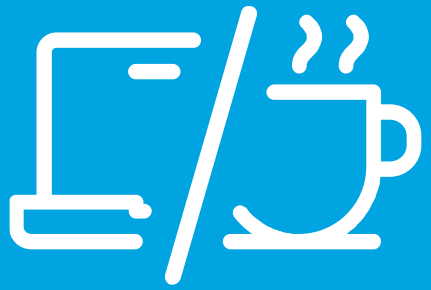


TECH breakFast

by asseco

CQRS (bez C) i MongoDB Aggregation Framework nie tylko dla programistów języka Java

Włodzimierz Kozłowski, Starszy Projektant

Wlodzimierz.Kozlowski@asseco.pl
wlodzimierz@kozlowscy.org
https://www.linkedin.com/in/wlodzimierz-kozlowski

Z jednej strony mamy dane, którymi zarządzamy zgodnie z jakimś modelem. MongoDB, przez wielu traktowane jako synonim bazy dokumentowej, daje nam “nieograniczone” możliwości modelowania tych danych bez sztywnych ograniczeń struktury i normalizacji relacyjnych baz danych. Z drugiej strony “Q” z genialnej koncepcji Younga i Dahana to “dobra odpowiedź na dobre zapytanie”. Jak wykorzystać *aggregation pipeline* do przygotowania odpowiedzi bez konieczności utrzymywania struktur zależnych od zapytań?!

CQS = Command and Query Separation

Asking a question should not change the answer.

A command (procedure) does something but does not return a result.
A query (function or attribute) returns a result but does not change the state.

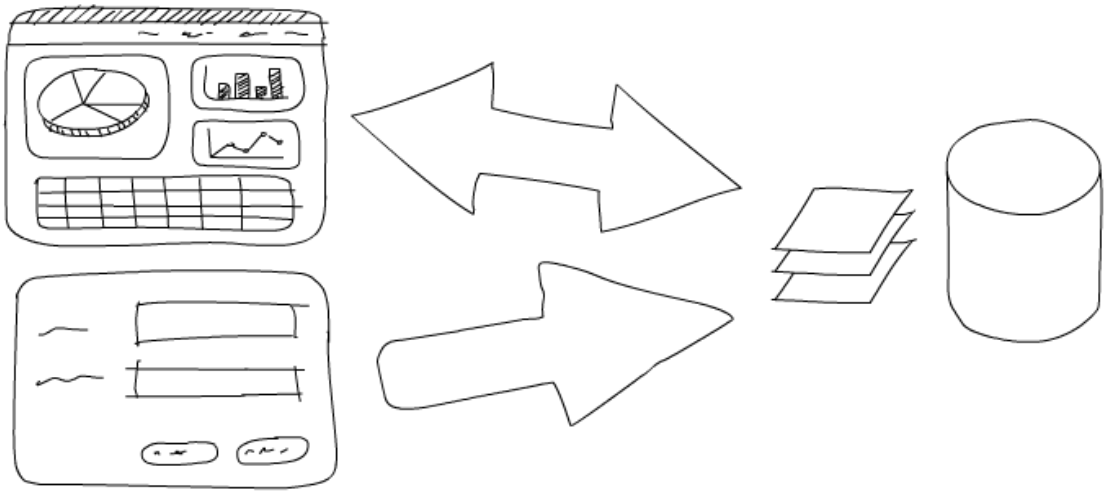
źródło: Bertrand Meyer, *Eiffel: a language for software engineering*
http://laser.inf.ethz.ch/2012/slides/Meyer/eiffel_laser_2012.pdf

