

# IBM Cloudant – търсене на съдържание

## I. Въведение

Търсенето на съдържание в IBM Cloudant (CouchDB) бази данни може да се реализира по два начина:

- Използване на дизайн документи (design) и ресурси, формиращи изгледа (view).
- Използване на Mango заявки.

Базовият подход за обработка на заявки към IBM Cloudant е създаването на View ресурси. Всеки дизайн документ съдържа един или колкото е необходимо view ресурси чрез които се формира отговора посредством функцията **emit**. Всеки view ресурс може да реализира MapReduce функционалност. За целта **map** функцията е стойността на поле "map", а **reduce** функцията е стойността на поле "reduce".

Предпочитаният подход е да използвате **Mango заявки**. Mango е акроним, който идва от "MongoDB inspired query language interface for Apache CouchDB". Целта е предоставяне на възможност за генериране на заявки с цел извличане на информация от базата данни, без да е необходимо да се пишат view ресурси и без да се използва MapReduce. Тези заявки се изпълняват от 2 до 10 пъти по-бързо отколкото обслужваните чрез Erlang заявки. Заявките Mango се обработват от Mango Query Server, който е част от всеки CouchDB сървър версия 2.0.0+. Този сървър позволява обработка на декларативен тип заявки, без да е необходимо да се пишат ресурси на JavaScript. Препоръчва се да се използват само Mango заявки при разработване на приложенията. Причините за това са няколко:

- Mango заявките се обработват по-бързо от Erlang заявките.
- Erlang Query Server не е достатъчно сигурен. За разлика от JavaScript Query Server той не функционира в sandbox режим. Следователно Erlang програмния код има пълен достъп до апаратните и програмни ресурси на хоста, в зависимост от правата на достъп на текущият потребител.

## II. Използване на view ресурси

За да може да се обработват заявките от страна на клиента, трябва да създадете необходимите дизайн документи и view ресурси, които генерират отговор, който се визуализира от брауъра. Базата данни CouchDB използва MapReduce механизъм, за да формира отговор. За целта всеки **view ресурс** може да има **map** и **reduce** функции. Те се стартират за всеки документ от базата данни и логиката им не зависи от нищо друго извън документа. Тази независимост позволява паралелната обработка на view ресурсите. Тъй като тези функции формират отговор във формат "key:value" те се индексират по ключ чрез B+ дърво. Това позволява бързо търсене на информация по стойността на ключ или по минимална и максимална стойност на ключа. На Фиг. 1 е показан синтаксиса на дизайн документ, който съдържа два view ресурса.

```

{
  "_id": "_design/name",
  "_rev": "1-11b803fe24ddfbcb6fd8d47dd7150225c",
  "language": "javascript",
  "views": {
    "viewFuncName1": {
      "map": "function(doc) {
        emit(key, value);
      }",
      "reduce": "function(key, value, rereduce) {
        return sum(value)/value.length;
      }"
    },
    "viewFuncName2": {
      "map": "function(doc) {
        emit(key, value);
      }",
      "reduce": "_stats"
    }
  }
}

```

Фиг. 1 Дизайн документ – структура и синтаксис

Както всеки друг документ, всеки дизайн документ има системни полета “\_id” и “\_rev”. Може да смените стойността на системния идентификатор, както е показано на Фиг. 1. За име на документа задайте низ, който ще ви подсеща за какво точно служи документа. На този етап поле “language” има стойност “javascript”. Това подсказва, че документа съдържа view ресурси в които има JavaScript код. Всички view ресурси са в тялото на поле “views”. Избирайте имената на view ресурсите така, че да са максимално информативни. Всеки ресурс трябва да има **map** функция и незадължителна **reduce** функция. Тези функции се декларират като анонимни JavaScript функции. Всяка map функция трябва да завършва с **emit** функция, която указва какво представлява ключа (key) и каква е стойността (value), която се връща (JSON обект). Функциите map получават един аргумент – обект, който е един от обработваните документи (doc). От тялото на map функцията може да проверите дали този документ трябва да се обработва или не (например, това може да е дизайн документ). Следва логика чрез която се реализира желаното филтриране. Ключът показва кое свойство ще се използва като ключ. Това позволява да филтрираме информацията в документите, в зависимост от стойността на един или няколко ключа. Ако трябва да се използват няколко ключа се използват няколко **emit** функции, или се използва съставен ключ (масив от ключове).

