



Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И
КОМПЈУТЕРСКО ИНЖЕНЕРСТВО



Проект по предметот:
Неструктурирани бази на податоци
Тема:
Document-Based

Ментор:
Проф. д-р Слободан Калајциски

Изработила:
Јана Иваноска 183009
Моника Веловска 183168
Катерина Станковска 151201

7 февруари 2024, Скопје

Вовед во document-based

NoSQL е пристап за управување со базата на податоци што може да прими широк спектар на модели на податоци, вклучувајќи key-value, document-based, column-family и graph formats. NoSQL базата на податоци генерално значи дека е не-релациона, дистрибуирана, флексибилна и скалабилна. Дополнителни заеднички карактеристики на базата на податоци NoSQL вклучуваат недостаток на шема на база на податоци, групирање податоци, поддршка за репликација и евентуална конзистентност, за разлика од типичната ACID (атомичност, конзистентност, изолација и издржливост) трансакциска конзистентност на релационите и SQL базите на податоци. Многу системи за бази на податоци NoSQL се исто така со отворен код.

Ваквите бази на податоци се особено интересни во современите апликации каде што податоците се разновидни и нередовно структурирани. Еден од пристапите за управување со овој вид на податоци е document-based моделот, кој се фокусира на зачувување на информации во документи кои се слични на JSON или XML формат. Обезбедува флексибилен и ефикасен начин за складирање и управување со полуструктурирани и неструктурирани податоци, овозможувајќи им на програмерите да работат со сложени структури на податоци и да го развиваат нивниот модел на податоци со текот на времето без ограничувања на однапред дефинирана шема.

Во контекстот на неструктурирани бази на податоци, document-based моделот претставува начин за организација на податоците каде што секоја информација се зачувува во документи кои се самостојни единици на податоци. Секој документ може да содржи повеќе полиња, вклучувајќи скаларни вредности, низи и поддокументи, што го олеснува претставувањето сложени, хиерархиски или неструктурирани податоци.

Податоците се организирани како документи

- XML, JSON, BSON, ..
- хиерархиски, самоопишувачки структури од: мапи, колекции, скаларни вредности
- документот се чува во value делот од key-value БП

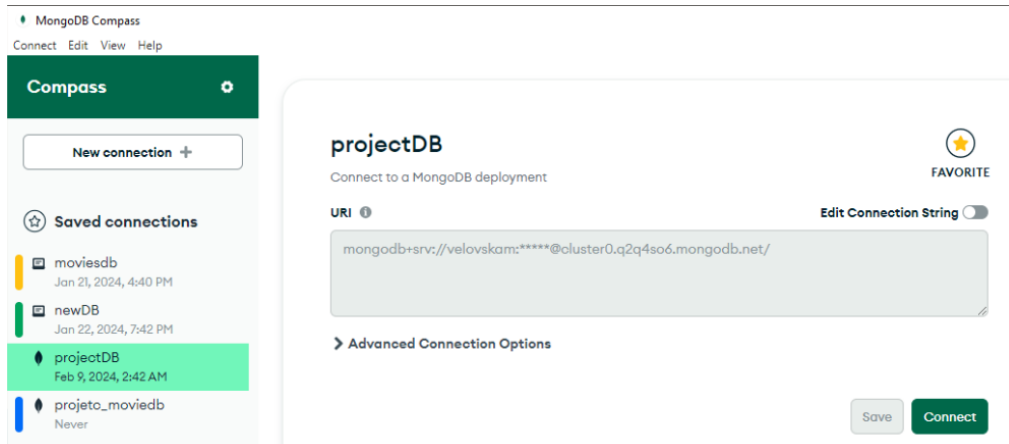
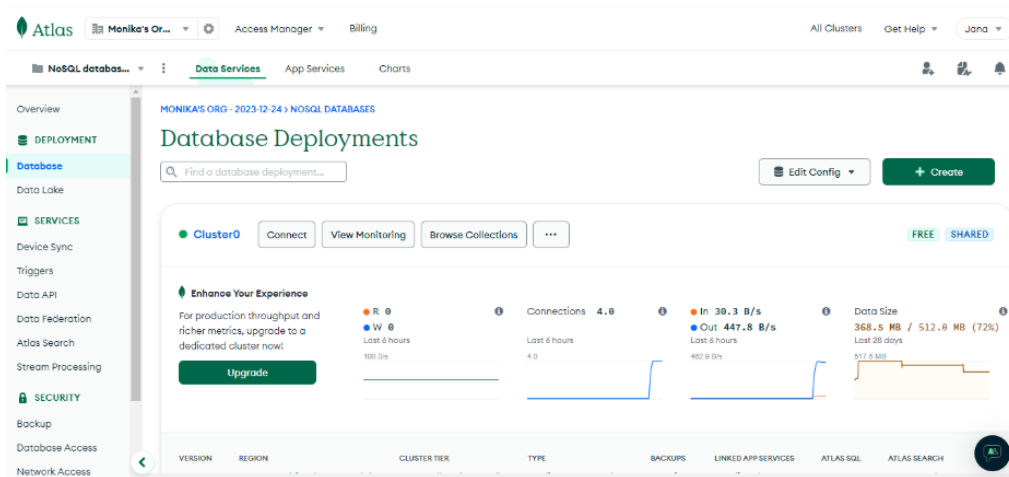
Документите се слични меѓусебе

