IOT-LAITTEIDEN DATAN KÄSITTELYN TEHOKKUUS ERI TIETOKANNOILLA

Lappeenrannan–Lahden teknillinen yliopisto LUT

Tietotekniikan kandidaatintyö

2024

Viljami Puro

Ohjaajat: Nuorempi tutkija Majad Qureshi

Apulaisprofessori Antti Knutas

Tarkastaja: Apulaisprofessori Antti Knutas

TIIVISTELMÄ

Lappeenrannan–Lahden teknillinen yliopisto LUT

LUT Teknis-luonnontieteellinen

Tietotekniikka

Viljami Puro

IoT-laitteiden datan käsittelyn tehokkuus eri tietokannoilla

Tietotekniikan kandidaatintyö

2024

27 sivua, 4 kuvaa ja 5 taulukkoa

Tarkastaja: Apulaisprofessori Antti Knutas

Avainsanat: esineiden internet, IoT, tietokannat, kontrolloitu tutkimus, suorituskyky, tehokas tiedonkäsittely, MySQL, PostgreSQL, MongoDB, Apache Cassandra

Nykyajan teknologia pystyy kannattelemaan suuria määriä IoT-laitteita 5G:n takia. Tämän IoT-datan tehokkaaseen tallentamiseen tarvitaan siihen soveltuvia tietokantoja. Sopivien tietokantojen avulla pystytään käsittelemään IoT-data nopeasti ja tehokkaasti.

Tämän kandidaatin työn tavoitteena on tutkia eri tietokantojen soveltuvuutta IoT-datan tehokkaaseen käsittelyyn. Tutkimusmenetelmä tässä työssä on kontrolloitu tutkimus. Työssä suoritetaan suorituskykytestit neljälle suositulle ja laajasti käytössä olevalle tietokannalle, jotka ovat MySQL, PostgreSQL, MongoDB ja Apache Cassandra.

Suorituskykytestejä varten rakennettiin asiakasohjelmisto ja palvelin. Asiakasohjelmisto pystyy lähettämään IoT-laitteiden dataa palvelimelle. Palvelin kykenee vastaanottamaan datan ja tallentamaan se eri tietokantoihin samalla mitaten ajan, mikä kului tietokannan sijoitusoperaatioon.

Työn suorituskykytestien perusteella parhaiten suoriutuivat tietokannat PostgreSQL ja Apache Cassandra. Työn suorituskykytestit eivät kuitenkaan kerro suoraan eri tietokantojen suorituskyvystä tuotantoympäristössä, sillä testit olivat siinä mielessä rajoittuneita, että ne eivät ottaneet huomioon samanaikaisia operaatioita tai mahdollista viivettä, joka syntyy asiakkaan ja palvelimen välisestä viestinnästä. Testien tulokset riippuvat myös käytetystä laitteistosta.

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

School of Engineering Science

Software Engineering

Viljami Puro

Efficiency of IoT data handling with different databases

Bachelor's thesis

2024

27 pages, 4 figures and 5 tables

Examiner: Associate Professor Antti Knutas

Keywords: Internet of Things, IoT, databases, controlled experiment, efficient data handling, MySQL, PostgreSQL, MongoDB, Apache Cassandra

Modern 5G technology can support masses of IoT devices. To be able to handle all the data outputted by the devices efficient databases up for the task are needed. With proper databases the data can be handled fast and efficiently.

The goal for this bachelor's thesis is to examine different databases suitability to handle IoT data efficiently. The research method of this thesis is a controlled experiment. Performance tests are done to four popular and widely used databases which are MySQL, PostgreSQL, MongoDB and Apache Cassandra.

Before the performance tests, client and server software was built. The client can send IoT data over to the server. The server can receive the data and store it to the different databases. The server is smart in that sense that it measures the time taken for every insert operation.

Based on this study's performance tests the best performing databases were PostgreSQL and Apache Cassandra. The performance tests don't tell the full truth especially regarding the databases' performance in a production scenario. This is because the tests conducted are limited and they do not consider asynchronous operations nor the latency that occurs when the client and server communicate. The result of the tests also depends heavily on the hardware used.

LYHENNELUETTELO

Lyhenteet

HTTP	Hypertext markup language
IoT	esineiden internet (Internet of Things)
NoSQL	Not only SQL. Ei-vain-SQL-kyselykieli
SQL	Structured Query Language. Standardoitu kyselykieli

Sisällysluettelo

Tiivistelmä

Abstract

Lyhenneluettelo

1 Johdanto ... 7
2 IoT ja muu tutkimus ... 9
2.1 IoT ... 9
2.1.1 IoT:n perusteet ... 9
2.1.2 IoT:n tärkeys ... 9
2.1.3 IoT:n sovellukset ... 9
2.2 Muu tutkimus ... 10
2.2.1 IoT-laitteet ... 10
2.2.2 MySQL ... 10
2.2.3 PostgreSQL ... 11
2.2.4 MongoDB ... 11
2.2.5 Apache Cassandra ... 11
2.2.6 Relaatiotietokantojen suorituskyky ... 12
2.2.7 NoSQL-tietokantojen suorituskyky ... 12
3 Menetelmäkuvaus ... 13
4 Testiympäristö ja tulokset ... 14
4.1 Testiympäristön kuvaus ... 14
4.2 Käytetyt laitteet ... 14
4.3 Käytetty ohjelmointikieli ... 15
4.4 Tietokannat ja niiden käyttö ... 16
4.5 Testin kulku ... 17
4.6 Tulokset ... 17
5 Keskustelu ... 21
5.1 Tulosten merkitys ... 21
5.2 Tietokannan valitseminen ... 21
5.3 Testien parantaminen ... 22

5.4 Tutkimuskysymykseen vastaaminen 22
5.5 Tulevaisuuden kehitys 23
5.6 Työn rajallisuudet 23
6 Yhteenveto 24
Lähteet 25

1 Johdanto

Nykyajan kehittyvässä maailmassa ei enää voi olla törmäämättä IoT-laitteisiin. Niitä näkyy jopa kauppojen hyllyillä hintalappujen muodossa. Laitteet toimivat myös piilossa erilaisten sensoreiden muodossa mitaten lämpötilaa, kosteutta tai laitteiden sijaintia. IoT-laitteiden eksponentiaalisen kasvun myötä tulee tarve niiden lähettämän datan tehokkaaseen käsittelyyn. Työ käsittelee siis IoT-datan tehokasta käsittelyä eri tietokannoilla.

Työn tavoitteena on selvittää eri tietokantojen vaikutuksen suuruutta IoT-datan käsittelyn tehokkuuteen. Työ tehdään, jotta tulevaisuudessa voidaan perustella eri tietokantojen sopivuutta IoT-datan käsittelyyn ja tallentamiseen. Työn tarkoituksena on erityisesti havainnollistaa eri tietokantojen sopivuutta IoT-ratkaisuihin, jotta voidaan tehdä valistuneita päätöksiä johtaen pienennettyihin kuluihin ja vähentyneeseen energian tarpeeseen.