За да използвате view ресурсите е необходимо да създадете URL низ в който е описано кой view ресурс да се използва и в кой дизайн документ е този ресурс, например:

[https://hostName/databaseName/\\_design/designDocName/\\_view/viewResName?key=keyValue](https://hostName/databaseName/_design/designDocName/_view/viewResName?key=keyValue)

Името на хоста “hostName” зависи дали работите локално с CouchDB или с IBM Cloudant. Към всеки view ресурс може да предавате конкретна стойност за ключа, “key=keyValue”. Стойността на ключа може да бъде число, низ или масив от елементи, ако ключа е съставен.

### III. Използване на Mango заявки

Основният начин за изпращане на Mango заявки е чрез ресурс **\_find**. Заявката до **\_find** се изпраща като се използва HTTP **POST** метод. Всяка заявка може да използва собствено индексирание или това, което е активно за “\_all\_docs”. В HTTP заглавния блок задължително трябва да се зададе “Content-type: application/json”, тъй като заявката е JSON обект. Този обект съдържа следните по-важни свойства:

- **selector** – Задължително поле, което задава критерия по които се търси информация в базата данни като JSON обект;
- **sort** – Незадължително поле - JSON обект, който описва декларативно какво сортиране трябва да се приложи към заявката;
- **fields** – Незадължително поле - JSON масив, чрез който се задава кои полета на документа трябва да формират отговора. При пропускане на това свойство се връща съдържанието на целия обект.
- **limit** – Незадължително поле - максимален брой на върнатите обекти. По подразбиране 25.
- **skip** – Незадължително поле - пропускане на първите *n* резултата, където *n* е зададената стойност след skip (skip: n).
- **use\_index** - Незадължително поле - инструктира сървъра, че заявката ще използва точно определен индекс. Индексът се описва като низ, съдържащ името на дизайн документа с индекса или масив съдържащ името на дизайн документа и името на индекса: ["design\_document", "index\_name"].
- **execution\_stats** – Незадължително поле - при стойност true в отговора се включва и статистическа информация за обслужената заявка, например: общ брой анализирани документи; брой документи, върнати като резултат, време за изпълнение на заявката в ms.

Ще опишем синтаксиса на така изброените свойства. За примерна база данни ще използваме “students”, която съдържа информация за студенти.

#### Свойство selector

Стойността на свойство selector (JSON) задава каква точно информация търси клиента, например:

```
{
  "specialty": "KCT"
}
```

В този случай ще се върне информация за студентите, които са от специалност KCT. Всеки селектор може да съдържа толкова полета, колкото са необходими, например:

```
{
  "specialty": "KCT",
  "course": 1
}
```

В този случай ще се върнат всички студенти от специалност KCT, които в момента са в първи курс. Някои от полетата от документите може да съдържат други JSON обекти или да са JSON масиви. В този случай трябва да се укаже кое вложено поле ще се използва с цел генериране на заявката. За да намерим всички студенти с фамилия Иванов трябва да използваме следния селектор:

```
{
  "name.family": "Иванов"
}
```

За да създадете по-сложни като логика заявки можете да използвате **Mango оператори** или комбинация от Mango оператори. Всеки оператор започва със знака за долар \$. Операторите се делят на:

- **Оператори за комбиниране:** \$and, \$or, \$not, \$nor, \$all, \$elemMatch, \$allMatch, \$keyMapMatch.
- **Условни оператори:** \$eq, \$ne, \$gt, \$gte, \$lt, \$lte, \$exist, \$type, \$in, \$min, \$mod, \$size, \$regex

Операторите за комбиниране се използват за комбиниране на селектори. В допълнение към общите логически оператори има три оператора (\$all, \$elemMatch и \$allMatch), които ви помагат да работите с JSON масиви, и един, който работи с JSON карти (\$keyMapMatch).