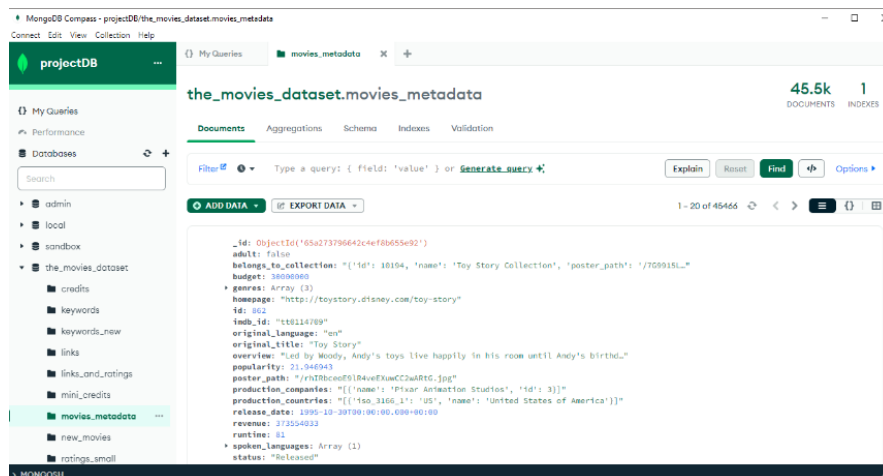
- секој документ може да содржи произволни атрибути
- шемата се разликува од документ до документ
- внесување нови атрибути без претходно дефинирање и менување на постоечките документи
- вгнездувањето на под-документи во документи нуди подобри перформанси

2. MongoDB

MongoDB е open-source NoSQL database систем кој користи документ ориентиран дата модел. Тој е еден од најпопуларните NoSQL системи за бази на податоци, познат по своите високи перформанси, приспособливост и флексибилност. Како база на податоци за документи, MongoDB им олеснува на програмерите да складираат структурирани или неструктурирани податоци. Користи формат сличен на JSON за складирање документи.

- Cloud базирана дата платформа
- Флексибилна шема
- Едноставен дизајн и кориснички интерфејс
- Лесно хоризонтално скалирање со sharding
- Лесна инсталација
- Техничка поддршка и документација