CQRS = Command and Query Responsibility Segregation

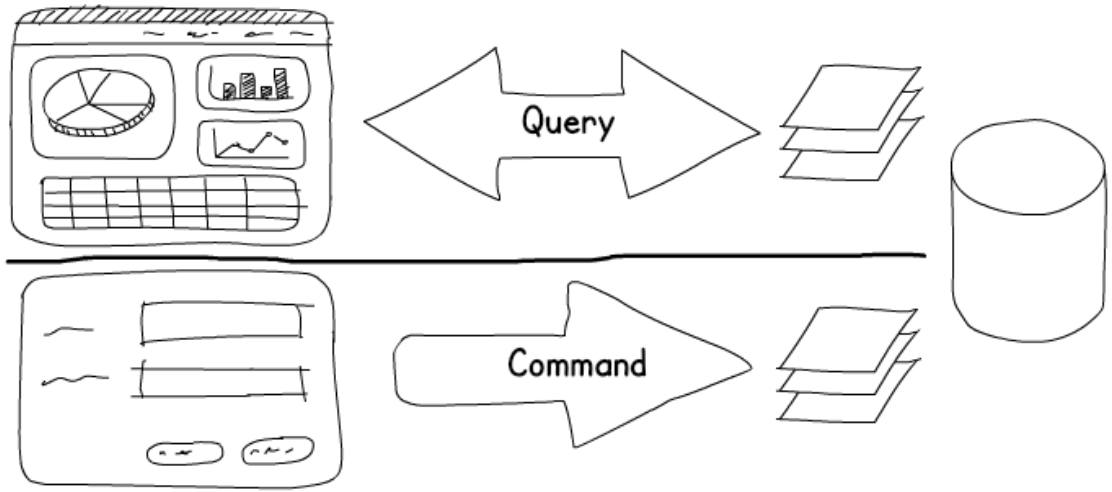
Command and Query Responsibility Segregation uses the same definition of Commands and Queries that Meyer used and maintains the viewpoint that they should be pure. The fundamental difference is that in CQRS objects are split into two objects, one containing the Commands one containing the Queries.

źródło: *CQRS Documents by Greg Young*
https://cqrs.files.wordpress.com/2010/11/cqrs_documents.pdf