Eri järjestelmien kokonaisuus testataan suorituskykytesteillä, joilla simuloidaan IoT-kokonaisuuksia ja niiden datan virtaa. Testien jälkeen tulokset arvioidaan ja laaditaan suositukset eri tietokantojen soveltuvuudesta IoT-datan käsittelyyn.

Aiemmat tutkimukset antavat runsaasti tietoa, esimerkiksi siitä kuinka yksittäiset tietokannat pärjäävät teollisuuden IoT kuormituksessa. Samantyyppisten tietokantojenkin (NoSQL) välillä on huomattavia eroja, jotka voivat olla jopa 4850-kertaisia. (Kruger and Hancke, 2014) Myös eri tietokantojen, jotka ovat tehty juuri aikasarjadatan käsittelyyn, välillä voi olla ympäristöstä, datavirrasta ja tietokantakyselystä riippuen moninkertaisia eroja (Liu et al., 2024).

Tämän työn tavoitteena on luoda testiympäristö, jolla pystytään testaamaan eri tietokantojen sopivuutta IoT-datan käsittelyyn. Testien tuloksista on tarkoitus myös laatia kattava yhteenveto ja suositukset eri tietokantojen eroista sekä sopivuudesta tehokkaaseen IoT-datan käsittelyyn. Tutkimuskysymys on seuraava:

- Mitkä tutkituista tietokannoista soveltuvat parhaiten IoT-datan tehokkaaseen käsittelyyn ja kuinka suuret niiden erot ovat?

Tämän työn ensimmäinen luku on johdanto. Toinen luku käsittelee IoT:ta yleisesti ja aikaisempia tutkimuksia, jotka liittyvät jotenkin aiheeseen. Kolmannessa luvussa käsitellään millä testialustalla ja miten suorituskykytestit aiotaan suorittaa. Neljännessä luvussa

esitellään testiympäristö, testien kulku sekä testien tulokset. Viides luku sisältää keskustelua testituloksista. Kuudes luku, joka on myös viimeinen, sisältää työn yhteenvedon.

2 IoT ja muu tutkimus

2.1 IoT

Tämä luku sisältää kaksi alalukua, joista ensimmäinen käsittelee aluksi lyhyesti IoT:n, miksi IoT on tärkeä asia ja sen mahdollisista sovellusaloista. Toinen alaluku käsittää katsauksen muista tutkimuksista ja niiden tuloksista.

2.1.1 IoT:n perusteet

Esineiden internet (engl. Internet of Things, lyh. IoT) on eri kokonaisuuksien, laitteiden tai asioiden yhdistämistä internetiin. Näin saadaan kerättyä suuri määrä dataa, vaikka jonkin laitteen toiminnasta ja tehokkuudesta. Asioiden muuntaminen älykkäiksi sekä niiden datan kerääminen ja sen hyödyntäminen onkin IoT:n perusta (Abdelaziz, 2022).

2.1.2 IoT:n tärkeys

IoT:n merkitys kasvaa koko ajan tiedon tarpeen kanssa sillä se on perimmiltään vain tiedon keräämistä. IoT:n avulla saadaan reaaliaikaista dataa historiallisen datan sijasta. Tämä tarkoittaa, että tehtyjä muutoksia ja niiden vaikutuksia pystytään seuraamaan reaaliajassa. Juuri tiedon reaaliaikaisuus avaa ovet monille mahdollisuuksille, jotka liittyvät yksityisten ihmisien, yritysten ja hallituksien jokapäiväiseen elämään. (Baiyere et al., 2020.)

2.1.3 IoT:n sovellukset

IoT mahdollistaa erilaiset lähestymistavat nykyisiin ratkaisuihin. Sovelluspaikkoja on tuotannossa, kodeissa ja kaupungeissa. Tuotannossa voidaan seurata tuotantolinjaa ja sen koneita reaaliaikaisesti, mikä auttaa kehittämään tuotteiden laatua ja linjan tehokkuutta. Kodeissa IoT mahdollistaa älykkyyden tuomisen taloon, mikä tarkoittaa, että valoja, ilmastointia ja ilmanvaihtoa on mahdollista hallita älykkäästi johtaen energian säästöön ja

lisääntyneeseen asumismukavuuteen. Kaupungeissa voidaan lisätä älyä valaistukseen, liikenteeseen ja jätteiden hallintaan. Jätteiden haku ja vienti voitaisiin toteuttaa vasta silloin, kun saataisiin kokonainen roska-auto täyteen, mikä vähentäisi turhien reissujen määrää ja polttoaineenkulutusta. Valaistus voitaisiin toteuttaa vain sille alueelle, jossa liikkuu sillä hetkellä ihmisiä. Liikenne voitaisiin optimoida älykkäillä liikennevaloilla, jotta pysähdyksien määrä vähenisi. (Abdelaziz, 2022.) Näiden kaikkien toteuttaminen olisi nykytekniikalla mahdollista kaupungeissa ja niiden lähialueilla, sillä yksi 5G:n sovellusaloista on juuri koneiden välinen viestintä (Cox, 2021).

2.2 Muu tutkimus

Tämä alaluku tarkentaa IoT-laitteiden määrittelyä, esittelee työssä käytetyt tietokannat sekä niiden valinnan perustelut. Alaluku sisältää myös muuta kirjallisuutta aiheesta.

2.2.1 IoT-laitteet

IoT-laitteet voidaan jakaa kolmeen eri luokkaan niiden ominaisuuksien perusteella. Alimpaan luokkaan voidaan jaotella IoT-laitteet, joissa on hyvin rajoittunut määrä laskentatehoa, hajasaantimuistia ja vain mahdollisuus hyvin pienitehoiseen viestintään, yleensä pelkän yhdyskäytävän (engl. gateway) kanssa. Keskimmäiseen luokkaan kuuluvat IoT-laitteet omaavat enemmän resursseja verrattuna alimpaan luokkaan ja pystyvät, esimerkiksi ajamaan itse alhaisen tason konenäköalgoritmeja. Ylimpään luokkaan kuuluvat IoT-laitteet omaavat suurimman määrän resursseja kaikista luokista. Runsaan resurssimäärän takia näitä laitteita käytetään yleensä yhdyskäytävinä, jotka lähettävät koottua dataa eteenpäin. Laitteet tukevat myös raskaita koneoppimisen algoritmeja sekä reunalaskentaa (engl. edge computing). (Ojo et al., 2018.)

2.2.2 MySQL

MySQL on ilmainen ja avoimen lähdekoodin tietokannan hallintajärjestelmä (engl. database management system). Se on julkaistu 1990-luvulla ja on yleisesti käytössä yritysmaailmassa

vielä tänäkin päivänä. MySQL on tunnettu luotettavuudestaan ja nopeudestaan.(Bell, 2018; Cabral and Murphy, 2009.)

2.2.3 PostgreSQL

PostgreSQL on MySQL:n tapaan ilmainen ja avoimen lähdekoodin tietokannan hallintajärjestelmä. Sekin on julkaistu 1990-luvulla ja on suora kilpailija MySQL-tietokannalle. Suuria yrityksiä, jotka ovat käyttäneet PostgreSQL-tietokantoja ovat, esimerkiksi Skype ja Instagram. Sillä on myös hyvin aktiivinen jäsenkanta, joka pyrkii kehittämään tietokantaa entistä paremmaksi. PostgreSQL-tietokannalla on hyviä ominaisuuksia, kuten nopeus, saatavuus ja kyky varastoida suuri määrä dataa. PostgreSQL tukee myös erilaisia laajennuksia tehden tietokannasta sopeutuvan eri tarpeisiin. (Ferrari and Pirozzi, 2020; Juba, 2017.)

