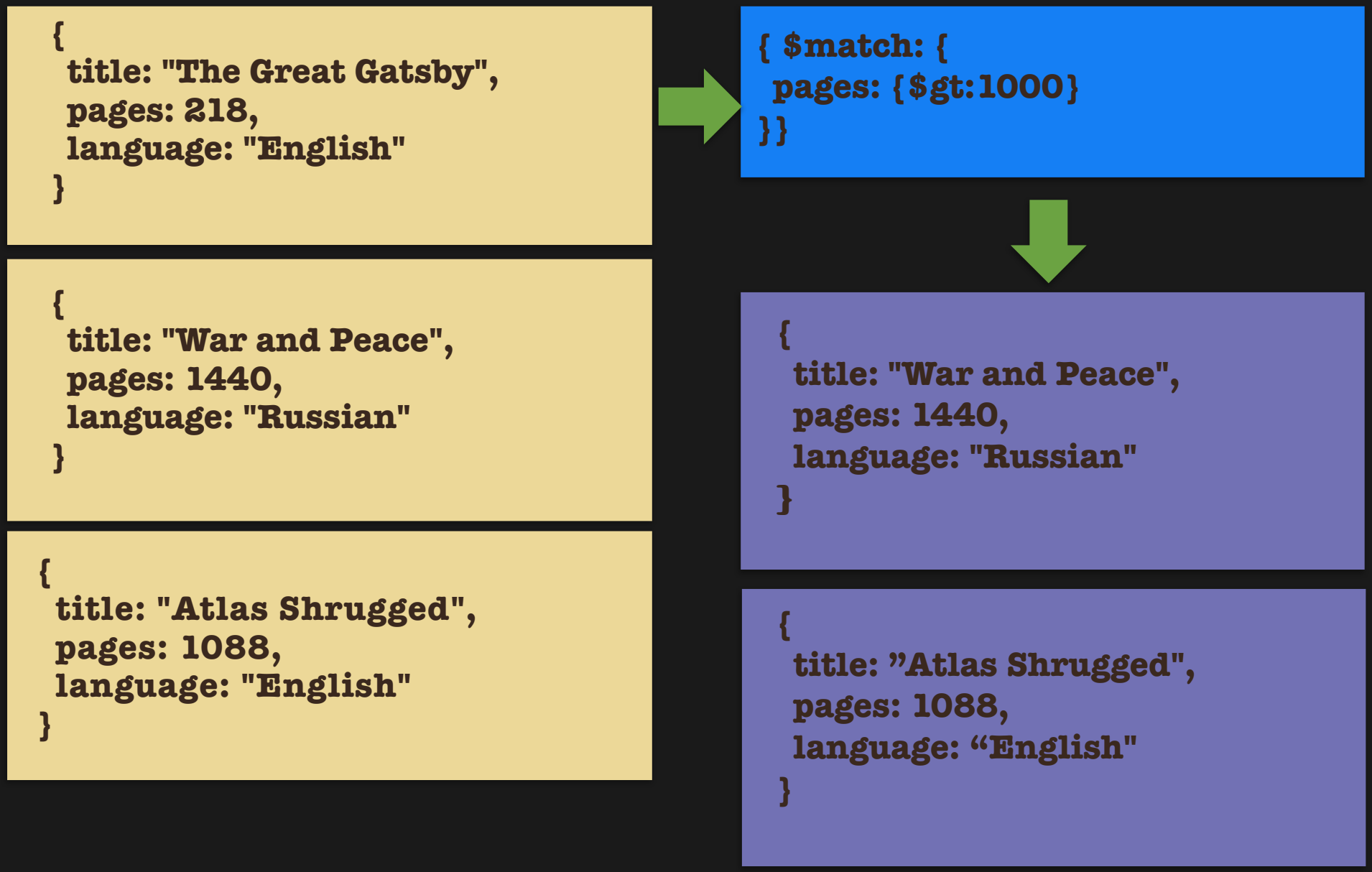
$match

- Filter documents
  - Uses existing query syntax
  - Can facilitate shard exclusion
  - No $where (server side Javascript)

