Examensarbete

HÖGSKOLAN I SKÖVDE

HANTERING AV ELPRIS DATA FÖR ETT SMART GRID SYSTEM

Prestanda analys vid hantering av elpris data mellan MySQL och MongoDB

MANAGING ELECTRICITY PRICE DATA FOR A SMART GRID SYSTEM

Performance analysis when handling electricity price data between MySQL and MongoDB

Examensarbete inom huvudområdet Informationsteknologi
Grundnivå 30 Högskolepoäng
Vårtermin 2022

Philip Johansson Thörn

Handledare: Johan Bjurén
Examinator: Henrik Gustavsson

Sammanfattning

Elektricitet är någonting vi alla använder dagligen och eftersom att elpriserna varierar över tid kan det bli dyrt för vissa hushåll. Det finns många tillvägagångssätt för att dra ner på elkonsumtionen och det finns även smarta tekniker som Smart Grid anpassar elkonsumtionen utifrån elpriser. Dessa smarta enheter kommunicerar med Smart Grid control centers databas för att koordinera dess körtid. Problemet är att för att denna lösning ska vidhålla en långtidslösning måste databaserna utföra operationer med låga svarstider. I studien utförs experiment som undersöker vilken av databaserna MySQL eller MongoDB är mest lämpad och ger svar på hypotesen *"För denna undersökning kommer MongoDB att prestera med lägst svarstider"*. Detta undersöks genom att jämföra svarstider mellan respektive databas och resultatet visar att MySQL presterar bättre än MongoDB vid hantering av elpris data. I framtiden vore det intressant att använda andra databaser samt andra tillvägagångssätt för att hantera datan.

Nyckelord: SmartGrid, Elpriser, MongoDB, MySQL

Innehållsförteckning

1 Introduktion

Elektricitet är någonting vi alla använder dagligen och det har blivit involverad i vårt samhälle att nästintill ingenting fungerar utan det. Men el är inte gratis och i vissa fall är det inte ens billigt, detta gjordes väldigt tydligt under december månad 2021 där en el-prognos utfördes av Energimarknadsbyrån. Prognosen berättar att under december månad 2021 har stigit upp till rekordpris ända sedan Sverige delade upp sig i elområden. Det finns många olika tillvägagångssätt för ett hushåll att eventuellt dra ner på allt för höga elräkningar och bland av de åtgärderna presenteras framtidsvision tekniken Smart Grid. Denna teknik tillämpar de senaste teknikerna och teorierna för att revolutionera världens elnät på ett effektivt, miljö positivt sätt, stabilisera och dra ner på elkonsumtionen. Hittills har denna framtidsvision inte helt blivit tillämpat i vårt samhälle förutom smarta enheter som vitvaror som tar del av elbörsen för att koordinera dess körtid utefter konsumentens egna konfigurationer. Den datan som nyttjas måste även kunna hämtas, ändra och beräknas utifrån elbörsens egna funktioner. Det vore därför vara optimalt att kunna utföra dessa operationer på ett effektivt och snabbt sätt att helhetslösningen fungerar på längre tid.

För att en sådan lösning ska fungera måste datan lagras vilket sköts bäst med databaser, i regel används antingen en relationsdatabas system (SQL) eller en icke rationell databassystem (NoSQL). Relation databassystem är en av de äldsta och populäraste databassystem dock har de påvisats att ha brister gällande större datamängd lagring eftersom att de inte är gjorde för de. Då presenterades fram icke-rationella databassystem som då ska tackla en av dessa problem.

Studien använder sig av en kvalitativ undersökningsmetod vilket innefattar att slutresultatet returnerar siffror och för detta fallet innefattar de svarstider. Respektive databassystem kommer genomgå olika prestandamätningar i form av experiment i en sluten miljö samt repeterbara för framtida studier Wohlin, C (2012). Utformandet av experimenten togs inspiration av olika studier som Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) och Abramova, V. & Bernardino, J. (2013) som använder experiment för att jämföra två databaser.

Kapitlen går igenom databas systemen som kommer användas i denna studie, nödvändiga tekniker som implementeras samt arbetets progression. Inkluderas även en pilotstudie som visar hur mätningarna kommer gå till i studien och få en första hands bild på vad som kan förvänta sig av mätningarna. Till sist utfördes tre experiment där vi mäter svarstider vid insättning av data, sökning på ren data och till sist hämtning av beräknad data. Resultatet av experimentet tyder på att MySQL är snabbare vid sökning av ren data och hämtning av beräknad data men påvisar svårigheter vid insättning av data där MongoDB är snabbare. Under experiment tillfället vid insättning av data upptäcktes något intressant där skillnaden på svarstiderna mellan MongoDB och MySQL var radikal att en utökning på experimentet var nödvändigt för att se om annorlunda insättningsmetod kan ha någon påverkan på resultatet. Eftersom att den ursplunglinga experimentet vid insättning av data inkluderar insättning av små dataposter skulle det vara intressant och se hur svarstiderna påverkas vid insättning av större dataposter. Resultatet tyder på att både MongoDB och MySQL är snabbare vid insättning av större dataposter, eftersom att vardera experiment av insättning innefattar annorlunda struktur vilket kan vara en påverkande faktor på svarstiderna.

2 Bakgrund

Under vintern 2021 genomfördes en prognos på sveriges elpriser där en jämförelse utfördes med december 2021 månads medelpriser för samtliga elområden slog rekord i högsta prisnivå sedan sverige delades in i elområden (Energimarknadsbyrån, 2022). Januari 2022 sjönk el-priserna nedåt igen. För samtliga elområden i Sverige har en drastisk minskning skett mellan december månad 2021 och januari månad 2022. Ett exempel i elområde 3 (Stockholm) där månadsmedel priset sjönk med 42% från 180,7 öre till 104,3 öre mellan december 2021 till januari 2022.

Under januari 2022 skriver föreslog regeringen med en el kompensation modell för hushåll som har drabbats av höga elräkningar (Regeringskansliet, 2022). Ersättningsmodellen är utformat som en trappa som börjar med en förbrukning på 700 kWh per månad upp till 2000 kWh där ersättningen varierar beroende på konsumtion. Med förslaget om en ersättningsmodell finns det en tanke att de flesta hushåll i sverige kommer att påverkas negativt av den drastiskt ökande elpriserna. Precis som många andra applikationer tillämpas ett databassystem för att lagra data och det gäller även för att lagra elpriser på elbörsen.

2.1 Smart grid

Elnätet idag är inte helt felfri och i vissa fall kan kan elnätet se exakt likadana ut som de gjorde förr i tiden. För de mesta utgår elnätet på samma teorier, teknologier och planlösning som för hundra år sedan Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012). För att tackla problemet samt att dra ner på eventuell elkonsumtion och minsta miljöavtrycket introducerades Smart grid som kombinerar avancerade tekniker för att tillhandahålla förbättrade tjänster för konsumenten.

I en artikel skriven av Office of Electricity (2011) om smart grid tillämpning i vårt samhälle där Smart Grid kommer att förbättra varje aspekt inom elektrisk leverans inklusive elektrisk produktion. Som då kommer att ge energi till de nytt initiativ som uppmuntrar konsumenter att ändra sitt mönster med sitt elanvändning. Med ett uppmuntrat initiativ kan inte bara elförbrukningen reduceras utan även miljöavtrycket genom att använda el på ett mer konservativt sätt. Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012) skriver vidare och förklarar att det finns primärt fyra stycken olika funktioner som Smart Grid förmedlar.

- **Load Adjustment** - Den totala belastningen anslutet till elnätet kan variera över tid och för att hantera större ändringar i elnätets belastning tillsätts standby ersättnings elgeneratorer.
- **Demand Response Support** - Funktionen att låta generatorer koordinera interaktionen med varandra för att undvika eventuella spikar i elkonsumtionen. Dra ner på konsumenternas el räkningar via meddelanden till lågprioriterade enheter att enbart nyttja elen när det är som billigast.
- **Decentralization Of Power Generation** - Ett kraftnät är uppbyggt på rutter där elen färdas för att koppla kraftnätet till konsumenter vilket funkar men är inte felfritt. Även fast Smart Grid kommer tillägga flera rutter återstår problemet vid en överbelastning vid ett visst område kan det orsaka en dominoeffekt och orsaka problem för relaterade områden. En teknik för att förhindra problemet är belastnings avlastning genom spänningsreduktion.

- **Price Signaling To Consumers** - I många länder, inklusive Belgien, Nederländerna Storbritannien har elbolag installerat elmätare för att uppmuntra folk att använda el under natten eller helgen. Genom att göra det smidigt för konsumenter att kontrollera elpriser för att då koordinera sina enheter i hushållet för att effektivt sätt dra ner på elkonsumtionen.

Dessa funktioner tar del av lösningen med smarta enheter som vitvaror som tar del av prisbilden och nyttjar el när elpriserna är rätt utefter hushållets egna konfigurationer. Men eftersom att dessa funktioner hos Smart Grid tillämpar någon form av internetuppkoppling för att analysera elpriser på elbörsen är låga svarstider en viktig faktor. Men för att tillhandahålla låga svarstider till konsumenten behövs ett databassystem som lagrar elpriserna och förmedlar informationen.

Smart Grid kontrollcenter har som huvuduppgift att realisera intelligenta förvägs varningar genom att övervaka kontrollera överförings nätverket i realtid J. Liu, X. Li (2011). Genom att samla in och analysera data för att fastställa bättre beslut och att säkerhetsställa säkerhet, tillförlitlighet, flexibilitet, ekonomin och effektiviteten hos kraftnäten och därför har ett databassystem en kritisk uppgift för att förverkliga intelligent utsändning och mottagning av data. Utöver att databassystem tillämpas i ett Smart Grid system helhetslösning utförde ett verklighets inspirerat experiment med hjälp av databassystem för att producera fram en prototyp för att hjälpa konsumenter analysera och kontrollera elkonsumtion för ett Smart Grid M. Schapranow, R. Kühne (2014)

2.2 Databaser

Under flera års tid har databaser används för att lagra stora mängder data där olika databaser har sina varierade fördelar och nackdelar. Abramova & Bernardino (2013) skriver att få tillgång till data som är lagrad på en databas är det vanligt att använda ett databassystem (DBMS) vilket är ett samlingsbegrepp på en kollektion på olika funktioner och tekniker för att hämta, redigera och lagra data. Hur data lagras spelar roll gällande val av databassystem, eftersom att olika databassystem lagrar och hanterar data på olika sätt därför att det kan skilja sig vilket databassystem som är mest lämpat för ändamålet Abramova & Bernardino (2013).

2.2.1 RDBMS

Med hjälp av relationsdatabaser har de blivit möjligt att lagra data i dess naturliga struktur utan någon underliggande struktur skriver Codd (1970). Där den naturliga struktur baseras på att lagra data i tabeller där tabellerna har relationer med varandra. Melton, J & Simon, A. (1993) tar upp att tabellerna i en relationsdatabas används för att lagra data som datatyper (string, int, float) och namn på den specifika data. Utöver datatypen och namn på data lagras behövs en identifierare för varje tabell vilket ska va de attributet som gör varje tabell unikt som ett personnummer, serienummer eller registreringsnummer, vilket som även används för att skapa en relation mellan tabellerna.

MySQL är en av de mest pålitliga och populäraste relationsdatabas systemen som används idag där de flesta högtrafikerade hemsidor tar användning av MySQL Satoto, K. I (2016).

Melton, J & Simon, A. (1993) skriver vidare och förklarar att relationsdatabaser använder språket SQL (Structured Query Language) vilket är frågespråk som gör det möjligt att kommunicera med databasen och få tillgång till data med hjälp av statements.

2.2.2 NOSQL

Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) menar på att dagen syn på databas val måste föraktas väldigt försiktigt då det inte var som för ett par års sedan där det var vanligt med några tusen användare på en och samma hemsida. Dagens läge vissa hemsidor kan besökas av miljontals användare dygnet runt vilket kan orsaka en stor belastning på vilken form av databassystem som skall nyttjas. Relationsdatabaser har varit ett bra val för prestanda när det gäller för limiterad datamängd men när det gäller för större mängder datahantering har de visat sig ineffektivt.

År 1998 introducerades termen "NO SQL" av Carlo Strozzi som refererades till non-relational databaser som senare blev återintroducerade 2009 av Eric Evans. NoSQL är ett samlingsord för "Not Only SQL" som till skillnad mot relationsdatabaser kan hantera ostrukturerad data. Eftersom att NoSQL databaser är non-relational databaser inte följer samma princip som relationsdatabaser där data lagras i tabeller. Scheman är inte fixerade och har en simpel data modell och istället använder ett id som unik identifierare för att hitta den specifika data till den specifika nyckeln.

Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) fortsätter och skriver att de fyra primära strategierna för NoSQL vid lagring av data i en non-relational databaser följer:

- **Key-value** databaser har inte ett fördefinierat schema, de är schema lösa. Baseras på att all data lagras parvis som nyckel och värde och databasen får tillgång till ett unikt värde genom att relatera nycklarna till dess värde.
- **Dokument** databaser lagrar på ett liknande sätt som key value databaser där varje insättning är unikt och nyckeln är kopplad till ett dokument. Flexibelt på grund av att de inte behöver något fördefinierat schema. Dokumentdatabaser hanterar datatyper som JSON, BSON, XML och BLOBs.
- **Kolumn** databaser lagrar data i kolumner där fördefinierade scheman är ett måste. Data är lagrade i celler och grupperade av kolumner där kolumnerna är i sin tur grupperade i familjer som då är större grupper av kolumner.
- **Graf-Orienterad** har stöd för komplexa data quiries som kan utföra i relativt små perioder jämfört med resterande av strategierna.

2.2.3 MongoDB

MongoDB är en open source NoSQL databas utvecklad år 2007 och är skrivet i C++ och hade sin första offentliga utgivning 2009.

```
{
    "_id": {
        "$oid": "620a4f0060c61b676d09f0e9" ← Autogenererade index
    },
    "name": "olof",
    "age": 28,
    "favcolor": "green",
    "pnr": "0000-0000-0001" ← Genererade index av admin
}
```

Figur 1 Exempel på BSON dokument och indexering

MongoDB är en dokumentdatabas där dokumenten lagras i formatet BSON - Binary JSON och grupperas i form av collections utefter deras struktur. Det går även att lagra dokument med olika strukturer dock för att vidhålla effektiviteten rekommenderas samma struktur på alla dokument Abramova, V. & Bernardino, J. (2013). Precis som relationsdatabaser använder MongoDB sig av indexering som innebär att varje dokument identifieras med ett _id-fält som fungerar som en nyckel där ett index skapas över varje unikt fält. Utöver automatiserat index skapande av _id-fält kan databasadministratörer skapa ytterligare index se Figur 1.

De viktigaste karaktärsdragen för MongoDB är hållbarhet (durability) och samtidighet (concurrency) där hållbarheten gör det möjligt att skapa kopior av data (Master-slave replication). Master-slave mekanismen gör det möjligt att definiera en Master och en eller flera Slaves där master kan läsa och skriva filer medans slaves agerar som kopior. När master försvinner blir slave med den senaste data befordrad till master.

2.3 Elpriser

Olika elavtal förmedlar olika priser för konsumenten där avtal med fast elpris innebär att konsumenten har ett bundet elpris under hela avtalets längd. Rörligt elavtal baseras priset på månadens medelvärde av spotpris (börspriset) vilket innebär att priset kan variera från månad till månad (Energimarknadsbyrå, 2022).

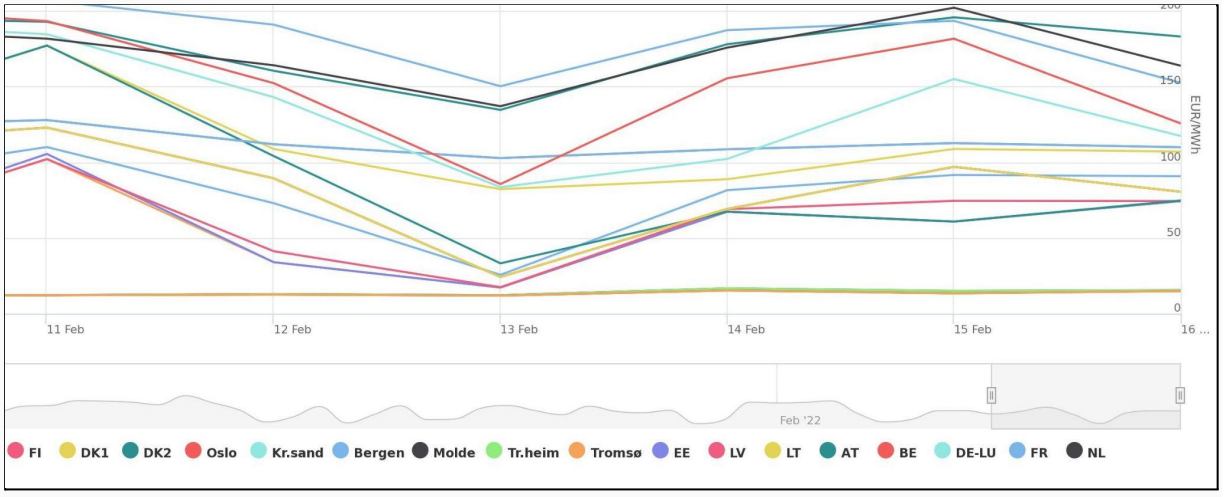
2.3.1 NordPoolGroup

I den dagliga konkurrensutsatta kraft marknaderna korrekt förutsägelse för elpriser är avgörande för alla deltagare i marknaderna. Genom att förmedla pålitlig daglig prisprognos information, producenter eller energitjänstföretag kan avgränsa bra ömsesidigt kontrakt och fatta bättre ekonomiska beslut M. Lei & Z. Feng. (2009). Marknader som inkluderas under kategorin kraft marknader som erbjuder prognoser och information om elpriser skulle kunna vara NordPoolGroup.