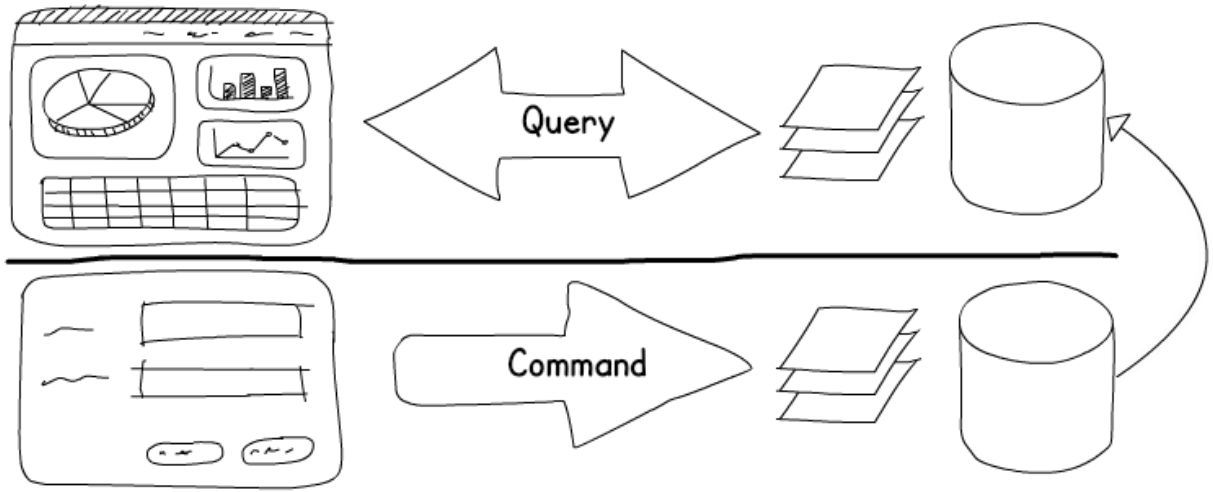
!CQRS



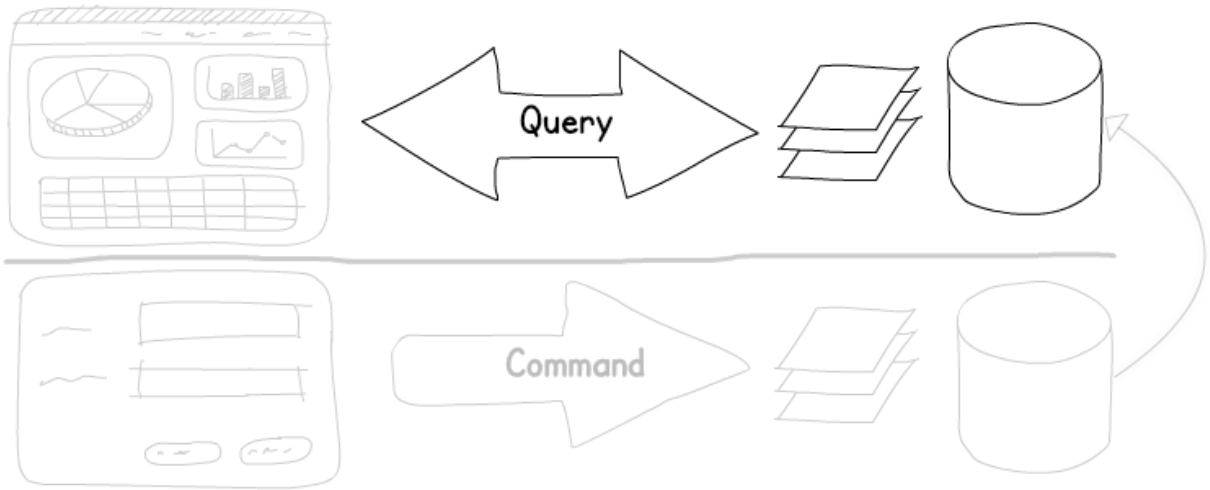
CQRS



CQRS/ES



CQRS (bez C) ...

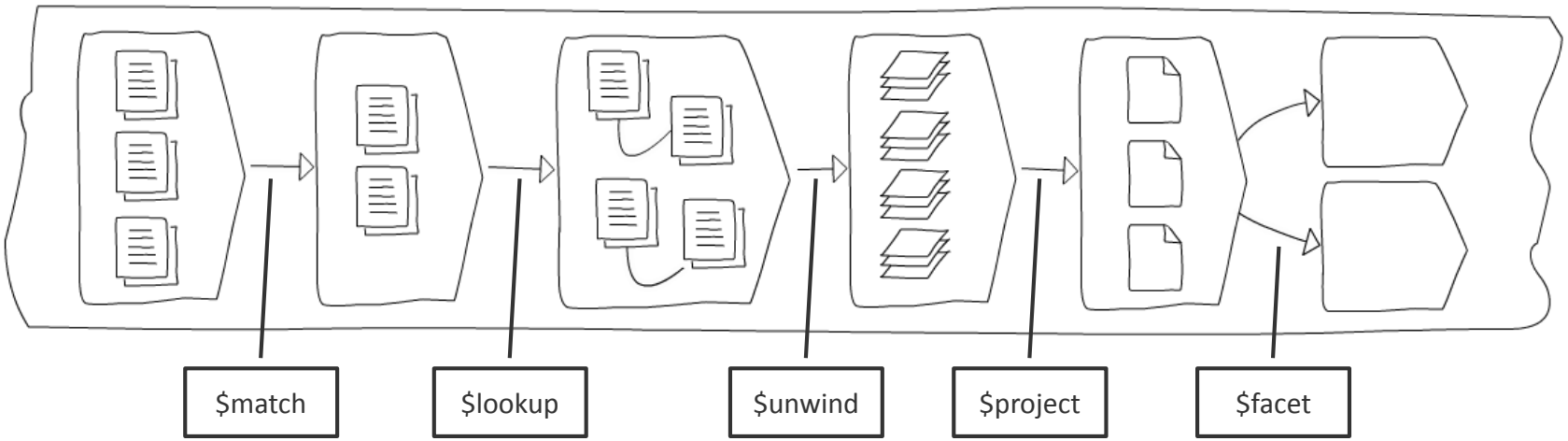


MongoDB / Aggregation Pipeline

The aggregation pipeline is a framework for data aggregation modeled on the concept of data processing pipelines. Documents enter a multi-stage pipeline that transforms the documents into aggregated results.

źródło: https://docs.mongodb.com/manual/core/aggregation-pipeline/

pipeline, stage



$match

/mætʃ/

Filters the document stream to allow only matching documents to pass unmodified into the next pipeline stage. $match uses standard MongoDB queries. For each input document, outputs either one document (a match) or zero documents (no match).