Matching Field Values

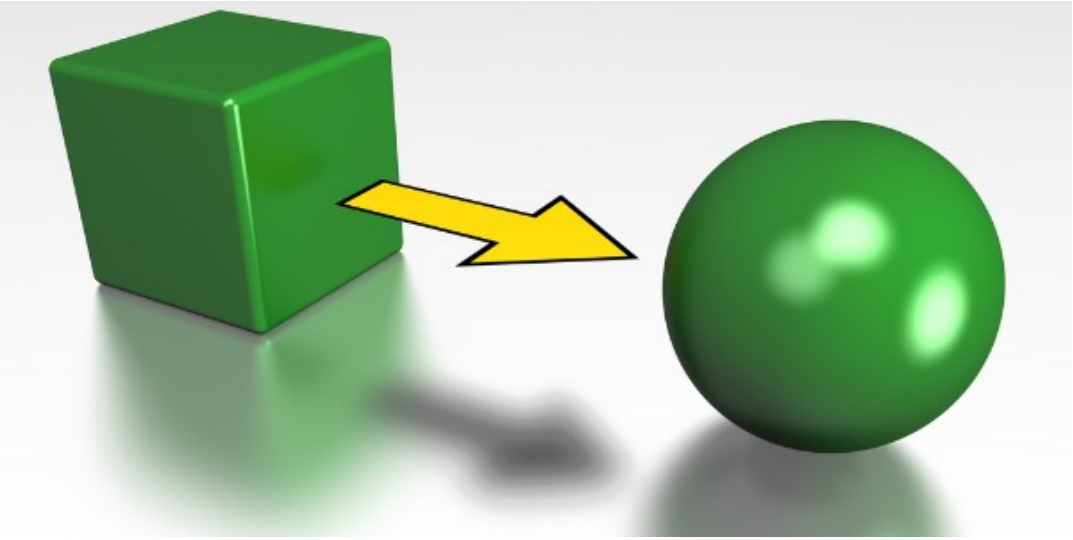


Matching with Query Operators



$project

- Reshape Documents
  - Include, exclude or rename fields
  - Inject computed fields
  - Create sub-document fields



Including and Excluding Fields

```
{
  _id: 375,
  title: "Great Gatsby",
  ISBN: "9781857150193",
  available: true,
  pages: 218,
  subjects: [
    "Long Island",
    "New York",
    "1920s"
  ],
  language: "English"
}
```

```
{ $project: {
  _id: 0,
  title: 1,
  language: 1
}}
```

```
{
  title: "Great Gatsby",
  language: "English"
}
```

Renaming and Computing Fields

```
{
  _id: 375,
  title: "Great Gatsby",
  ISBN: "9781857150193",
  available: true,
  pages: 218,
  chapters: 9,
  subjects: [
    "Long Island",
    "New York",
    "1920s"
  ],
  language: "English"
}
```

```
{ $project: {
  avgChapterLength: {
    $divide: ["$pages",
              "$chapters"]
  },
  lang: "$language"
}}
```

```
{
  _id: 375,
  avgChapterLength:
24.2222,
  lang: "English"
}
```

Creating Sub-Document Fields

```
{
  _id: 375,
  title: "Great Gatsby",
  ISBN: "9781857150193",
  available: true,
  pages: 218,
  chapters: 9,
  subjects: [
    "Long Island",
    "New York",
    "1920s"
  ],
  language: "English"
}
```