	SYS	SE1	SE2	SE3	SE4	FI	DK1	DK2	Oslo
16-02-2022	90,81	15,50	15,50	74,37	74,37	74,37	107,07	74,79	109,97
15-02-2022	91,64	15,05	15,05	60,83	60,83	74,54	108,81	60,83	112,68
14-02-2022	81,59	16,74	16,74	67,39	67,39	69,05	88,82	67,39	108,60
13-02-2022	25,68	12,20	12,20	17,28	17,28	17,49	82,28	33,25	102,78
12-02-2022	73,00	13,00	13,00	34,02	34,02	41,27	108,79	104,13	111,93
11-02-2022	110,06	12,20	12,20	102,12	105,51	102,12	177,00	177,15	127,87
10-02-2022	87,70	12,38	12,38	55,06	55,06	54,53	130,34	130,34	124,10
09-02-2022	75,60	13,10	13,10	57,14	57,14	78,89	117,47	102,89	120,69
08-02-2022	92,33	15,12	15,12	37,32	37,32	95,85	114,21	89,45	122,91
07-02-2022	92,02	16,29	16,29	80,37	80,37	105,23	81,20	80,84	121,81
06-02-2022	47,57	13,50	13,50	40,32	40,32	40,70	53,38	47,00	111,54
05-02-2022	33,97	14,68	14,68	23,63	23,63	58,98	49,59	28,09	115,73
04-02-2022	77,52	14,76	14,76	57,61	57,61	76,30	119,26	118,13	123,43
03-02-2022	143,41	101,22	101,22	176,43	176,43	176,93	173,64	178,13	147,61
02-02-2022	135,05	76,12	76,12	139,70	139,70	165,92	132,03	135,83	135,37

Figur 2 Exempel på day-ahead tabell (Bild tagen från NordPoolGroup hemsida)

NordPoolGroup skriver på sin hemsida att de är europas ledande elbörs som erbjuder handel, clearing, avveckling och tillhörande tjänster i 16 europeiska länder på både day-ahead vilket är en elpris tablå för dagens elpriser men även om morgondagens elpriser se Figur 2.



Figur 3 Exempel på intraday linjetabell (Bild tagen från NordPoolGroup hemsida)

Intraday-marknader vilket är pris tablå för hela dagen timvis se Figur 3 NordPoolGroup (2022). NordPoolGroup skriver själva att de strävar efter att erbjuda effektiv och säker day-ahead och intraday marknader för sina kunder.

Genom att erbjuda bland annat day-ahead marknader för sina kunder har NordPoolGroup möjligheten att presentera market data i form av day-ahead priser som summerar dagspriser för el i MWh enheter. Privatpersoner har då möjligheten att kontrollera distrikt elpriser. NordPoolGroup erbjuder en day-ahead tablå som gör det möjligt för privatpersoner att kontrollera elpriser för varje timme, varje dag.

3 Problemformulering

Prognosen som Energimarknadsbyrå tog fram i början av 2022 kom som en chock för de flesta svenskar i sverige, regeringskansliet förslag om kompensation för elräkningar förtydligar allvaret för situationen Regeringskansliet (2022). Även fast prognosen enbart täcker mellan december 2021 och januari 2022 Energimarknadsbyrå (2022) återstår problemet att el kan va väldigt dyrt för hushållen. Smart grid är en lösning som redan har implementeras i vissa hem som en smart elmätare Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012) vilket är en mätare som samlar in information som elkonsumtion, denna enhet inkluderas under Smart grid funktionen Price Signaling To Consumers. Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012) skriver vidare och förklarar att det finns andra enheter som använder sig av Smart grid funktionen som värmepumpar och andra vitvaror som kontrollerar el priserna på elbörsen och utefter hushållets egna konfigurationer kan enheterna konsumera el när det är som billigast.

Problemet med denna teknik är att enheterna i hushållet är uppkopplad mot internet för att hämta prisinformation som är lagrad på elbörsens databas som exempelvis NordPoolGroup är det väldigt viktigt med låga svarstider. Denna punkt förtydligats under artikeln skriven av J. Liu, X. Li (2011) där ett databassystem fyller en viktig roll för ett Smart Grid kontrollsystem vid analysering av utsändning och mottagning av data. Vid kommunikationen mellan ett Smart Grid system och tjänsterna som förmedlar elbörs priser är ett databassystem ett kritisk del av lösningen och eftersom att enheterna i hushållet i vissa fall står standby i väntan på information för att starta igång är snabbare svarstider ett måste för att kunna använda energi på ett miljövänligt och kostnadseffektivt sätt på grund av att elbörsens elpriser kan variera över tid Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012). Utöver att Smart Grid nyttjar och kommunicerar med ett databassystem förtydligats denna punkt med undersökningen skriven av M. Schapranow, R. Kühne (2014) där ett verklighets inspirerat experiment utfördes med hjälp av kommunikation mellan smart grid och ett databassystem.

För att en applikation som presenterar sådan data för ändamålet att förmedla elbörs data är de viktigt att applikationen ska kunna utföra operationer med snabba svarstider som möjligt. Därför har det viktigt på att analysera de olika databaserna via olika experiment för att då se vilket av de som har snabbast svarstider. När det gäller undersökningen kring vilka databaser som ska inkluderas i denna studie undersöktes artikeln skriven av Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) där en prestanda jämförelse utfördes mellan MySQL och MongoDB. Resultatet av undersökningen påvisade MongoDB presterade över marginellt snabbare än vad MySQL under alla experiment. Liknande studie skriven av Abramova, V. & Bernardino, J. (2013) där en prestanda jämförelse mellan MongoDB och Cassandra utfördes och resultatet påvisade en liten skillnad svarstider jämförelse med marginella skillnaden på undersökningen mellan MySQL och MongoDB skriven av Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015).

Med denna vetskap om slutresultaten från båda undersökningarna beslutas att för denna studie kommer MongoDB och MySQL inkluderas även fast MySQL presterade något långsammare än vad MongoDB presterande men kan det ändå vara intressant att inkludera MySQL.

Frågeställningen: *Om en dokumentdatabas representerad av MongoDB kan få kortare svarstider än en traditionell relationsdatabas MySQL vid insättning av ren data, hämtningar av ren data och hämtning beräknad data.*

Hypotesen för denna undersökning är att MongoDB får kortare svarstider. Denna hypotes utifrån undersökningen etablerad i Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) där en undersökning utfördes med MongoDB och MySQL där resultatet från undersökningen påvisar att MongoDB erhöll övermarginelt lägre svarstider än MySQL under alla experiment genomgångar.

3.1 Metodbeskrivning

Denna studie kommer utforma sig efter tekniska experiment eftersom att denna studie fokuserar sig mer på den tekniska delen istället för mänskliga faktorn Wohlin, C (2012). Anledningen är på grund av att experiment är byggt för att mäta prestandan via svarstider för att se vilket databassystem som har lägst svarstider förhållande till de datasetet som används. Eftersom att denna studie kommer inkludera tekniska experiment där resultaten returnerar svarstider kommer den mänskliga åsikten inte inkluderas.

Experimentet för denna studie kommer basera sig på liknande studier utförda av Abramova, V. & Bernardino, J. (2013) och Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015). Metoden för deras studie innefattar en jämförelse i svarstider mellan två olika datasystem där slutsatasen baseras på slutresultaten av experminetet. Experimentet är uppbyggd på olika tekniska experiment för att se vilket databassystem som har lägst svarstider. Baserat på liknande studier kan en slutsats dra att tekniska experiment som nyttjar en kvantitativ metod för att undersöka prestanda hos olika databassystem är att föredra. Experimentet kommer ske i en kontrollerad miljö via olika virtuella maskiner för att minimera felfaktorer som kan påverka slutresultat. Vid varje iteration av experimentet kommer svarstiderna samlas in i en loggbok som kommer nyttjas för att presentera fram slutersultaten.

Alternativa metoder för en sådan studie utöver tillämpning av ett experiment skulle kunna vara utformning av en användarstudie där användare testar en applikation för att sedan ge input via tankar och åsikter om applikationen. En användarstudie kan vara väldigt användbar om ett företag producerar en prototyp av en produkt där användar åsikter nyttjas för att utveckla och förbättra en produkt för intresset av användarna är viktigt. På grund av att denna studie kommer jämföra svarstider för de olika datasystemen kommer en användarstudie inte inkluderas då det är väldigt svårt för användare att ge detaljerade och säkra svar angående svarstider. Eftersom att svarstider kan variera mellan sekunder till millisekunder och att dra en förutfattad förväntan att användare ska kunna se skillnad på minsta lilla millisekund är orealistiskt.

Nackdelen med att använda en kvantitativ metod för att utforma experiment är att slutresultaten kan variera per experiment tillfälle och att fastställa att resultatet från experimentet kommer gälla för alla återupprepningar inte möjligt. På grund av att resultatet på experimenten kan påverkas av olika faktorer som internethastighet eller hårdvara, därför

kommer virtuella maskiner tillämpas för att kunna minimera på eventuella fel faktorer som kan påverka slutresultatet vid återupprepning Wohlin, C (2012).

3.1.2 Etiska Aspekter

Eftersom att NordPoolGroup skriver på sin hemsida att de erbjuder en pristablå för konsumenter att kunna ta del av samt även tillgång till en simpel nedladdnings möjlighet för att nyttja den datan som hemsidan erbjuder (NordPoolGroup, 2022), vilket va idén till en början att just den specifika data skulle användas till denna undersökning. Men efter vidare granskning över datan som presenteras, granskning kring eventuella kopplingar till privatpersoner samt hänsyn till återupprepning av experimentet har beslutet lagt att datan som nyttjas i experiment kommer att genereras via ett script baserat på struktur av NordPoolGroup egna data. Utöver datagenerering scriptet kommer all kod som skrivs under denna undersökning publiceras på GitHub för att underlätta en eventuell återupprepning av denna undersökning.

4 Genomförande/Implementation/ Projektbeskrivning

Kapitel 4 kommer gå igenom viktiga komponenter som implementeras för att göra denna studie möjlig. För att kunna besvara frågeställningen och hypotesen krävs två databaser varav en är en relationsdatabas, MySQL och den andra är en NoSQL databas, MongoDB. För att interagera med databaserna finns det många olika tillvägagångssätt men inför denna studie har vi valt att använda programmeringsspråket PHP för kommunikationen och en webbserver som hostar databaserna. Med hjälp av PHP blir det möjligt att utveckla olika scripts som kommunicerar och interagera med databaserna där applikationen sedan ska hämta och presentera data.

4.1 Research / Förstudie

Grund idén till denna studie inspireras utifrån NordPoolGroup's pristablå då de erbjuder olika funktioner för att inspektera elpriser över hela norden och andra länder. Utöver det togs även hänsyn till Smart Grid för själva studien och hur denna framtidsvision tar del av databassystem för att hämta och skicka ut data. Då studien använder sig av på olika programmeringsspråk och metoder har inspirationen hämtats från flera olika källor beroende på vad som skulle utföras och vilken del av scripten som skulle utvecklas.

Skillnaden mellan NoSQL och MySQL togs upp i boken Getting Started with NoSQL Vaish, Gaurav. (2013) som även var en bra startpunkt och introduktion till NoSQL. Installationen utgicks efter MongoDB egna dokumentation (MongoDB docs, 2022) som även inkluderar genomgående instruktioner på hur en databas byggs upp med kollektioner. Inför arbetet med MongoDB togs boken The Definitive Guide to MongoDB av Hows, D (2015) som hjälp eftersom att de gav stor mängd information om just MongoDB. Boken gav en bra grund till hur bland annat MongoDB fungerar gällande uppsättning av databas samt även hur PHP fungerar med MongoDB. Men eftersom att boken är relativt utdaterad när det kom till PHP aspekten med stora förändringar med metoder som används för att koppla MongoDB med PHP. En uppdaterad beskrivning hittades på PHP egna hemsida Php.net(The PHP Group 2022) där det finns tydlig information på vad som behövs göra och vilka resurser som kan involvera. En av dessa resurser är Composer verktyget som finns hos getcomposer.org vilket ger tillgång till viktiga bibliotek som krävs för att bygga upp en koppling mellan MongoDB och PHP.

För MySQL utgick installationen från MySQL egna dokumentation (MySQL docs, 2022) vilket involverar installation, uppsättning av databaser samt även tabell uppställning för att matcha struktur från MongoDB egna kollektion. Vid uppsättningen av databasen användes även boken Learning MySQL av Tahaghoghi et al. (2006) som beskriver utförligt hur en grundläggande databas kan byggas, inkluderar även vilka datatyper som är lämplig för vilken data och andra viktiga delar av en databas. Utöver uppsättning av databasen går boken även genom grundliga begrepp och tumregler som är viktigt och veta för att sätta upp en MySQL databas på ett korrekt sätt, bland annat förklarar boken hur en webbmiljö kan sättas upp lokalt för att koppla databasen med en webbapplikation. Genom att koppla databasen med en

webbapplikation som använder sig utav programmeringsspråket PHP vilket är nödvändigt för att genomföra studiens experiment.

Utöver installationen för vardera databas behövdes en webbserver där PHP scripten kan köras genom. Inför denna studie installerades XAMPP vilket är en Apache-distribution som inkluderar dem vanligaste webbutvecklings teknologierna i en och samma paket Dalibor, D Dvorski. (2007). XAMPP förmedlar bland annat Apache HTTP Server, PHP, MySQL och phpmyadmin för att tillhandahålla smidig utveckling av webbapplikationer.

4.2 Implementation och progression

Detta underkapitel kommer gå igenom alla de viktigaste delmomenten som ingick under implementationen för båda databaserna som inkluderar hämtning, insättning och mätningen av experimenten.

4.2.1 Databaser

Den viktigaste delen vid skapandet av respektive databas är gemensam struktur som innefattar gemensamma attribut och sedan gemensama datatyper eftersom att båda databaserna innehåller exakt samma data.

```
Create table elpriser(
        distrikt varchar(6),
        EURMWh float,
        datum date,
        startHour int,
        endHour int,
        primary key (distrikt, datum, startHour, endHour)
)ENGINE=INNODB;
```

Figur 4 Exempel på datastruktur i MySQL

Figur 4 visar ett exempel på hur data struktureras i MySQL #95622b07[1]

Största skillnaden mellan MongoDB och MySQL när det gäller för implementationen av databaserna är att MongoDB skapar själva kollektionen via PHP scriptet jämförelse med MySQL där själva SQL filen importeras in direkt till själva webbservern via PhpMyAdmin tablån. Andra tankar gällande databas implementation var att skapa två separata webbservrar för båda databaserna eftersom det dök upp misstanke att det skulle orsaka problem när de körs på samma webbserver. Men eftersom att båda databaserna körs på två olika portar orsakade det inga problem med funktionaliteten.

4.2.2 Hämta data

Vid hämtningen av data fastställdes ideen om att enbart använda PHP för att utföra kommunikationen och operationerna mot databaserna eftersom att PHP föll väldigt naturligt mot MySQL och för att ingen av databaserna ska ha någon fördel vilket programmeringsspråk som nyttjas, därför kommer PHP användas för MongoDB istället för Javascript. För att göra denna tillämpning möjlig behövs Mongodb PHP drivrutiner som laddas ner från MongoDB egna PHP dokumentation (MongoDB docs PHP, 2022).

[1] https://github.com/a19phith/Examensarbete/commit/95622b07

```
    for ($t=0 ; $t < 50 ; $t++) {
       $document = $elprisercollection-> find(
           ['datum' => Date("$day/$month/2022")]
       );
     }
     foreach($document as $doc){
       var_dump($doc);
       var_dump($counter);

   }
```

Figur 5 Exempel på hämtning av data för MongoDB

Vid hämtning av data för MongoDB nyttjas find() operationen #e5846edb[2]. För att sedan skriva ut resultatet för MongoDB erhålls tanken om att datan som returneras är i array struktur (JSON format). Då nyttjas en simpel foreach function med en var_dump för att skriva ut datan i en och samma expression se Figur 5 för MongoDB exempel för en simpel hämtning av data.

```
        $sql = "SELECT * FROM elpriser ORDER BY RAND () LIMIT 1; ";
         $result = $conn->query($sql);

         if ($result->num_rows > 0) {
             // output data of each row
             while($row = $result->fetch_assoc()) {
                 echo "<br>". $row["distrikt"]. " ". $row["EURMWh"]. " " . $row["datum"]
. " " . $row["startHour"]. " - " . $row["endHour"]."<br>";
             }
         } else {
             echo "0 results";
         }
```

Figur 6 Exempel på hämtning av data för MySQL

Genomförandet av en sökning måste den specificerade data hämtas från databaserna och för att göra detta möjligt öppnas en kommunikation mellan applikation och databas. För att hämta data nyttjas ett SQL statement #ef056036[3] för MySQL se Figur 6

[2] https://github.com/a19phith/Examensarbete/commit/e5846edb
[3] https://github.com/a19phith/Examensarbete/commit/ef056036

4.2.3 Generera data

Innan genomförandet av en sökning på data behövs en insättning av data i sin databas. Idén till en början var att skapa ett script som generera data i ett formulär där insättningen sker manuellt utan automatisering. Men eftersom att en av experimenten utformar sig efter att jämföra prestanda vid olika former av insättningar ändrades idén till att scriptet utför separata insättningar direkt in till databasen samt att datamängden blir mer unik då datan genereras på nytt efter varje omgång. Scripten används för att fylla databaserna med data från en och samma funktion med hjälp av olika for loopar. Värderna varierar från distrikt, elpris, datum, starttimme och sluttimme där varje data insättning representerar en timme i databasen. Data insättningen börjar med datumet 01-01-2022 i tidsspannet kl 00 - 01. Varje insättning ökar först på tidsspannet med värdet 1 vilket innebär att 24 insättningar representerar ett dygn. Andra attribut såsom elpriset nyttjar en slump funktion som genererar fram ett slump värde utefter min och maxvärdet, och vid granskning av NordPools egna data som denna data genererings script är baserat på är min värdet på 35 och maxvärdet på 350. För att inkludera decimaler för funktionen multipliceras värdena med 9 och att sedan dividera med 10.

Enda skillnaden mellan MySQL och MongoDB vid genereringen av data är att MongoDB är det väldigt viktigt att attributet datum skall skrivas in i ISO Date format. Anledningen är att utan ISODate formatet känner MongoDB inte igen att datumet som skrivs in är ett datum vilket resulterar i att värdet skrivs in som en string. Vilket förhindra experiment som inkluderar operationer kring datum och tidpunkter.