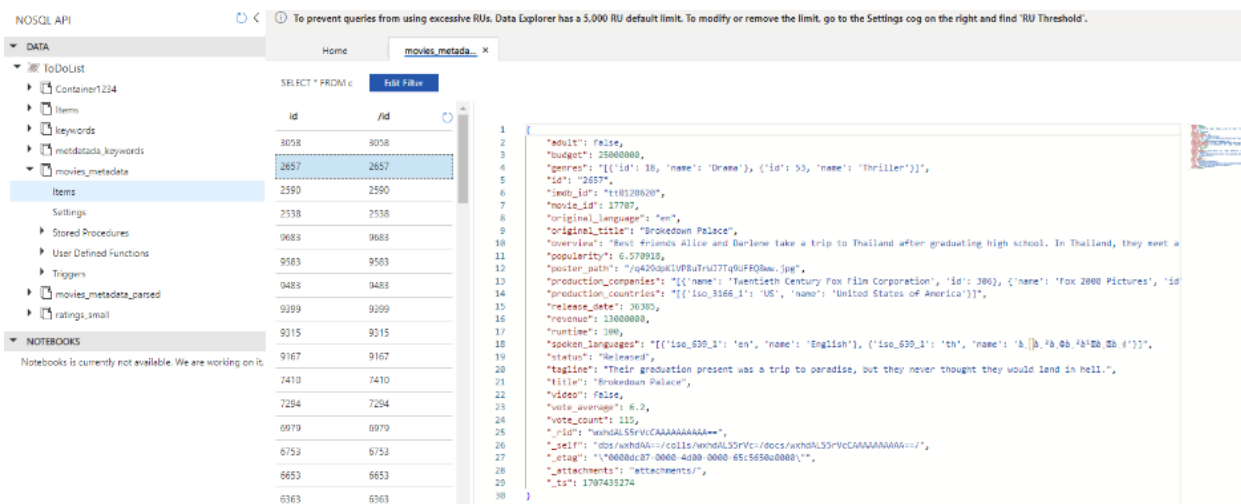




3. AZURE Cosmos DB

Azure Cosmos DB е целосно управувана, глобално дистрибуиран сервис за NoSQL и релациона база на податоци обезбедена од Microsoft Azure, дизајнирана за модерни апликации во обем на интернет. Тој нуди флексибилни модели на податоци, автоматска приспособливост и високи перформанси, што го прави погоден за широк опсег на сценарија за примена.

- Platform-as-a-service (PaaS)
- Real-time пристап со брза латенција за читање и запишување глобално
- независност и еластично скалирање низ Azure region



EXPLORER Version 1.32.1 of Storage Explorer is available. [Install Now](#) [Install on Close](#) [View Release Notes](#) [Close](#)

Search for resources

Collapse All Refresh All

Quick Access

- Emulator & Attached
- Storage Accounts
 - Data Lake Storage Gen1 (deprecated)
 - Azure for Students (katerina.stankovska@stud...)
 - Storage Accounts
 - nosqlstorage
 - Blob Containers
 - \$logs
 - merged
 - raw
 - File Shares
 - Queues
 - Tables
 - Disks
 - nosql-project
 - Data Lake Storage Gen1 (deprecated)

Active blobs (default) raw

Name	Access Tier	Access Tier Last Modified	Last Modified	Blob Type	Content Type	Size	Status
new_merged_metadata_keywords.json	Hot (inferred)		2/9/2024 3:17 AM	Block Blob	application/octet-stream	56.53 MiB	Active
new_parsed_movies_metadata.json	Hot (inferred)		2/9/2024 2:14 AM	Block Blob	application/octet-stream	48.51 MiB	Active
movies_metadata_mongodb.json	Hot (inferred)		2/9/2024 1:25 AM	Block Blob	application/json	56.41 MiB	Active
keywords.json	Hot (inferred)		2/8/2024 11:39 PM	Block Blob	application/json	8.54 MiB	Active
movies_metadata.json	Hot (inferred)		2/8/2024 11:36 PM	Block Blob	application/json	46.13 MiB	Active
ratings_small.json	Hot (inferred)		2/8/2024 4:17 PM	Block Blob	application/json	10.57 MiB	Active
the_movies_dataset.movies_metadata.json	Hot (inferred)		2/7/2024 9:18 PM	Block Blob	application/json	53.22 MiB	Active
small_metadata.json	Hot (inferred)		2/7/2024 9:06 PM	Block Blob	application/json	11.75 KiB	Active

Copy data

Copy data1

General Source Sink Mapping Settings User properties

Source dataset: metadata [Open](#) [New](#) [Preview data](#) [Learn more](#)

File path type: ☒ File path in dataset ☐ Prefix ☐ Wildcard file path ☐ List of files

Filter by last modified: Start time (UTC) End time (UTC)

Recursively: ☒

Enable partitions discovery: ☐

Max concurrent connections:

Additional columns: + New



4. Информации за податочното множество

Линк до податочно множество: https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?resource=download&select=movies_metadata.csv

Овие датотеки содржат метаподатоци за сите 45.000 филмови наведени во Full MovieLens Dataset. Базата на податоци се состои од филмови објавени на или пред јули 2017 година. Податоците вклучуваат актерска екипа, екипа, клучни зборови за заплетот, буџет, приходи, постери, датуми на објавување, јазици, продукциски компании, земји, број на гласови на TMDB и просечни гласови.

Базата на податоци се состои од следниве датотеки:

movies_metadata.csv: Главната датотека со метаподатоци за филмови. Содржи информации за 45.000 филмови вклучени во комплетната база на податоци на MovieLens. Карактеристиките вклучуваат постери, заднини, буџет, приходи, датуми на издавање, јазици, земји за производство и компании.

keywords.csv: Ги содржи клучните зборови за заплетот на филмот за нашите филмови на MovieLens. Достапно во форма на стрингиран JSON објект.

credits.csv: Се состои од информации за актерската екипа и екипата за сите наши филмови. Достапно во форма на стрингиран JSON објект.

links.csv: Датотеката што ги содржи идентификаторите на TMDB и IMDb на сите филмови вклучени во комплетната база на податоци на MovieLens.

links_small.csv: Ги содржи TMDB и IMDb ID на мала подгрупа од 9.000 филмови од Целосната група на податоци.

ratings_small.csv: Подгрупа од 100.000 оценки од 700 корисници на 9.000 филмови.