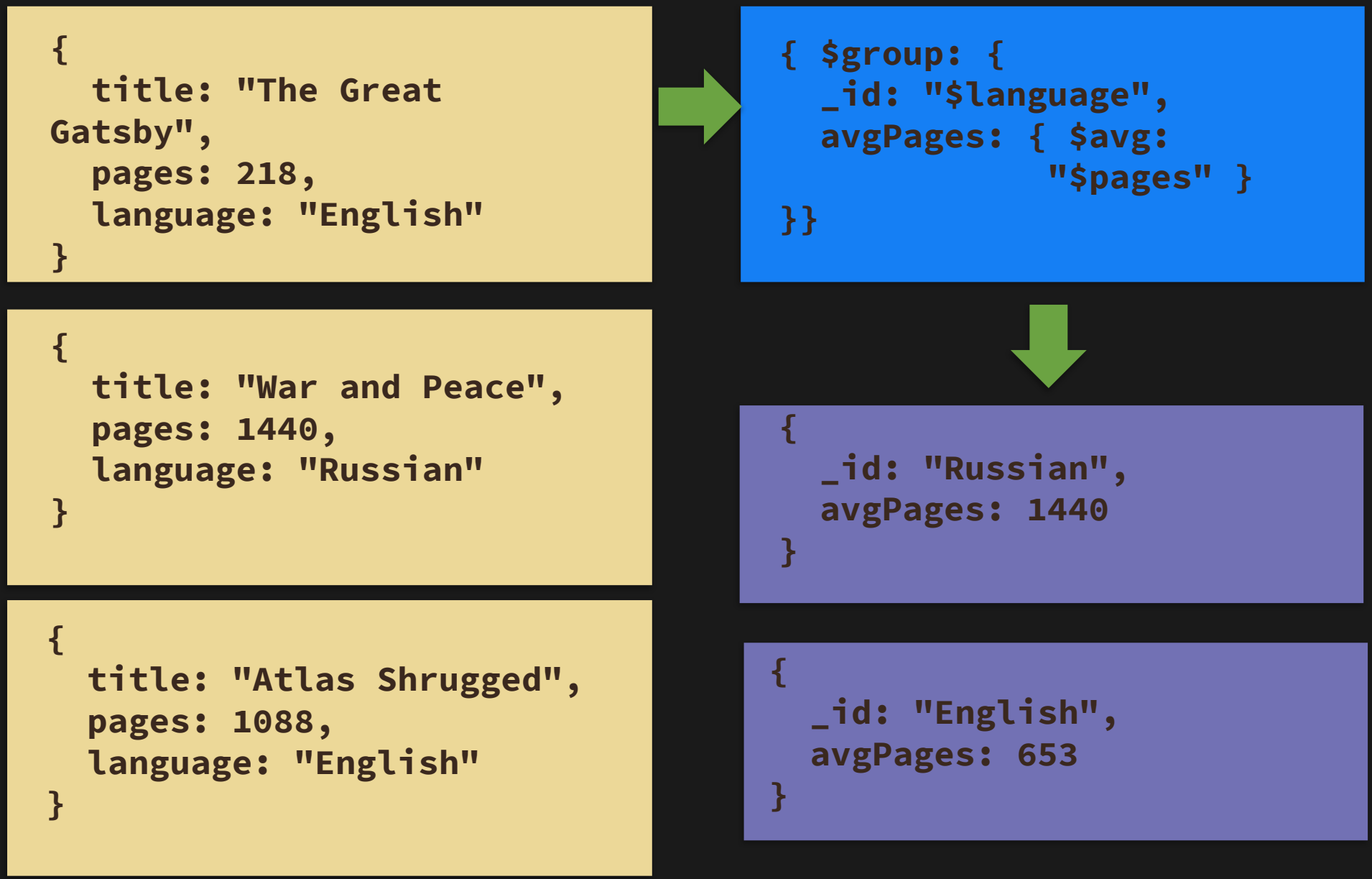
```
{ $project: {
  title: 1,
  stats: {
    pages: "$pages",
    language: "$language",
  }
}}
```

```
{
  _id: 375,
  title: "Great Gatsby",
  stats: {
    pages: 218,
    language: "English"
  }
}
```

