Distributed Data Systems

DIANE WOODBRIDGE, PH.D

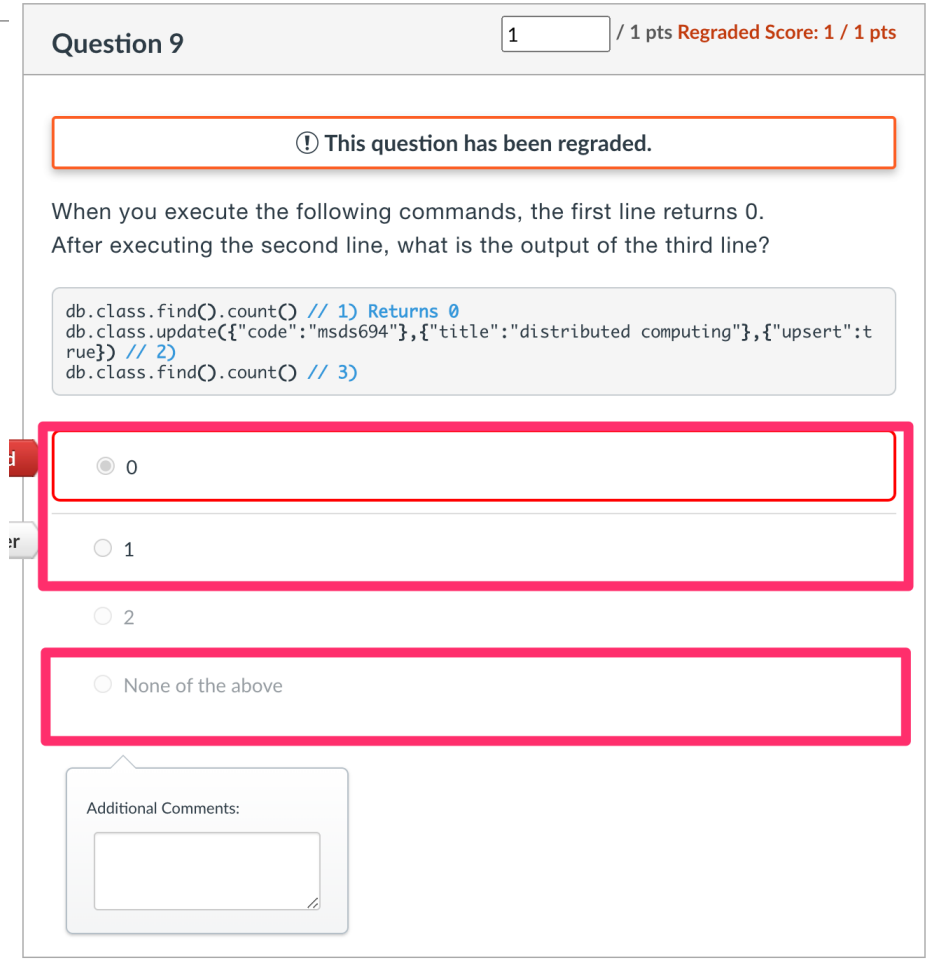
Announcement

Homework

- Individual HW 3 - Feb 13 Midnight (Cannot be extended)
- Group HW2 - Feb 25th
  - See the instructions and videos
- Make sure to watch Replica/Sharding Video by Feb 26th