ГРУПА I:

1. Кои се 5-те најдолги филмови ?

MongoDB:

```
var longestMovies = db.movies_metadata.find().sort({ "runtime": -1 }).limit(5);
print("Најдолги филмови:");
longestMovies.forEach(function (doc) {
  print("Наслов: " + doc.title + ", Времетраење: " + doc.runtime);
});
```

```

< Најдолги филмови:
< Наслов: Centennial, Времетраење: 1256
< Наслов: Jazz, Времетраење: 1140
< Наслов: Baseball, Времетраење: 1140
< Наслов: Berlin Alexanderplatz, Времетраење: 931
< Наслов: Heimat: A Chronicle of Germany, Времетраење: 925
< Извршувањето на барањето траеше 186 милисекунди.

```

Cosmos DB:

```

SELECT TOP 5 c.title AS title, c.runtime AS runtime
FROM movies_metadata c
ORDER BY c.runtime DESC

```

movies_metada...

Query 5 x

```

1 SELECT TOP 5 c.title AS title, c.runtime AS runtime
2 FROM movies_metadata c
3 ORDER BY c.runtime DESC

```

Results Query Stats

Output document size

642 bytes

Index hit document count

5

Index lookup time

0.26 ms

Document load time

0.05 ms

Query engine execution time

0.02 ms

System function execution time

0 ms

↓ Per-partition query metrics (CSV)

Results Query Stats

1 - 5

```

[
  {
    "title": "Centennial",
    "runtime": 1256
  },
  {
    "title": "Baseball",
    "runtime": 1140
  },
  {
    "title": "Jazz",
    "runtime": 1140
  }
]

```

2. Колку филмови има во секоја година ?

MongoDB:

```

var result = db.movies_metadata.aggregate([
  {
    $match: {
      $and: [
        { release_date: { $type: "date" } },
        { release_date: { $ne: null } }
      ]
    }
  },
  {

```

```

    $group: {
      _id: { $year: "$release_date" },
      count: { $sum: 1 }
    }
  },
  {
    $sort: { _id: 1 }
  },
  {
    $project: {
      _id: 0, // Exclude the _id field
      Year: "$_id", // Rename _id to Year
      Count: "$count"
    }
  }
];
result.forEach(function(doc) {
  printjson(doc);
});

```

```

< { Year: 2014, Count: 1974 }
< { Year: 2015, Count: 1905 }
< { Year: 2016, Count: 1604 }
< { Year: 2017, Count: 532 }
< { Year: 2018, Count: 5 }
< { Year: 2020, Count: 1 }
< Извршувањето на барањето траеше 24765 милисекунди.

```

Cosmos DB:

```

SELECT
SUBSTRING(c.release_date, 0, 4),
COUNT(1)
FROM movies_metadata c
GROUP BY SUBSTRING(c.release_date, 0, 4)

```



```

1 SELECT
2 SUBSTRING(c.release_date, 0, 4), -- Extract the year portion of the date string
3 COUNT(1)
4 FROM movies_metadata c
5 GROUP BY SUBSTRING(c.release_date, 0, 4)

```

Results Query Stats

① Document load time	0 ms
① Query engine execution time	8.99 ms
① System function execution time	0.48000000000000004 ms
① User defined function execution time	0 ms
① Document write time	0.01 ms
① Round Trips	1

```

1 SELECT
2 SUBSTRING(c.release_date, 0, 4), -- Extract the
3 COUNT(1)
4 FROM movies_metadata c
5 GROUP BY SUBSTRING(c.release_date, 0, 4)

```

Results Query Stats

1 - 16

[
{
"\$1": "1899",
"\$2": 5
},
{
"\$1": "1898",
"\$2": 13
},
{
"\$1": "1897",
"\$2": 8
},
]

3. Каков статус имаат филмовите и колку филмови припаѓаат во секој од тие статуси ? MongoDB:

```
var startTime = new Date();
```

```

var pipeline = [
  {
    $group: {
      _id: "$status",
      count: { $sum: 1 }
    }
  },
  {
    $replaceRoot: {
      newRoot: {
        Status: "$_id",
        Count: "$count"
      }
    }
  }
];

```

```
var explainResult = db.movies_metadata.aggregate(pipeline, { explain: true });
```

```
var result = db.movies_metadata.aggregate(pipeline);
```

```
print("Статус на филмови и број на филмови по статус:");
```

```

result.forEach(function (doc) {
  print("Статус: " + doc.Status + ", Број на филмови: " + doc.Count);
}