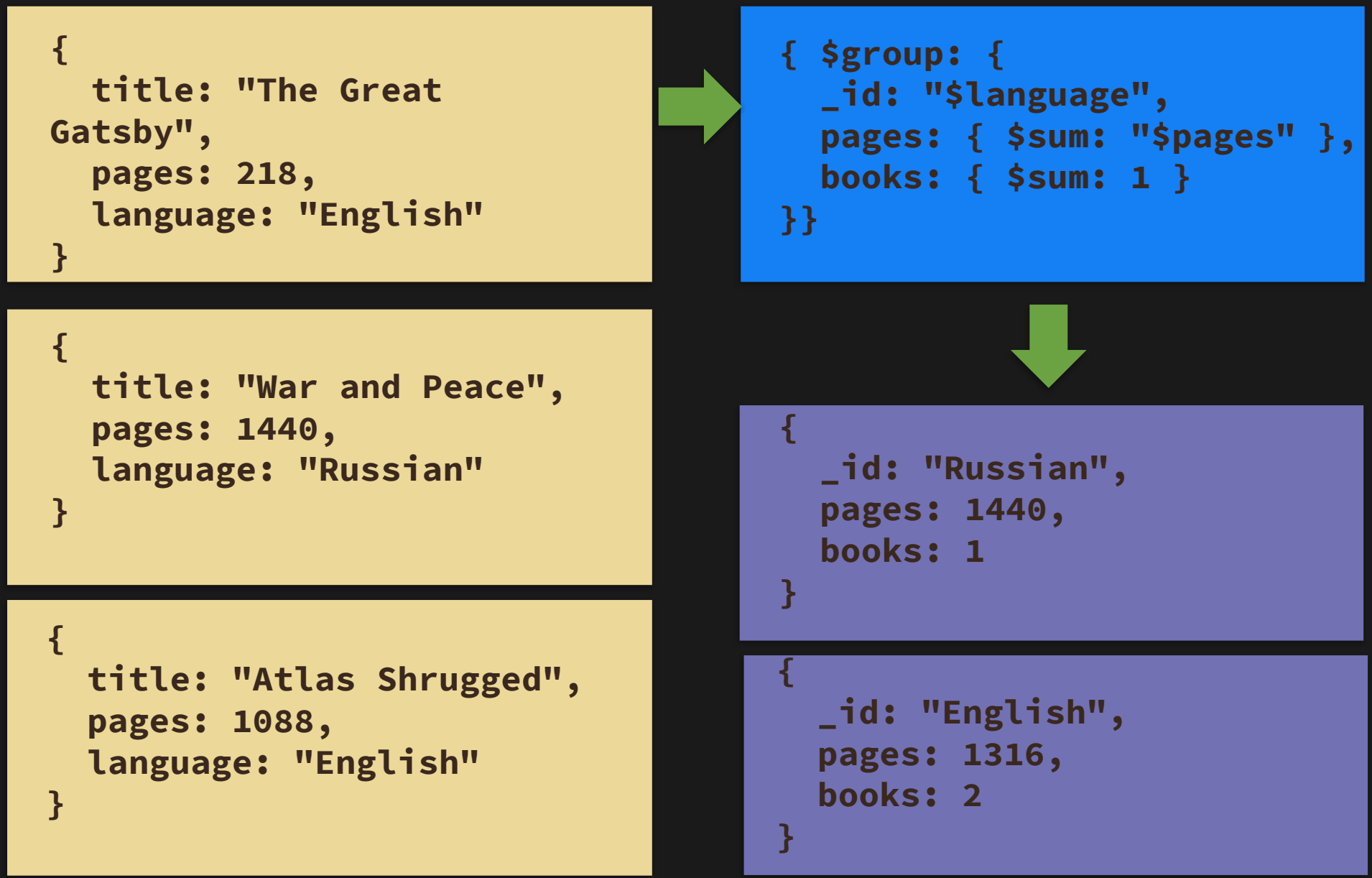
$group

- Group documents by value
  - Field reference, object, constant
  - Other output fields are computed
    - $max, $min, $avg, $sum
    - $addToSet, $push
    - $first, $last
  - Processes all data in memory by default