Quiz 1

- Multiple Choice



Content

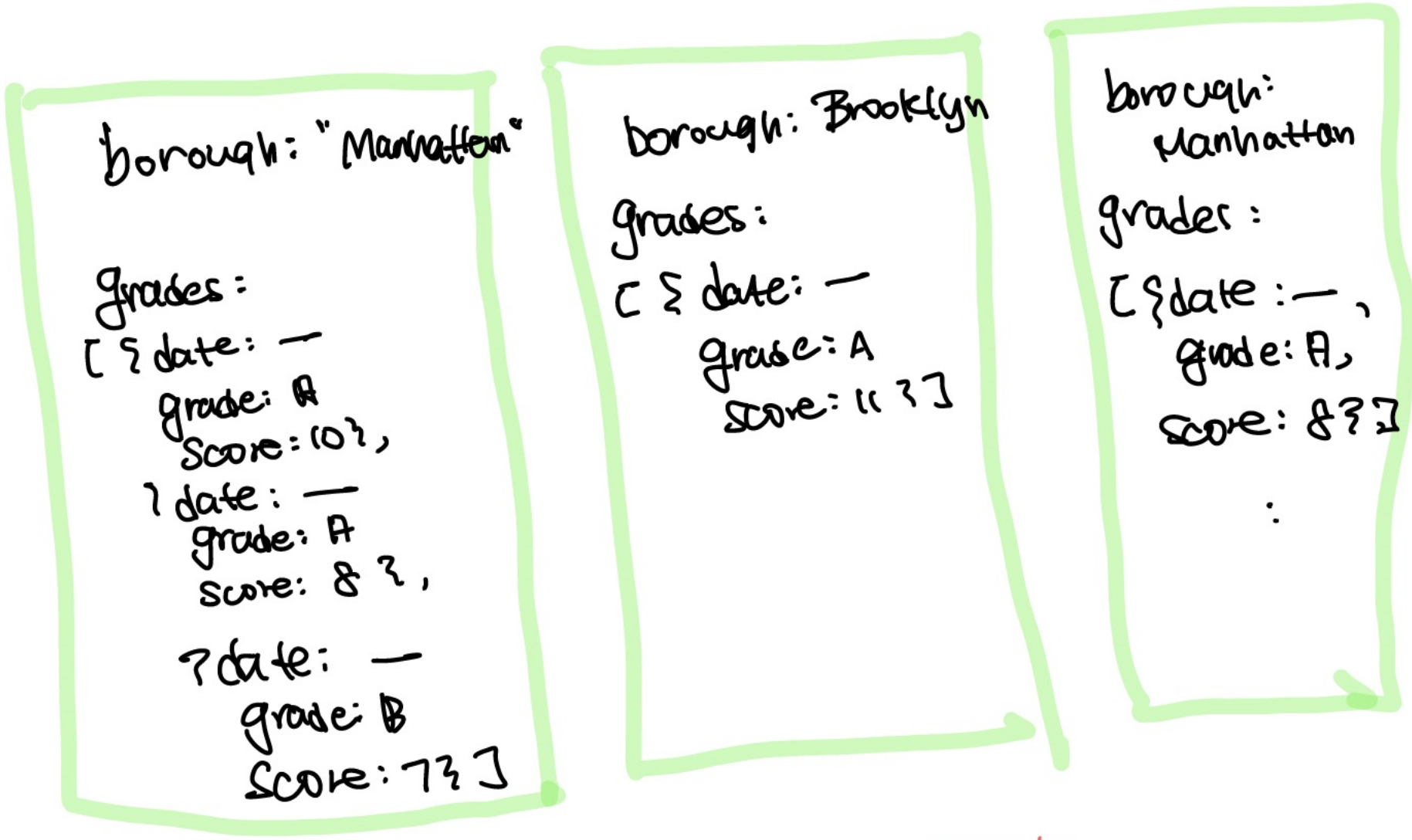
MongoDB

- Aggregation Operations (aggregation pipeline)
- Index

Content

MongoDB

- **Aggregation Operations (aggregation pipeline)**
- Index



What's the avg score per grade for businesses in "Manhattan"?

MongoDB

Aggregation Pipeline

- Perform several stages of operations.
  - Each stage performs an operation on the input documents, and its outputs are passed to the next stage
  - An aggregation pipeline can return results for groups of documents.
  - Syntax
    - **`db.collecion_name.aggregate([{transform_operator_1 : criteria_1}, {transform_operator_2 : criteria_2}, …])`**
      - **`transform_operators`** performs operations including filtering, transformation, grouping, sorting using **`$match, $project, $group, $unwind, $sort, $lookup`** `etc.`

https://docs.mongodb.com/manual/aggregation/

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/

More Operators : https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/

MongoDB

Aggregation Pipeline

- Perform several stages of operations.
  - Each stage performs an operation on the input documents, and its outputs are passed to the next stage
  - An aggregation pipeline can return results for groups of documents.
  - **db.collecion_name.aggregate([{transform_operator_1 : criteria_1}, {transform_operator_2 : criteria_2}, …])**

Transform Operators	Description	Example
$match	Filters the documents to allow only matching documents to pass the the next pipeline stage.	db.collection.aggregate({$match : {"size":{$gt : 0}}})
$project	Reshapes each document in the stream, such as by adding new fields or removing existing fields.	db.collection.aggregate({$match : {"size":{$gt : 0}}}, {$project: {"size": false}})
$group	Groups input documents by a specified identifier expression.	db.collection.aggregate({$group:{_id :"$date", count: {$sum:1}}})
$unwind	Deconstructs an array field from the input documents to output a document for each element.	db.collection.aggregate({$unwind : "$grade"})
$sort	Reorders documents by a specified sort key.	db.collection.aggregate({$sort:{"date":-1}})
$lookup	Performs a left outer join to another collection in the same database.	db.world_bank_project.aggregate({$lookup: {from:"country_code", localField:"countrycode", foreignField:"code", as:"country_phone_info"}})