źródło: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$match

```
{ $match: { <query> } }

// WHERE, HAVING
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/match/#pipe._S_match

$match

```
// zawartość kolekcji
{ "_id" : ObjectId("512bc95fe835e68f199c8686"), "author" : "dave", "score" : 80, "views" : 100 }
{ "_id" : ObjectId("512bc962e835e68f199c8687"), "author" : "dave", "score" : 85, "views" : 521 }
{ "_id" : ObjectId("55f5a192d4bede9ac365b257"), "author" : "ahn", "score" : 60, "views" : 1000 }
{ "_id" : ObjectId("55f5a192d4bede9ac365b258"), "author" : "li", "score" : 55, "views" : 5000 }
{ "_id" : ObjectId("55f5a1d3d4bede9ac365b259"), "author" : "annT", "score" : 60, "views" : 50 }
{ "_id" : ObjectId("55f5a1d3d4bede9ac365b25a"), "author" : "li", "score" : 94, "views" : 999 }
{ "_id" : ObjectId("55f5a1d3d4bede9ac365b25b"), "author" : "ty", "score" : 95, "views" : 1000 }

// dopasowanie dla author = "dave"
db.articles.aggregate(
    [ { $match : { author : "dave" } } ] // dla "LIKE" $match: { $regex: "dave", $options: "i" }
);

{ "_id" : ObjectId("512bc95fe835e68f199c8686"), "author" : "dave", "score" : 80, "views" : 100 }
{ "_id" : ObjectId("512bc962e835e68f199c8687"), "author" : "dave", "score" : 85, "views" : 521 }

// nieco bardziej "wyuzdane„ warunki wyszukiwania
db.articles.aggregate( [
            { $match: { $or: [ { score: { $gt: 70, $lt: 90 } }, { views: { $gte: 1000 } } ] } },
            { $group: { _id: null, count: { $sum: 1 } } } // albo { $count: "count" }
] );

{ "_id" : null, "count" : 5 }
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/match/#pipe._S_match

$project

/prəˈdʒekt/

Reshapes each document in the stream, such as by adding new fields or removing existing fields. For each input document, outputs one document.

źródło: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$project

```
{ $project: { <specification(s)> } }

// SELECT
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/project/#pipe._S_project

$project

```
// project "name" and remove "_id"
db.solarSystem.aggregate([{ "$project": { "_id": 0, "name": 1 } }]);
// project "name" and "gravity" fields, including default "_id"
db.solarSystem.aggregate([{ "$project": { "name": 1, "gravity": 1 } }]);
// using dot-notation to express the projection fields
db.solarSystem.aggregate([{ "$project": { "_id": 0, "name": 1, "gravity.value": 1 } }]);
// reassing "gravity" field with value from "gravity.value" embeded field
db.solarSystem.aggregate([{"$project": { "_id": 0, "name": 1, "gravity": "$gravity.value" }}]);
// creating a document new field "surfaceGravity"
db.solarSystem.aggregate([{"$project": {
           "_id": 0,
           "name": 1,
           "surfaceGravity": "$gravity.value" }}]);
// creating a new field "myWeight" using expressions
db.solarSystem.aggregate([{"$project": {
           "_id": 0,
           "name": 1,
           "myWeight": { "$multiply": [ { "$divide": [ "$gravity.value", 9.8 ] }, 86 ] } }}]);
```

źródło: https://university.mongodb.com/courses/M121

$lookup

/lʊkʌp/

Performs a left outer join to another collection in the same database to filter in documents from the “joined” collection for processing.

więcej: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$lookup

```
{
   $lookup:
   {
       from: <collection to join>,
       localField: <field from the input documents>,
       foreignField: <field from the documents of the "from" collection>,
       as: <output array field>
   }
}

// JOIN
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/#pipe._S_lookup

$lookup

```
// "orders" collection (master)
db.orders.insert([
   { "_id" : 1, "item" : "almonds", "price" : 12, "quantity" : 2 },
   { "_id" : 2, "item" : "pecans", "price" : 20, "quantity" : 1 },
   { "_id" : 3  }
])

// "inventory" collection (details)
db.inventory.insert([
   { "_id" : 1, "sku" : "almonds", description: "product 1", "instock" : 120 },
   { "_id" : 2, "sku" : "bread", description: "product 2", "instock" : 80 },
   { "_id" : 3, "sku" : "cashews", description: "product 3", "instock" : 60 },
   { "_id" : 4, "sku" : "pecans", description: "product 4", "instock" : 70 },
   { "_id" : 5, "sku": null, description: "Incomplete" },
   { "_id" : 6 }
])
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/#pipe._S_lookup

$lookup