```

```
});
```

```
var executionTime = new Date() - startTime;  
print("Извршувањето на барањето траеше: " + executionTime + " милисекунди.");
```

```
< Статус на филмови и број на филмови по статус:  
< Статус: Released, Број на филмови: 45014  
< Статус: In Production, Број на филмови: 20  
< Статус: Rumored, Број на филмови: 230  
< Статус: Planned, Број на филмови: 15  
< Статус: Post Production, Број на филмови: 98  
< Статус: Canceled, Број на филмови: 2  
< Статус: null, Број на филмови: 87  
< Извршувањето на барањето траеше: 332 милисекунди.
```

Cosmos DB:

```
SELECT c.status as status,  
COUNT(1) as count  
FROM movies_metadata c  
GROUP BY c.status
```

```
1 SELECT c.status as status,  
2 COUNT(1) as count  
3 FROM movies_metadata c  
4 GROUP BY c.status
```

Results Query Stats

Index the document count	0 ms
Index lookup time	0.33 ms
Document load time	0 ms
Query engine execution time	0.060000000000000005 ms
System function execution time	0 ms
User defined function execution time	0 ms

```
1 SELECT c.status as status,  
2 COUNT(1) as count  
3 FROM movies_metadata c  
4 GROUP BY c.status
```

Results Query Stats

1 - 7

```
{  
  "status": "Rumored",  
  "count": 230  
},  
{  
  "status": "Released",  
  "count": 45014  
},  
{  
  "status": "Post Production",  
  "count": 98  
}
```

4. Колку филмови има во секој жанр ?

```
var pipeline = [
  { $unwind: "$genres" },
  {
    $group: {
      _id: "$genres.name",
      count: { $sum: 1 }
    }
  },
  { $sort: { count: -1 } },
  {
    $replaceRoot: {
      newRoot: {
        Genre: "$_id",
        Count: "$count"
      }
    }
  }
];

var explainResult = db.movies_metadata.aggregate(pipeline, { explain: true });

var result = db.movies_metadata.aggregate(pipeline);

result.forEach(function (doc) {
  print("Genre: " + doc.Genre + ", Count: " + doc.Count);
});
```

```
< Genre: Carousel Productions, Count: 1
< Genre: BROSTA TV, Count: 1
< Genre: Pulser Productions, Count: 1
< Genre: GoHands, Count: 1
< Genre: The Cartel, Count: 1
< Genre: Sentai Filmworks, Count: 1
< Извршувањето на барањето траеше: 210 милисекунди.
```

ГРУПА II:

1. Кои се филмовите со најмногу гласање ?

MongoDB:

```
var pipeline = [
  {
    $sort: { "vote_count": -1 }
  }
```

```

    },
    {
      $limit: 5
    },
    {
      $project: {
        _id: 0,
        movie: "$title",
        vote_count: 1
      }
    }
  ]
};

```

```

var explainResult = db.movies_metadata.aggregate(pipeline, { explain: true });
var executionTime = new Date() - startTime;

```

```

var result = db.movies_metadata.aggregate(pipeline);

```

```

print("Филмови со најмногу гласови:");
result.forEach(function (doc) {
  print("Movie: " + doc.movie + ", Vote Count: " + doc.vote_count);
});

```

```

< Најпопуларни филмови:
< Movie: Inception, Vote Count: 14075
< Movie: The Dark Knight, Vote Count: 12269
< Movie: Avatar, Vote Count: 12114
< Movie: The Avengers, Vote Count: 12000
< Movie: Deadpool, Vote Count: 11444
< Извршувањето на барањето траеше: 108 милисекунди.

```

Cosmos DB:

```

SELECT TOP 5 c.title AS movie, c.vote_count
FROM movies_metadata as c
ORDER BY c.vote_count DESC

```

```

1 SELECT TOP 5 c.title AS movie, c.vote_count
2 FROM movies_metadata as c
3 ORDER BY c.vote_count DESC
4

```

Results		Query Stats	
Results	Query Stats		
1 - 5		①	Index lookup time 0.16 ms
		①	Document load time 3.39 ms
		①	Query engine execution time 0.04 ms
		①	System function execution time 0 ms
		①	User defined function execution time 0 ms
		①	...

	{
	"movie": "Inception",
	"vote_count": 14075
	},
	{
	"movie": "The Dark Knight",
	"vote_count": 12269
	},
	{
	"movie": "Avatar",
	"vote_count": 12114

2. Кои се најдобрите 10 филмови по популарност и профит ? Проверува дали насловот на секој филм во листата со врвна популарност постои во листата со врвни профити.