MongoDB

Aggregation Pipeline

- Perform several stages of operations.
  - Each stage performs an operation on the input documents, and its outputs are passed to the next stage
  - An aggregation pipeline can return results for groups of documents.
  - Syntax
    - **`db.collecion_name.aggregate([{transform_operator_1 : criteria_1},`**
      `{transform_operator_2 : criteria_2}, …])`
    - Field Path (**`$<field>`**) in criteria
      - Aggregation pipelines use field path to access fields in the input documents.
        - "`$<field>`" is equivalent to "`$$CURRENT.<field>`" where the CURRENT is the root of the current object.
      - To access the value of the field, prefix the variable name with double dollar signs ($$)

https://docs.mongodb.com/manual/meta/aggregation-quick-reference/#field-paths
https://www.mongodb.com/docs/manual/reference/aggregation-variables/

MongoDB

Aggregation Pipeline

- Pipeline transform operators
  - **`$match`** : Filter document with criteria.
    - `Format : { $match: { <query> } }`
      - The `query` syntax is identical to the read operation query syntax.

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/match/

Example 1

Using aggregate(), return documents where "id" is "P130164"

Example 1

Using aggregate(), return documents where "id" is "P130164"

```
105 db.world_bank_project.aggregate({$match:{"id":"P130164"}})
```

Find × | Aggregate × | Aggregate × | Find × | Find × | Aggregate × | Text × | Aggregate × | Aggregate × | Aggregate × | Aggregate × | Aggregate × | »99+

50 | Documents 1 to 1 | JSON View


```
{
    "_id" : ObjectId("52b213b38594d8a2be17c788"),
    "approvalfy" : "2014",
    "board_approval_month" : "October",
    "boardapprovaldate" : "2013-10-29T00:00:00Z",
    "borrower" : "THE GOVERNMENT OF INDIA",
    "closingdate" : "2018-12-31T00:00:00Z",
    "country_namecode" : "Republic of India!$!IN",
    "countrycode" : "IN",
    "countryname" : "Republic of India",
    "countryshortname" : "India",
    "docty" : "Project Appraisal Document,Integrated Safeguards Data Sheet,Procurement Plan,Procurement F
```

MongoDB

Aggregation Pipeline

- Pipeline transform operators
  - **`$project`** : Extract field. Rename the projected field.
    - Project expressions
      - **Specify the inclusion of fields, the addition of new fields, and the resetting the values of existing fields.**
        - Syntax:`{ $project : { <specification(s)> } }`
          - **`specification`**
            - **`{'field' : 'true'/'false'}`**
            - **`{'new_field' : {$operator: expression}}`**

              New fields can be included by various expressions including operations arithmetic, array, boolean, string expression operators and accumulators.
              Ex. `{$project : {new_field : {$log:[number, base]}}}`

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/project/
https://docs.mongodb.com/manual/reference/operator/aggregation/

MongoDB

Aggregation Pipeline

- Operators which can be used to construct expression in transform operations.

Operators	Type	Description	Syntax
$add	Arithmetic	Adds numbers to return the sum, or adds numbers and a date to return a new date.	{ $add: ["$<field1>", “$<field2>"] }
$eq/gt/lt	Comparison	Returns true if the values are equivalent/greater or less than the <value>.	{ $gt: ["$<field>", <value>] }
$filter	Array	Selects a subset of an array to return based on the specified condition.	{ $filter: { input:”$<array_field>” , as: “<new-varname>”, cond: <expression> } }
$size	Array	Counts and returns the total number of items in an array.	{ $size: “$<array_field>" }
$cond	Conditional	A ternary operator that evaluates one expression, and depending on the result, returns the value of one of the other two expressions.	{ $cond: [<boolean-expression>, <true-case>, <false-case>] }
$concat	String	Concatenates strings.	{ $concat: ["$<field1>", “$<field2>", ...] }
$dateDiff	Date	Returns the difference between two dates. Unit : year, month, day, hour, minute, etc.	{ $dateDiff: { startDate: “$<field1>", endDate: “$<field2>", unit: "day"}
$convert	Type	Converts a value to a specified type.	{$convert : {input : “$<field>”, to: “data_type”}}

https://docs.mongodb.com/manual/reference/operator/aggregation/

Example 2

In world_bank_project, return "id" and a non-empty “human_development_project” array which represents "mjtheme_namecode" only containing documents which name is "Human development”.