```
db.orders.aggregate([
   {
     $lookup:
       {
         from: "inventory",
         localField: "item",
         foreignField: "sku",
         as: "inventory_docs"
       }
  }
])
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/#pipe._S_lookup

$lookup

```
{
   "_id" : 1,
   "item" : "almonds",
   "price" : 12,
   "quantity" : 2,
   "inventory_docs" : [
      { "_id" : 1, "sku" : "almonds", "description" : "product 1", "instock" : 120 }
   ]
}
{
   "_id" : 2,
   "item" : "pecans",
   "price" : 20,
   "quantity" : 1,
   "inventory_docs" : [
      { "_id" : 4, "sku" : "pecans", "description" : "product 4", "instock" : 70 }
   ]
}
{
   "_id" : 3,
   "inventory_docs" : [
      { "_id" : 5, "sku" : null, "description" : "Incomplete" },
      { "_id" : 6 }
   ]
}
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/lookup/#pipe._S_lookup

$unwind

/ʌnˈwaɪnd/

Deconstructs an array field from the input documents to output a document for each element. Each output document replaces the array with an element value. For each input document, outputs n documents where n is the number of array elements and can be zero for an empty array.

więcej: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$unwind

```
{
  $unwind:
  {
      path: <field path>,
      includeArrayIndex: <string>,
      preserveNullAndEmptyArrays: <boolean>
  }
}
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/unwind/#pipe._S_unwind

$unwind

```
// kolekcja wejściowa "inventory"
{ "_id" : 1, "item" : "ABC1", sizes: [ "S", "M", "L"] }

// $unwind
db.inventory.aggregate( [ { $unwind : "$sizes" } ] )

// wynik
{ "_id" : 1, "item" : "ABC1", "sizes" : "S" }
{ "_id" : 1, "item" : "ABC1", "sizes" : "M" }
{ "_id" : 1, "item" : "ABC1", "sizes" : "L" }
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/unwind/#pipe._S_unwind

$group

/gru:p/

Groups input documents by a specified identifier expression and applies the accumulator expression(s), if specified, to each group. Consumes all input documents and outputs one document per each distinct group. The output documents only contain the identifier field and, if specified, accumulated fields.

więcej: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$group

```
{
  $group:
  {
    _id: <expression>,
    <field1>: { <accumulator1> : <expression1> },
    ...
  }
}

// GROUP BY
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group

$group

```
// dane wejściowe w kolekcji "sales"
{ "_id": 1, "item": "abc", "price": 10, "quantity": 2, "date": ISODate("2014-03-01T08:00:00Z") }
{ "_id": 2, "item": "jkl", "price": 20, "quantity": 1, "date": ISODate("2014-03-01T09:00:00Z") }
{ "_id": 3, "item": "xyz", "price": 5, "quantity": 10, "date": ISODate("2014-03-15T09:00:00Z") }
{ "_id": 4, "item": "xyz", "price": 5, "quantity": 20, "date": ISODate("2014-04-04T11:21:39.736Z") }
{ "_id": 5, "item": "abc", "price": 10, "quantity": 10, "date": ISODate("2014-04-04T21:23:13.331Z") }

// oczekiwany wynik grupowania
{ "_id" : { "month" : 3, "day" : 15, "year" : 2014 },
            "totalPrice" : 50, "averageQuantity" : 10, "count" : 1 }
{ "_id" : { "month" : 4, "day" : 4, "year" : 2014 },
            "totalPrice" : 200, "averageQuantity" : 15, "count" : 2 }
{ "_id" : { "month" : 3, "day" : 1, "year" : 2014 },
            "totalPrice" : 40, "averageQuantity" : 1.5, "count" : 2 }
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group

$group

```
db.sales.aggregate(
   [
      {
        $group : {
           _id : {
                      month: { $month: "$date" },
                      day: { $dayOfMonth: "$date" },
                      year: { $year: "$date" }
           },
           totalPrice: { $sum: { $multiply: [ "$price", "$quantity" ] } },
           averageQuantity: { $avg: "$quantity" },
           count: { $sum: 1 }
           // $addToSet, $first, $last, $push, $stdDevSamp ...
        }
      }
   ]
)
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/group/#pipe._S_group

