Tehtävä 6: NoSQL-tietokannat

MongoDB on suosittu NoSQL-tietokanta, joka ei perustu relaatiomalliin ja SQL-kieleen kuten useimmat perinteiset tietokannat. Etsi tietoa MongoDB:stä netistä ja vastaa sitten seuraaviin tehtäviin.

MongoDB vs. SQL-tietokanta

Miten MongoDB eroaa SQL-tietokannasta? Mitä hyvää ja huonoa siinä on?

SQL VS MONGOBD
- SQL databases are used to store structured data (tables with fixed rows and columns) while NoSQL databases like MongoDB are used to save unstructured data
- MongoDB is used to save unstructured data in JSON format (document: JSON documents, key-value: key-value pairs, wide-column: tables with rows and dynamic columns, graph: nodes and edges)
- MongoDB does not support advanced analytics and joins like SQL databases support
- SQL databases are table based databases whereas NoSQL databases can be document based, key-value pairs, graph databases.
- There are many good solutions available to support MongoDB analytics, including: data virtualization, translation, the MongoDB connector, and data warehousing with an ETL or ELT process.
- SQL requires specialized DB hardware for better performance while NoSQL uses commodity hardware.
- SQL databases are vertically scalable while NoSQL databases are horizontally scalable.
- SQL has ACID(Atomicity, Consistency, Isolation, and Durability) standard while NoSQL has Base (Basically Available, Soft state, Eventually Consistent) standard.

* MongoDB pros:
- Manageability: user-friendly and easy to use
- Scalability: highly and easily scalable
- Less expensive in maintaining cost, lesser server cost, and open-source
- Flexible tool and dynamic schema
- High performance and fast queries
- Flexibility: add new columns or fields without affecting existing rows or application performance

* MongoDB cons:
- NoSQL does not support ACID
- NoSQL databases can be larger than SQL databases
- There are not many defined standards for NoSQL
- No Stored Procedures

MongoDB:n käyttäminen

Seuraavassa on keskustelu SQLite-tulkin kanssa, joka luo taulun, lisää siihen sisältöä ja hakee tietoa:

```
sqlite> CREATE TABLE Tyontekijat (id INTEGER PRIMARY KEY, nimi TEXT, yritys
TEXT, palkka INTEGER);
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Maija',
'Google', 8000);
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Uolevi',
'Amazon', 5000);
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Kotivalo',
'Google', 7000);
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Kaaleppi',
'Facebook', 6000);
```

```
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Liisa',
'Amazon', 9000);
sqlite> INSERT INTO Tyontekijat (nimi, yritys, palkka) VALUES ('Anna', 'Amazon',
6500);
sqlite> SELECT * FROM Tyontekijat;
id          nimi        yritys      palkka
----------  ----------  ----------  ----------
1           Maija       Google      8000
2           Uolevi      Amazon      5000
3           Kotivalo    Google      7000
4           Kaaleppi    Facebook    6000
5           Liisa       Amazon      9000
6           Anna        Amazon      6500
sqlite> UPDATE Tyontekijat SET palkka=5500 WHERE id=2;
sqlite> SELECT * FROM Tyontekijat;
id          nimi        yritys      palkka
----------  ----------  ----------  ----------
1           Maija       Google      8000
2           Uolevi      Amazon      5500
3           Kotivalo    Google      7000
4           Kaaleppi    Facebook    6000
5           Liisa       Amazon      9000
6           Anna        Amazon      6500
sqlite> SELECT nimi, palkka FROM Tyontekijat WHERE yritys='Amazon';
nimi        palkka
----------  ----------
Uolevi      5500
Liisa       9000
Anna        6500
sqlite> SELECT COUNT(*) FROM Tyontekijat WHERE yritys='Google';
COUNT(*)
----------
2
sqlite> SELECT nimi, yritys FROM Tyontekijat WHERE palkka>6000;
nimi        yritys
----------  ----------
Maija       Google
Kotivalo    Google
Liisa       Amazon
Anna        Amazon
sqlite> SELECT yritys, COUNT(*), MAX(palkka) FROM Tyontekijat GROUP BY yritys;
yritys      COUNT(*)    MAX(palkka)
----------  ----------  -----------
Amazon      3           9000
Facebook    1           6000
Google      2           8000
```

Ota selvää, miten vastaavan voi tehdä MongoDB:ssä ja kirjoita keskustelu MongoDB-tulkin kanssa alla olevaan tekstikenttään.

```
1. SELECT * FROM Tyontekijat;

https://mongoplayground.net/p/4zQpGEX9xil

db.Tyontekijat.find()


2. UPDATE Workers SET salary = 5500 WHERE id = 2;

https://mongoplayground.net/p/oc5IoP8Uofx
```

```
db.Tyontekijat.update({
  _id: 2
},
{
  "$set": {
    "palkka": 5500
  }
})
```

3. SELECT * FROM Tyontekijat;

https://mongoplayground.net/p/ojD9IykhTgm

```
db.Tyontekijat.find()
```

4. SELECT nimi, palkka FROM Tyontekijat WHERE yritys='Amazon';

https://mongoplayground.net/p/jciLIcMGmMi

```
db.Tyontekijat.find({
  "yritys": "Amazon"
},
{
  "_id": 0,
  "nimi": 1,
  "palkka": 1,
  "yritys": 1
})
```

5. SELECT COUNT(*) FROM Tyontekijat WHERE yritys='Google';

https://mongoplayground.net/p/lEf2uMs63T1

```
db.Tyontekijat.aggregate([
  {
    "$match": {
      "yritys": "Google"
    }
  },
  {
    "$count": "Tyontekijat"
  }
])
```

6. SELECT nimi, yritys FROM Tyontekijat WHERE palkka>6000;

https://mongoplayground.net/p/2tqk8CFTFkr

```
db.Tyontekijat.find({
  "palkka": {
    "$gt": 6000
  }
},
{
  "_id": 0,
  "nimi": 1,
  "yritys": 1
```

```
})

7. SELECT yritys, COUNT(*), MAX(palkka) FROM Tyontekijat GROUP BY yritys;

https://mongoplayground.net/p/YCIkF36-eiX

db.Tyontekijat.aggregate([
  {
    $group: {
      _id: "$yritys",
      maxpalkka: {
        $max: "$palkka"
      },
      count: {
        "$sum": 1
      }
    }
  },
  {
    $project: {
      _id: 0,
      count: 1,
      yritys: "$_id",
      maxpalkka: 1
    }
  }
])
```