Expected Output

```
{
    "id" : "P129828",
    "human_development_project" : [
        {
            "name" : "Human development",
            "code" : "8"
        }
    ]
}
{
    "id" : "P132616",
    "human_development_project" : [
        {
            "name" : "Human development",
            "code" : "8"
        }
    ]
}
```

Example 2

In world_bank_project, return "id" and a non-empty “human_development_project” array which represents "mjtheme_namecode" only containing documents which name is "Human development”.

```
db.world_bank_project.aggregate(
    {$project: {"_id": false, "id": true,
                "human_development_project" :
                {$filter : {input: "$mjtheme_namecode",
                            as : "mjtheme_namecode_a",
                            cond: {$eq:["$$mjtheme_namecode_a.name",
                                        "Human development"]}}},
                }}
    )
```

Example 2

In world_bank_project, return "id" and a non-empty “human_development_project” array which represents "mjtheme_namecode" only containing documents which name is "Human development”.

```
db.world_bank_project.aggregate(
    {$project: {"_id": false, "id": true,
                "human_development_project" :
                {$filter : {input: "$mjtheme_namecode",
                            as : "mjtheme_namecode_a",
                            cond: {$eq:["$$mjtheme_namecode_a.name",
                                        "Human development"]}}},
                }},
     {$project: {"id":true, "human_development_project":true,
                 "size" : {$size : "$human_development_project"}}}
    )
```

Example 2

In world_bank_project, return "id" and a non-empty “human_development_project” array which represents "mjtheme_namecode" only containing documents which name is "Human development”.

```
db.world_bank_project.aggregate(
    {$project: {"_id": false, "id": true,
                "human_development_project" :
                {$filter : {input: "$mjtheme_namecode",
                            as : "mjtheme_namecode_a",
                            cond: {$eq:["$$mjtheme_namecode_a.name",
                                        "Human development"]}}},
                }},
     {$project: {"id":true, "human_development_project":true,
                 "size" : {$size : "$human_development_project"}}},
     {$match :  {"size":{$gt : 0}}},
     {$project: {"size":  false}}
    )
```

Output | Aggregate Query (line 17)

50 | Documents 1 to 50 | Pin Result | Query Code | Explain Query

```
{
    "id" : "P129828",
    "human_development_project" : [
        {
            "name" : "Human development",
            "code" : "8"
        }
    ]
}
```

MongoDB

Aggregation Pipeline

- Pipeline operators
  - **`$group`** : Group documents based on certain fields and combine their values.
    - Create/contain an _id field which contains the distinct group by key.
    - Syntax
      - `{ $group: { _id: “$<field>”,`
        `<new_field_name>: { <accumulator1> : <expression1> }, ... } }`
    - Accumulator operators can be used together. (also use with $<field path>)
      - `Ex. $sum, $avg, $min, $max, $first, $last, etc.`

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/group/

Example 3

For each countrycode, return the number of documents in the collection in a format of {"_id" : "code", "count" : value}

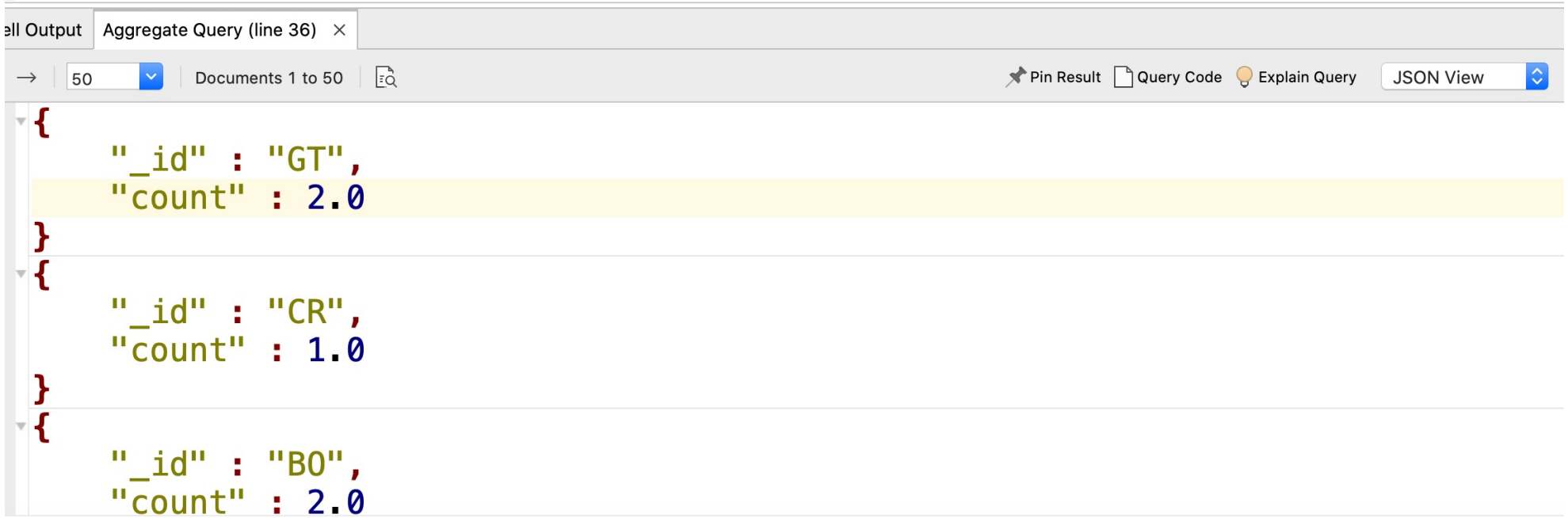
Expected Output


```
{
    "_id" : "GT",
    "count" : 2.0
}
{
    "_id" : "CR",
    "count" : 1.0
}
```