Условните оператори са специфични за дадено поле и се използват за оценка на стойността, съхранена в това поле.

Можете да създавате по-сложни изрази за селектор чрез комбиниране на оператори. За най-добра производителност е най-добре да обединявате оператори за комбиниране или условни оператори, например оператора за търсене чрез регулярни изрази \$regex с условни оператори, като например \$eq, \$gt, \$gte, \$lt и \$lte (но не и \$ne). Не използвайте оператор \$regex с големи масиви с данни, тъй като производителността ще е ниска.

Ако трябва да намерим всички студенти от специалност КСТ, които са във втори курс, може да използваме следния селектор:

```
"$and": [
  {
    "specialty": { "$eq": "КСТ" }
  },
  {
    "course": { "$eq": 2 }
  }
]
```

Ако трябва да получите имената на студентите, които са от няколко курса, например 2-ри и 4-ти, може да използвате оператор \$in. Стойността трябва да е масив, всеки елемент от който са номерата на курсовете от които се интересувате:

```
"course": { "$in": [2, 4] }
```

Когато трябва да извлечем информация за параметър, който приема минимална и максимална стойност можем да използваме условни оператори, например:

```
"course": { "$gte": 1 }, "course": { "$lte": 3 }
```

Този селектор връща всички студенти от 1-ви до 3-ти курс.

Ако трябва да намерим всички студенти, които имат фамилия, която започва с буква "И" можем да използваме регулярен израз, например:

```
"name.family": { "$regex": "^И" }
```

Нека да намерим имената на всички студенти, които имат двойки по дисциплината НБД. За целта можем да използваме следния селектор:

```
"grade": {
  $elemMatch: {
    "$and": [
      {
        "subject": { "$eq": "КСТ" }
      },
      {
        "value": { "$eq": 2 }
      }
    ]
  }
}
```

### Свойство sort

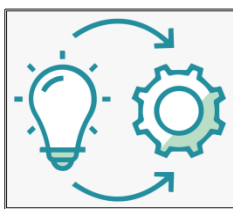
Стойността на поле sort съдържа масив от JSON обекти или имена на полета чрез които се задава какво сортиране на данните трябва да се приложи. Може да се използват множество полета при сортирането. Данните се сортират с приоритет, от първото към последното поле. Подредбата на резултата може да бъде във низходяща или възходяща посока. Посоката се задава като стойност на полето:

- asc – възходяща посока;
- desc – низходяща посока.

Нека да сортираме трите имена на всички студенти с фамилия Иванов. Следва съдържанието на Mango заявката:

```
{
  "selector": {
    "name.family": "Иванов"
  },
  "sort": [ { "name": "desc" } ],
  "fields": [ "name" ]
}
```

## IV. Задачи за изпълнение



**Задача 1:** Създайте база данни “students2”, която позволява извличане на информация за студентите от дадена специалност: факултетен номер, трите имена, курс на обучение, оценки по всяка дисциплина за всеки семестър на обучение. Напишете необходимите view ресурси, които позволяват получаване на следната информация:

1. Име на студента и курс на обучение по зададен факултетен номер.
2. Факултетен номер на студента по зададено име.
3. Имената на студентите от зададена специалност.
4. Оценките на студент по зададено име за даден семестър.
5. Статистика на оценките за зададена специалност.
6. Статистика за оценките за зададен семестър.
7. Статистика на оценките за зададена дисциплина.

```

{
  "_id": "student-0001",
  "_rev": "1-96fa0a50e00ef79eaf3efacb39012b38",
  "type": "student",
  "name": {
    "firstname": "Стоян",
    "family": "Иванов",
    "lastname": "Стоянов"
  },
  "id": "123457",
  "specialty": "АИУТ",
  "course": 1,
  "semester": [
    {
      "number": 1,
      "subjects": [
        {
          "name": "Висша математика I",
          "grade": 2
        },
        {
          "name": "Химия",
          "grade": 3
        },
        // Други дисциплини от Семестър 1
      ]
    },
    {
      // Информация за следващ семестър
    }
  ]
}