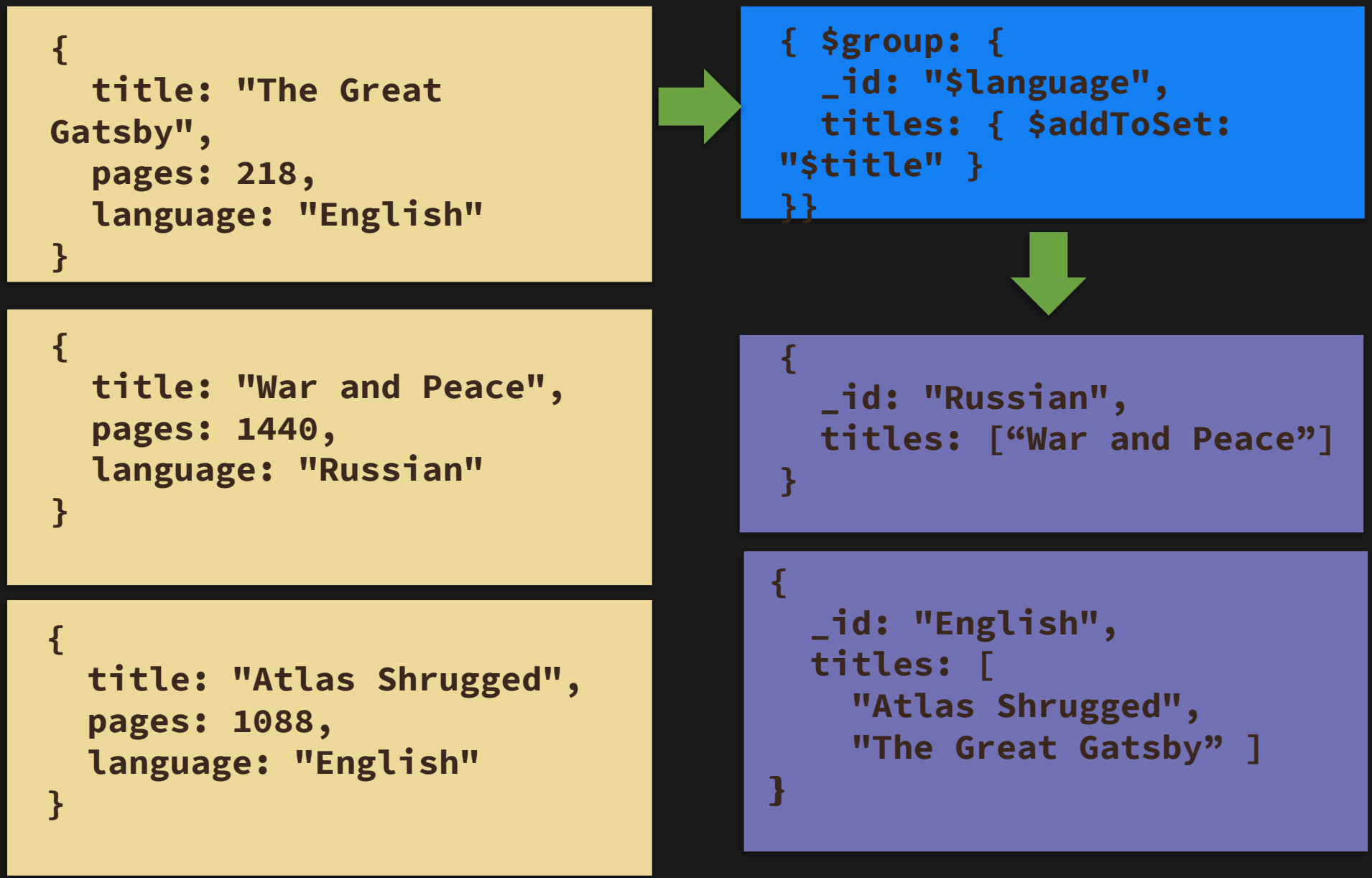
Calculating An Average



Summing Fields and Counting



Collecting Distinct Values



$unwind

- Operate on an array field
  - Create documents from array elements
    - Array replaced by element value
    - Missing/empty fields → no output
    - Non-array fields → error
  - Pipe to `$group` to aggregate

Collecting Distinct Values

```
{
  title: "The Great
Gatsby",
  ISBN: "9781857150193",
  subjects: [
    "Long Island",
    "New York",
    "1920s"
  ]
}
```

```
{ $unwind: "$subjects" }
```

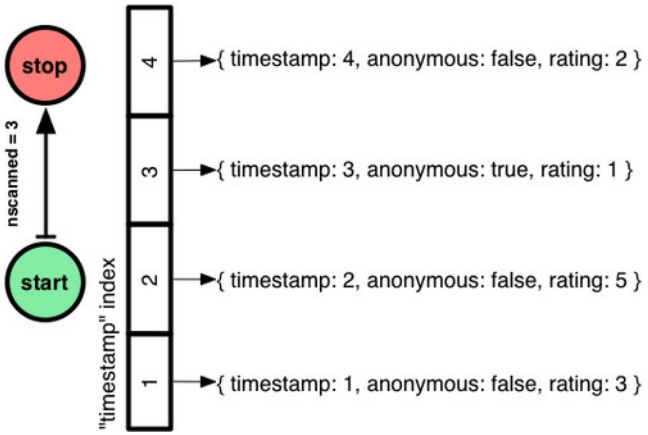
```
{ title: "The Great Gatsby",
  ISBN: "9781857150193",
  subjects: "Long Island” }
```

```
{ title: "The Great Gatsby",
  ISBN: "9781857150193",
  subjects: "New York” }
```

```
{ title: "The Great Gatsby",
  ISBN: "9781857150193",
  subjects: "1920s” }
```