MongoDB:

```

var topPopularityMovies = db.movies_metadata.find({ popularity: { $type: "number" } }).sort({
popularity: -1 }).limit(10).toArray();

```

```

var topProfitMovies = db.movies_metadata.aggregate([
{
  $match: {
    budget: { $exists: true, $ne: null },
    revenue: { $exists: true, $ne: null }
  }
},
{
  $addFields: {
    profit: { $subtract: ["$revenue", "$budget"] }
  }
},
{
  $sort: {
    profit: -1
  }
},
{
  $limit: 10
}
]).toArray();

```

```
var commonMovies = topPopularityMovies.filter(popularityMovie =>
  topProfitMovies.some(profitMovie => profitMovie.title === popularityMovie.title)
);
printjson(commonMovies);
```

```
release_date: 2009-12-10T00:00:00.000Z,
revenue: Long('2787965087'),
runtime: 162,
spoken_languages: [ [Object], [Object] ],
status: 'Released',
tagline: 'Enter the World of Pandora.',
title: 'Avatar',
video: false,
vote_average: 7.2,
vote_count: 12114
}
]
< Извршувањето на барањето траеше: 454 милисекунди.
```

COSMOS DB:

```
SELECT c.title, c.popularity, (c.revenue-c.budget) as profit
FROM c
ORDER BY c.popularity desc
```

```
SELECT c.title,c.popularity, (c.revenue-c.budget) as profit
FROM c
order by c.popularity desc
```

Results

Query Stats

- 100 | Load more

```
[
  {
    "title": "Minions",
    "popularity": 547.488298,
    "profit": 1082730962
  },
  {
    "title": "Wonder Woman",
    "popularity": 294.337037,
    "profit": 671580447
  },
  {

```

Results

Query Stats

①

Index lookup time

0.48000000000000004 ms

①

Document load time

1.74 ms

①

Query engine execution time

0.51 ms

①

System function execution time

0 ms

①

User defined function execution time

0 ms

①

Document write time

0.5 ms

①

Round Trips

1

3. Кои се филмовите со просечни гласови над 9 издадени во последните 5 години ?

MongoDB:

```
var highlyRatedMoviesLast5Years = db.movies_metadata.aggregate([
  {
    $match: {
      vote_average: { $gt: 9 },
      release_date: { $gte: new Date("2017-01-01") }
    }
  },
  {
    $group: {
      _id: null,
      totalMovies: { $sum: 1 },
      titles: { $push: "$title" }
    }
  },
  {
    $project: {
      _id: 0,
      totalMovies: 1,
      titles: 1
    }
  }
]);
print("Број на филмови со рејтинг над 9 издадени во последните 5 години и нивните наслови:");
highlyRatedMoviesLast5Years.forEach(function (doc) {
  printjson(doc);
});
```

```
< Број на филмови со рејтинг над 9 издадени во последните 5 години и нивните наслови:
< {
  totalMovies: 7,
  titles: [
    'Kizumonogatari Part 3: Reiketsu',
    'Zombie Pizza',
    'The Reagan Show',
    'Long Strange Trip',
    'Tokyo Ghoul',
    'First Round Down',
    'LEGO DC Super Hero Girls: Brain Drain'
  ]
}
< Извршувањето на барањето траеше 578 милисекунди.
```

4. Кој е најпопуларниот жанр ?

MongoDB:

```
var pipeline = [
  { $unwind: "$genres" },
  {
    $group: {
      _id: "$genres.name",
      count: { $sum: 1 }
    }
  },
  { $sort: { count: -1 } },
  { $limit: 1 },
  {
    $project: {
      _id: 0,
      result: {
        $concat: [
          "The most popular genre is ",
          "$_id",
          ". "
        ]
      }
    }
  }
];
```

```
var explainResult = db.movies_metadata.aggregate(pipeline, { explain: true });
```

```
var result = db.movies_metadata.aggregate(pipeline);
```

```
result.forEach(function (doc) {
  print(doc.result);
});
```

```
< The most popular genre is Drama.
< Извршувањето на барањето траеше: 192 милисекунди.
```

ГРУПА III:

1. Кои се директорите на најпопуларниот филм ?