```

Фиг. 2 Примерно съдържание на документите от база данни “students2”

За всеки студент ще създадем документ, който ще описва неговите имена, специалност, курс и оценките по дисциплините по семестри. Идентификационният код на документите може да съвпада с факултетния номер на студента, тъй като той е уникален за всеки студент в дадено учебно заведение. Тъй като е възможно един студент да промени факултетния си номер по време на следването (издаване на нова студентска книжка) или да има две студентски книжки (изучава две специалности едновременно) е по-добре факултетният номер да е свойство, а не уникален ключ.

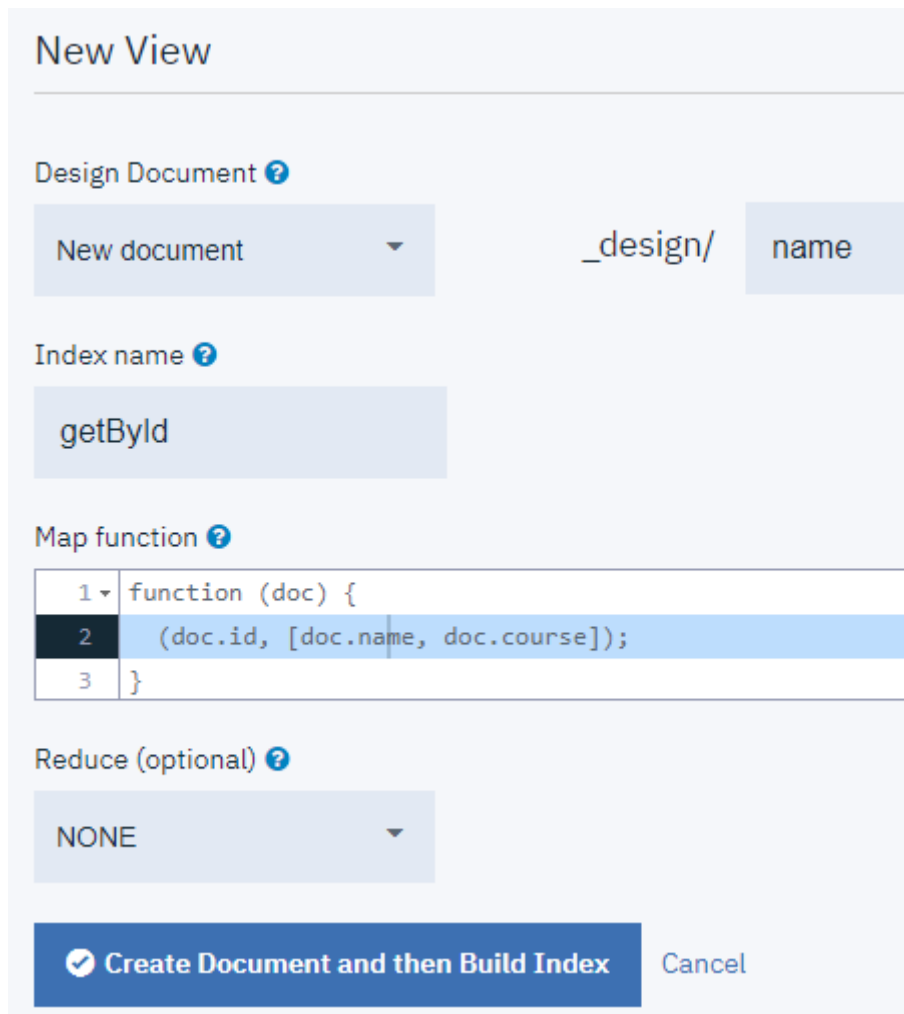
Създайте нова база данни “students2” в IBM Cloud. Получете двойката “ключ-стойност” за достъп до базата данни с цел четене на информация от нея. Създайте поне три документа, всеки от които описва студент, съгласно заданието. На Фиг. 2 е показано примерно съдържание на един документ от тази база данни.