Example 3

For each countrycode, return the number of documents in the collection in a format of {"_id" : "code", "count" : value}

```
db.world_bank_project.aggregate({$group:{_id :"$countrycode", count:{$sum:1}}})
```



MongoDB

Aggregation Pipeline

- Pipeline operators
  - **`$unwind`** : Return each of an array element into a separate document.
    - Syntax
      - `{ $unwind: '$<field_path>' }`
        - **To specify a field path, prefix the field name with a dollar sign $ and enclose in quotes.**

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/unwind/

Example 4

Return theme_namecode as an _id and the number of project has the corresponding theme in the entire collection.

Expected Output


```
{
    "_id" : {
        "name" : "Judicial and other dispute resolution mechanisms",
        "code" : "32"
    },
    "count" : 2.0
}
{
    "_id" : {
        "name" : "Other communicable diseases",
        "code" : "64"
    },
    "count" : 3.0
}
```

Example 4

Return theme_namecode as an _id and the number of project has the corresponding theme in the entire collection.

```
db.world_bank_project.aggregate({$unwind : "$theme_namecode"},
                                {$group : {_id : "$theme_namecode",
                                           count:{$sum:1}}},
                               )
// Example 3
```

ll Output | Aggregate Query (line 38) ×

→ | 50 | Documents 1 to 50 | Pin Result | Query Code | Explain Query


```
{
    "_id" : {
        "name" : "Judicial and other dispute resolution mechanisms",
        "code" : "32"
    },
    "count" : 2.0
}
```

MongoDB

Aggregation Pipeline

- **`$sort`** : Collect all document and properly sort them.
  - Syntax
    - `{ $sort: { <field1>: <sort order>, <field2>: <sort order> ... } }`
      - `sort order : 1` (ascending), `-1` (descending)

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/sort/

Example 5

In world_bank_project, return the ‘approvalfy’ in double as ‘approvalfy_num’ with ‘borrower’, and ‘id’ ordered by approvalfy in descending order.

Expected Output

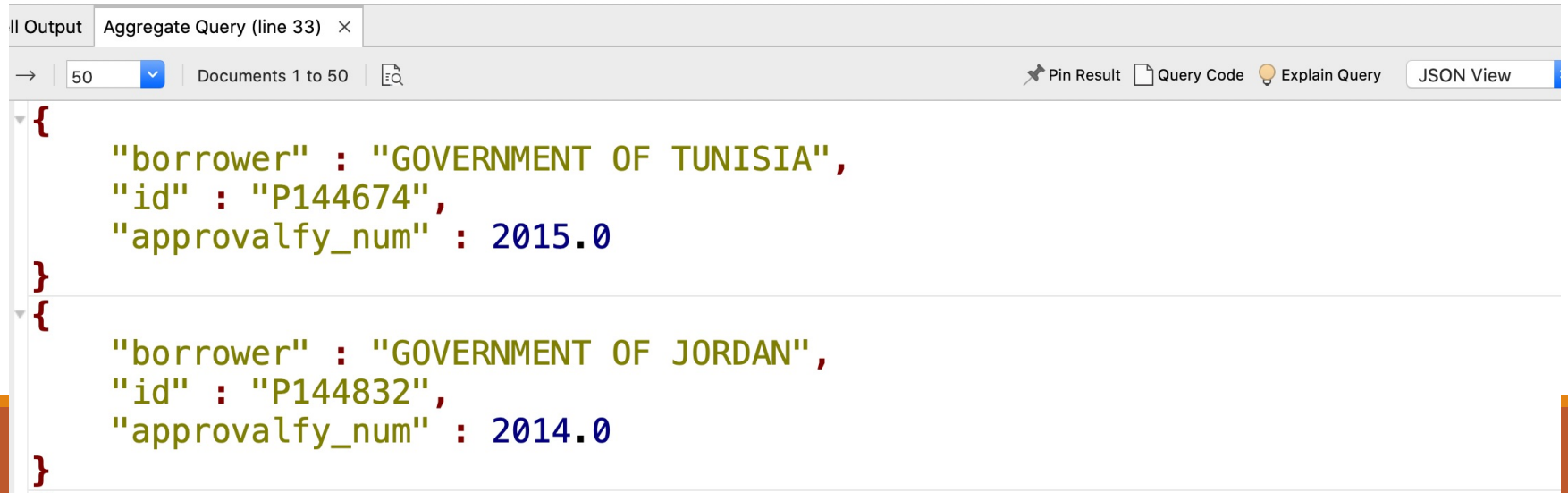

```
{
    "borrower" : "GOVERNMENT OF TUNISIA",
    "id" : "P144674",
    "approvalfy_num" : 2015.0
}
{
    "borrower" : "GOVERNMENT OF JORDAN",
    "id" : "P144832",
    "approvalfy_num" : 2014.0
}
```