MongoDB:

```
var theMostPopularMovies = db.movies_metadata.find({ popularity: { $type: "number" } }).sort({
  popularity: -1 }).limit(1).toArray();
var theMostPopularMovieIds = theMostPopularMovies.map(movie => movie.id);
var creditsForTheMostPopularMovies = db.credits.find({ 'id': { $in: theMostPopularMovieIds }
}).toArray();
```



```

var extractedDirectors = [];

theMostPopularMovies.forEach(movie => {
  var correspondingCredit = creditsForTheMostPopularMovies.find(credit => credit.id ===
movie.id);

  if (correspondingCredit && correspondingCredit.crew) {
    try {
      var cleanedCrew = correspondingCredit.crew.replace(/None/g, 'null');

      var crewArray = JSON.parse(cleanedCrew.replace(/"/g, ''));

      var directors = crewArray.filter(person => person.job === 'Director');

      extractedDirectors = directors.map(director => director.name);

      print(`The most popular movie is: ${movie.title}`);
      print(`Directors: ${extractedDirectors.join(', ')}`);
    } catch (error) {
      console.error('Error parsing JSON in cleaned crew field:', error.message);
    }
  }
});

```

```

< The most popular movie is: Minions
< Directors: Kyle Balda, Pierre Coffin
< Извршувањето на барањето траеше 2034 милисекунди.

```

2. Кој е главниот актер на најпопуларниот филм ?

MongoDB:

```

var theMostPopularMovies = db.movies_metadata.find({ popularity: { $type: "number" } }).sort({
popularity: -1 }).limit(1).toArray();
var theMostPopularMovieIds = theMostPopularMovies.map(movie => movie.id);
var creditsForTheMostPopularMovies = db.credits.find({ 'id': { $in: theMostPopularMovieIds }
}).toArray();

```

```

theMostPopularMovies.forEach(movie => {
  var correspondingCredit = creditsForTheMostPopularMovies.find(credit => credit.id ===
movie.id);

```

```

  if (correspondingCredit && correspondingCredit.cast) {
    try {
      var cleanedCast = correspondingCredit.cast.replace(/None/g, 'null');
      var castArray = JSON.parse(cleanedCast.replace(/"/g, ''));
      var firstNameOfFirstCast = castArray[0] ? castArray[0].name : "";

```

```

    print(`The most popular movie is: ${movie.title}`);
    print(`First name of the cast: ${firstNameOfFirstCast}`);
  } catch (error) {
    console.error('Error parsing JSON in cleaned cast field:', error.message);
  }
}
});

```

```

< The most popular movie is: Minions
< First name of the cast: Sandra Bullock
< Извршувањето на барањето траеше 454 милисекунди.

```

3. Кои се филмовите со рејтинг 5 ?

MongoDB:

```

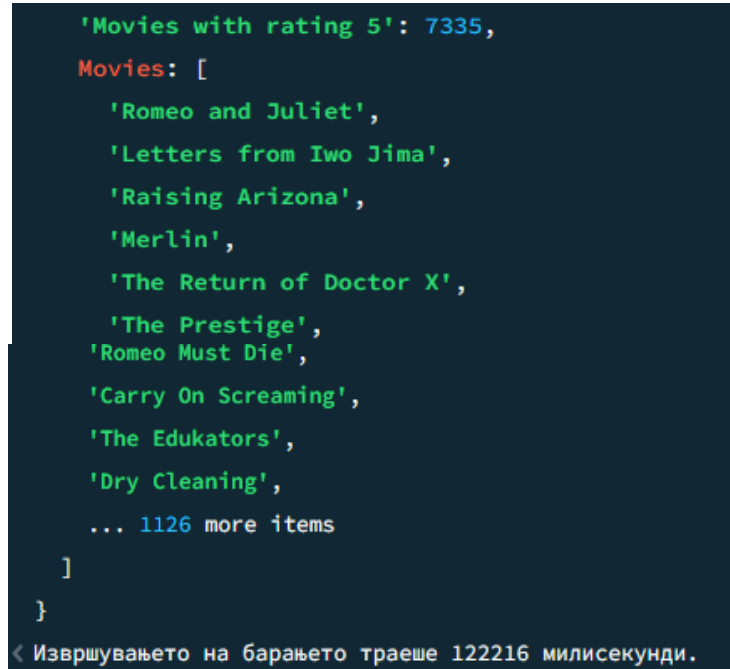
var result = db.ratings_small.aggregate([
  {
    $match: { rating: 5 }
  },
  {
    $group: {
      _id: "$movieId",
      count: { $sum: 1 }
    }
  },
  {
    $lookup: {
      from: "movies_metadata",
      localField: "_id",
      foreignField: "id",
      as: "movie_info"
    }
  },
  {
    $unwind: "$movie_info"
  },
  {
    $project: {
      _id: 0,
      title: "$movie_info.title",
      count: 1
    }
  }
]).toArray();

```

```

var movieTitles = result.map(item => item.title);
var totalCount = result.reduce((acc, item) => acc + item.count, 0);
printjson({
  ["Movies with rating 5"]: totalCount,
  ["Movies"]: movieTitles
});

```



```

'Movies with rating 5': 7335,
Movies: [
  'Romeo and Juliet',
  'Letters from Iwo Jima',
  'Raising Arizona',
  'Merlin',
  'The Return of Doctor X',
  'The Prestige',
  'Romeo Must Die',
  'Carry On Screaming',
  'The Edukators',
  'Dry Cleaning',
  ... 1126 more items
]
}
< Извршувањето на барањето траеше 122216 милисекунди.