Основните полета и тяхното предназначение са следните:

- Поле **“type”** – чрез това поле view ресурсите могат бързо да филтрират документите, които не съдържат описание на студенти.
- Поле **“name”** – съдържа трите имена на студента.
- Поле **“semester”** – масив от JSON обекти - съдържа информация за номера на семестъра (поле “number”) и информация за изучаваните дисциплини (поле “subjects”). Това поле е масив, елементите от който съдържат информация за име на дисциплина (поле “name”) и оценката (поле “grade”) по тази дисциплина.

### 1. Получаване на името на студента и курса на обучение по зададен факултетен номер

Ще създаден *дизайн документ* с име `name` и `view` ресурс с име **getByld**. За целта изберете от менюто на IBM Cloudant Dashboard “Design Document” и след това “New View” (виж Фиг. 3). Задайте “New document” за “Design Document” и име на документа “name”. Името на `view` документа е “getByld”. Редактирайте съдържанието на **map** функцията. Задайте за ключ “doc.id”, а за стойност масив, който съдържа името на студента “doc.name” и курса на обучение “doc.course”. Създайте документа чрез натискане на бутон “Create document and then build index”.



New View

Design Document ?

New document      \_design/      name

Index name ?

getByld

Map function ?

```
1 function (doc) {  
2   (doc.id, [doc.name, doc.course]);  
3 }
```

Reduce (optional) ?

NONE

✓ Create Document and then Build Index      Cancel

Фиг. 3 Създаване на нов `view` ресурс

Съдържанието на новия документ “\_design/name” е следното:

```
{  
  "_id": "_design/name",  
  "_rev": "1-274dc0094371ce7984bab146e5a242f",  
  "views": {  
    "getById": {  
      "map": "function (doc) {\n (doc.id, [doc.name, doc.course]);\n}"  
    }  
  },  
  "language": "javascript"  
}
```

Остава да изпратим заявката, като използваме създадения view ресурс, например:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/name/\\_view/getById](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/name/_view/getById)

При тази заявка ще получим информация за всички студенти във формат ключ-стойност:

```
{ "total_rows": 3, "offset": 0, "rows": [
  { "id": "student-0002", "key": "123456", "value": [{ "firstname": "Иван", "family": "Иванов", "lastname": "Иванов" }, 2] },
  { "id": "student-0001", "key": "123457", "value": [{ "firstname": "Стоян", "family": "Иванов", "lastname": "Стоянов" }, 1] },
  { "id": "student-0003", "key": "123458", "value": [{ "firstname": "Надя", "family": "Иванова", "lastname": "Петкова" }, 2] }
] }
```

Ако желаете тази информация за конкретен студент трябва да предадете към view ресурса конкретна стойност за ключа, например:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/name/\\_view/getById?key='123457'](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/name/_view/getById?key='123457')

В този случай сървърът ще върне информация само за студента със зададения идентификатор (факултетен номер):

```
{ "total_rows": 3, "offset": 1, "rows": [
  { "id": "student-0001", "key": "123457", "value": [{ "firstname": "Стоян", "family": "Иванов", "lastname": "Стоянов" }, 1] }
] }
```

Ако искате да получите данните за няколко студента може да използвате параметър **keys**, например:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/name/\\_view/getById?keys=\['123457', '123456'\]](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/name/_view/getById?keys=['123457', '123456'])

Ако искате да получите данните на студенти с факултетни номера попадащи в интервал трябва да използвате параметри **startkey** и **endkey**, например:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/name/\\_view/getById?startkey='123456'&endkey='123458'](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/name/_view/getById?startkey='123456'&endkey='123458')