Operators	Type	Description	Syntax
$convert	Type	Converts a value to a specified type.	{$convert : {input : “$<field>”, to: “data_type”}}

Example 5

In world_bank_project, return the ‘approvalfy’ in double as ‘approvalfy_num’ with ‘borrower’, and ‘id’ ordered by approvalfy in descending order.

```
db.world_bank_project.aggregate({$project:
                                    {"id" : true,
                                    "borrower" : true,
                                    "_id" : false,
                                    "approvalfy_num" : {$convert:
                                        {input:"$approvalfy", to:"double"}}
                                    }
                                },
                                {$sort:{"approvalfy_num":-1}})
```



Example 6 - Extra

In world_bank_project, 1) find a doucment where totalcommamt is greater than 75000000 and 2) find the unique names of projectdocs' DocType and the number of projectdocs belonging to the sorted by DocType.

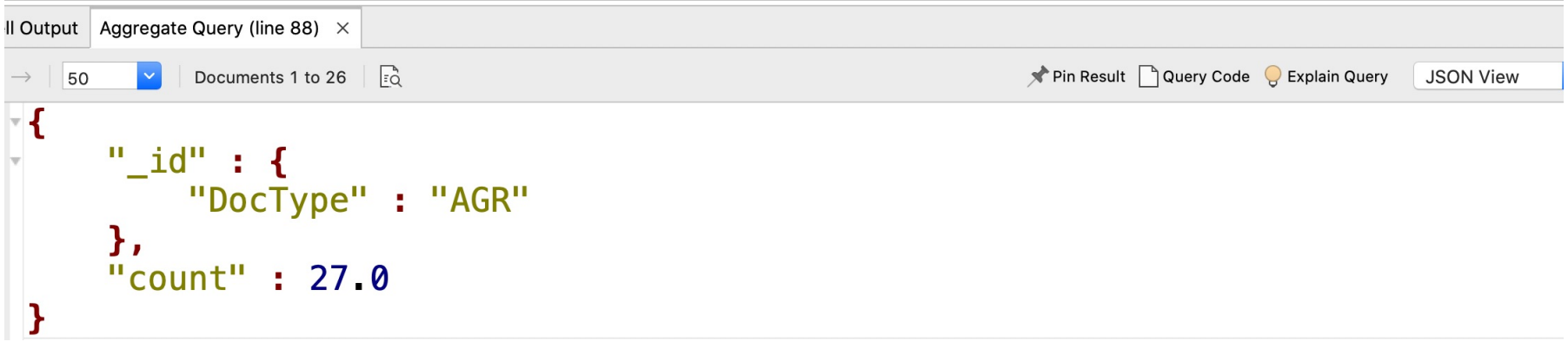
Expected Output


```
{
    "_id" : {
        "DocType" : "AGR"
    },
    "count" : 27.0
}
{
    "_id" : {
        "DocType" : "AGRS"
    },
    "count" : 1.0
}
```