2.2.4 MongoDB

MongoDB on dokumenttipohjainen NoSQL-tietokanta. MongoDB on kehitetty kahdella pääperiaattella, jotka ovat horisontaalinen skaalautuvuus ja datan dokumenttimainen rakenne, jonka haluttiin muistuttavan tavallista ja tunnettua JSON-rakennetta. MongoDB on suosituin dokumenttipohjainen tietokanta, jota on käyttänyt myös suuret toimijat, kuten eBay. (Haviv, 2016.)

2.2.5 Apache Cassandra

Apache Cassandra on pohjimmiltaan sarakepohjainen NoSQL-tietokanta. Cassandran arkkitehtuuri perustuu pääsolmukohdattomaan rinki arkkitehtuuriin, minkä takia Cassandran saatavuus on vaarantunut vasta silloin, kun monet solmukohdat kaatuvat samanaikaisesti. Eri solmukohdat juttelevat vertaisverkossa (engl. peer-to-peer network, lyh. P2P) gossip-protokollan avulla. Tämän takia Cassandra skaalautuu horisontaalisesti ja kestää virheitä huomattavasti paremmin kuin perinteiset yhden pääsolmukohdan omaavat tietokannat. Cassandran skaalautuvuudesta kertoo myös se, että sen suoritusteho nousee lineaarisesti, kun solmukohtia lisätään klusteriin. Cassandraa on käyttänyt myös palveluissaan suuret yritykset, kuten Apple ja Netflix. (Estrada and Ruiz, 2016; Lakshman and Malik, 2010.)

2.2.6 Relaatiotietokantojen suorituskyky

Yhdessä tutkimuksessa huomattiin, että PostgreSQL oli noin 9 kertaa nopeampi kuin MySQL, kun kyseessä oli valintakysely tarkentavilla parametreilla. Monimutkaisissa testeissä, jossa oli yhdenaikaisia operaatioita, PostgreSQL säilytti suorituskykynsä, joka oli hieman alle 1 ms, kun taas MySQL:n suorituskyky vaihteli hiukan 10 ms molemmin puolin. (Salunke and Ouda, 2024.)

2.2.7 NoSQL-tietokantojen suorituskyky

Monet tutkimukset näyttävät, että MongoDB on moninkertaisesti nopeampi kuin Apache Cassandra etenkin tiedon tallentamisessa ja haussa, mutta Cassandra on huomattavasti nopeampi tietojen päivittämisessä. Erot johtuvat siitä, että tietokannat käyttävät järjestelmän resursseja eri tavalla. Eroja selittää myös se, että MongoDB on dokumenttipohjainen ja Cassandra on sarakepohjainen tietokanta. (Matallah et al., 2020; Seghier and Kazar, 2021.)

3 Menetelmäkuvaus

Tämän työn tutkimuksen kohteena on eri tietokantojen tehokkuus IoT-datan käsittelyssä. Tutkimus suoritetaan kontrolloiduilla suorituskykytesteillä, joten menetelmäksi sopii kontrolloitu tutkimus.

Kontrolloitu tutkimus on tapa tutkia asioita, muuttamalla testissä olevia muuttujia nähdäkseen niiden vaikutuksen lopputulokseen ja mahdollisesti muihin muuttujiin, kuten tässä työssä eri tietokantojen vaikutusta tiedon käsittelyn tehokkuuteen. Näin saadaan tarkasti selville muuttujien suhteet ja vaikutukset toisiinsa. Jotta kontrolloitu tutkimus voidaan tehdä, täytyy ennen tutkimusta olla selvillä selkeä hypoteesi, joka on tässä työssä se kuinka eri tietokannat vaikuttavat IoT-datan tehokkaaseen käsittelyyn. (Easterbrook et al., 2008.)

Kontrolloituja tutkimuksia tehdään, kun halutaan hallita tutkimustilannetta ja sen käyttäytymistä suurissa määrin tarkasti ja systemaattisesti. Tutkimustilanteen käyttäytymistä ohjataan muuttamalla tiettyjen muuttujien arvoja samalla pitäen muut muuttujat samoina, aivan kuten tässä työssä pidetään muu ympäristö samana, mutta vaihdetaan pelkästään tietokanta. Kontrolloituja tutkimuksia voi olla ihmis- ja teknologiapohjaisia. Luonnollisesti ihmispohjaisissa tutkimuksissa on vähemmän kykyä hallita tutkimustilannetta, sillä ihmiset käyttäytyvät samoissa tilanteissa eri tavalla. Tämän työn tutkimus on teknologiapohjainen, sillä eri työkaluja, tietokantoja, käytetään samaan tehtävään. (Wohlin et al., 2012.)

Kontrolloidussa tutkimuksessa valitaan, jokin alkutilanne tai kontrollitapaus, johon voidaan myöhemmin vertailla tuloksia. Tulosten saaminen saadaan riippumattoman muuttujan tai muuttujien arvojen muokkaamisella samalla muun ympäristön pysyessä samana, tässä työssä vain tietokannat muuttuvat. Tuloksia verrataan kontrollitapaukseen ja siitä saadaan tehtyjen muutosten suuruus. Tässä työssä verrataan eri tietokantojen suorituskykyä toisiinsa. (Wohlin et al., 2012.)

4 Testiympäristö ja tulokset

Tämä luku kertoo työssä toteutettujen testien ympäristöstä ja testien tuloksista. Luvun ensimmäiset aliluvut käsittelevät testiympäristössä käytetyt laitteet ja ohjelmistot. Testiympäristön esittelyn jälkeiset alaluvut esittelevät testien tulokset eri tietokannoilla.

4.1 Testiympäristön kuvaus

Testit suoritettiin kahden tietokoneen ja reitittimen avulla. Toinen tietokoneista toimi palvelimena, jossa pyöri alkukantainen HTTP-palvelin sekä testien eri tietokannat asennettuina lokaalisti. Muita sovelluksia tai ohjelmia ei ollut käynnissä palvelimella, kun testejä suoritettiin. Palvelimena toimineen tietokoneen keskusyksikön käyttöä seurattiin käyttöjärjestelmän omalla resurssien seurantatyökalulla. Jäljelle jääneessä tietokoneessa pyöri asiakasohjelma (engl. client), joka lähetti palvelimelle pyyntöjä synkronisesti. Pyyntöjen sisältämä data simuloi eri IoT-laitteiden lähettämiä paketteja niiden tilatiedostaan. Testin tietokoneet olivat yhteydessä toisiinsa kiinteällä yhteydellä.