$facet

/ˈfæs.ɪt/

Processes multiple aggregation pipelines within a single stage on the same set of input documents. Enables the creation of multi-faceted aggregations capable of characterizing data across multiple dimensions, or facets, in a single stage.

więcej: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

$facet

```
{
  $facet:
  {
    <outputField1>: [ <stage1>, <stage2>, ... ],
    <outputField2>: [ <stage1>, <stage2>, ... ],
    ...
  }
}
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/facet/#pipe._S_facet

$facet

```
db.artwork.aggregate( [
  {
    $facet: {
      "categorizedByTags": [ // pipeline 1
        { $unwind: "$tags" },
        { $sortByCount: "$tags" }
      ],
      "categorizedByPrice": [ // pipeline 2
        { $match: { price: { $exists: 1 } } },
        {
          $bucket: { // "wiaderka" z granicami
            groupBy: "$price",
            boundaries: [  0, 150, 200, 300, 400 ],
            default: "Other",
            output: {
              "count": { $sum: 1 },
              "titles": { $push: "$title" }
            }
          }
        }
      ],
      "categorizedByYears(Auto)": [ // pipeline n
        {
          $bucketAuto: { // automatyczne "wiaderko"
            groupBy: "$year",
            buckets: 4
          }
        }
      ]
    }
  }
])
```

źródło: https://docs.mongodb.com/manual/reference/operator/aggregation/facet/#pipe._S_facet

(...)

$addFields

$geoNear

$sortByCount

$count

$indexStats

$listSessions

$out

$sort

$graphLookup

$bucket

$limit

$skip

$bucketAuto

$collStats

$redact

więcej: https://docs.mongodb.com/manual/meta/aggregation-quick-reference/

/src/main/java

```
// dokument z kolekcji "controlled_points"
{
  "_id": "5cfa180bf64a4a3e40e66c4c",
  "droId": ObjectId("5cf66733f64a4a50cc7e1789„),
  "controlledPointNo": "A0938727",
  "controlledPointName": "klimatyzator",
  "controlledPointStatus": "ACTIVE",
  "controlledPointType": "HVAC840",
  "controlledPointSort": "DEVICE",
  "address": {
    "postalCode": "85-752",
    "cityCode": 0,
    "cityName": "Bydgoszcz",
    "streetCode": 0,
    "streetName": "Fordońska",
    "buildingNo": "2",
    "apartmentNo": "7.13",
    "postOffice": "string"
  },
  "direction": "DELIVERED",
  "droMeterNo": "123456.8",
  "meterChannel": 1,
  "commProtocolName": "ZigBee v2.8",
  "urlControlledPoint": "172.123.234"
}
```

```
// dokument z kolekcji "demand_reduction_objects"
{
  "_id": "5cf66733f64a4a50cc7e1789",
  "validityCertDateFrom": "2018-02-01T00:00:00.000Z",
  "address": {
    "postalCode": "85-752",
    "cityCode": 0,
    "cityName": "Bydgoszcz",
    "streetCode": 0,
    "streetName": "Fordońska",
    "buildingNo": "2",
    "apartmentNo": "7.13",
    "postOffice": "string"
  },
  "droName": "Betoniarnia",
  "certificateNo": "2018/01/02",
  "droNo": "MRID01",
  "droStatus": "INACTIVE",
  "droType": "RECEIVING_POINT"
}
```

/src/main/java

```
// odpowiedź na zapytanie - "dobra odpowiedź na dobre zapytanie"
@Data
@ApiModel(description = "Response for find controlled points query")
public class FindControlledPointsResponse {
  String controlledPointId;
  String controlledPointNo;
  String controlledPointName;
  String droId;
  String droNo;
  Integer meterChannel;
  BigDecimal dsrMinusPower;
  BigDecimal dsrPlusPower;
}
```

/src/main/java