Example 6 - Extra

In world_bank_project, 1) find a doucment where totalcommamt is greater than 75000000 and 2) find the unique names of projectdocs' DocType and the number of projectdocs belonging to the sorted by DocType.

```
db.world_bank_project.aggregate({$match:{"totalcommamt": {$gte : 75000000}}},
                                {$project:{"projectdocs":1}},
                                {$unwind:"$projectdocs"},
                                {$group:{"_id":{"DocType":
                                                "$projectdocs.DocType"},
                                         "count": {$sum: 1}}},
                                {$sort:{"_id.DocType":1}})
```



MongoDB

Aggregation Pipeline

- **`$lookup`** : Performs a left outer join to a collection.
  It adds a new array field whose elements are the matching documents from the "joined" collection.
  - Syntax
    - `db.<collection_1>.aggregate( { $lookup:{ localField: “<field_1>”, from: “<collection_2>”, foreignField: “<field_2>”, as: “<output_field>”}} )`

https://docs.mongodb.com/manual/reference/operator/aggregation-pipeline/
https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/

Example 7

Create a collection called country_code using country_codes.json.

For all the projects whose "countrycode" is "CN", create a field called country_phone_info which includes data from country_code (inclulding "name", "dial_code", "code", and “_id").

```
    "url" : "http://www.worldbank.org/projects/P127033/china-renewab
    "country_phone_info" : [
        {
            "_id" : ObjectId("61fbc97af5e71beac2ee0ce2"),
            "name" : "China",
            "dial_code" : "+86",
            "code" : "CN"
        }
    ]
```

Example 7

Create a collection called country_code using country_codes.json.

For all the projects whose "countrycode" is "CN", create a field called country_phone_info which includes data from country_code (inclulding "name", "dial_code", "code", and “_id").

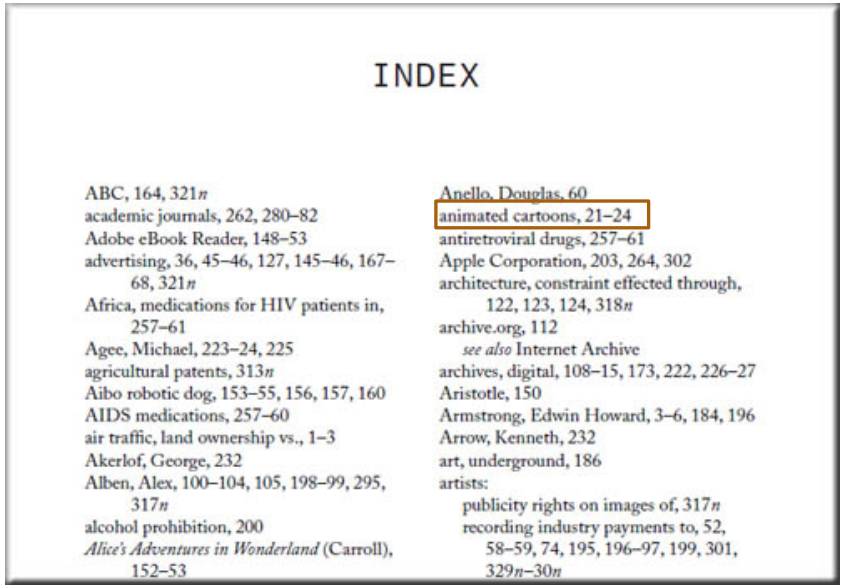
```
db.world_bank_project.aggregate([{$match:{"countrycode":"CN"}},
                                 {$lookup:{
                                     from:"country_code",
                                     localField:"countrycode",
                                     foreignField:"code",
                                     as:"country_phone_info"}
                                 }]
                                )
```

Content

MongoDB

- Aggregation Operations (aggregation pipeline)
- **Index**

Index



INDEX

ABC, 164, 321*n*
academic journals, 262, 280–82
Adobe eBook Reader, 148–53
advertising, 36, 45–46, 127, 145–46, 167–68, 321*n*
Africa, medications for HIV patients in, 257–61
Agee, Michael, 223–24, 225
agricultural patents, 313*n*
Aibo robotic dog, 153–55, 156, 157, 160
AIDS medications, 257–60
air traffic, land ownership vs., 1–3
Akerlof, George, 232
Alben, Alex, 100–104, 105, 198–99, 295, 317*n*
alcohol prohibition, 200
Alice's Adventures in Wonderland (Carroll), 152–53
Anello, Douglas, 60
animated cartoons, 21–24
antiretroviral drugs, 257–61
Apple Corporation, 203, 264, 302
architecture, constraint effected through, 122, 123, 124, 318*n*
archive.org, 112
see also Internet Archive
archives, digital, 108–15, 173, 222, 226–27
Aristotle, 150
Armstrong, Edwin Howard, 3–6, 184, 196
Arrow, Kenneth, 232
art, underground, 186
artists:
publicity rights on images of, 317*n*
recording industry payments to, 52, 58–59, 74, 195, 196–97, 199, 301, 329*n*–30*n*

- Organize data by a given field. ➔ Optimize queries and enhance query performance.
- Things to consider.
  - Which fields to index?
  - Index scheme : balanced search tree (B-tree), hash, etc.