Med denna information kan en uträkning på antalet dataposter genomföras där en månad är 744 utifrån vilken månad datan genereras. När en data rad är genererad implementeras denna rad in till databasen och de som skiljer sig mellan MongoDB #1fce3803[4] och MySQL #ed31d611[5] är hur insert operationen utförs. MySQL utgår från sql INSERT statement och MongoDB utgår insertOne operationen där varje insättning sker via en array.

4.2.4 Prestandamätning

Den frågeställningen som denna studie grundas på kräver att prestandan mäts för båda databaserna, MongoDB och MySQL. Efter mätningen är utförd samlas resultaten in för analysering och jämförelse för att kunna dra en slutsats. För att mäta svarstider för varje experiment nyttjas PHP funktionen micro time vilket tar en tidsstämpel vid varje gång den körs, genom att ta en tidsstämpel innan experimentet körs samt en till tidsstämpel när experimentet är utförd erhålls tiden det tog att utföra experimentet genom att subtrahera första tidsstämpeln med den andra tidsstämpeln. Tidsenheterna mäts i millisekunder vilket kommer kommer va enheten som syns på resultat diagrammen.

[4] https://github.com/a19phith/Examensarbete/commit/1fce3803
[5] https://github.com/a19phith/Examensarbete/commit/ed31d611

```
$timestampList = [];
//Get a random document from the record with the operator $sample
for ($i=0; $i < 100 ; $i++) {
    //Start time
    $time_start = microtime(true);

    $document = $elprisercollection-> aggregate([
        [ '$sample' => ['size' => 1] ]
    ]);

    $time_end = microtime(true);

    //The time for the insert
    $time = $time_end - $time_start;

    //Push in the time in the array
    array_push($timestampList, $time);
}
//Create a CSV file
$file = fopen('Time-for-random-mongodb.csv', 'w');
foreach ($timestampList as $line) {
    //put data into csv file
    fputcsv ($file, (array)$line);
}
fclose($file);
```

Figur 7 Exempel på ett experiment där man hämtar slumpad rad från MongoDB databas

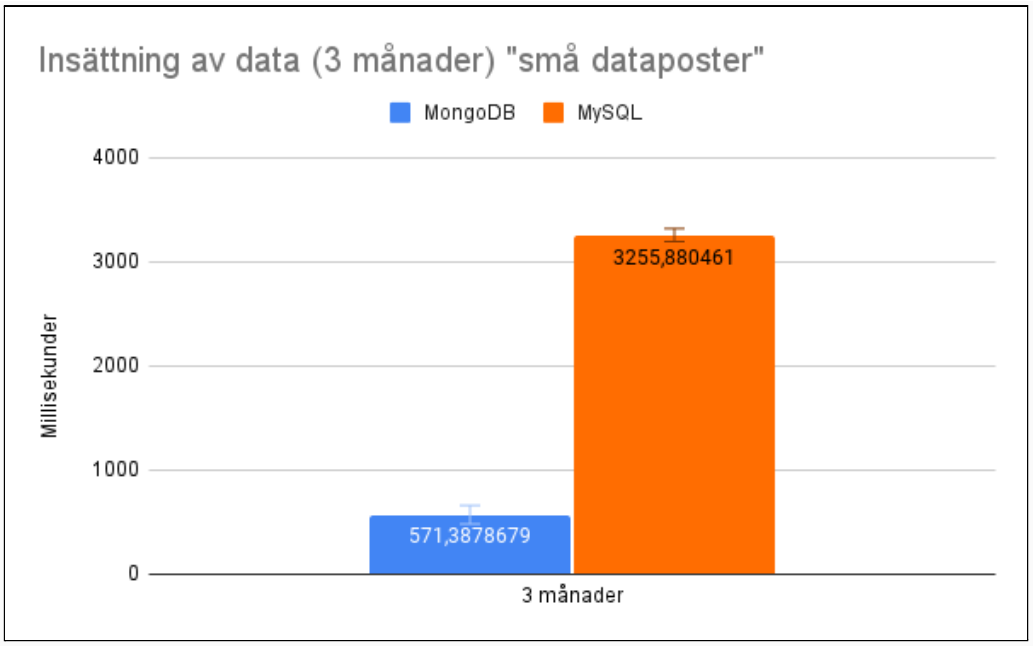
I Figur 7 syns scriptet som mäter svarstiden vid hämtning av ett random dokument från kollektionen. Start tidsstämpeln påbörjas i början av funktionen och slut tidsstämpeln påbörjas i slutet av funktionen. Efter att starttiden subtraheras med sluttiden läggs tiden in i tidsstämpel arrayen med hjälp av array_push där alla tiderna läggs in i form av en lista. Efter att alla experiment iterationer är fullföljda skapas ett csv dokument med alla sparade svarstiderna där datan från dessa mätpunkter används för att skapa ett diagram över svarstider. Varje experiment sparas i separata PHP filer som genererar fram egna csv filer med svarstider och varje experiment har två versioner, Mysql- version #66bb82cc[6] och Mongodb- version #e5846edb[7]. Varje experiment utförs inom loppet av 100 iterationer med hjälp av en for loop där varje iteration sparar en svarstid.

[6] https://github.com/a19phith/Examensarbete/commit/66bb82cc
[7] https://github.com/a19phith/Examensarbete/commit/e5846edb

4.3 Pilotstudie

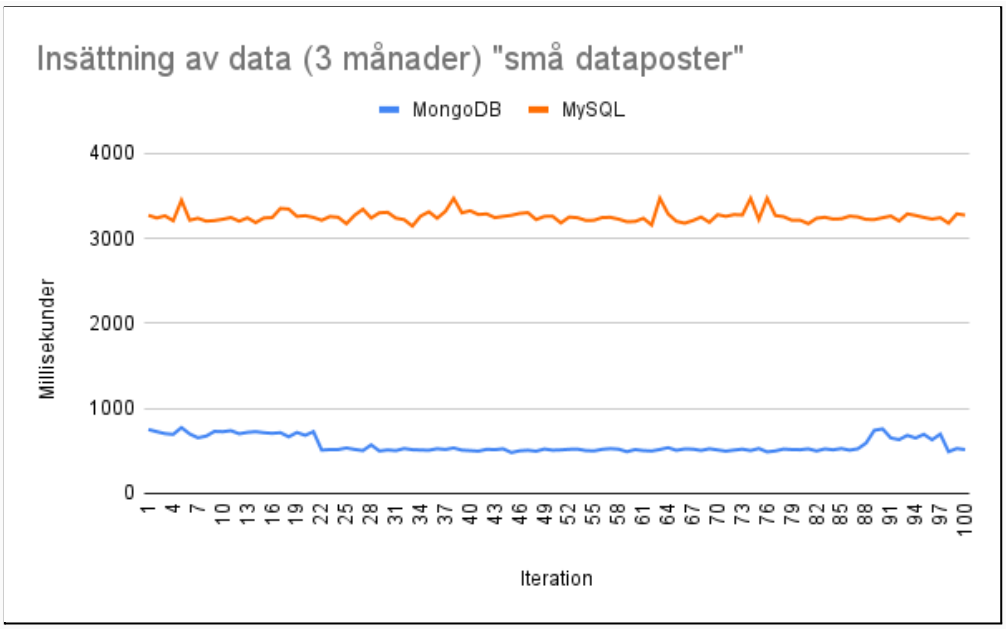
För att se till att mätningarna sker på ett korrekt sätt utfördes en pilotstudie som även gav en baseline för framtida experiment. Pilotstudien utgick på fem stycken experiment som innefattade insättning, sökning och beräkning av data för båda databaserna vilket innefattar tio stycken experiment. Mätserier som innefattar insättning av data kommer delas upp i två experiment, ena mäter svarstider på flera små insättningar och den andra mäter svarstider på en stor insättning.

Varje experiment bestod av 50 stycken mätningar med den datan som sätts in och den förutbestämda mängd av data poster kommer bli 2,160 vilket representerar de första 3 månaderna på året. Alla experiment utfördes på en och samma dator och webbläsaren som nyttjas är Google Chrome.



Figur 8 Stapeldiagram av 50 små insättningar i millisekunder

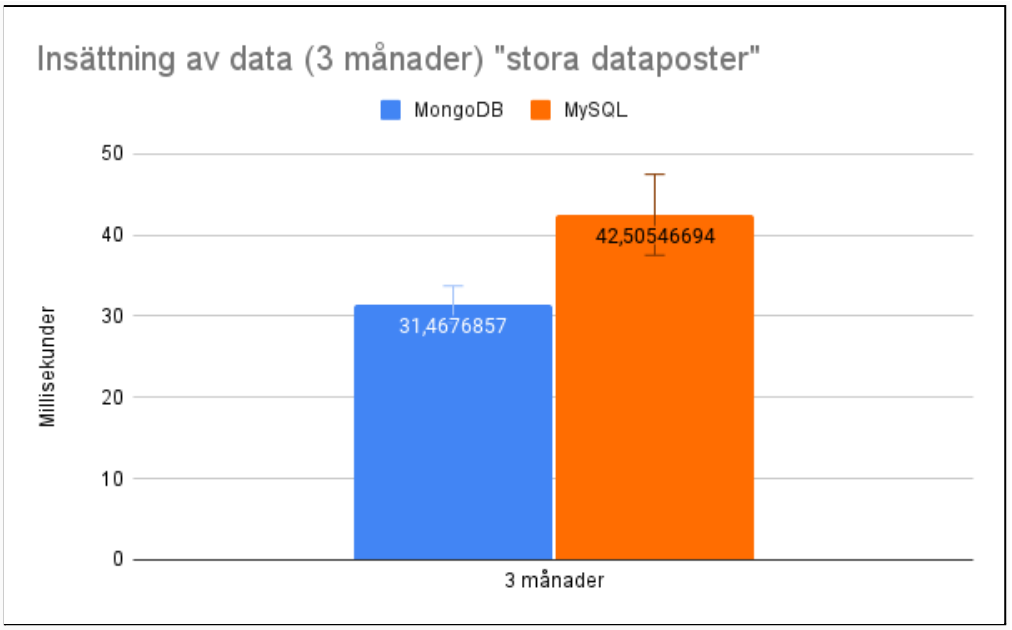
I Figur 8 presenteras resultatet för pilotstudien för insättningar av stora dataposter för respektive databas. Figur 10 visar att MongoDB presterar något snabbare än Mysql med 88,5% skillnad.



Figur 9 linjediagram av 50 små insättningar i millisekunder

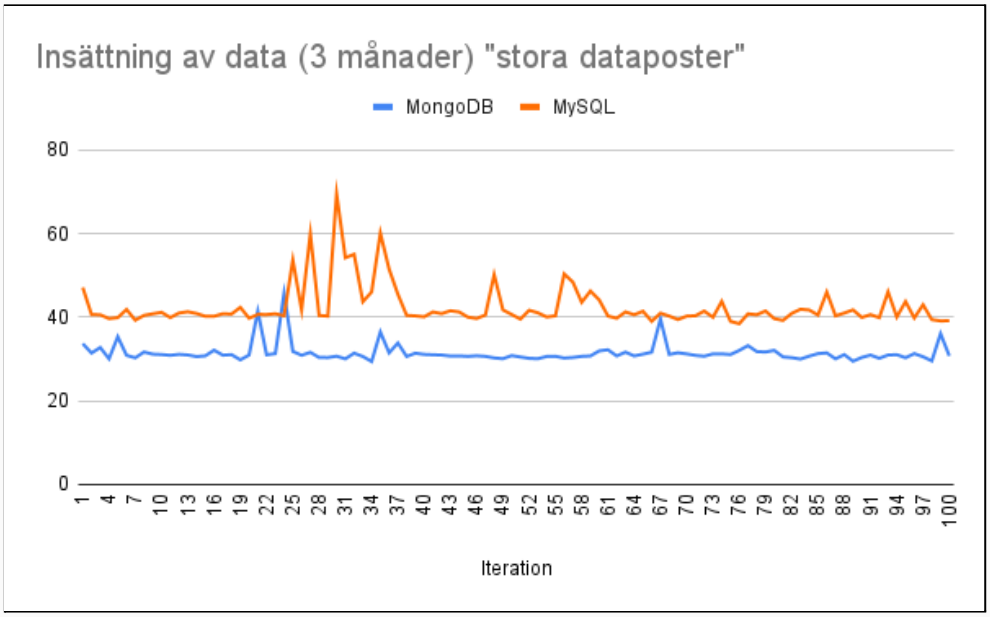
Figur 9 presenteras linjediagrammet vid insättning av data för respektive databas där både mysql och mongodb inte returnerar några specifika spikar som kan vara värt att undersöka mer. Resultatet tyder på att MongoDB är snabbare än MySQL gällande datamängden som nyttjades under pilotstudien. Inför de senare pilotstudie experimenten kommer den totala

datamängden öka för att se om det blir någon skillnad i slutresultatet samt att en tydligare bild på deras prestanda.



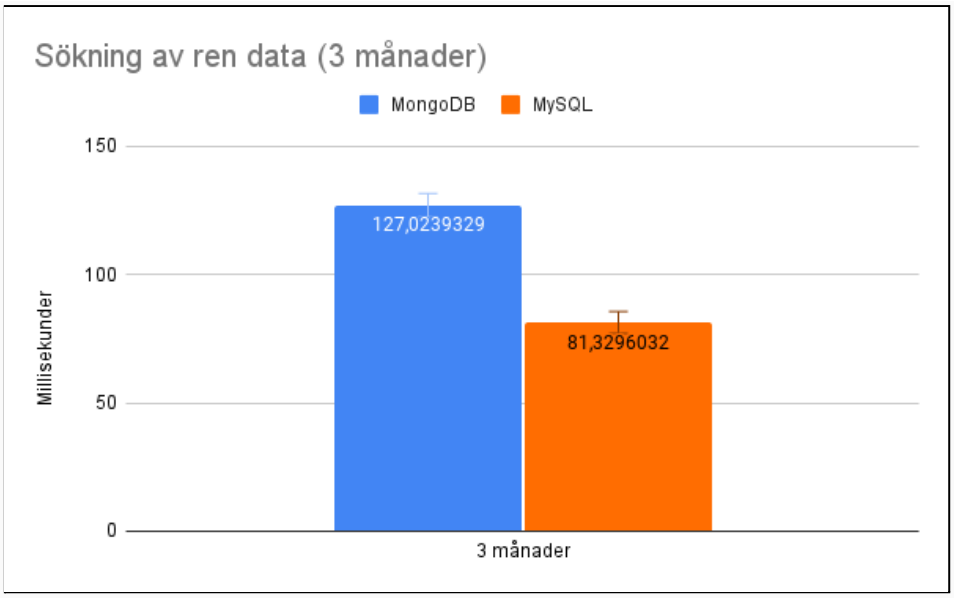
Figur 10 Stapeldiagram av 50 stora insättningar i millisekunder

I Figur 10 presenteras stapeldiagrammet på resultatet för pilotstudien för insättningar av stora dataposter för respektive databas. Figur 10 visar att MongoDB presterar något snabbare än Mysql med 25,9% skillnad. Resultatet tyder på att MongoDB är snabbare än MySQL gällande datamängden som nyttjades under pilotstudien. Inför de senare pilotstudie experimenten kommer den totala datamängden öka för att se om det blir någon skillnad i slutresultatet samt att en tydligare bild på deras prestanda.



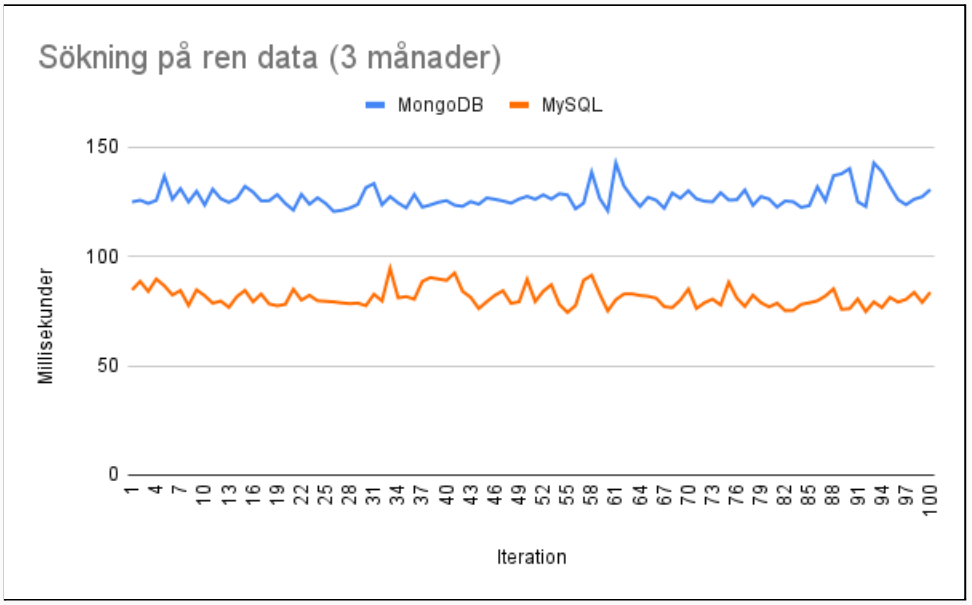
Figur 11 linjediagram av 50 stora insättningar i millisekunder

Figur 11 presenteras linjediagrammet vid insättning av data för respektive databas. Till skillnad från föregående pilotstudie där små dataposter genereras och sätts in successivt erhölls lägre svarstider för respektive databas med större insättningar.