## 2. Получаване на факултетен номер на студента по зададено име

Създайте нов view ресурс при който ключът е името на студента, а стойността - факултетния му номер:

```
{
  "_id": "_design/id",
  "_rev": "1-11b803fe24ddfbcb6fd8d47dd7150225c",
  "language": "javascript",
  "views": {
    "getByName": {
      "map": "function(doc) {emit (doc.name, doc.id);}"
    }
  }
}
```



### 3. Получаване на името на студентите от зададена специалност

Създайте нов view ресурс при който ключът е името на студента, а стойността - специалността която учи:

```
{
  "_id": "_design/name",
  "_rev": "1-d861ad1dc20a8df29cbfed9f1a3c062b",
  "language": "javascript",
  "views": {
    "getBySpecialty": {
      "map": "function(doc) { emit(doc.specialty, doc.name); }"
    }
  }
}
```

### 4. Получаване на оценките на студент за даден семестър по зададено име

В този случай трябва да използваме съставен ключ, който се състои от името на студента и номера на семестъра. Стойността, която връща функцията **emit** трябва да е JSON обект, който съдържа името на дисциплината и оценката по тази дисциплина. За целта създайте view ресурс **getByNameAndSemester**. От тялото на map функцията трябва да обходим програмно два масива - "semester" и "subjects":

```
{
  "_id": "_design/grade",
  "_rev": "1-d861ad1dc20a8df29cbfed9f1a3c062b",
  "language": "javascript",
  "views": {
    "getByNameAndSemester": {
      "map": "function(doc) {
        doc.semester.forEach(function(semester) {
          semester.subjects.forEach(function(subject) {
            emit([doc.name, semester.number], subject);
          });
        });
      }"
    }
  }
}
```

Следва примерна заявка, която връща оценките на студента Иван Иванов за първи семестър:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-blueimix.cloudant.com/students2/\\_design/grade/\\_view/getByNameAndSemester?key=\[{"firstname":"Иван","family":"Иванов","lastname":"Иванов"},1\]](https://dc3b328b-545c-4c02-9457-531fdc110a3e-blueimix.cloudant.com/students2/_design/grade/_view/getByNameAndSemester?key=[{)

Резултатът, който се получава, е следния:

```
{
  "total_rows": 33,
  "offset": 0,
  "rows": [
    {
      "id": "student-0002",
      "key": [{"firstname": "Иван", "family": "Иванов", "lastname": "Иванов"}, 1],
      "value": {"name": "Химия", "grade": 4}
    },
    {
      "id": "student-0002",
      "key": [{"firstname": "Иван", "family": "Иванов", "lastname": "Иванов"}, 1],
      "value": {"name": "Физика I", "grade": 4}
    },
    ...
  ]
}
```

## 5. Получаване на статистика на оценките за зададена специалност

При тази задача за ключ трябва да използваме специалността на студента, а за стойност оценките по отделните дисциплини. Напишете нов view ресурс с име **getBySpecialty**, който да е част от вече създадения дизайн документ “\_design/grade”. Освен необходимата **map** функция, трябва да напишете и **reduce** функция чрез която да се върне статистика за оценките:

```
{
  "_id": "_design/grade",
  "_rev": "1-d861ad1dc20a8df29cbfed9f1a3c062b",
  "language": "javascript",
  "views": {
    "getBySpecialty": {
      "map": "function(doc) {
        doc.semester.forEach(function(semester) {
          semester.subjects.forEach(function(subject) {
            emit(doc.specialty, subject.grade);
          });
        });
      }",
      "reduce": "_stats"
    }
  }
}
```

В IBM Cloudant има вградени reduce функции като **\_sum**, **\_count** и **\_stat**. Използвайте тях винаги, когато е възможно, за да намалите времето необходимо за формиране на отговор. При конкретния пример, стойността на reduce функцията е “\_stats”. Това означава, че ще се върне статистическа информация (сума на оценките, брой оценки, минимална оценка, максимална оценка и сумата от квадрата на оценките). Ако искаме да получим статистика за оценките на всички студенти от катедра КСТ трябва да генерираме следната заявка:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySpecialty?key='KCT'](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySpecialty?key='KCT')