Index

Mongo DB

- Without indexes, MongoDB must perform a *collection scan*, i.e. scan every document in a collection, to select those documents that match the query statement.
  - Index : Optimize query performance.
    - **Creating** Index
      - Syntax
        - `createIndex({“field” : value })`
          - **value**
            - Range-based indexes - Direction : 1 (Ascending), -1 (Descending)
              - MongoDB indexes use a B-tree data structure, by default.
            - Hashed indexes – “`hashed`”
            - Also supports other index types including text index, 2d index, etc.

https://docs.mongodb.com/manual/indexes/

Index

Mongo DB

- Without indexes, MongoDB must perform a *collection scan*, i.e. scan every document in a collection, to select those documents that match the query statement.
  - Index : Optimize query performance.
    - **Describing** the existing indexes on the collection.
      - Syntax
        - `getIndexes()`
          - Check which indexes exist on a collection.
          - Default : `_id` (Cannot be deleted)

https://docs.mongodb.com/manual/indexes/

Index

Mongo DB

- Without indexes, MongoDB must perform a *collection scan*, i.e. scan every document in a collection, to select those documents that match the query statement.
  - Index : Optimize query performance.
    - **Dropping or removing** the specified index from a collection.
      - Syntax
        - `dropIndex({“field” : direction})`

https://docs.mongodb.com/manual/indexes/

Example 8

For world_bank_project,

1) Find a document where "countrycode" is "AO" and see its executionStats using `cursor.explain("executionStats").`

2) Create a hashed-index on "countrycode".

3) Find a document where "countrycode" is "AO" and see its executionStats using `cursor.explain("executionStats").`

4) Drop the index.

Without an index, how many documents were examined for .find()?

- 500 (All)
- Less than 500, but higher than 1
- 1

Without an index, how many documents were examined for .find()?

500 (All)

Less than 500, but higher than 1

1

Without an index, how many documents were examined for .find()?

500 (All)

Less than 500, but higher than 1

1

Example 8

For world_bank_project,

1) Find a document where "countrycode" is "AO" and see its executionStats using `cursor.explain("executionStats").`

```
181 db.world_bank_project.find({"countrycode" : "AO"}).explain("executionStats") /
182
```

Aggregate | Aggregate | Aggregate | Aggregate | Aggregate | Aggregate | Aggregate | Document | Docume

50 Documents 1 to 1

```
36            },
37            "nReturned" : 1.0,
38            "executionTimeMillisEstimate" : 0.0,
39            "works" : 502.0,
40            "advanced" : 1.0,
41            "needTime" : 500.0,
42            "needYield" : 0.0,
43            "saveState" : 3.0,
44            "restoreState" : 3.0,
45            "isEOF" : 1.0,
46            "invalidates" : 0.0,
47            "direction" : "forward",
48            "docsExamined" : 500.0
49        }
```

Example 8

For world_bank_project,

2) Create a hashed-index on “countrycode”.

3) Find a document where "countrycode" is "AO” and see its executionStats using `cursor.explain(“executionStats").`

```
183 // 2) Create a hashed-index  on “countrycode”.
189 // 4) Drop the index
190 db.world_bank_project.dropIndex({"countrycode":"hashed"})
```

Aggregate × | Aggregate × | Aggregate × | Aggregate × | Document × | Document × | Docume

50 | Documents 1 to 1

```
1 {
2     "nIndexesWas" : 2.0,
3     "ok" : 1.0
4 }
5
```

```
46         "totalDocsExamined" : 1.0,
47         "executionStages" : {
48             "stage" : "FETCH",
49             "filter" : {
50                 "countrycode" : {
51                     "$eq" : "AO"
52                 }
53             },
```

Example 8

For world_bank_project,

4) Drop the index.

```
189 // 4) Drop the index
190 db.world_bank_project.dropIndex({"countrycode":"hashed"})
```

Aggregate | Aggregate | Aggregate | Aggregate | Document | Document | Docume

50 Documents 1 to 1

```
1 {
2     "nIndexesWas" : 2.0,
3     "ok" : 1.0
4 }
5
```

Reference

1. MongoDB Online Documentation : https://docs.mongodb.com/manual/tutorial/query-documents/