$sort, $limit, $skip

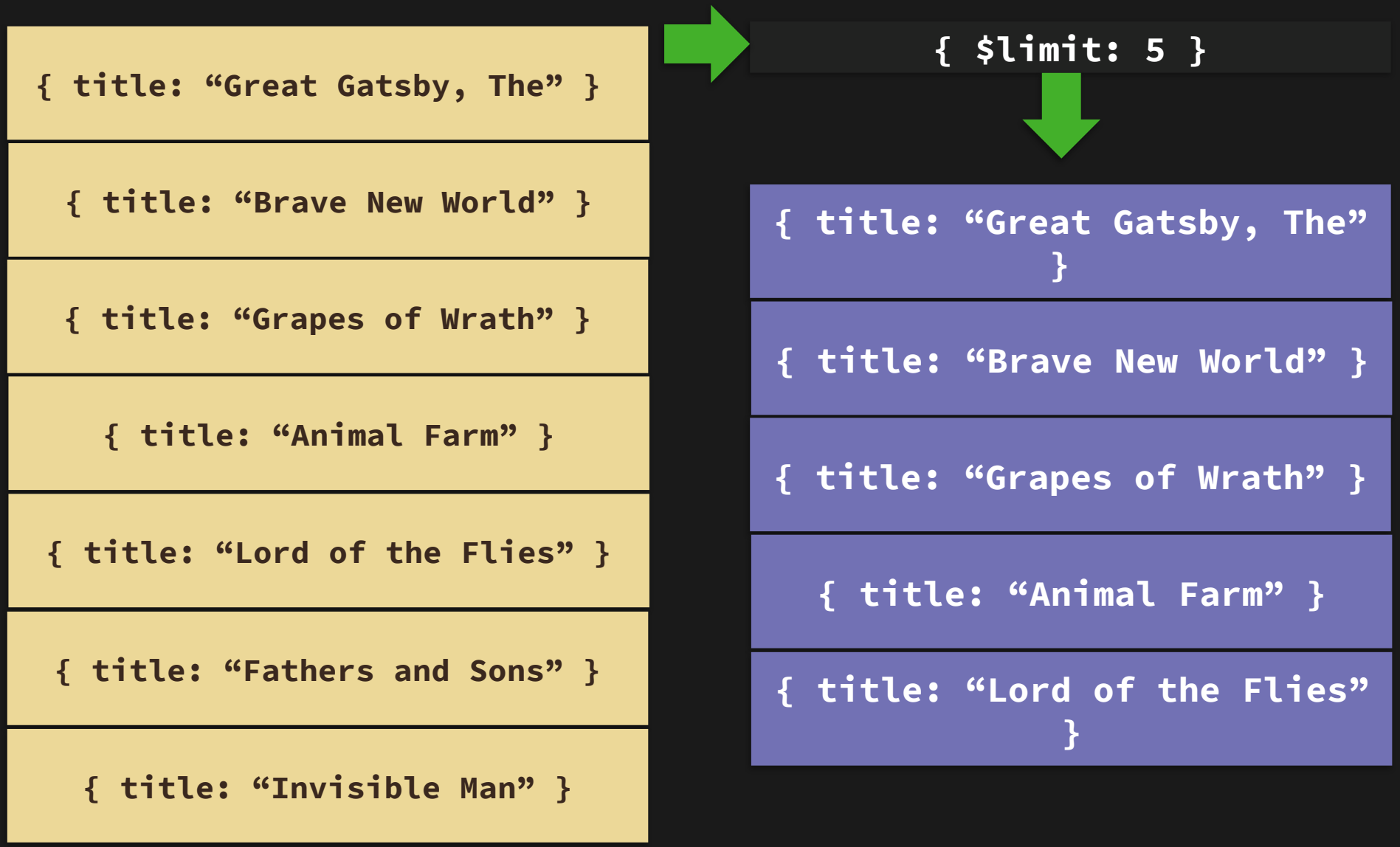
- Sort documents by one or more fields
  - Same order syntax as cursors
  - Waits for earlier pipeline operator to return
  - In-memory unless early and indexed