4.2 Käytetyt laitteet

Reitittimenä toimi Huawein WiFi WS5200 WiFi-ac reititin. Palvelimena toimi vuoden 2022 MacBook Air 13", jossa oli M2 siru ja asiakasohjelma pyöri HP 250 G6 -kannettavassa tietokoneessa. Laitteet valikoituivat siten, että laitteet olivat ainoat, joiden avulla pystyttiin toteuttamaan järkevästi suorituskykytestit. Alta löytyvät taulukot laitteiden tarkemmista ominaisuuksista.

Taulukko 1: Testiympäristön reitittimen ominaisuudet

Selite	Arvo
Merkki	Huawei
Malli	Huawei 53037204
Verkkoyhteys	RJ45 (Ethernet)
Verkkoyhteyden maksimi nopeus	1000 Mbit/s

Taulukko 2: Testiympäristön asiakkaan ominaisuudet

Selite	Arvo
Merkki	HP
Malli	250 G6
Käyttöjärjestelmä	Windows 10 Pro 21H2
Suoritin	i5-7200U 2,5–3,1 GHz Intel Turbo Boost -tekniikalla
Muisti	8 Gt DDR4-2133 SDRAM

Taulukko 3: Testiympäristön palvelimen ominaisuudet

Selite	Arvo
Merkki	Apple
Malli	MacBook Air 13"
Käyttöjärjestelmä	Sonoma 14.6.1 (23G93)
Suoritin	Apple M2
Muisti	16 Gt LPDDR5

4.3 Käytetty ohjelmointikieli

Ohjelmointikieleksi valikoitui Python 3.10.15. Kieli valittiin sen helppouden takia. Versio valittiin, koska se oli viimeisin, joka tuki kaikkia ajureita eri tietokannoille. Kaikki riippuvuudet asennettiin Python Venv -virtuaaliympäristöön. Testeissä käytettiin myös Pythonin sisäänrakennettua random-moduulia, timeit-moduulia sekä time-moduulia. Random-moduulia käytettiin laitteiden tunnisteiden ja mittadatan luomiseen. Timeit-moduulia käytettiin laskemaan tietokannan operaation kesto ja time-moduulia käytettiin aikaleimojen (engl.

timestamp) luomiseen. Seuraavaksi taulukko, joka pitää sisällään palvelimen sekä asiakasohjelmiston riippuvuudet ja syyt minkä takia ne asennettiin.

Taulukko 4: Asennetut Python riippuvuudet

Riippuvuus ja versio	Syy
PyMySQL 1.1.1	Ajuri Pythonin ja MySQL-tietokannan yhdistämiseen
psycopg2 2.9.10	Ajuri Pythonin ja PostgreSQL-tietokannan yhdistämiseen
pymongo 4.10.1	Ajuri Pythonin ja MongoDB-tietokannan yhdistämiseen
cassandra-driver 3.29.2	Ajuri Pythonin ja Apache Cassandra -tietokannan yhdistämiseen
requests 2.32.3	Kirjasto HTTP-pyyntöjen tekemiseen asiakasohjelmistossa
Flask 3.1.0	Kirjasto alkukantaisen HTTP-palvelimen pyörittämiseen palvelimella

4.4 Tietokannat ja niiden käyttö

Työssä testattiin neljää eri tietokantaa. Niistä kaksi oli relaatiotietokantoja, MySQL ja PostgreSQL, yksi oli dokumenttipohjainen NoSQL-tietokanta, MongoDB, ja yksi oli sarakepohjainen NoSQL-tietokanta, Apache Cassandra. Seuraavaksi taulukko tietokantojen versioista.

Taulukko 5: Tietokantojen versiot

Tietokanta	Versio
MySQL	8.0.40
PostgreSQL	12.21
MongoDB	8.0.1
Apache Cassandra	5.0.2

Tietokantojen versiot määräytyivät Pythonin ajureiden tuen mukaan. Versiot ovat viimeisimmät versiot, jotka ovat olleet työn kirjoittamisen aikaan saatavilla joko tietokannoille tai Pythonin ajureille.

Kaikki tietokannat asennettiin paikallisesti vuorotellen Homebrew-paketinhallintajärjestelmän avulla ja jokainen tietokanta pyöri taustaprosessina testien aikana. Tietokannat olivat myös testin alussa tyhjiä, mutta taulukot luotiin ennen kuin mitään dataa tallennettiin tietokantoihin.