Figur 12 Stapeldiagram av 50 sökningar i millisekunder

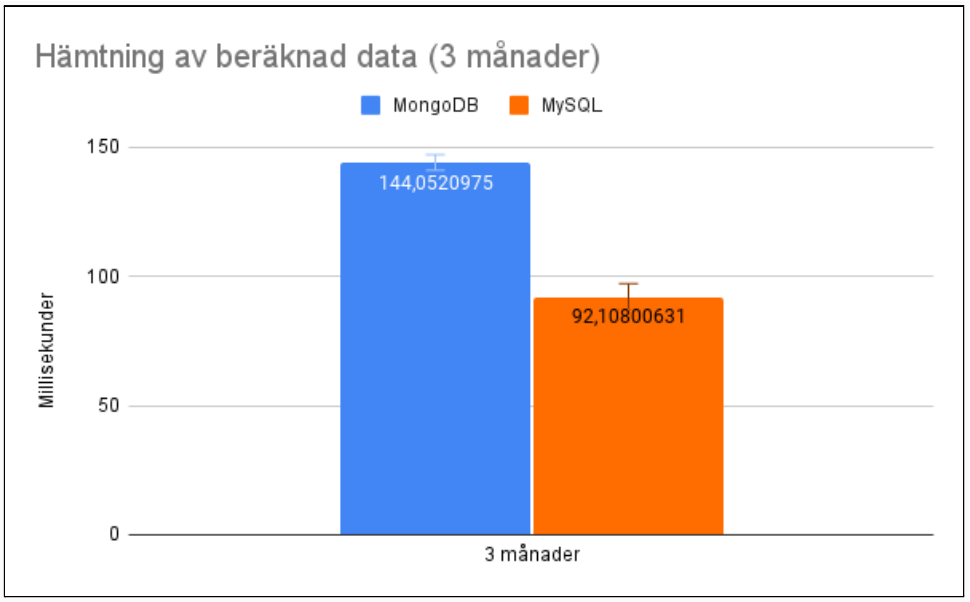
I Figur 12 presenteras stapeldiagrammet vid sökning för respektive databas där MySQL presterar i snitt snabbare än MongoDB gällande sökningar med 56,2% skillnad.



Figur 13 Linjediagram av 50 sökningar i millisekunder

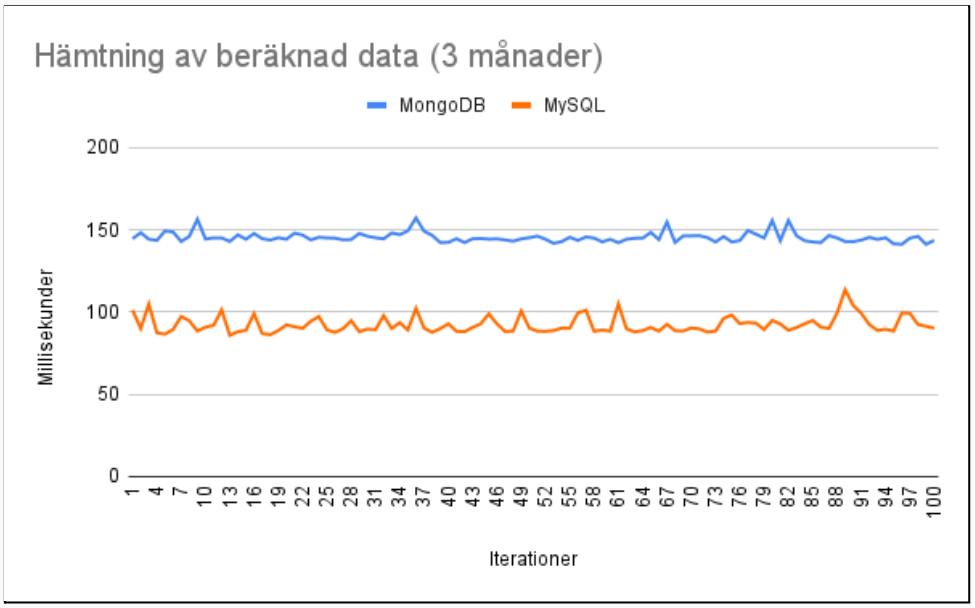
Figur 13 presenteras linjediagrammet vid sökning för respektive databas där pilotstudien var relativt stabilt. Resultatet tyder på att MySQL är snabbare vid sökning jämförelse med MongoDB gällande denna datamängd. Inför nästa experiment kommer datamängden öka och efter denna pilotstudie är det lite otydligt vilka som kommer returnera lägst svarstider

men utefter vad som etablerades i hypotesen finns det en chans att mongodb kommer ha snabbare svarstider.



Figur 14 Stapeldiagram av 50 hämtningar av beräknad data i millisekunder

I Figur 14 presenteras stapeldiagrammet med resultatet för experimentet hämtning av snittvärde för respektive databas.



Figur 15 Linjediagram av 50 hämtningar av beräknad data i millisekunder

Figur 15 presenteras linjediagrammet med resultatet för experimentet hämtning av snittvärde för respektive databas. För detta experimentet är MySQL snabbare än MongoDB med 40,1% skillnad. Resultatet tyder på att MySQL är snabbare vid hämtning av beräknad data jämförelse med MongoDB gällande denna datamängd och denna operation. Som

tidigare sagt under andra experimenten kommer datamängden öka för att se om det blir någon skillnad på resultat.

5 Utvärdering

I pilotstudien utfördes experimentet på svarstiderna hos MySQL och Mongodb via både insättning, sökning och beräkning av data för att se om scripten fungerar som det ska och får ut mätvärden för att jämföra de båda. Under pilotstudien vid insättningen av små dataposter presterade MongoDB snabbare än MySQL med 88,5% skillnad med jämförelse med sökningen där MySQL presterade snabbare än Mongodb fast med några fåtal millisekunder. Utifrån resultaten som tyder på att MongoDB enbart ligger lite snabbare i sökningen jämförelse med insättningen där skillnaderna är mycket mera. Inför nästa experiment kommer datamängderna öka för att se om totala datamängden kan påverka slutresultatet.

Huvud Mätningen kommer ske med tre stycken datamängder som bestäms vid insättning. första insättningen kommer vara på 2160 dataposter som representerar 3 månader , den andra kommer vara på 4368 som representerar 6 månader och till sist tredje kommer vara på 8760 dataposter som representerar 12 månader.

5.1 Hård- och mjukvara

Denna mätning kommer att ske på en personlig dator, i tabell 1 nedan står alla hårdvaru specifikationerna.

Processor	Intel i5-8600k 3.60GHZ
RAM	16 GB 2133 MHz
Lagring	1TB HDD
Operativsystem	Windows 10 pro version 1903

Tabell 1: Datorspecifikationer

Mätningarna kommer utföras på en virtuell maskin där specifikationerna för hårdvaran finns i tabell 2.

Processor (antal)	4 st
RAM	4096 MB
Lagring	50 GB

Tabell 2: Virtuella Maskiner

Utöver hårdvara kommer specifik hårdvara behövas för att göra denna studie möjlig, mjukvaran och dess versioner kan ses under tabell 3.

Mjukvara	Version
Apache	2.4.41
PHP	8.1.4
MySQL	8.0.28
MongoDB	5.0.4
Google Chrome	100.0.4896.75

Tabell 3: Mjukvara och version

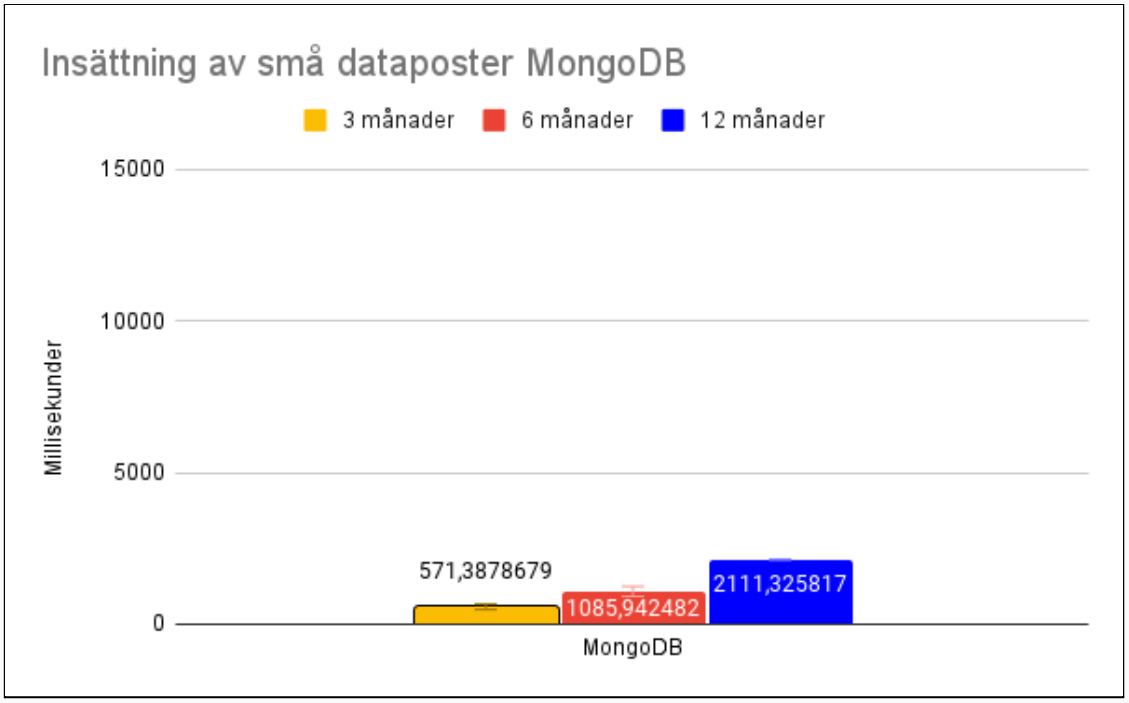
5.2 Presentation av undersökning

Experimentera kommer bestå av tre olika sorters mätningar, insättning av data , sökning av ren data och sedan hämtning av beräknad data. Vid insättning av data kommer experimenten delas upp i två del experiment för att kontrollera om det finns någon skillnad på insättning av flera små dokument parallellt med ett stort dokument. Varje experiment kommer genomgå 100 intervaller där varje intervall inkluderar 50 stycken operationer beroende på vilket experiment som utförs.

Varje intervall sparas tidsstämpel på hur lång tid det tog att utföra dessa 50 operationer. Mätserien undersöker hur väl databaserna presterar med olika datamängd. I varje mätserie kommer det finnas tabeller som presenterar den genomsnittliga tiden för mätningarna i millisekunder.

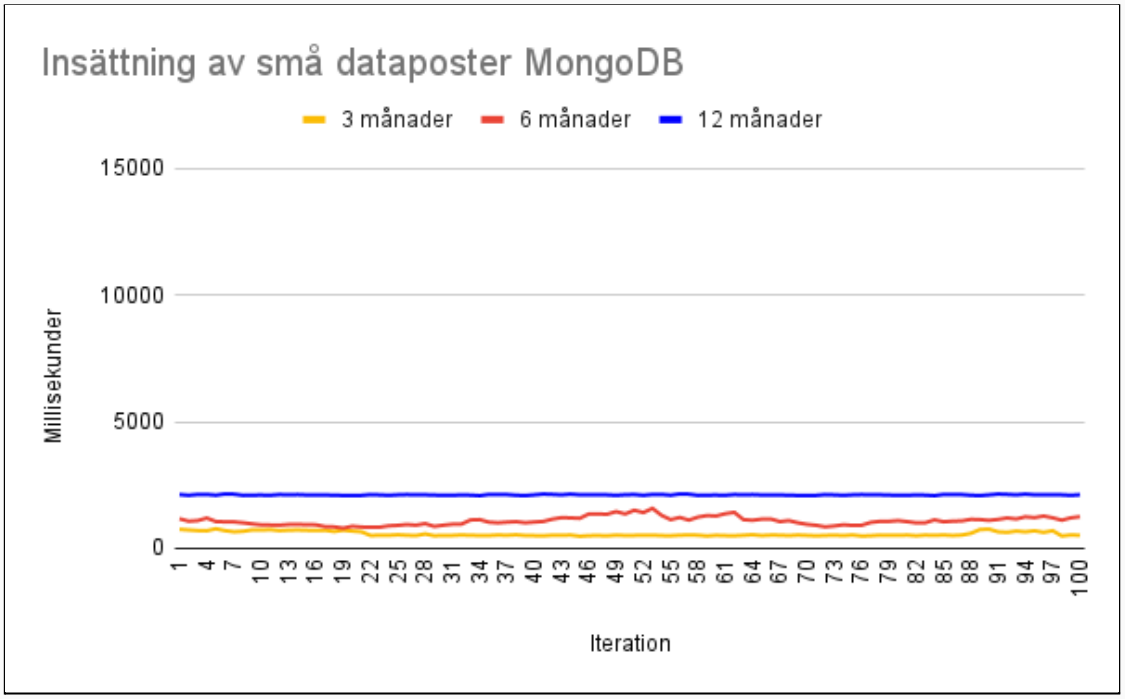
5.2.1 Mätserie 1 - Insättning av data

Mätserie 1 mäter vi svarstiderna vid insättning av data. Datan som nyttjas genereras fram via ett PHP script och tiden mäts utifrån att scriptet startar sin generering tills att alla datamängd är genererad och insatt i databaserna. Första del experiment mäts svarstiden på insättning av små dataposter och andra del experiment mäter svarstider på en stor datapost insättning.



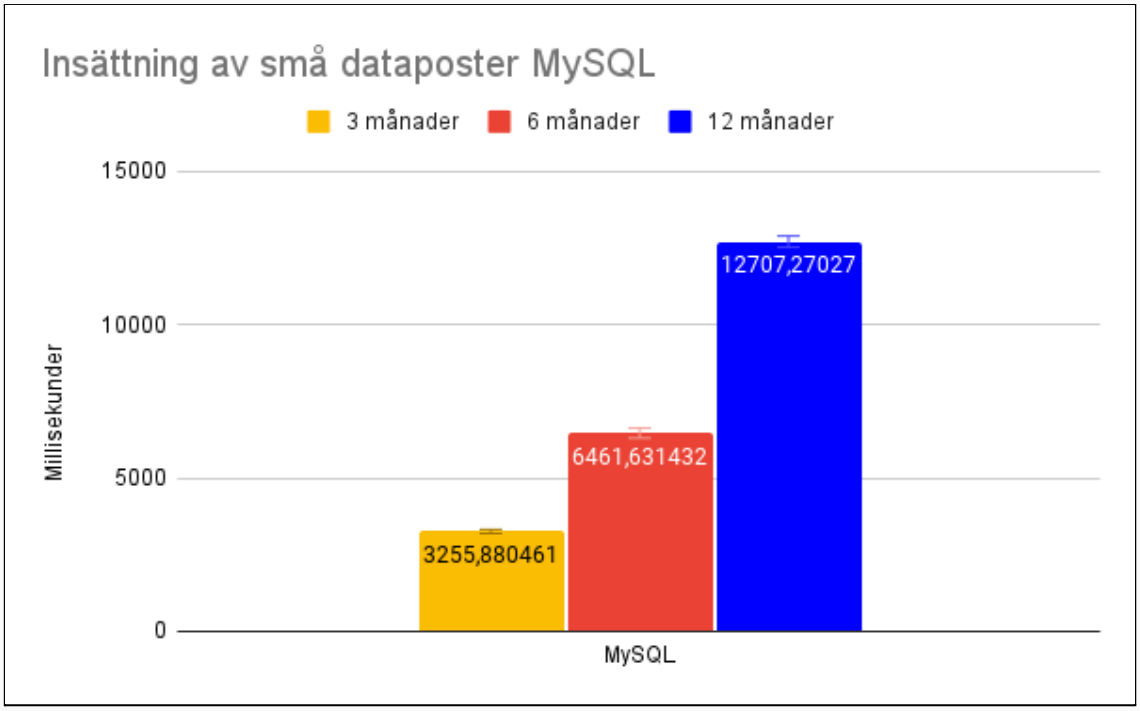
Figur 16 Mätserie 1 - Stapeldiagram för insättning av data för små dataposter MongoDB

I Figur 16 presenteras stapeldiagrammet med resultatet för MongoDB insättning av små dataposter.



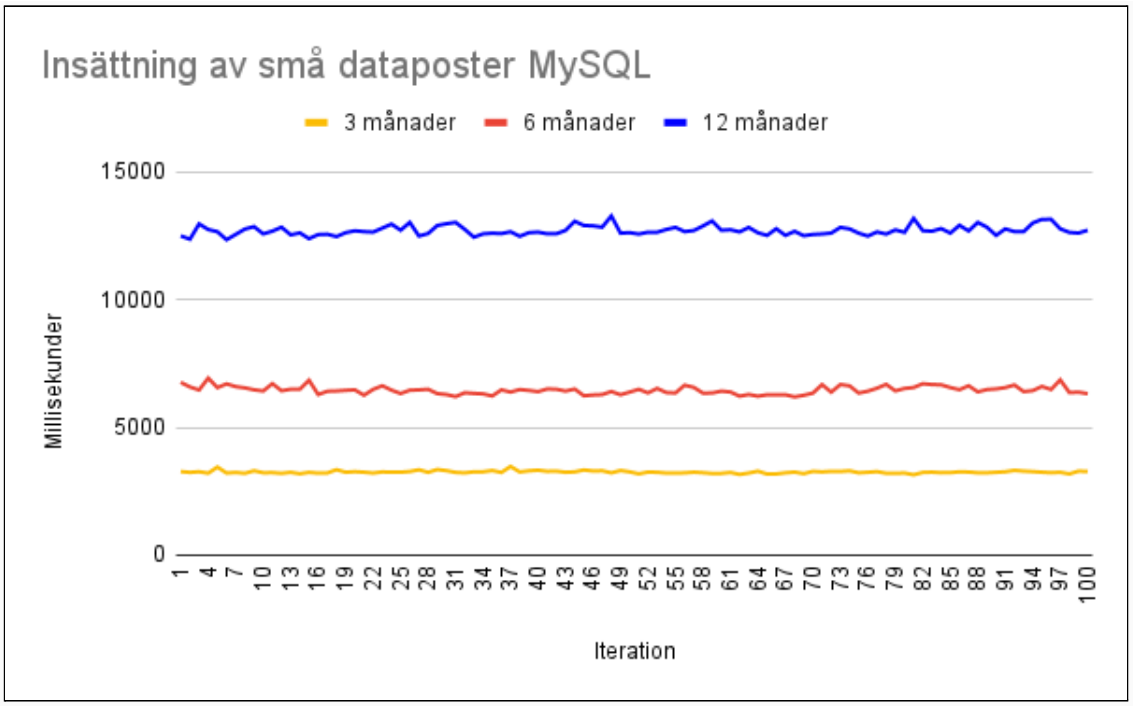
Figur 17 Mätserie 1 - Linjediagram för insättning av data för små dataposter MongoDB

Figur 17 presenteras linjediagrammet med resultatet för MongoDB insättningar av små dataposter, resultatet visar stabilitet genom hela experimentet.