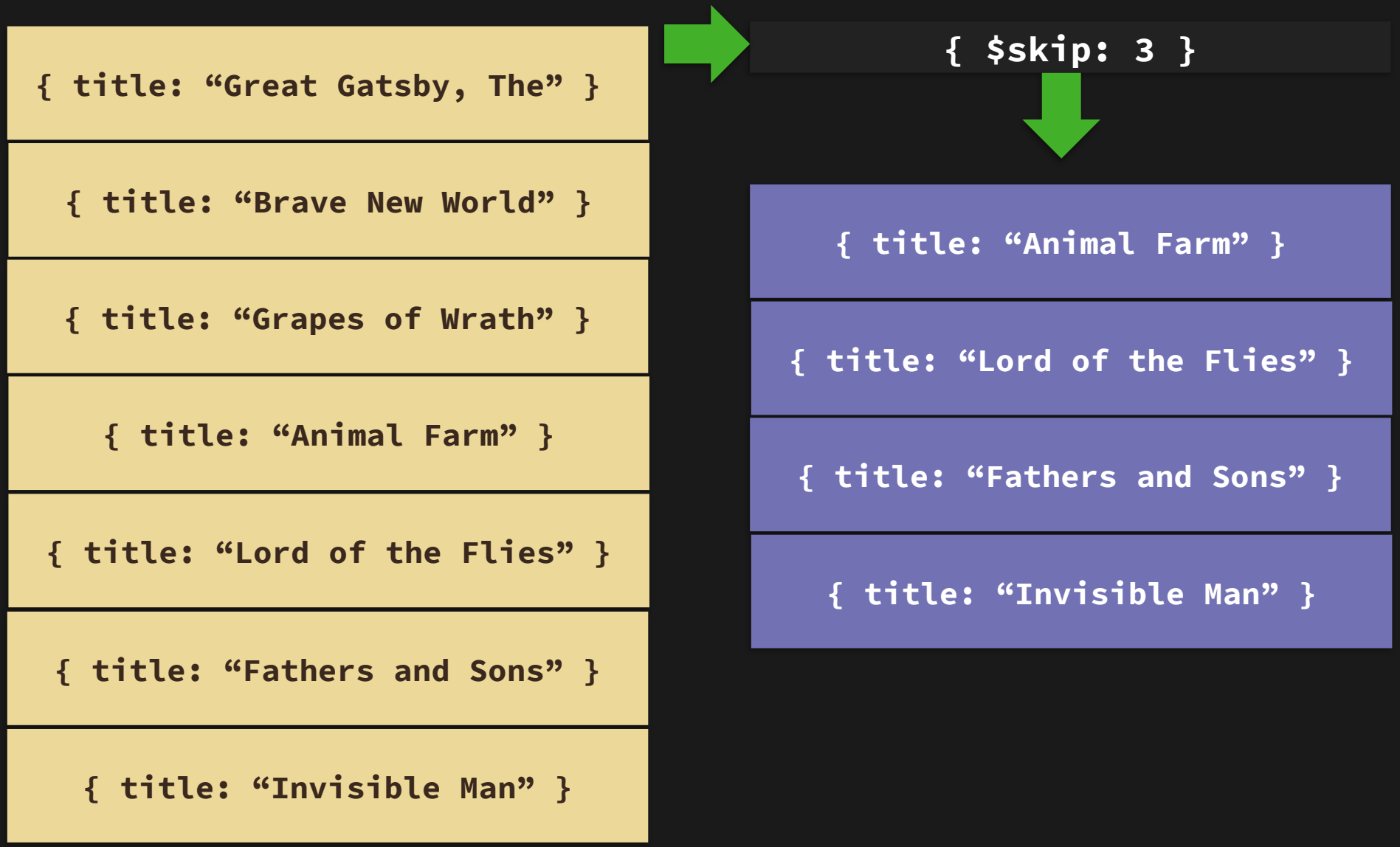
- Limit and skip follow cursor behavior

Sort All the Documents in the Pipeline

Limit Documents Through the Pipeline



Skip Documents in the Pipeline



Herramientas

- El cliente Compass ofrece una interfaz gráfica que facilita la creación de estos pipelines. Más info
- MongoDB Atlas: MongoDB en la nube. Tiene una interfaz para consultas similar a Compass.
- Para probar consultas y validar sintaxis también se puede usar esta página

Material adicional

- El manual de MongoDB, y específicamente la sección sobre Agregación
- Los cursos online gratuitos de MongoDB University y en particular el curso The MongoDB Aggregation Framework (M121)

- El libro Practical MongoDB Aggregations