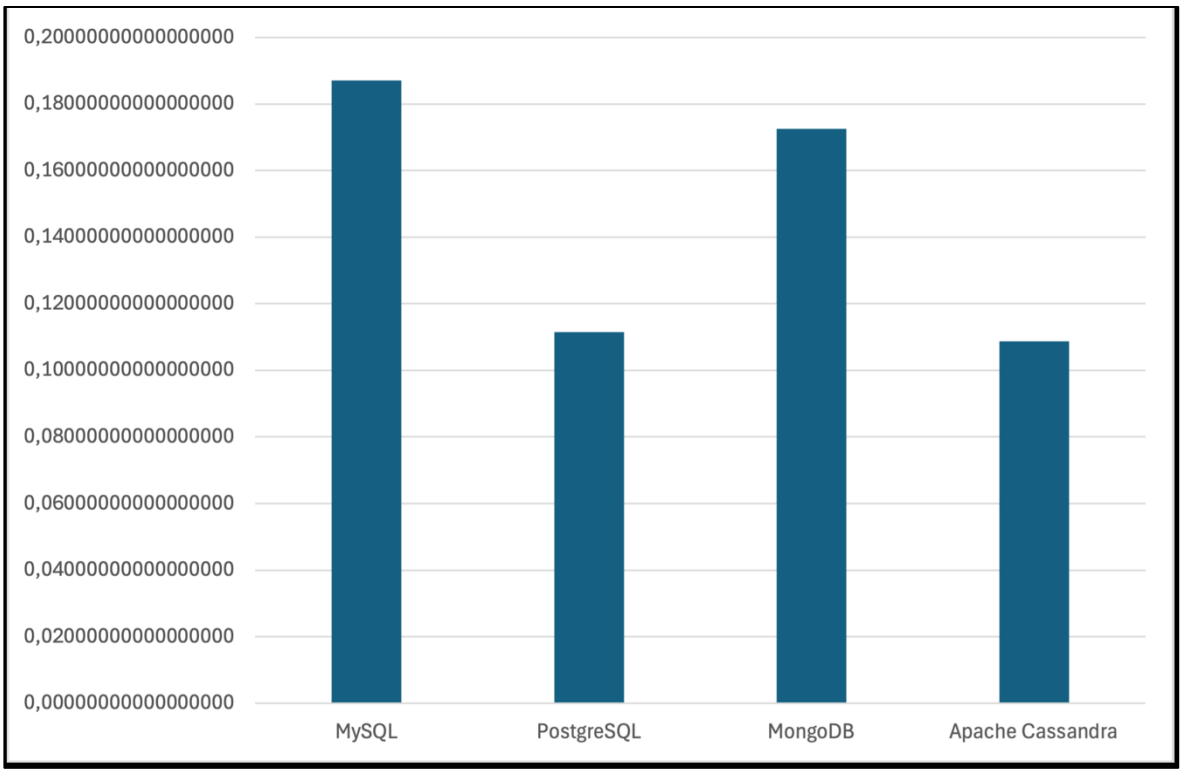
4.5 Testin kulku

Testit toteutettiin samalla tavalla muuttaen vain tietokantaa välissä. Ensiksi käynnistettiin testattava tietokanta. Sitten käynnistettiin palvelin, joka yhdistyy käynnissä olevaan tietokantaan ja avaa ulkoisen portin, jotta palvelin pystyy vastaanottamaan pyyntöjä asiakasohjelmistolta. Kun palvelin on täysin pystyssä, lähettää asiakasohjelmisto 100 pyyntöä yksi kerrallaan. Jokainen pyyntö sisältää laitteen tunnisteen, joka on kokonaisluku väliltä 0–1000, ja laitteen lähettämän arvon, joka on liukuluku väliltä 0–100. Palvelin vastaanottaa pyynnön ja luo aikaleiman Unix-aikana millisekunteina, jonka jälkeen se tallentaa tunnisteen, arvon ja aikaleiman tietokantaan. Palvelin myös laskee ajan, joka kuluu pelkästään tietokantaan tallentamiseen ja kirjoittaa tiedon paikalliseen tiedostoon. Kaiken tämän jälkeen hyväksyy palvelin taas uuden pyynnön, joka toistaa saman ketjun, kunnes kaikki 100 pyyntöä on käsitelty.

4.6 Tulokset

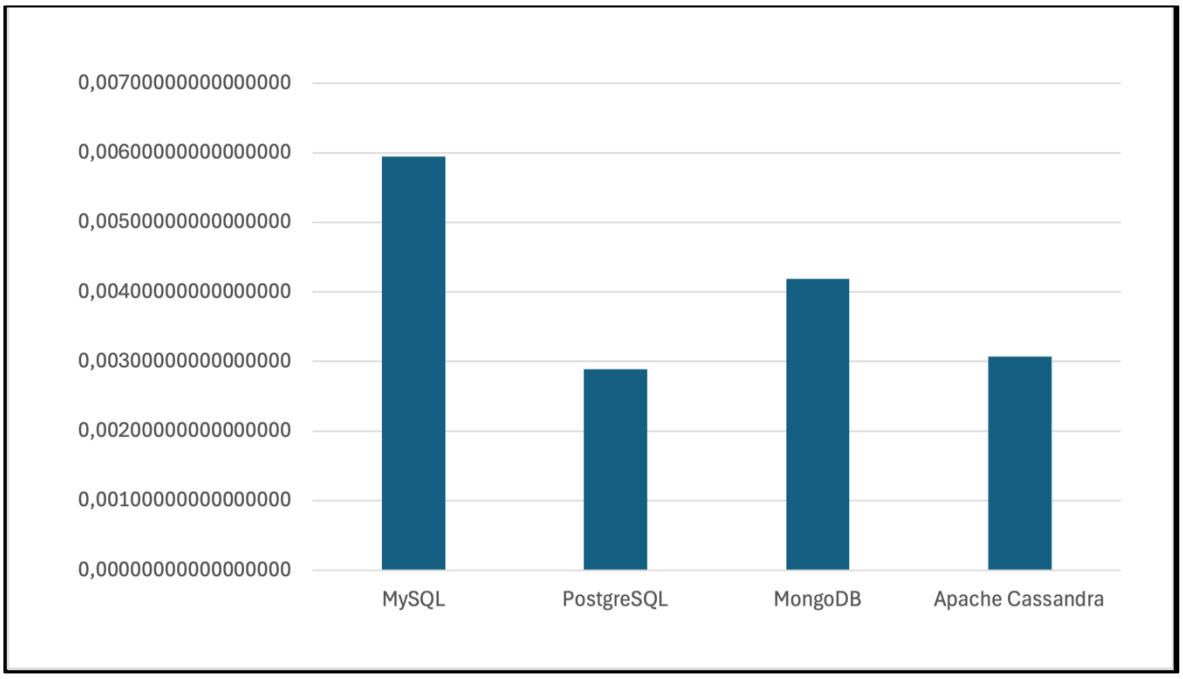
Suorituskykytestejä tehtiin yhteensä neljä. Jokainen testi suoritettiin eri tietokannalla. Testit suoritettiin, jotta nähtäisiin eri tietokantojen suorituskyky IoT-datan käsittelyssä. Testeissä sijoitettiin tietokantoihin 100 satunnaistettua kokonaisuutta, jotka sisälsivät liukuluvun, kokonaisluvun ja aikaleiman, joka oli 13 tavua pitkä suurkokonaisluku (engl. BigInt).

Kuva 1 näyttää, että eri tietokantojen kokonaisajoaikojen erot ovat pienet. Sillä suurimman ja pienimmän arvon väli on karkeasti vain 78 ms. Tietokannoista PostgreSQL ja Apache Cassandra voittivat muut tietokannat ajoilla 111 ms ja 109 ms. Huonoiten pärjäsivät MySQL-tietokanta ajalla 187 ms sekä MongoDB ajalla 172 ms.