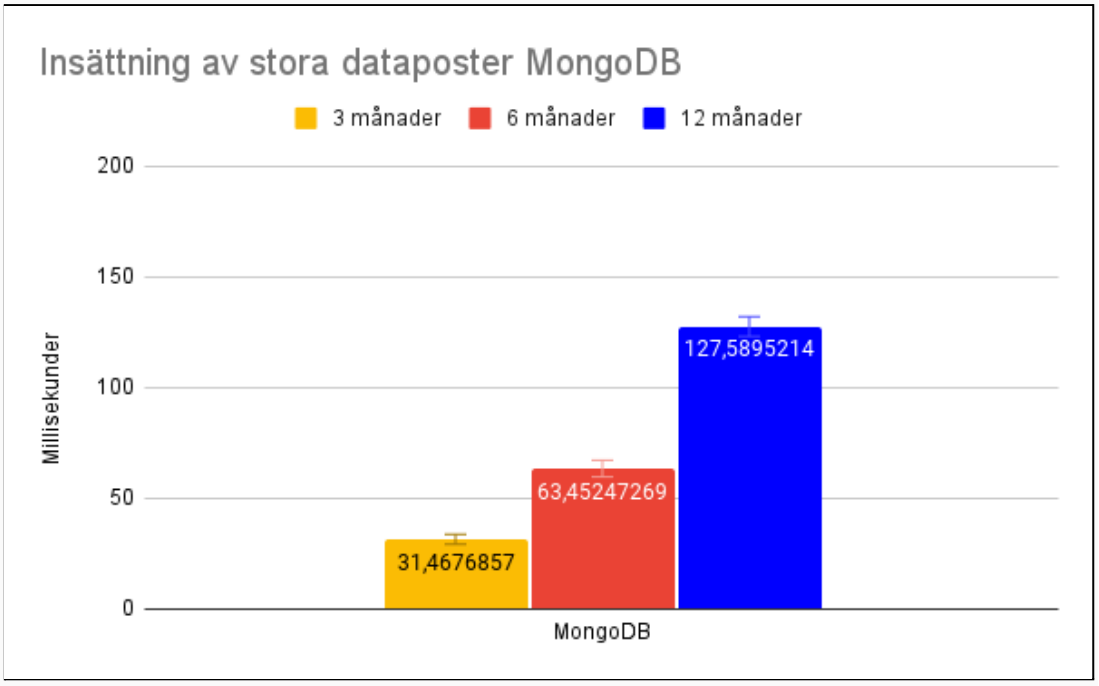
Figur 18 Mätserie 1 - Stapeldiagram för insättning av data för små dataposter MySQL

Figur 18 presenteras stapeldiagrammet med resultatet för MySQL insättning av små dataposter.



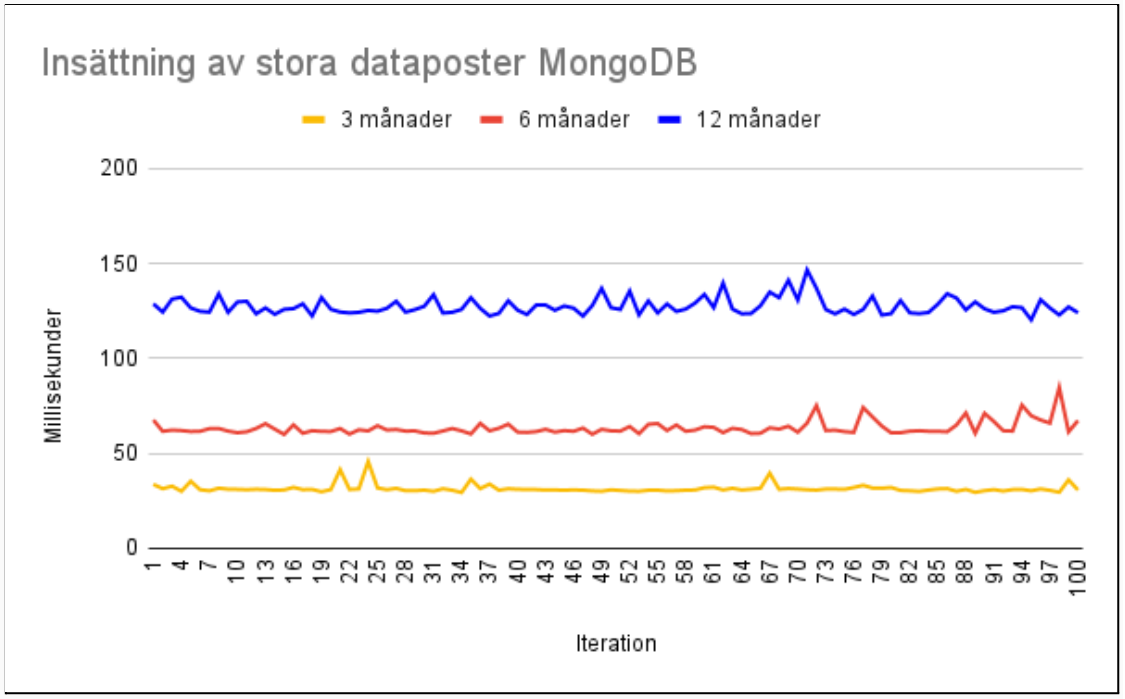
Figur 19 Mätserie 1 - Linjediagram för insättning av data för små dataposter MySQL

Figur 19 presenteras linjediagrammet med resultatet för MySQL insättningar av små dataposter, resultatet visar mindre stabilitet jämförelsevis med MongoDB experimentet. Vid insättning av av små dataposter presterar MongoDB i snitt snabbare än MySQL, med tydliga skillnader gällande tid där den största skillnaden gick från 2000ms till 12000ms. Med slutresultatet är den beräknade skillnaden för MySQL och MongoDB på 83.1%.



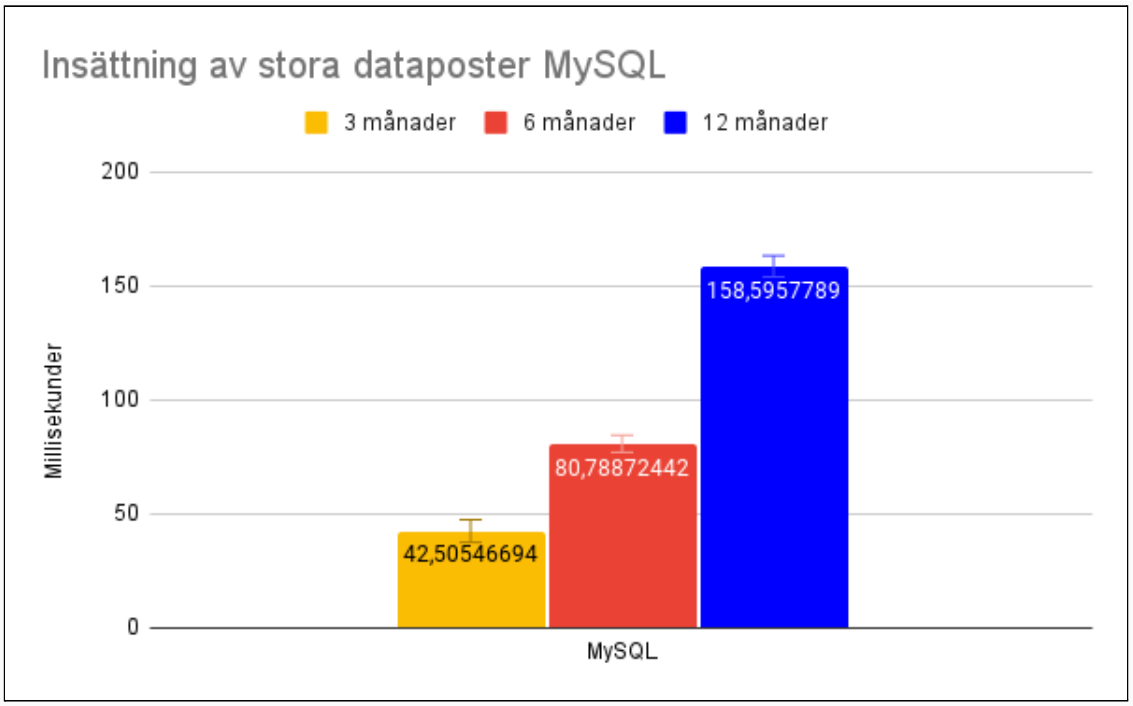
Figur 20 Mätserie 1 - Stapeldiagram för insättning av data för stora dataposter MongoDB

I Figur 20 presenteras stapeldiagrammet med svarstiderna för MongoDB vid insättning av stora dataposter och till förhållandevis med experimentet med små datapost insättningar presterar MongoDB med lägre svarstider.



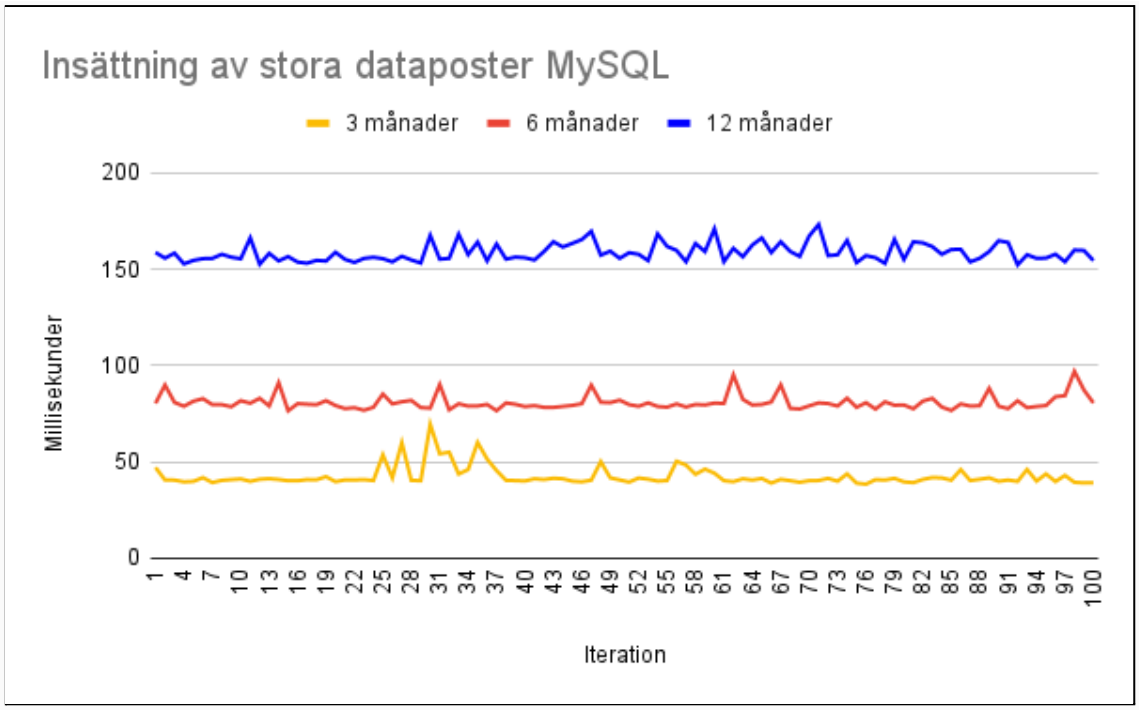
Figur 21 Mätserie 1 - Linjediagram för insättning av data för stora dataposter MongoDB

I Figur 21 presenteras linjediagrammet med resultatet för MongoDB insättning av stora dataposter. Förhållandevis med linjediagrammet från små dataposter returnerar Figur 17 mindre stabilitet men det kan bero på annorlunda upplösning vilket visuellt sätt kan ses som mindre stabilt.



Figur 22 Mätserie 1 - Stapeldiagram för insättning av data för stora dataposter MySQL

Figur 22 presenteras stapeldiagrammet med resultatet för MySQL vid insättning av stora dataposter där resultatet från figur 20 och 22 dras slutsatsen om att MySQL presterar med lägre svarstider än MongoDB.

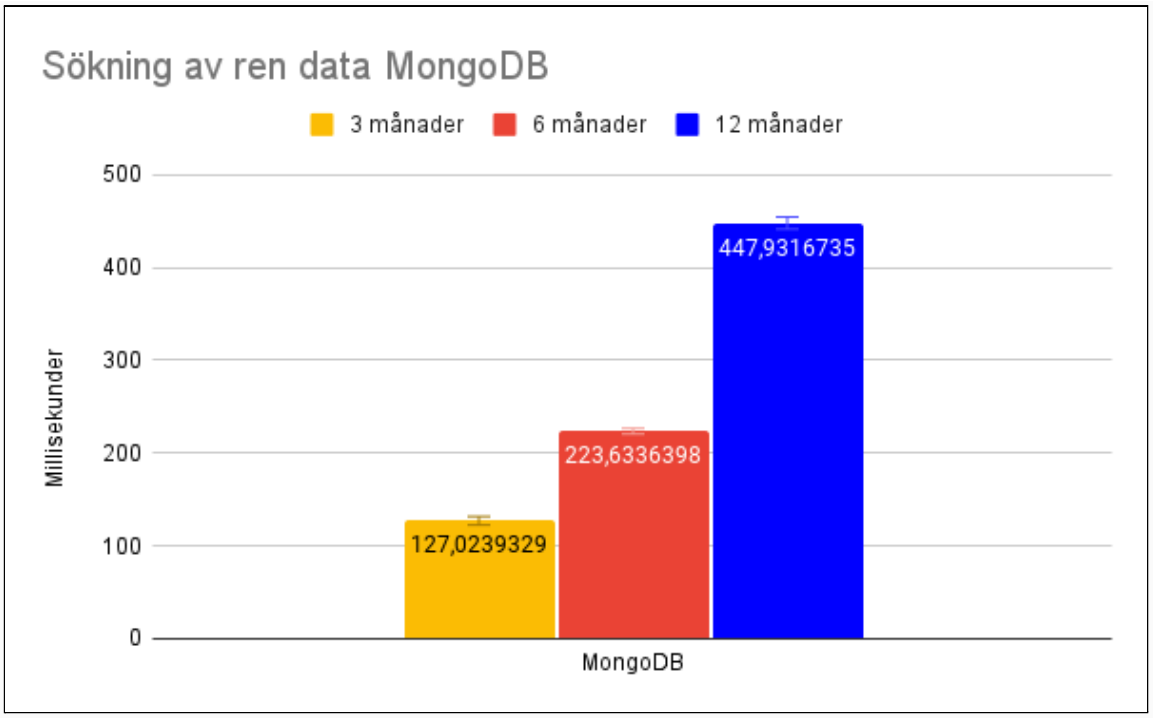


Figur 23 Mätserie 1 - Linjediagram för insättning av data för stora dataposter MySQL

Vid insättning av av stora dataposter presterar MongoDB i snitt snabbare än MySQL. Med resultatet från Figur 22 och 24 är den beräknade skillnaden för MongoDB och MySQL 21% vid insättning av stora dataposter.

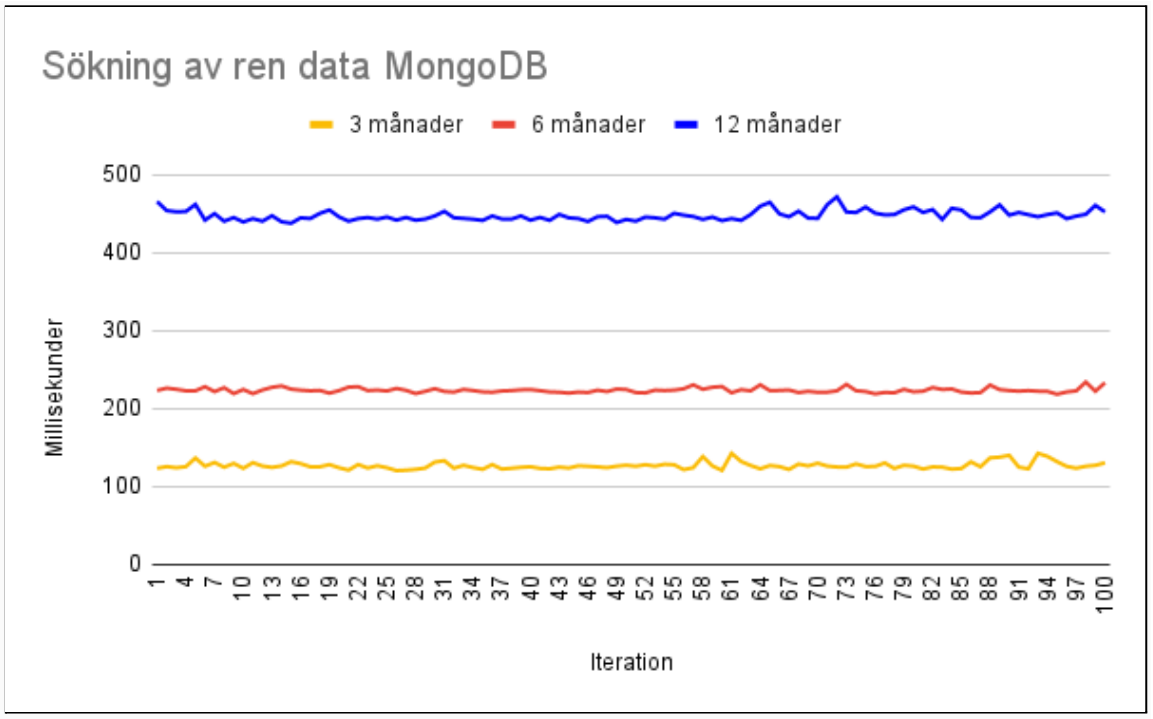
5.2.2 Mätserie 2 - Sökning av ren data

I Mätserie 2 mäts tiden genom att söka på olika datum för att få all information om just de datumet vilket inkluderar alla under rader för varje timme, varje datum som som söks är slumpgenererade.