```

Дополнителни query за трансформација :

- Парсирање на стринг во низа за spoken_languages атрибутот

```

db.small_test_metadata.find({}).forEach(function(doc) {
  try {
    print("Processing document with _id: " + doc._id);
    printjson(doc);
    var cleanedLanguages = doc.spoken_languages.replace(/"/g, "");
    var parsedLanguages = JSON.parse(cleanedLanguages);

    if (Array.isArray(parsedLanguages)) {
      db.small_test_metadata.update(
        { _id: doc._id },
        { $set: { spoken_languages: parsedLanguages } }
      );
      print("Updated spoken_languages for document with _id: " + doc._id);
      printjson(parsedLanguages);
    } else {
      print("Skipping document with _id: " + doc._id + " - 'spoken_languages' is not an array.");
    }
  }
});

```

```

    } catch (e) {
      print("Error parsing 'spoken_languages' for document with _id: " + doc._id + " - " + e.message);
    }
  });

```

```

< DeprecationWarning: Collection.update() is deprecated. Use updateOne, updateMany, or bulkWrite.
< Total execution time: 2124679 milliseconds

```

- Парсирање на стринг во низа за keywords атрибути

```

db.keywords.find({}).forEach(function(doc) {
  try {
    print("Processing document with _id: " + doc._id);
    printjson(doc);
    var cleanedKeywords = doc.keywords.replace(/"/g, "");
    var parsedKeywords = JSON.parse(cleanedKeywords);

    if (Array.isArray(parsedKeywords)) {
      db.keywords.update(
        { _id: doc._id },
        { $set: { keywords: parsedKeywords } }
      );
      print("Updated keywords for document with _id: " + doc._id);
      printjson(parsedKeywords);
    } else {
      print("Skipping document with _id: " + doc._id + " - 'keywords' is not an array.");
    }
  } catch (e) {
    print("Error parsing 'keywords' for document with _id: " + doc._id + " - " + e.message);
  }
});

```

```

> print("Total execution time: " + executionTime + " milliseconds");
  print("or: " + (executionTime/60000).toFixed(2) + " minutes");
< Total execution time: 2343079 milliseconds
< or: 39.05 minutes

```

ЗАКЛУЧОК

MongoDB и Cosmos DB се популарни NoSQL бази на податоци, но тие имаат некои разлики во однос на карактеристиките, архитектурата и случаите на употреба. MongoDB обезбедува силна поддршка за ACID трансакции во повеќе документи во една збирка, обезбедувајќи интегритет и конзистентност на податоците, додека ограничената поддршка за трансакции на Azure Cosmos DB за операции со повеќе документи и ограничувања за единечни збирки може да влијае на апликациите со сложени трансакциски барања со повеќе колекции, барајќи внимателно разгледување и алтернативни пристапи за дизајн.

Врз основа на набљудувањето дека прашалниците во Cosmos DB имаат помало време на извршување во споредба со истите прашалници во MongoDB, може да се заклучи дека Cosmos DB може да понуди подобри перформанси во одредени сценарија. Сепак, важно е да се забележи дека перформансите може да варираат во зависност од фактори како што се волуменот на податоците, сложеноста на барањето, стратегиите за индексирање и целокупната конфигурација на базата на податоци. Оваа опсервација сугерира дека архитектурата или техниките за оптимизација на Cosmos DB може да бидат поефикасни во справувањето со одредени типови на прашања во споредба со MongoDB.

Mongo db

- MongoDB обезбедува силна поддршка за ACID трансакции
- Прашалниците во Cosmos DB имаат поголемо време на извршување
- Користи флексибилен модел на податоци базиран на документи, каде што податоците се складираат во документи слични на JSON.
- Користи свој јазик за пребарување кој е сличен на JSON.

Cosmos db

- Ограничената поддршка за трансакции на Azure Cosmos DB за операции со повеќе документи и ограничувања за единечни збирки може да влијае на апликациите со сложени трансакциски барања со повеќе колекции, барајќи внимателно разгледување и алтернативни пристапи за дизајн.
- Прашалниците во Cosmos DB имаат помало време на извршување
- Поддржува повеќе модели на податоци
- Поддржува пребарување слично на SQL преку неговиот SQL API.