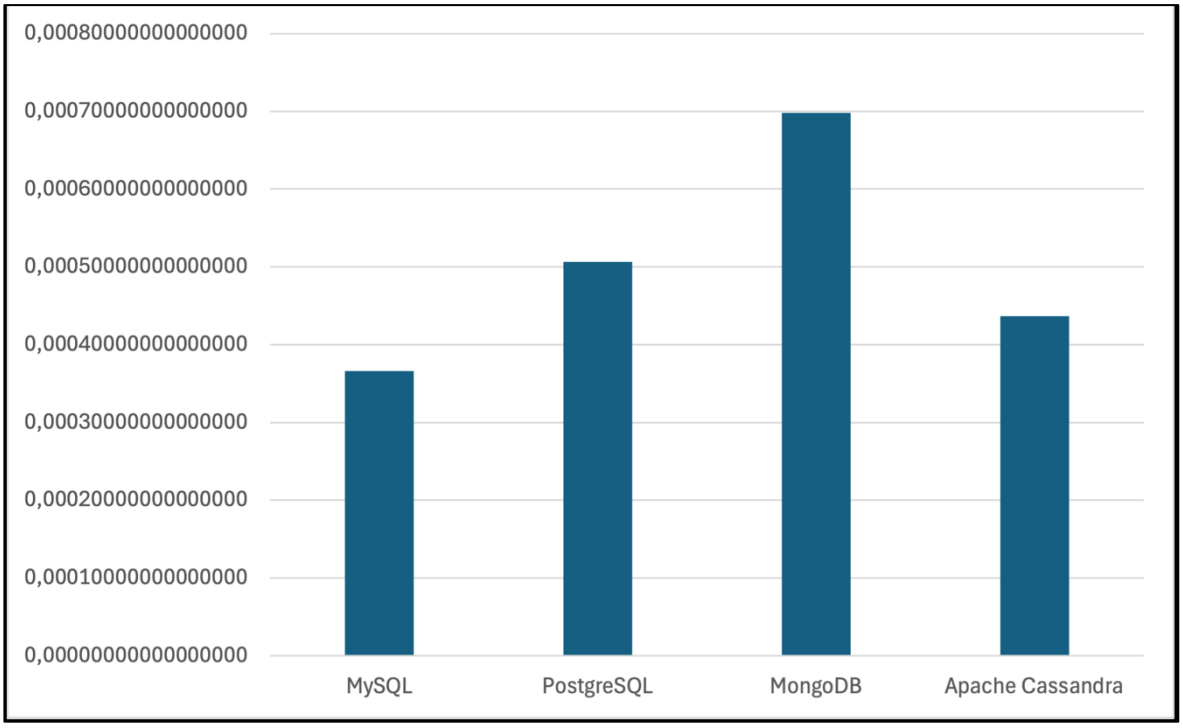


Kuva 1: Kaavio tietokantojen kokonaisajoajasta sekunteina

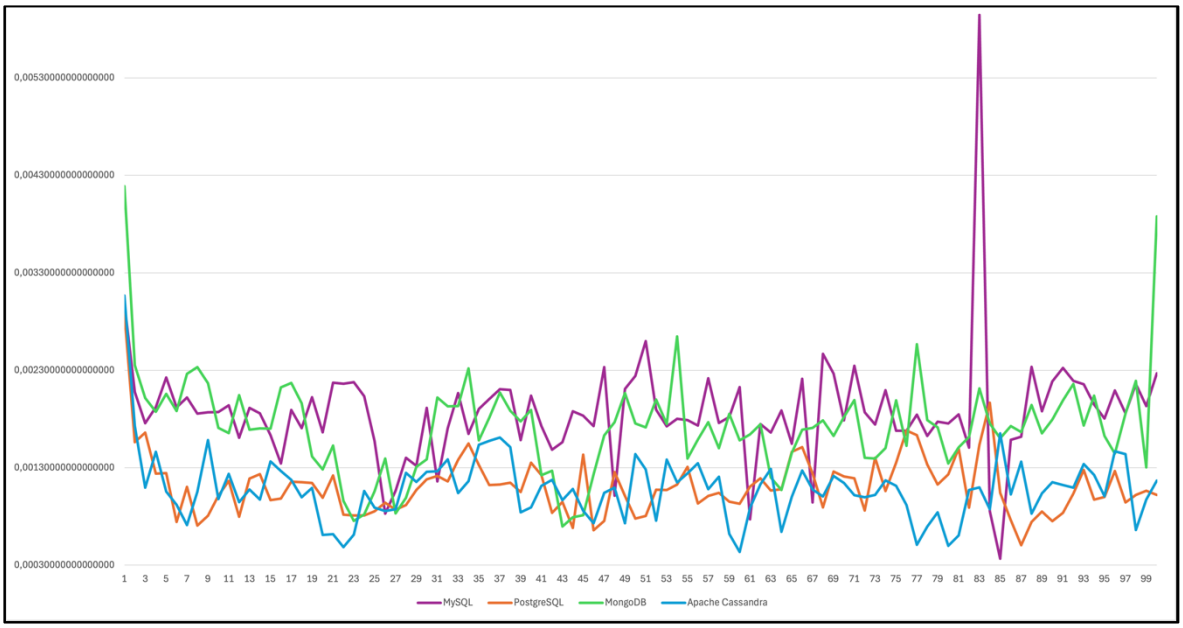
Kuva 2 esittää tietokantojen pisimmän ajan, joka vaadittiin sijoitusoperaatioon ja Kuva 3 lyhimmän ajan vastaavasti. Eri sijoitusoperaatioiden vaihteluväliksi tuli MySQL-tietokannalla 6 ms, PostgreSQL-tietokannalla 2 ms ja MongoDB- sekä Cassandra-tietokannalla 3 ms. Suuri ero MySQL-tietokannan ja muiden välillä selittyy 83. sijoitusoperaation kohdalla olevasta piikistä. Piikki voi johtua monesta eri syystä, kuten että palvelimella käynnistyi jokin toinen prosessi juuri sillä hetkellä. Ilman piikkiä on MySQL-tietokannan vaihteluväli myös 3 ms luokkaa.



Kuva 2: Kaavio testien aikana eniten aikaa vieneiden sijoitusoperaatioiden arvoista sekunteina



Kuva 3: Kaavio testien aikana vähiten aikaa vieneiden sijoitusoperaatioiden arvoista sekunteina



Kuva 4: Kaavio tietokantojen jokaisen sijoitusoperaation vaatimasta ajasta sekunteina

Kuva 4 esittää joka sijoitusoperaatioiden viemän ajan viivakaaviona. Kaaviosta huomataan, että tietokannat suoriutuvat tehtävästä melkein identtisesti, jos 83. sijoitusoperaation piikkiä ei huomioida MySQL-tietokannan kohdalla. Kaaviosta kuitenkin huomaa, että PostgreSQL- ja Apache Cassandra -tietokannat suoriutuivat paremmin kuin MySQL- ja MongoDB-tietokannat kuten Kuvasta 1 huomattiin. Vaikka eron huomaa kaaviossa on se kuitenkin suurimmillaan karkeasti 5 ms, kun tarkastellaan yksittäisiä sijoitusoperaatioita eri tietokannoissa eikä jätetä MySQL-tietokannan piikkiä huomioimatta. Erot ovat pieniä, mutta otoksen kasvaessa, kasvaa myös tietokantojen kokonaisajoaikojen erot, jolloin erot suorituskyvyssä näkyisivät selvemmin.

5 Keskustelu

Tämä luku tarkastelee testien tuloksia, kuinka testejä pystyisi parantamaan entisestään sekä tietokannan valintaa. Luku käsittelee myös viimeiseksi tutkimuskysymyksen ja siihen saadun vastauksen, tulevaisuuden suunnista sekä työn rajoituksista.

5.1 Tulosten merkitys

Tietokantoja testattiin 100 satunnaistetun sijoitusoperaation avulla. Eri tietokantojen tulokset olivat hyvinkin samankaltaiset erojen ollessa vain muutamia millisekunteja, kuten Kuvasta 4 voidaan huomata. Erot ovat pieniä, sillä testattiin hyvin perustavanlaatuista sijoitusoperaatiota, joka on oletettavasti hyvin optimoitu jokaisella tietokannalla. Eri operaatioihin siirryttäessä voitaisiin huomata lisää eroja tietokantojen välillä. Myös samanaikaisten operaatioiden vaikutus suorituskykyyn voisi olla huomattava.

Tämän työn testien tulokset ovat kuitenkin vain suuntaa antavia, sillä testit suoritettiin hyvin eristetyssä ympäristössä. Testit eivät ota, esimerkiksi huomioon verkon viivettä eikä samanaikaisia yhteyksiä tai operaatioita. Tulokset myös eroavat merkittävästi suhteellisen samankaltaisten töiden tuloksista, jotka on esitelty luvussa 2. Esimerkiksi aikaisempien tutkimusten mukaan erot olisivat moninkertaisia, vaikka tämän työn testien tulosten perusteella erot olisivat minimaaliset.