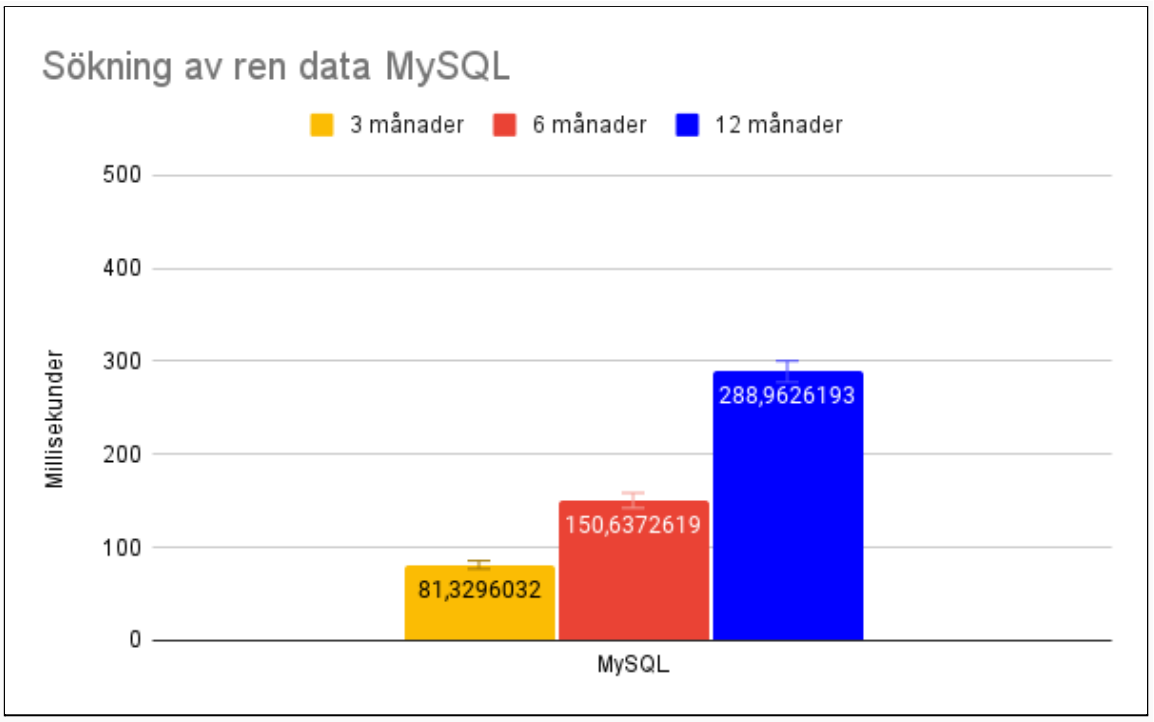
Figur 24 Mätserie 2 - Stapeldiagram för sökning av ren data MongoDB

Figur 24 presenteras stapeldiagrammet med resultatet för MongoDB vid sökning av ren data som påvisar en exponentiell tillväxt med den ökade datamängden med en liten större skillnad mellan 6 månader och 12 månader datamängden.



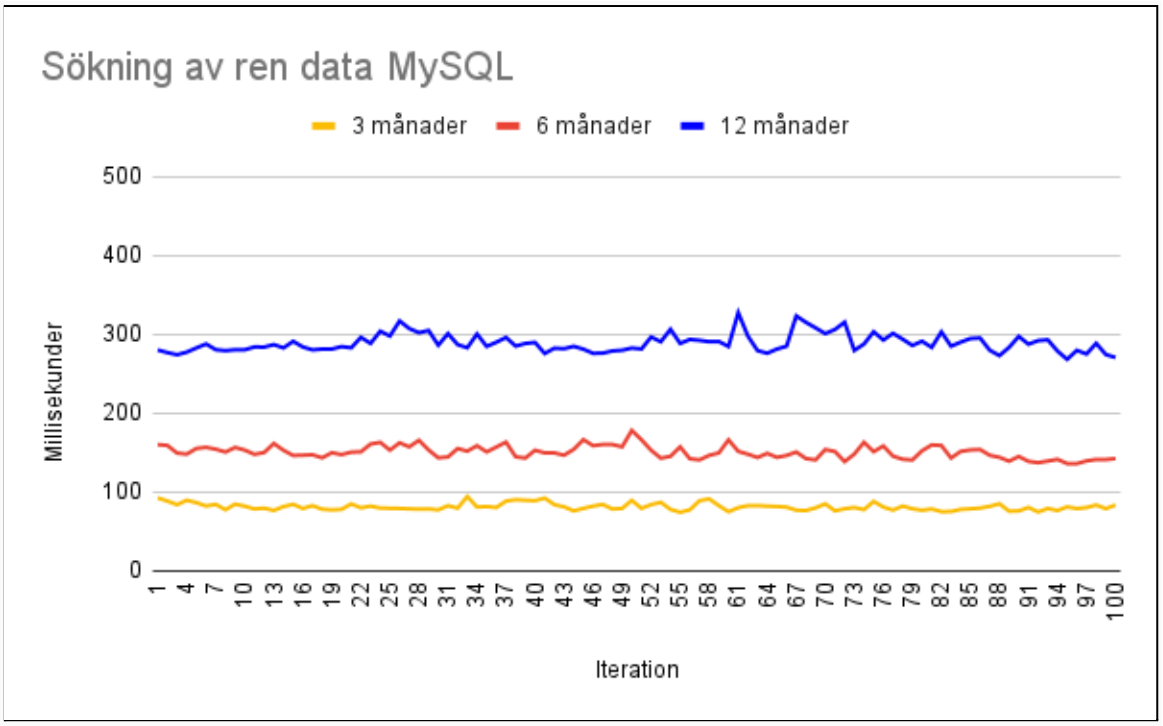
Figur 25 mätserie2 - Linjediagram för sökning av ren data MongoDB

Figur 25 presenteras linjediagrammet med resultatet för MongoDB vid sökning av ren data där varje datamängd påvisas ha gemensam stabilitet men med viss skillnad för 12 månader spannet jämförelsevis med 6 och 3 månaders datamängden.



Figur 26 Mätserie 2 -Stapeldiagram för sökning av ren data MySQL

Figur 26 presenteras stapeldiagrammet med resultatet för MySQL vid sökning av ren data som påvisar en exponentiell tillväxt med den ökade datamängden med en liten större skillnad mellan 6 månader och 12 månader datamängden men inte lika mycket skillnad som de var för MongoDB.



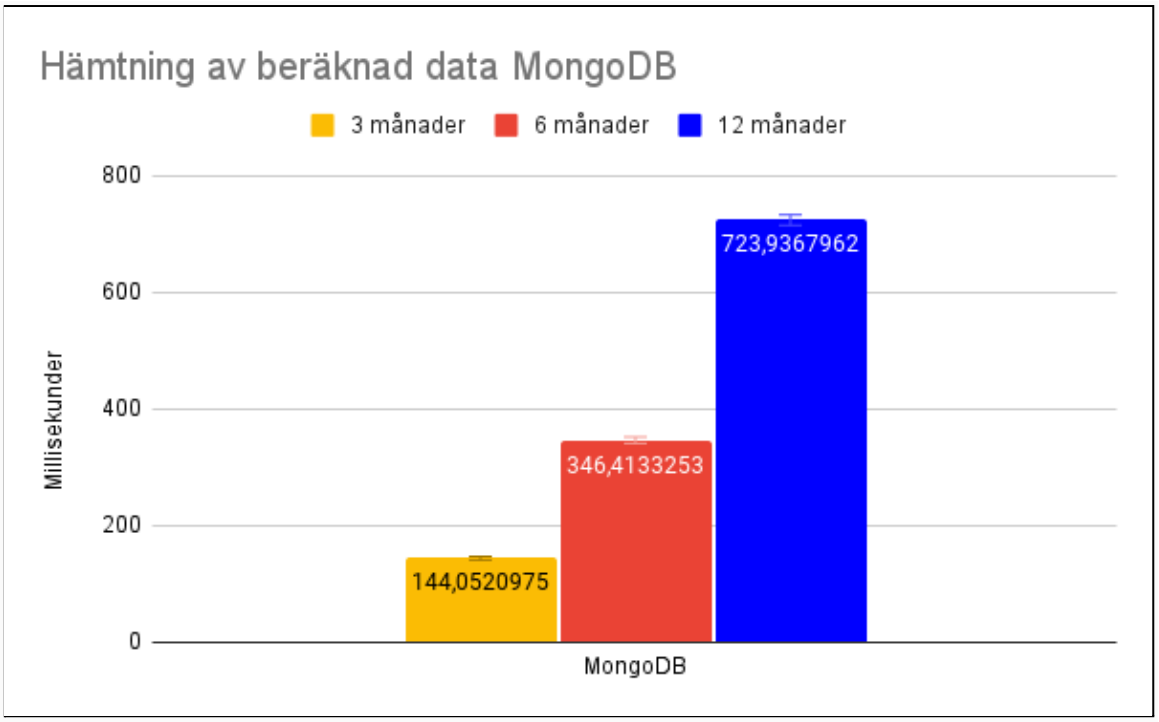
Figur 27 Mätserie 2 - Linjediagram för sökning av ren data MySQL

Figur 27 presenteras linjediagrammet med resultatet för MySQL vid sökning av ren data där varje datamängd inkluderar viss gemensam stabilitet men vid jämförelse med MongoDB linjediagram se Figur 25 där MySQL påvisar mindre stabilitet än MongoDB.

För denna mätserie presterar MySQL lite snabbare än MongoDB för varje datamängd vilket även blir tydligare vid 12 månaders datamängden där skillnaden är på 150ms se Figur 24 och 26. Något som är lite intressant är att båda databaserna ökar identiskt lika procentuellt i svarstider för varje datamängd. Med resultatet från Figur 24 och 26 är den beräknade skillnaden för MongoDB och MySQL 35% vid sökning av ren data.

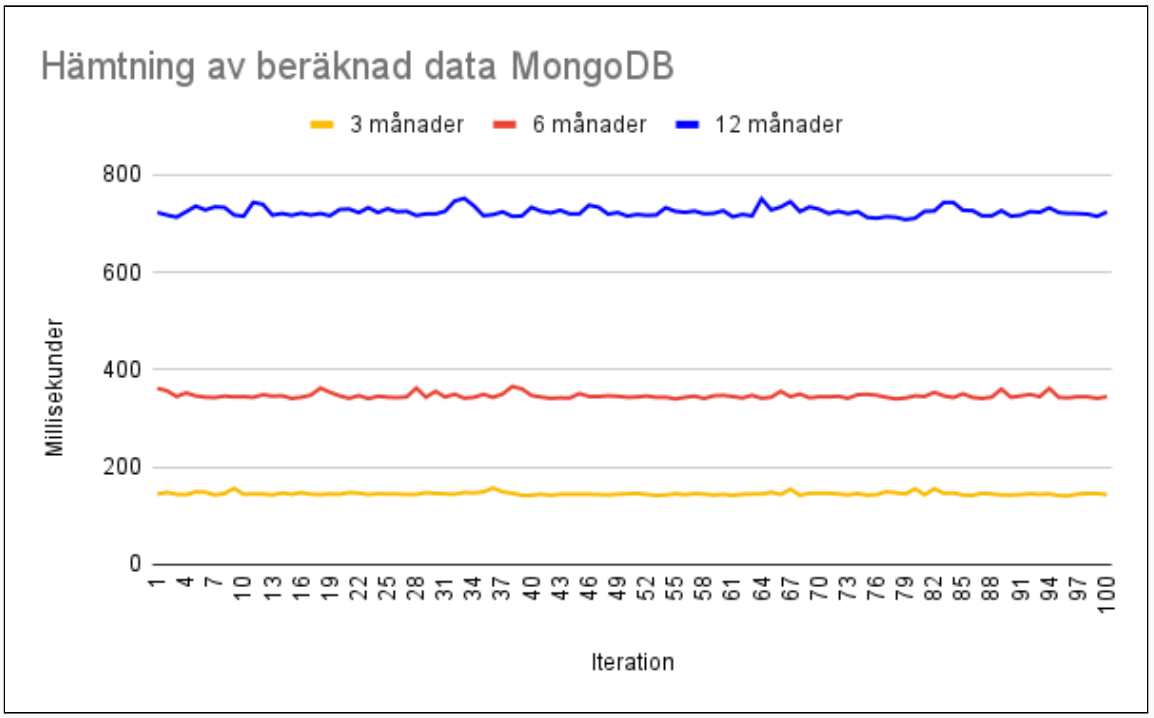
5.2.3 Mätserie 3 - Hämtning av beräknad data

I mätserie 3 sker en hämtning på elpris snittvärde på 100 lägsta elpris värden från respektive databas. Vilket innefattar sökning och hämtning av de 100 lägsta elpris värden, grupperar värden och till sist kalkylera dess snittvärde.



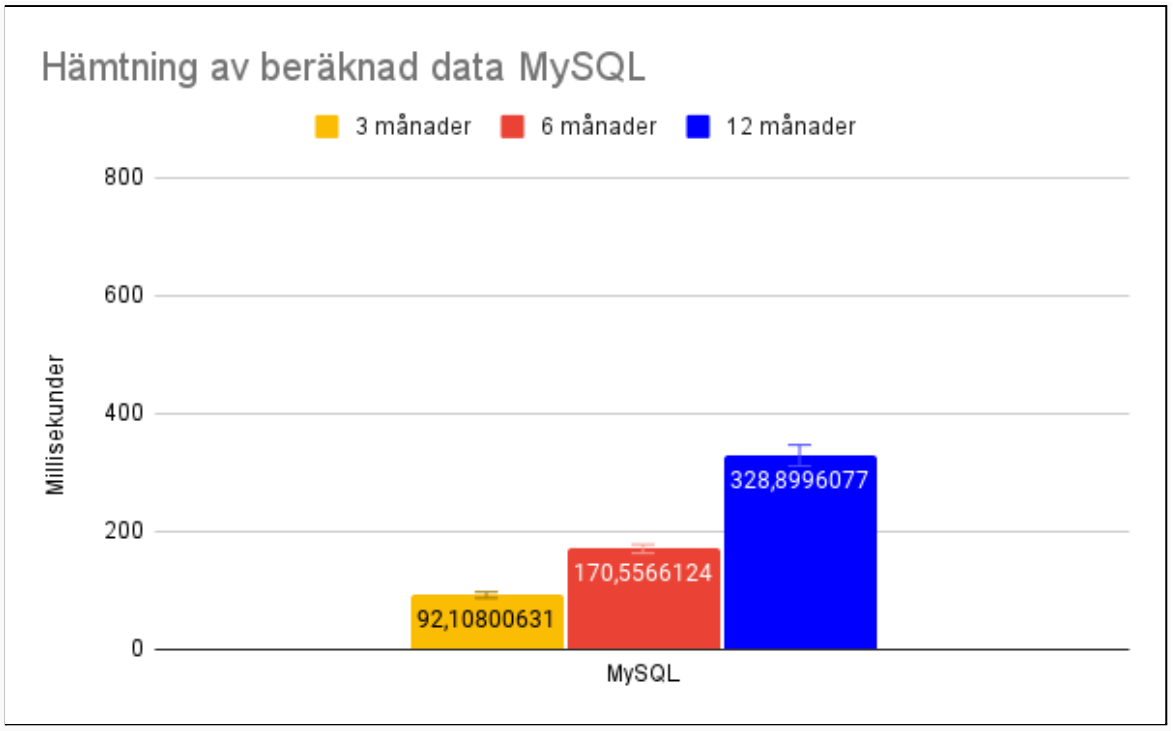
Figur 28 Mätserie 3 - Stapeldiagram för hämtning av beräknad data i MongoDB

Figur 28 presenteras stapeldiagrammet med resultatet för MongoDB vid hämtning av beräknad data som påvisar en jämn exponentiell tillväxt där varje ökad datamängd dubbleras utifrån tidigare datamängd.



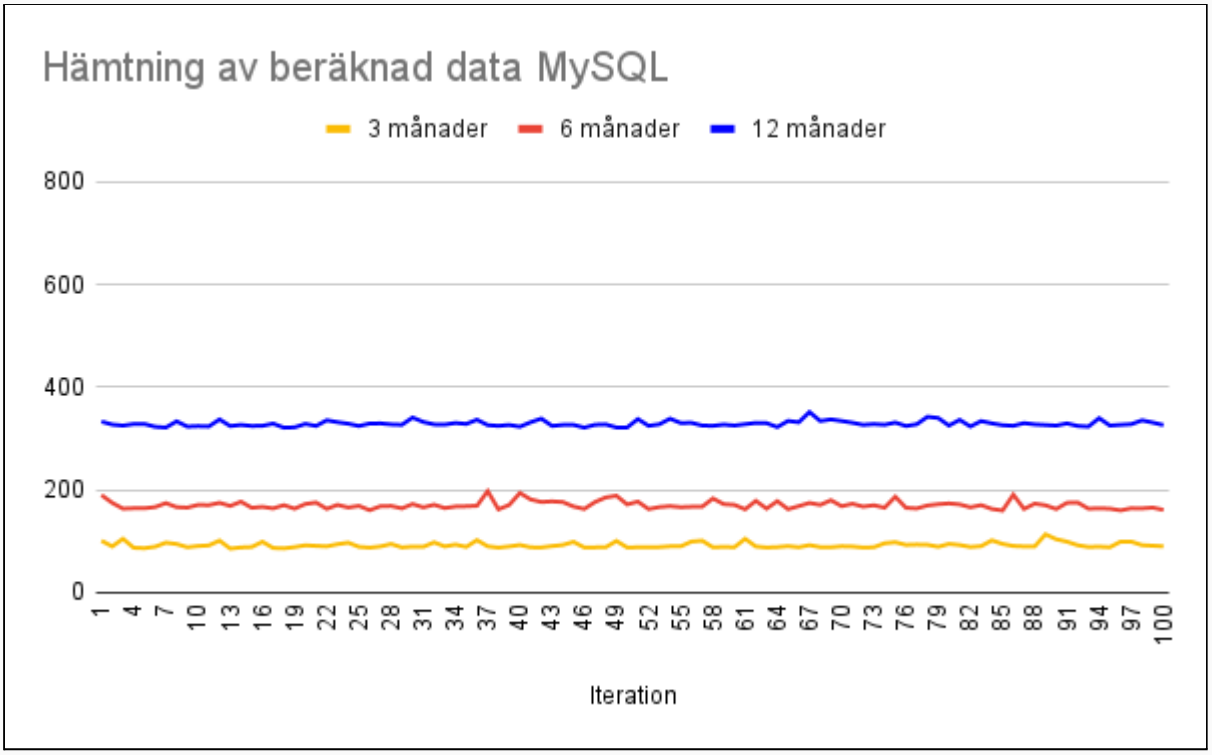
Figur 29 Mätserie 3 - Linjediagram för hämtning av beräknad data i MongoDB

Figur 29 presenteras linjediagrammet med resultatet för MongoDB vid hämtning av beräknad data där varje datamängd inkluderar viss gemensam stabilitet men med viss skillnad för 12 månader spannet jämförelsevis med 6 och 3 månaders datamängden.