Резултатът, който се получава, е следния:

```
{ "rows": [
  { "key": null, "value": { "sum": 100, "count": 22, "min": 2, "max": 6, "sumsq": 476 } }
]
```

Ако не зададем стойност за параметър **key** ще получим статистика за оценките за студентите от всички специалности:

```
{ "rows": [
  { "key": null, "value": { "sum": 144, "count": 33, "min": 2, "max": 6, "sumsq": 662 } }
]
```

Ако искате да получите статистика на оценките за всяка специалност по отделно чрез една заявка трябва да използвате параметър **group** със стойност **true**:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySpecialty?group=true](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySpecialty?group=true)

Резултатът, който се получава, е следния:

```
{ "rows": [
  { "key": "АИУТ", "value": { "sum": 44, "count": 11, "min": 2, "max": 5, "sumsqr": 186 } },
  { "key": "КСТ", "value": { "sum": 100, "count": 22, "min": 2, "max": 6, "sumsqr": 476 } }
]
```

## 6. Получаване на статистика за оценките за зададен семестър

В този случай трябва да добавим нов view ресурс с име **getBySemester** към дизайн документа с име "grade":

```
{
  "_id": "_design/grade",
  "_rev": "1-d861ad1dc20a8df29cbfed9f1a3c062b",
  "language": "javascript",
  "views": {
    "getBySemester": {
      "map": "function(doc) {
        doc.semester.forEach(function(semester) {
          semester.subjects.forEach(function(subject) {
            emit(semester.number, subject.grade);
          });
        });
      }",
      "reduce": "_stats"
    }
  }
}
```

Ако трябва да получим статистика за оценките за всеки семестър поотделно ще използваме следната заявка:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySemester?group=true](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySemester?group=true)

Резултатът, който се получава, е следния:

```
{ "rows": [
  { "key": 1, "value": { "sum": 73, "count": 18, "min": 2, "max": 6, "sumsqr": 317 } },
  { "key": 2, "value": { "sum": 71, "count": 15, "min": 3, "max": 6, "sumsqr": 345 } }
]
```

Ако интерес представлява конкретен семестър (например 1-ви), задайте номера му като стойност на параметър key:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySemester?key=1](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySemester?key=1)

Ако трябва да получим статистика на оценките за точно определени семестри, трябва да се използва параметър keys. Стойността на параметъра е масив, който в случая трябва да съдържа номерата на семестрите за които желаем статистика на оценките. Трябва да зададете и параметър group със стойност true:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySemester?keys=\[1,3,2\]&group=true](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySemester?keys=[1,3,2]&group=true)

## 7. Получаване на статистика на оценките за зададена дисциплина

В този случай трябва да добавим нова view ресурс с име **getBySubject** към дизайн документа с име "grade". Ключът в този пример трябва да бъде името на дисциплина, а стойността – оценката на студента:

```
{
  "_id": "_design/grade",
  "_rev": "1-d861ad1dc20a8df29cbfed9f1a3c062b",
  "language": "javascript",
  "views": {
    "getBySubject": {
      "map": "function(doc) {
        doc.semester.forEach(function(semester) {
          semester.subjects.forEach(function(subject) {
            emit(subject.name, subject.grade);
          });
        });
      }",
      "reduce": "_stats"
    }
  }
}
```

Ако трябва да получите статистика за оценките за конкретна дисциплина, например "Висша математика I", използвайте следния синтаксис на заявката:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySubject?key=Висша математика I](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySubject?key=Висша%20математика%20I)

Резултатът, който се получава, е следния:

```
{ "rows": [
  { "key": null, "value": { "sum": 11, "count": 3, "min": 2, "max": 6, "sumsqr": 49 } }
]
```