5.2 Tietokannan valitseminen

Pelkästään testien tulosten perusteella selkeät voittajat ovat PostgreSQL ja Apache Cassandra. Todellisuus voi kuitenkin olla hyvinkin erilainen todellisessa tuotantoympäristössä, jossa samanaikaisia yhteyksiä esiintyy ja datan määrä on huomattavasti suurempi. Erojen vähäisyys tarkoittaakin sitä, että tietokannan valitsemisessa tärkeintä on tarkastella mihin tarkoitukseen tietokanta valitaan.

Tietokantoihin vaikuttavat myös muut tekijät, esimerkiksi SQL-tietokantojen nopeus ja tehokkuus riippuu taulukoiden normalisoinnista, mitä ongelmaa NoSQL-tietokannoilla ei ole,

sillä ne tallentavat tiedot yhtenä kokonaisuutena ilman suhteita (Albarak et al., 2020). NoSQL-tietokannat on myös suunniteltu skaalautumaan, mikä ei taas ole ollut mielessä, kun SQL-tietokantoja ollaan alun perin kehitetty (Hassan, Mohamed, 2024). Saatavuus on myös eri luokkaa, sillä esimerkiksi Apache Cassandran klusterissa ei ole pääsolmukohtaa, vaan jokainen on samanarvoinen, kun taas MySQL-klusterissa on yksi pääsolmukohta, jonka pettäessä ilmenee häiriöaikaa (engl. downtime) (Bell, 2018; Estrada and Ruiz, 2016). Turvallisuus on myös tärkeä ominaisuus. Turvallisuutta tarkasteltaessa ovat selkeät voittajat SQL-tietokannat, sillä ne ovat kypsempiä ja ovat olleet markkinoilla pidemmän aikaa (Hassan, Mohamed, 2024).

5.3 Testien parantaminen

Testit ovat nykyisessä muodossaan hyvin yksinkertaisia. Testejä voisi parantaa siten, että lisättäisiin eri tilanteita, kuten lisäämällä hakuoperaatioita sijoitusoperaatioiden väliin. Testejä voisi muokata myös lisäämällä samanaikaisuutta imitoimaan oikeata tuotantoympäristöä. Testit olisi voitu myös ajaa monta kertaa putkeen eikä vain kerran, kuten nyt tehtiin. Tämän tekeminen lisäisi testien tuloksien tarkkuutta. Testit voitaisiin myös toistaa samankaltaisena tietokantoihin, joissa on jo jonkin verran dataa sisällään. Näin tulokset heijastaisivat enemmän tuotantoympäristön arvoja.

5.4 Tutkimuskysymykseen vastaaminen

Johdantokappaleessa esiteltiin työn tutkimuskysymys. Tutkimuskysymys oli seuraavanlainen: Mitkä tutkituista tietokannoista soveltuvat parhaiten IoT-datan tehokkaaseen käsittelyyn ja kuinka suuret niiden erot ovat? Tämä aliluku vastaa tähän kysymykseen.

Työssä tehtiin suorituskykytestit neljälle eri tietokannalle, jotka olivat MySQL, PostgreSQL, MongoDB ja Apache Cassandra. Kaksi ensimmäistä luetelluista tietokannoista olivat perinteisiä SQL-tietokantoja ja jälkimmäiset kaksi olivat uudenlaisia NoSQL-tietokantoja. Suorituskykytestien perusteella voittajiksi valikoituivat PostgreSQL ja Apache Cassandra vain muutaman millisekunnin erolla kokonaisajoajassa, kun niitä verrataan toisiinsa. Muihin tietokantoihin ero oli hyvin karkeasti 70 ms luokkaa. Tulosten perusteella

tutkimuskysymykseen vastattiin vähintään tyydyttävästi, vaikkakin tuotantoympäristön tietokannan valinnassa tulee ottaa myös muita muuttujia huomioon kuin pelkkä sijoitusoperaatioiden tehokkuus.

5.5 Tulevaisuuden kehitys

Testien parantamisen lisäksi voitaisiin tulevaisuudessa tutkia lisää eri tietokantoja, kuten Redis, TimescaleDB ja MariaDB. Tulevaisuudessa voitaisiin myös vertailla perinteisten SQL- ja NoSQL-tietokantojen tehokkuutta aikasarjadatan käsittelyyn erikoistuneiden tietokantojen kanssa juuri IoT:n kontekstissa. Yksi mahdollinen tulevaisuuden suunta on myös eri tietokantojen suorituskyvyn testaaminen pilvipalvelualustalla tai -alustoilla. Toinen kehityssuunta on eri tietokantojen resurssienkäytön vertailu, mikä on erityisen tärkeää, jos tietokanta pyörii alustalla, jossa resurssit ovat hyvin rajalliset.

5.6 Työn rajallisuudet

Työ keskittyi vain SQL- ja NoSQL-tietokantoihin, joten sitä ei pysty soveltamaan NewSQL-tietokantoihin. Työ simuloi myös heikosti todellista tuotantoympäristöä, sillä se ei ota kaikkia oikean elämän muuttujia huomioon, kuten operaatioiden samanaikaisuutta tai viivettä. Työn testeissä käytetty sijoitusoperaatio on myös hyvin yksinkertainen, eikä testit kata esimerkiksi tiedonhakua ollenkaan tai monimutkaisia JOIN-operaatioita. Testien tulokset vaihtelevat myös eri laitteistoilla, esimerkiksi keskusytimen ja muistin nopeus vaikuttavat tietokantojen tehokkuuteen ja sitä kautta testeissä saatuihin tuloksiin. Testit suoritettiin myös hyvin alkeellisella HTTP-palvelimella, mikä puolestaan voi vaikuttaa testituloksiin.

6 Yhteenveto

Tässä työssä tutkittiin neljän eri tietokannan tehokkuutta, kun kyseessä on IoT-datan käsittely. Työhön valitut tietokannat olivat seuraavat: MySQL, PostgreSQL, MongoDB ja Apache Cassandra. Tehokkuus mitattiin suorituskykytestinä, jossa sijoitettiin jokaiseen tietokantaan 100 satunnaistettua kokonaisuutta ja mitattiin sijoituksiin kulunut aika.

Eri tietokantojen tehokkuutta tutkittiin, jotta tulevaisuudessa voidaan tehdä perusteltu päätös tietokannan valinnassa, kun halutaan tallentaa IoT-laitteiden tuottamaa dataa. Tietokannan tehokkuus on tärkeä etenkin IoT:n kontekstissa, jossa datan virta kasvaa nopeasti IoT-laitteiden lisääntyessä. Myös reaaliaikaiset IoT-sovellukset vaativat tehokkaan tietokannan, jotta sovellus näyttää oikeasti reaaliaikaista tietoa. Datan käsittelyn nopeuden lisäksi, oikean tietokannan valinta säästää resursseja ja kustannuksia sekä sitä kautta ympäristöä.

Työn testeistä saadut tulokset osoittavat, että parhaiten tietokannoista suoriutuivat PostgreSQL sekä Apache Cassandra, jotka suoriutuivat kokonaisuutena sekä yksittäisten sijoitusoperaatioiden kohdalla vain muutaman millisekunnin erolla verrattuna toisiinsa. Jos tarkastellaan yksittäisiä sijoitusoperaatioita, on kaikkien tietokantojen välillä eroa maksimissaan 4 ms. Tämä ero on mahdollisesti huomattava, jos otoksen suuruutta kasvatettaisiin, vaikka yhden operaation kohdalla on ero mitätön.