Figur 30 Mätserie 3 - Stapeldiagram för hämtning av beräknad data i MySQL

Figur 30 presenteras stapeldiagrammet med resultatet för MySQL vid hämtning av beräknad data som påvisar en jämn exponentiell tillväxt och returnerar längre svarstider än MongoDB på varje datamängd.



Figur 31 Mätserie 3 - Linjediagram för hämtning av beräknad data i MySQL

Figur 31 presenteras linjediagrammet med resultatet för MySQL vid hämtning av beräknad data där varje datamängd inkluderar viss gemensam stabilitet men med ojämna svarstider jämförelsevis med resultatet från MongoDB

Vid hämtning av snittpris är det en viss skillnad på MongoDB och MySQL, där MySQL presterade i snitt snabbare än MongoDB när det gäller att hämta snittpris för varje delmoment. Skillnaden syns tydligast vid 12 månaders datamängden där skillnaden är på 395 ms se Figur 28 och 30. Med resultatet från Figur 28 och 30 är den beräknade skillnaden på för MongoDB och MySQL 51% vid hämtning av beräknad data.

5.2.4 Presentation av alla svarstider

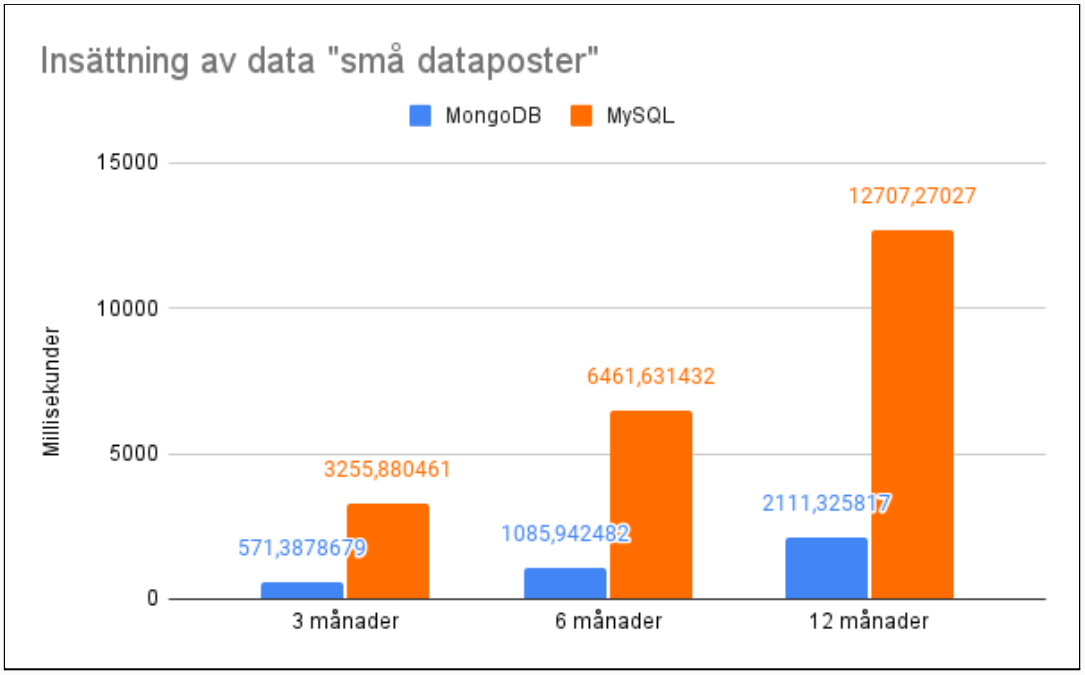
Under kapitel presenteras alla svarstider som nyttjades för att presentera tidigare Figurer. Utöver svarstider presenteras även vilken datamängd svarstiden representerar, vilken databas och även vilken av databaserna som erhöll lägst svarstider under experimenten.

Mätserie 1 (Små dataposter)				**Lägst svarstid**
	3 månader	6 månader	12 månader	MongoDB
MongoDB	571,4 ms	1085,9 ms	2111,3 ms	
MySQL	3255,9 ms	6461,6 ms	12707,3 ms	
Mätserie 1 (Stora dataposter)				**Lägst svarstid**
	3 månader	6 månader	12 månader	MongoDB
MongoDB	31,5 ms	63,5 ms	127,6 ms	
MySQL	42,5 ms	80,8 ms	158,6 ms	
Mätserie 2 (Sökning av ren data)				**Lägst svarstid**
	3 månader	6 månader	12 månader	MySQL
MongoDB	127,1 ms	223,6 ms	447,9 ms	
MySQL	81,3 ms	150,6 ms	288,9 ms	
Mätserie 3 (Hämtning av beräknad data)				**Lägst svarstid**
	3 månader	6 månader	12 månader	MySQL
MongoDB	144,1 ms	346,4 ms	723,9 ms	
MySQL	92,1 ms	170,5 ms	328,9 ms	

Tabell 4: Presentation av alla svarstider

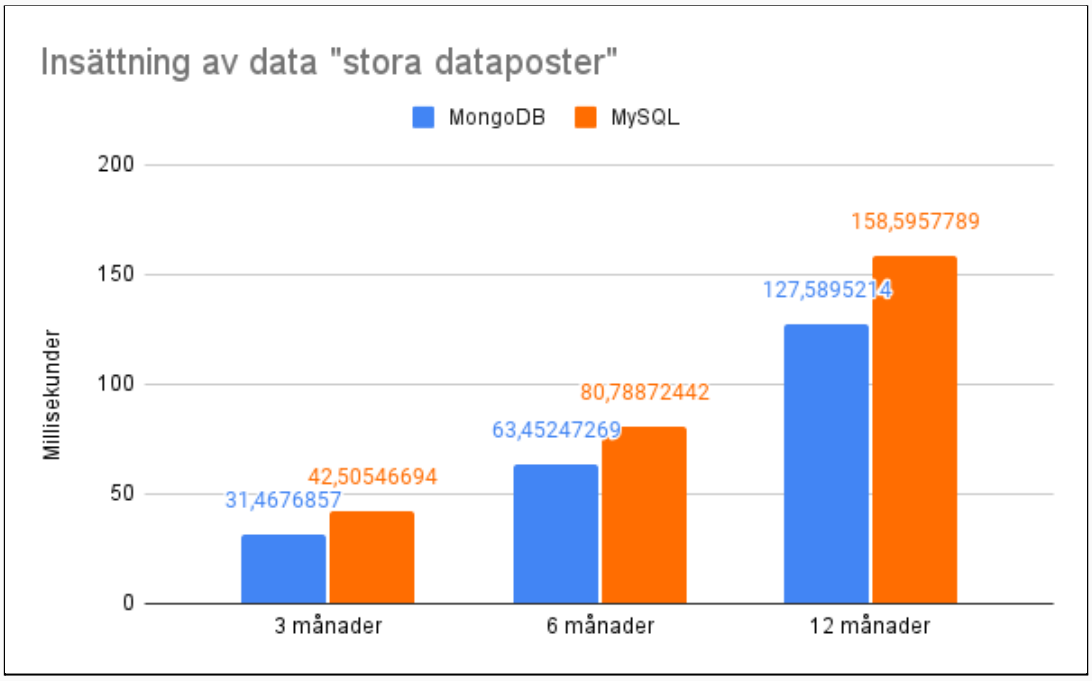
5.3 Analys

I analysen kommer resultatet från alla mätserier att jämföras för att se vilken databas erhållit lägst svarstider. Resultatet kommer kombinera svarstiderna från båda databaserna för att visa helhet av mätningarna som har gjorts.



Figur 32 Mätserie 1 - Insättning av data för små dataposter

I Figur 32 presenteras resultatet från mätserie 1 vid små dataposter där MongoDB presterade i snitt snabbare än MySQL vid insättning av små dataposter.

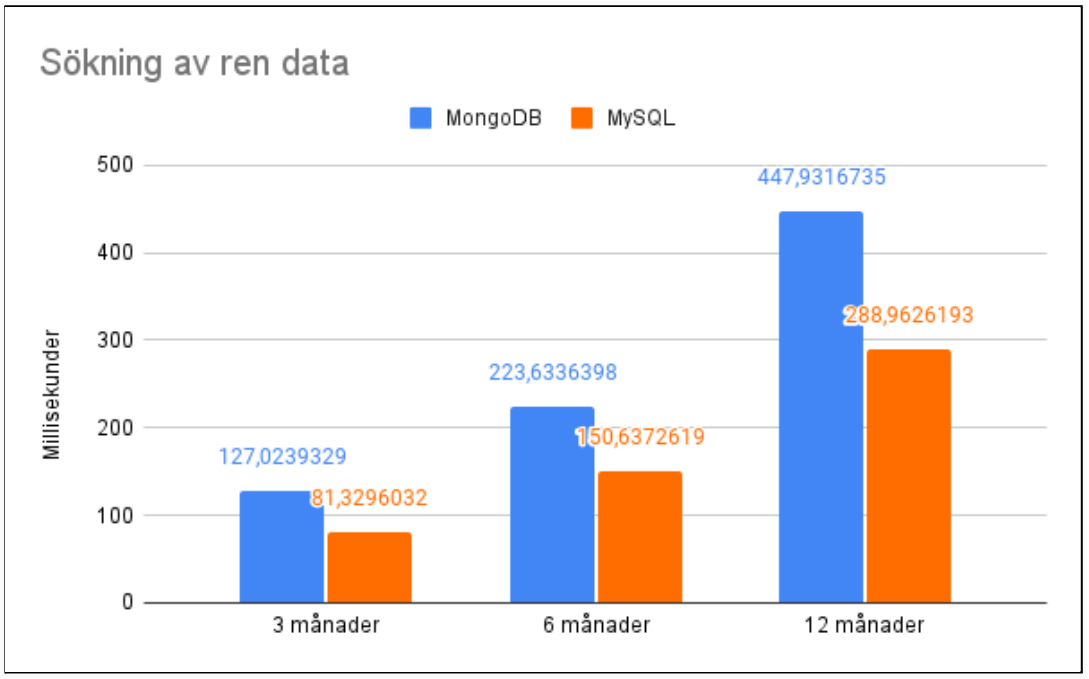


Figur 33 Mätserie 1 - Insättning av data för stora dataposter

Figur 33 presenteras resultatet från mätserie 1 vid stora dataposter. Från resultatet kan vi se att MongoDB presterade med lägre svarstider än MySQL vid insättning av stora dataposter. Resultatet visar på att båda databaserna verkar hålla en jämn prestanda. Olika faktorer kan spela roll när det gäller för både prestanda och svarshastighet men en av faktorerna som spelar roll för MongoDB's låga svarstider är att insättning av data via JSON data format visar sig vara mer effektivare än att data i tabeller vid insättning.

Vid insättning av stora dataposter erhålls inte bara lägre svarstider utan mer jämnare exponentiell tillväxt vid varje datamängd hos båda databaserna. Vid jämförelse på resultatet för MySQL i Figur 32 och 33 dras slutsatsen om att varje insättning av små dataposter tar en hel del processorkraft eftersom att svarstiderna i Figur 34 är långa. Alternativa förklaringar till svarstiderna skulle kunna vara att data strukturen för vardera experiment inte är densamma eftersom att insättning av små dataposter inkluderar flera små dokument och insättning av stora dataposter inkluderar ett stort dokument. Viktigt att notera att både Figur 32 och 33 har olika upplösningar på x axeln, även om båda ser ut som att de har samma stapel värden är det på grund av olika upplösningar på x axeln.

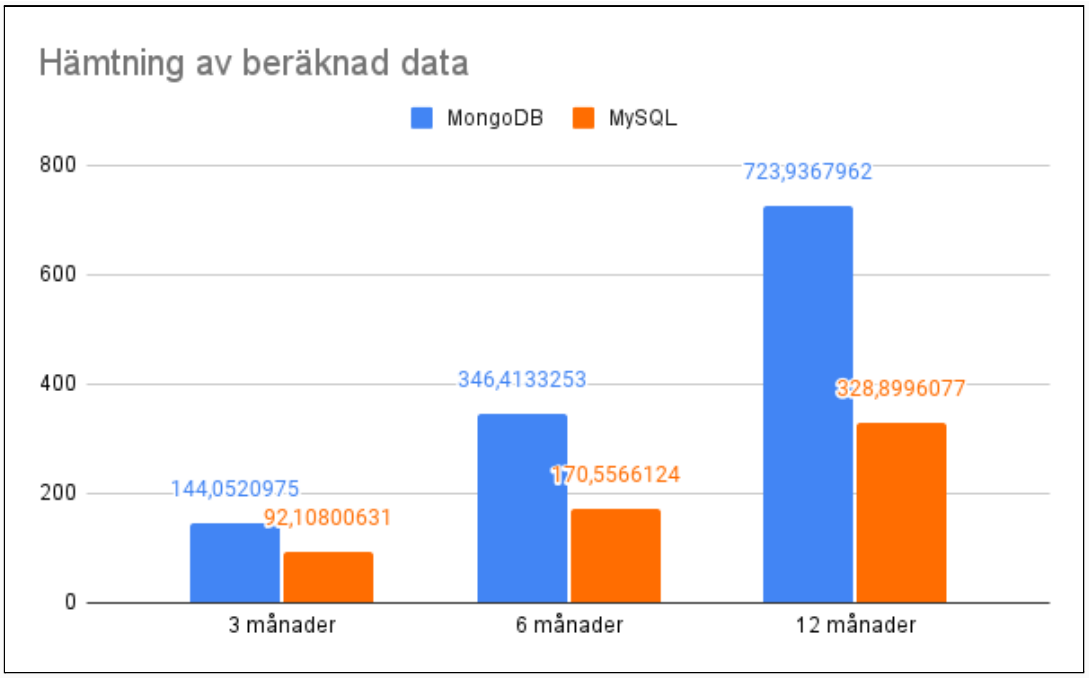
Utifrån mätserie 1 kan vi då säga att vid insättning av data är MongoDB är de bättre valet för ett Smart grid system.



Figur 34 Mätserie 2 - Sökning av ren data

I mätserie 2 presterade MySQL snabbare än MongoDB och skillnaden blir mer tydligare när datamängden ökar se i Figur 34. Anledningen till att MySQL är snabbare kan bero på att indexeringen för varje datapost inte utgick efter datumen utan efter själva objekt-id nyckeln för mongodb. På grund av att MySQL har satt att datumen är indexerat kan det vara att MongoDB behöver gå igenom varje rad för rad eftersom att indexeringen inte var satt på datumet utan hela dokumentet. Utifrån mätserien drar slutsstasen om att MySQL är bättre

valet som databassystem för ett Smart grid system då skillnaden mellan MySQL och MongoDB kombinerade svarstiden var på 35%.



Figur 35 Mätserie 3 - Hämtning av beräknad data

Under mätserie 3 kan vi se att MySQL presterade med snabbare svarstider än MongoDB under alla operationerna med en skillnad på 51% på svarstiderna, med en jämn exponentiell tillväxt för varje datamängd se i Figur 35. Utöver det ökade svarstiderna med datamängden som tidigare mätserie med sökning på ren data där MySQL presterade i snitt snabbare än MongoDB. Anledningen till att MongoDB är långsammare kan bero på att MySQL använder sig av sql statements för att utföra själva grupperingen av datamängden samt själva snitt operationen under samma statement. Till skillnad från MongoDB som nyttjar aggregation där varje operation sker individuellt, vilket kan leda till tyngre processer för att utföra data grupperingen och beräkningen.

5.4 Slutsatser

Frågeställningen för denna studie var:

"Om en dokumentdatabas representerad av MongoDB kan få kortare svarstider än en traditionell relationsdatabas MySQL vid insättning av ren data, hämtningar av ren data och hämtning beräknad data."

och studiens hypotes var:

"För denna undersökning kommer MongoDB att prestera med lägst svarstider."

Utifrån de resultat som har presenterats dras slutsatsen om att den etablerade hypotesen motbevisas. MongoDB påvisade svårigheter i två experiment av tre i svarstider, vilket syns mest under mätserie 3 där skillnaden mellan MongoDB och MySQL svarstider låg på 51%.

Vid insättning av små dataposter håller MongoDB en jämn prestanda som alltid är snabbare än MySQL. MongoDB har en genomsnittlig insättningstid på 1256,2 ms till skillnad från MySQL som har en genomsnitt insättningstid på 7474,9 ms, utifrån slutresultatet är den beräknade skillnaden mellan databasernas svarstider 83.1%

Vid insättning av stora dataposter som i tidigare del experiment håller MongoDB en jämn prestanda som alltid är snabbare än MySQL förhållande till svarstider. Eftersom att experimentet mäter svarstider vid insättning av stora dataposter hade både MongoDB och MySQL i snitt lägre svarstider jämförelsevis med små insättningar. MongoDB har en genomsnitt insättningstid på 74,2 ms till skillnad från MySQL som har en genomsnitt insättningstid på 93,9 ms, utifrån slutresultatet är den beräknade skillnaden mellan databasernas svarstider 21%. Någonting att ha till hänsyn när det gäller för insättning av både små och stora dataposter är att vardera experiment har annorlunda struktur på data vilket kan resultera i skillnaderna mellan experimenten.