```
db.controlled_points.aggregate([
  {
    "$match": {
      "controlledPointNo": {
        "$regex": "xxx",
        "$options": "i"
      },
      "controlledPointName": {
        "$regex": "yyy",
        "$options": "i"
      }
    }
  },
  {
    "$lookup": {
      "from": "demand_reduction_objects",
      "localField": "droId",
      "foreignField": "_id",
      "as": "demandReductionObject"
    }
  },
  {
    "$unwind": "$demandReductionObject"
  },
  {
    "$project": {
      "controlledPointNo": 1,
      "controlledPointName": 1,
      "droId": 1,
      "meterChannel": 1,
      "controlledPointId": "$_id",
      "droNo": "$demandReductionObject.droNo"
    }
  },
  {
    "$match": {
      "droNo": {
        "$regex": "zzz",
        "$options": "i"
      }
    }
  },
  {
    "$sort": {
      "controlledPointNo": 1
    }
  },
  {
    "$skip": 0
  },
  {
    "$limit": 10
  }
])
```

README

- https://www.google.com/search?q=cqrs
- Martin Fowler, *CQRS*
  https://martinfowler.com/bliki/CQRS.html
- Maciej Aniserowicz, *CQRS pragmatycznie*, Programista 2/2016 (45)
- *M121: The MongoDB Aggregation Framework*
  https://university.mongodb.com/courses/M121/about

git clone

- https://github.com/wkozi/cqrs-bez-c-i-mongodb-aggregation-framework
- Bydgoszcz, ul. Fordońska 2, p. **7.16**

@author

„Programujący projektant” w Pionie Energetyki i Gazownictwa Asseco Poland S.A.

Od kilkunastu lat związany z największym systemem bilingowym w polskiej energetyce a przede wszystkim ludźmi, którzy go tworzą. Ostatnio oddelegowany także do realizacji projektu „Badania nad zastosowaniem technologii Internet-of-Things oraz opracowaniem modeli regulacji przy użyciu różnych urządzeń, w tym akumulatorów domowych i pojazdów elektrycznych…”.

Od jakiegoś czasu żywo zainteresowany technologiami „dużych danych”. Absolwent studiów podyplomowych „Big Data – przetwarzanie i analiza dużych zbiorów danych” na Wydziale Elektroniki i Technik Informacyjnych Politechniki Warszawskiej.

Prywatnie, oprócz standardowego zestawu: praca, żona (1) i dzieci (3), instruktor ZHR (inowroclaw.zhr.pl) i „wędrownik” (wedrownicy.org), który zabiera plecak i samotnie wędruje zimą po Uralu albo rowerem po Górskim Karabachu.

Solutions for demanding business.

Zastrzeżenia prawne

Zawartość dostępna w prezentacji jest chroniona prawem autorskim i stanowi przedmiot własności. Teksty, grafika, fotografie, dźwięk, animacje i filmy, a także sposób ich rozmieszczenia w prezentacji podlegają ochronie na mocy Ustawy o prawach autorskich i prawach pokrewnych oraz innych przepisów z tym związanych. Jakiekolwiek nieautoryzowane zastosowanie jakichkolwiek materiałów zawartych w prezentacji może stanowić naruszenie praw autorskich, znaków firmowych lub innych przepisów. Materiały dostępne w prezentacji nie mogą być modyfikowane, powielane, przedstawiane publicznie, wykonywane, rozprowadzane lub wykorzystywane w innych celach publicznych lub komercyjnych, chyba że Asseco Poland S.A. wydał na to wyraźną zgodę na piśmie. Kopiowanie w celach komercyjnych, rozpowszechnianie, modyfikacja lub przejmowanie zawartości niniejszej prezentacji przez osoby trzecie jest niedozwolone. W prezentacji mogą być prezentowane również materiały zawierające odesłania do ofert i usług podmiotów trzecich. Warunki korzystania z ofert i usług podmiotów trzecich są określone przez te podmioty. Asseco Poland S.A. nie ponosi żadnej odpowiedzialności za warunki i skutki korzystania z ofert i usług tychże podmiotów. Dane i informacje zawarte w prezentacji mają jedynie charakter ogólnoinformacyjny. Prezentacja przygotowana została w oparciu i przy użyciu produktów firmy Inscale.

Nazwa oraz logo Asseco Poland S.A. są zarejestrowanymi znakami towarowymi. Korzystanie z tych znaków wymaga wyraźnej zgody ze strony Asseco Poland S.A.