Използвайте параметър group със стойност true, за да получите статистика за всички дисциплини:

[https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/\\_design/grade/\\_view/getBySubject?group=true](https://dc3b328b-545c-4c02-9457-531fdc110a3e-bluemix.cloudant.com/students2/_design/grade/_view/getBySubject?group=true)

Резултатът, който се получава, е следния:

```
{ "rows": [
  { "key": "Висша математика I", "value": { "sum": 11, "count": 3, "min": 2, "max": 6, "sumsqr": 49 } },
  { "key": "Висша математика II", "value": { "sum": 12, "count": 3, "min": 3, "max": 5, "sumsqr": 50 } },
  { "key": "Електроматериалознание", "value": { "sum": 16, "count": 3, "min": 5, "max": 6, "sumsqr": 86 } },
  { "key": "Инженерна графика", "value": { "sum": 15, "count": 3, "min": 5, "max": 5, "sumsqr": 75 } },
  { "key": "Програмиране и използване на компютри", "value": { "sum": 9, "count": 3, "min": 2, "max": 4, "sumsqr": 29 } },
  { "key": "Теоретична електротехника I", "value": { "sum": 13, "count": 3, "min": 4, "max": 5, "sumsqr": 57 } },
  { "key": "Физика I", "value": { "sum": 13, "count": 3, "min": 4, "max": 5, "sumsqr": 57 } },
  ...
]
```

**Задача 2:** Разпечатайте имената на всички студенти, които имат Добър(4) по дисциплината „Физика I“. Използвайте Mango заявка.

Трябва да проверим стойностите на полета “name” и “grade”. Тъй като те са в масив “subjects”, а той от своя страна в масив “semester” трябва да използваме два пъти оператор “\$elemMatch”.

Можете да симулирате Mango заявки от ниво програмен интерфейс на IBM Cloudant. Изберете желаната база данни (“students2”), а от менюто - “Query”. Въведете желания Mango код в поле “Cloudant Query” (виж Фиг. 4).

```
{
  "selector": {
    "semester": {
      "$elemMatch": {
        "subjects": {
          "$elemMatch": {
            "$and": [
              {
                "grade": {
                  "$eq": 4
                }
              },
              {
                "name": "Физика I"
              }
            ]
          }
        }
      }
    }
  },
  "fields": [
    "name"
  ]
}
```

Фиг. 4 Симулиране на Mango заявки чрез интерфейса на IBM Cloudant

Натиснете бутон <Run query>, за да видите резултата. Студентите, които имат оценка Добър(4) по дисциплината “Физика I” са двама (виж Фиг. 5).

```
{
  "name": {
    "firstname": "Стоян",
    "family": "Иванов",
    "lastname": "Стоянов"
  }
}

{
  "name": {
    "firstname": "Иван",
    "family": "Иванов",
    "lastname": "Иванов"
  }
}
```

Фиг. 5 Резултат, получен след изпълнение на кода от Фиг. 4

**Задача 3:** Разпечатайте имената на всички студенти, които имат поне една оценка Отличен(6). Използвайте Mango заявка. За колко време се изпълнява заявката?

Тъй като оценките се намират в масив “subjects”, а той от своя страна в масив “semester” ще използваме два пъти оператор “\$elemMatch” (виж Фиг. 6). За да получим автоматично статистика за изпълнението на заявката ще използваме поле “execution\_stats” със стойност true.

```
{
  "selector": {
    "semester": {
      $elemMatch: {
        "subjects": {
          $elemMatch: {
            "grade": {"$eq": 6}
          }
        }
      }
    }
  },
  "fields": ["name"],
  "execution_stats": true
}
```

Фиг. 6 Mango заявка – Задача 3

Резултатът, който се получава, е следния:

```
. { "firstname": "Иван", "family": "Иванов", "lastname": "Иванов" }
```

```
. { "firstname": "Надя", "family": "Иванова", "lastname": "Петкова" }
```

Времето за изпълнение на заявката е 2.028 ms.