Työn testien tulokset antaisivat olettaa, että tutkituista tietokannoista suorituskyvyltään parhaimmat sijoitusoperaatioissa olisivat PostgreSQL ja Apache Cassandra. Tämä tarkoittaa sitä, että kyseiset tietokannat soveltuisivat parhaiten IoT-datan käsittelyyn neljästä tutkitusta tietokannasta. Erot ovat kuitenkin suhteellisen pienet etenkin, jos IoT-laitteiden määrä on vähäinen, mikä tarkoittaa, ettei ole järkevää tehdä päätöstä pelkästään suorituskyvyn perusteella vaan ottaa huomioon, esimerkiksi turvallisuus, skaalautuvuus ja saatavuus.

Työn tuloksia voidaan hyödyntää jatkossa valittaessa tietokantaa IoT-projektille. Tietokantojen suorituskykyä tulee kuitenkin tutkia lisää etenkin tuotantoympäristössä, jossa datan määrä ja virta on suurempi sekä on mahdollisuus moniin yhdenaikaisiin operaatioihin. Myös muiden tietokantojen kuin nyt tutkittujen suorituskykyä olisi hyvä tutkia.

Lähteet

Abdelaziz, M., 2022. Designing Production-Grade and Large-Scale IoT Solutions: A comprehensive and practical guide to implementing end-to-end IoT solutions, 1st ed. Packt Publishing Limited, Birmingham.

Albarak, M., Bahsoon, R., Ozkaya, I., Nord, R.L., 2020. Managing Technical Debt in Database Normalization. IIEEE Trans. Software Eng. 1–1. [Verkkoaineisto]. [Viitattu 27.11.2024]. Saatavissa: https://doi.org/10.1109/TSE.2020.3001339

Baiyere, A., Topi, H., Venkatesh, V., Wyatt, J., Donnellan, B., 2020. Internet of Things (IoT) – A Research Agenda for Information Systems. CAIS 47, 524–549. [Verkkoaineisto]. [Viitattu 7.11.2024]. Saatavissa: https://doi.org/10.17705/1CAIS.04725

Bell, C., 2018. Introducing the MySQL 8 Document Store. Apress, Berkeley, CA. [Verkkoaineisto]. [Viitattu 9.11.2024]. Saatavissa: https://doi.org/10.1007/978-1-4842-2725-1

Cabral, S., Murphy, K., 2009. MySQL administrator's bible, Online-Ausg. ed, Bible Ser. Wiley, Indianapolis, IN. pp. 4-7, 727-748

Cox, C., 2021. An introduction to 5G: the new radio, 5G network and beyond. Wiley, Hoboken, NJ. [Verkkoaineisto]. [Viitattu 10.11.2024]. Saatavissa: https://doi.org/10.1002/9781119602682

Easterbrook, S., Singer, J., Storey, M.-A., Damian, D., 2008. Selecting Empirical Methods for Software Engineering Research, in: Shull, F., Singer, J., Sjøberg, D.I.K. (Eds.), Guide to Advanced Empirical Software Engineering. Springer London, London, pp. 285–311. [Verkkoaineisto]. [Viitattu 18.11.2024]. Saatavissa: https://doi.org/10.1007/978-1-84800-044-5_11

Estrada, R., Ruiz, I., 2016. Big Data SMACK. Apress, Berkeley, CA. [Verkkoaineisto]. [Viitattu 10.11.2024]. Saatavissa: https://doi.org/10.1007/978-1-4842-2175-4

Ferrari, L., Pirozzi, E., 2020. Learn PostgreSQL: build and manage high-performance database solutions using PostgreSQL 12 and 13. Packt, Birmingham Mumbai.

Hassan, Mohamed, 2024. Big Data, Big Decisions Choosing the Right Database. arXiv 0–5. [Verkkoaineisto]. [Viitattu 27.11.2024]. Saatavissa: https://doi.org/10.48550/ARXIV.2405.02506

Haviv, A.Q., 2016. Web application development with MEAN: unlock the power of the MEAN stack by creating attractive and real-world projects: a course in three modules, Learning path. Packt Publishing, Birmingham, UK.

Juba, S., 2017. Learning PostgreSQL 10 - Second Edition: A beginner's guide to building high-performance PostgreSQL database solutions, 2nd ed. Packt Publishing Limited, Birmingham.

Kruger, C.P., Hancke, G.P., 2014. Benchmarking Internet of things devices, in: 2014 12th IEEE International Conference on Industrial Informatics (INDIN). Presented at the 2014 12th IEEE International Conference on Industrial Informatics (INDIN), IEEE, Porto Alegre RS, Brazil, pp. 611–616. [Verkkoaineisto]. [Viitattu 1.11.2024]. Saatavissa: https://doi.org/10.1109/INDIN.2014.6945583

Lakshman, A., Malik, P., 2010. Cassandra: a decentralized structured storage system. SIGOPS Oper. Syst. Rev. 44, 35–40. [Verkkoaineisto]. [Viitattu 3.11.2024]. Saatavissa: https://doi.org/10.1145/1773912.1773922

Liu, R., Yuan, J., Huang, X., 2024. Benchmarking Time Series Databases with IoTDB-Benchmark for IoT Scenarios. [Verkkoaineisto]. [Viitattu 2.11.2024]. Saatavissa: https://arxiv.org/abs/1901.08304

Matallah, H., Belalem, G., Bouamrane, K., 2020. Evaluation of NoSQL Databases: MongoDB, Cassandra, HBase, Redis, Couchbase, OrientDB. International Journal of Software Science and Computational Intelligence 12, 71–91. [Verkkoaineisto]. [Viitattu 10.11.2024]. Saatavissa: https://doi.org/10.4018/IJSSCI.2020100105

Ojo, M.O., Giordano, S., Procissi, G., Seitanidis, I.N., 2018. A Review of Low-End, Middle-End, and High-End Iot Devices. IEEE Access 6, 70528–70554. [Verkkoaineisto]. [Viitattu 7.11.2024]. Saatavissa: https://doi.org/10.1109/ACCESS.2018.2879615

Salunke, S.V., Ouda, A., 2024. A Performance Benchmark for the PostgreSQL and MySQL Databases. Future Internet 16, 382. [Verkkoaineisto]. [Viitattu 3.11.2024]. Saatavissa: https://doi.org/10.3390/fi16100382

Seghier, N.B., Kazar, O., 2021. Performance Benchmarking and Comparison of NoSQL Databases: Redis vs MongoDB vs Cassandra Using YCSB Tool, in: 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI). Presented at the 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI), IEEE, Tebessa, Algeria, pp. 1–6. [Verkkoaineisto]. [Viitattu 10.11.2024]. Saatavissa: https://doi.org/10.1109/ICRAMI52622.2021.9585956

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesslén, A., 2012. Experimentation in Software Engineering. Springer Berlin Heidelberg, Berlin, Heidelberg. [Verkkoaineisto]. [Viitattu 18.11.2024]. Saatavissa: https://doi.org/10.1007/978-3-642-29044-2