Mätserie 2 vilket inkluderade sökning och hämtning av ren data där MySQL presterade med lägst svarstider genom alla datamängderna. Slutsatserna som dras från resultaten är att både MongoDB och MySQL vidhåller en jämn nivå i prestanda och en jämn tillväxt i svarstider i samband till den ökade datamängden. I snitt presterade MySQL med 173,6 ms parallellt med MongoDB som presterade i snitt på 266,2 ms vilket resulterar i att skillnaden mellan databaserna är på 35% med den kombinerade svarstiderna

Till sist för mätserie 3 vilket inkluderade hämtning av beräknad data där MySQL presterade med lägst svarstider genom alla datamängder och operationer. Eftersom att tidigare experiment som innefattade beräknad data resulterade i varierande, instabil och mer oförutsägbara svarstider och erhöll ingen exponentiell tillväxt med den tillökade datamängden som de andra experiment under denna studie. Vilket resulterade i att experimentet ändrades för att var mer mätbart och värt att undersöka. Precis som mätserie 2 presterade MySQL med lägst svarstider med ett snitt 197,2 ms på jämförelsevis med MongoDB med 404,8 ms i snitt vilket blir 51% skillnad på svarstider.

Med resultatet från alla mätserier presterar MySQL i snitt med lägre svarstider än MongoDB till förhållande till den valda datamängden och till den mätserien som utfördes. Med resultatet besvaras även frågeställningen, utifrån resultaten är MySQL mer effektiv än

MongoDB vid interaktionen av data för ett Smart Grid system. Även fast skillnaden på svarstiderna mellan varje mätserie inte var över marginellt är skillnaden tillräckligt stort för att se den mer passande databasen för ett Smart Grid system.

6 Avslutande diskussion

Kapitlet går igenom avslutande diskussioner, tankar och ideer som har dykt upp under denna studiens gång och även öppna upp andra ideer för framtida arbeten eller eventuell fortsättning för att bygga vidare på denna studie.

6.1 Sammanfattning

Under vintern 2021 genomfördes en prognos på sveriges elpriser där en jämförelse utfördes med december 2021 månads medelpriser för samtliga elområden slog rekord i högsta prisnivå sedan sverige delades in i elområden. Efter denna chock fick svenska medborga själva ta egna initiativ för att hålla nere elkonsumtionen för att förhindra allt för höga elräkningar. Då det finns många olika sätt för att dra ner elkonsumtionen finns det även tekniker och enheter som är konstruerade för att förhindra och dra ner på elkonsumtionen som exempelvis Smart Grid. Denna framtidsvision teknik är en tillämpning på hela vårt elnät för att utsluta gamla tekniker och teorier som vårt elnät är uppbyggd på. Men eftersom att denna teknik är en framtidsvision är den inte helt implementerad i vårt samhälle men smarta enheter som tvättmaskiner, diskmaskiner och värmepannor är någonting som har implementerat i många av våra hushåll. Dessa enheter har direkt kommunikation med ett smart grid control center och med hjälp av ett databassystem kan kommunicera och koordinera smart enheterna att köras när elpriserna är låga. Men för att denna lösning ska kunna uppfylla långtids kraven behövs ett databassystem som hanteras elpris datan med låga svarstider som möjligt då varje millisekund och sekund spelar roll.

Det första steget för att hitta den optimala databasen för ett smart grid system är att jämföra två olika databaser , en SQL databas mot en NoSQL databas, vilket är det studien undersöker. Den frågeställning som studien undersöker är *“Om en dokumentdatabas representerad av MongoDB kan få kortare svarstider än en traditionell relationsdatabas MySQL vid insättning av ren data, hämtningar av ren data och hämtning beräknad elpris data”*.

Under studien utförs experiment för att svara på frågeställningen. MySQL och MongoDB installerades på en server och sedan sker olika mätserier för att mäta deras prestanda och svarstider. Första mätserien undersöker svarstiderna vid insättning av data, både många små insättningar och en stor insättning av data mäts. Datan som genereras baseras på NordPoolGroup egna elpis tablå, denna data nyttjas i varje databas och för varje experiment ökas datamängden för att se om det blir någon skillnad i prestanda. Mätserien 2 undersöker svarstiderna vid sökning av ren data då varje sökterm är datum vilket inkluderar all data under 24 timmar spannet för just de datumet. Den sista mätserien undersöker svarstiderna vid hämtning av beräknad data där 100 lägsta elpriserna grupperas för att sedan kalkylera dess snittvärde.

Det resultat som studien tagit fram, och presenteras i kapitel 5, visar att MySQL är det bättre valet av databas för ett smart grid system. MySQL presterar bättre vid hämtning av ren och beräknad data men MongoDB presterade bättre än MySQL vid båda del experimenten av insättning av data.

6.2 Diskussion

Denna studie undersöker vilken av två databaser som är mest lämpad för ett Smart Grid system. Vid hantering av databaser ligger prestanda och låga svarstider i fokus vilket gör en kvantitativ undersökningsmetod de bättre alternativet för att undersöka och jämföra databaser. Kvantitativ undersökningsmetod nyttjas kontrollerade experiment för att få fram resultat i form av siffror och för detta fallet inkluderas svarstider som reslutat. Experimenten i studien sker på en sluten miljö i form av virtuella maskiner vilket minimerar eventuella fel faktorer som kan påverka slutresultatet vid återupprepning Wohlin, C (2012). Med experiment är det viktigt att ha i åtanke alla olika faktorer kan påverka slutersultatet antingen till de negativa eller till de positiva. Vid jämförelse av slutresultaten från ett experiment returneras oftas ett klart resultat, men om skillnaden är endast ett några millisekunder måste man fråga sig själv om det gör någon nytta eller är någon skillnad Wohlin, C (2012).

Intressanta aspekter att ta ifrån denna studie är hur data inmatningsmetoden påverkar svarstiderna för vardera databas. Från mätserie 1 där vi mäter svarstiderna vid insättning av data för MongoDB och MySQL resulterade i att svarstiderna var långsammare än resterande mätserier. Under experiment tillfället togs beslutet att en utökning på experimentet var nödvändigt då det skulle vara intressant för att se hur data inmatningsmetoden påverkar slutresultatet, vilket resulterade i skapandet av experimentet med insättning av stora dataposter eftersom att första data insättnings experimentet enbart utförde insättning av flera små dataposter. Vad som upptäcktes var att både MongoDB och MySQL svarstider påverkades positivt när data inmatningsmetoden involverade en stor insättning av data jämförelsevis med flera små insättningar. Någonting att ta till hänsyn för mätserie 1 är att en möjlig faktor till att svarstiderna varierar från varje insättningsmetod skulle kunna vara att vid insättning av små dataposter skapas flera små dokument jämförelsevis med insättning av stora dataposter där ett helt stort dokument skapas. Eftersom att vardera inmatningsmetod genererar olika antal dokument betyder det då att datastrukturen inte är densamma för varje mätserie.

Studien av Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) där vissa av dess experiment påminner om experimenten under denna studie men eftersom att Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) studie innefattade mindre komplexa experiment för att kompensera med studiens större tabellstruktur resulterade slutvärderna lite annorlunda än denna studie. Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) studie inkluderar 5 tabeller och dokument jämförelsevis med denna studie då enbart en tabell och dokument visar tydliga skillnader vid insert experimentet där MongoDB presterade betydligt snabbare än MySQL med 440 sekunder och 0,29 sekunder för MongoDB sammanlagd tid vid insättning. I denna studie var skillnaden vid insättning av data (1256,21 ms för MongoDB och 7474,92 ms för MySQL vid små insättningar) och sedan (74,16 ms för MongoDB och 93,96 ms för MySQL vid stora insättningar). Utifrån resultatet från denna studie som returnerade varierande resultat beroende på hur datan matas in och förhållandevis med Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) studie påvisar skillnaden på svarstider blir större med ökad komplexitet.

Databaserna som använts i denna studie representerar en liten del av ett komplext system, om studien byggs vidare, ökar komplexiteten hos tabellerna, även inkludera mer avancerade beräknar och förfrågningar mot databaserna kan resultatet påverkas. Viktig sak att ha i åtanke vid analys av slutresultatet från denna studie för att avgöra vilket av databaserna är de bättre alternativet. En ökad komplexitet och eller fördjupning inom olika aspekter på experimenten ökar antalet faktorer som kan påverka slutresultatet.

Det man kan ta ifrån denna studie är hur respektive databas presterar vid experiment genomgångarna och vad man bör tänka på vid implementering av den rätta databasen för ett Smart grid system. Först och främst ska man ha till hänsyn att vid insättning av flera små dataposter påverkas svarstiderna till det negativa för bägge databaserna. Specifikt för MySQL där skillnaderna på svarstiderna inte riktigt är optimalt förhållandevis med resterande av mätserierna där räckvidden på svarstiderna är upp till 800 ms. Eftersom att elbörs hemsidor som NordPool Group uppdaterar sin hemsida regelbundet med ny data måste man se över olika alternativ för att se till att svarstiderna inte blir för långa men ändå uppdatera hemsidan regelbundet. Med hänsyn till mätserien som inkluderar insättning av större dataposter skulle en optimal lösning vara att mata in medelstora dataposter för att inkludera större datamängder men ändå uppdatera och mata in ny information regelbundet som möjligt utan större tidsrum mellan insättningarna. Genom denna lösning erhålls låga svarstider men även en successiv insättning av data istället för att vänta på att större dokument måste genereras för att till slut matas in i databasen.

Sammanfattat är samhällsnyttan med denna studie erhålls vissa för- och nackdelar med databaserna men att man sedan ska kunna avgöra och fatta beslut om vilken av databaserna som är mest lämplig.

6.2.1 Etik

Eftersom att både hård- och mjukvara samt all programkod publiceras i denna rapport men även på github som används utger möjligheten att upprepa denna studie ur ett forskningsetiskt perspektiv. Det ger även möjligheten att fortsätta och förbättra studien till att antingen fokusera på andra områden förhållandevis till temat men även finslipa på eventuella felaktigheter. Dock kan det orsaka skapandet av andra etniska problem eftersom att denna studie inte kan garantera samma resultat vid tillämpning av t.ex annan version av mjukvara, variation av hårdvaran som minneskapacitet och liknande. Men alternativa tillvägagångssätt är någonting som uppmuntras då det kan vara intressant och se ifall nya version av mjukvara eller annan form av hårdvara kan påverka slutresultatet.

Något som är viktigt att ha i åtanke är att denna studie nyttjar datamängd upp till 8760 dataposter vilket representerar 12 månaders datamängd. Denna studie representerar en liten del av ett Smart grid lösning bör en till ökad datamängd vara mer representativ för att sätta ett mer fast beslut om vilket databassystem bör mest lämpad. Även om resultatet som erhålls från denna studie påvisar att MySQL bör prestera bättre än MongoDB är det ingen garanti att resultatet blir detsamma i fullt utvecklad applikation som går att jämföra med ett helt Smart grid system. Andra grejer att ha i åtanke är att MongoDB haft en historia med säkerhetsbrister vilket inte är förvånande eftersom att MongoDB släpptes för första gången 2009, förhållande till MySQL är MongoDB en relativ ny databas. Dock har MongoDB utvecklats genom åren men till skillnad från MySQL som har särskilda inbyggt

säkerhetsskydd bör man lägga lite tid på att kontrollera MongoDB säkerhet vid lagring av eventuell känslig data.

Denna studie baserar sig ifrån att hitta den mest lämpade databasen för ett smart grid system vid hantering av elpris data där själva datasetet har tagit inspiration av nord pool groups egna elpris tablå. Men eftersom att denna studie fokuserar på att testa vardera databassystem via experiment då betyder det inte att slutresultatet inte kan nyttjas mot någonting annat. Exempelvis skulle slutresultatet användas till beslutet om vilken databas som skulle vara lämpad för ett sjukhus journalsystem. För precis som för ett smart grid system är det viktigt att ha låga svarstider vid hämtning eller sökning på viktiga sjukhusdata då det är människors liv som står på spel vilket innebär att varje sekund räknas.

Vid diskussion om teknisk hållbarhet är det värt att nämna att för vidare tillämpning av MongoDB och MySQL håller ena databasen en fördel jämfört med den andra. Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015) skriver i sin artikel att relationsdatabaser som MySQL har varit ett bra val för prestanda när det gäller för limiterad datamängd men när det gäller för större mängder datahantering har de visat sig ineffektivt. Om man skulle basera beslutet om att använda den databasen utifrån slutresultatet från denna studie måste man ha i åtanke att en långsiktig tillämpning av en MySQL databas kan orsaka problem, däremot har MongoDB påvisat sig själv vara mer effektiv vid högre datamängds belastning.

6.3 Framtida arbete

Syftet med denna undersökning var att hitta den mest lämpade databasen för en Smart Grid lösning med fokus på att jämföra MongoDB och MySQL. Resultatet som från denna studie pekar mest emot att MySQL skulle vara den mest lämpad men eftersom att det inte finns någon garanti på att SQL databas är alltid mest lämpad då det finns många olika databasen som kanske skulle prestera bättre och effektivare. En alternativ studie skulle kunna vara att jämföra en annan SQL och NoSQL databas som PostgreSQL och Couchbase. Alternativa tekniker som skulle kunna bytas ut är en annan webbläsare och webbserver för att se om det blir någon skillnad i svarstider och även se om det finns någon lämpligare teknik i den fronten.

Andra tillvägagångssätt skulle kunna vara att istället för att utföra all kod med PHP för både MySQL och MongoDB skulle det vara intressant och se om det blir någon skillnad och utföra utformningen av MongoDB databasen med Javascript. Eftersom att MongoDB skrivs naturligt sätt med javascript kanske det finns någon prestanda vinst genom att utforma MongoDB databasen via javascript. Istället för att installera MongoDB PHP drivrutiner för att utöka möjligheten att skriva allt i PHP finns det en chans att experimenten returnerar annorlunda resultat.

Den intressanta aspekten från denna studie som togs upp under kapitel 6.2 var hur annorlunda data inmatningsmetod påverkar svarstiderna. Som tidigare nämn skulle en anledning till den varierade svarstiderna vara att datastrukturen på vardera inmatningsmetod inte är densamma men det betyder då inte att det inte finns flera intressanta aspekter att undersöka kring alternativa inmatningsmetoder påverkar svarstiderna hos databaserna. Någonting värt att notera för denna studie är att varje experiment tillfället utförs enbart en koppling per mätserie vilket innebär att för varje

mätserie skapas bara en kopplingen till databasen. Utifrån de perspektivet är det värt att notera att experimenten inte vidhåller en verklig simuleringen då det sällan sker en koppling till databaser för att utföra flera hundratals operationer i verkligheten. Inför en framtida utveckling på denna studie skulle det vara värt att modifiera experimenten att det sker en koppling till databasen per experiment genomgång för addera någon form av verklig simulering.

I framtiden lär det dyka upp andra intressanta aspekter att undersöka då nya tekniker och metoder utvecklas hela tiden som kan göra denna studie bättre.

Referenser/References

Győrödi, C & Gyorodi, R & Pecherle, G & Olah, A. (2015). *A comparative study: MongoDB vs. MySQL*. 2015 13th International Conference on Engineering of Modern Electric Systems (EMES). 1-6.

Vaish, G. (2013). *Getting Started with NoSQL*.

Hows, D & Membrey, P & Plugge, E & Hawkins, T. (2015). *The Definitive Guide to MongoDB*. 10.1007/978-1-4842-1182-3.

Converse, T & Park, J & Morgan, C. (2004) *PHP5 and MySQL Bible*: Wiley Publishing inc.

Abramova, V. & Bernardino, J. (2013). *NoSQL databases: MongoDB vs cassandra*. Proceedings of the International Conference on Computer Science and Software Engineering, s. 14-22.

Codd, E. F. (1970). *A relational model of data for large shared data banks*. Communications of the ACM Volume 13 Issue 6, s. 377-387.

Melton, J & Simon, A. (1993). *Understanding the new SQL: A Complete Guide*. Burlington: Morgan Kaufmann Publishers. s. 4-8.

Prajapati, J. K & Ghadiali, S. & Vora, D. R. (2012). *Smart grid -A vision for the future*. IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012), 2012, pp. s. 672-677.

Wohlin, C & Runeson, P & Höst, M & Ohlsson, M & Regnell, B & Wesslén, A. (2012). *Experimentation in Software Engineering*. 10.1007/978-3-642-29044-2_10.

Office of Electricity (2011) *The Smart Grid An Introduction*. s. 5-12

K. I. Satoto, R. R. Isnanto, R. Kridalukmana and K. T. Martono. (2016). *Optimizing MySQL database system on information systems research, publications and community service*. 3rd International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE), 2016, pp. 1-5, doi: 10.1109/ICITACEE.2016.7892476.

M. Lei & Z. Feng. (2009). *A novel grey model to short-term electricity price forecasting for NordPool power market*, IEEE International Conference on Systems, Man and Cybernetics, 2009, pp. 4347-4352, doi: 10.1109/ICSMC.2009.5346948.

J. Liu, X. Li, D. Liu, H. Liu & P. Mao. (2011). *Study on Data Management of Fundamental Model in Control Center for Smart Grid Operation*. IEEE Transactions on Smart Grid, vol. 2, no. 4, pp. 573-579, Dec. 2011, doi: 10.1109/TSG.2011.2160571.

M. Schapranow, R. Kühne, A. Zeier & H. Plattner.(2014). *Enabling real-time charging for smart grid infrastructures using in-memory databases*. IEEE Local Computer Network Conference, 2010, pp. 1040-1045, doi: 10.1109/LCN.2010.5735677.

Tahaghoghi, S.M. and Williams, H.E., (2006), *Learning MySQL: Get a Handle on Your Data, O'Reilly Media*, Inc.

Dalibor, D Dvorski. (2007). *INSTALLING, CONFIGURING, AND DEVELOPING WITH XAMPP*. Skills Canada, Ontario.

Energimarknadsbyrå. 2022. *Elpriser - prognos och utveckling* https://www.energimarknadsbyran.se/el/dina-avtal-och-kostnader/elpriser-statistik/elpriser-prognos-och-utveckling/ (Hämtad 2022-02-22)

Regeringskansliet. 2022. *Kompensation för höga elpriser* https://www.regeringen.se/pressmeddelanden/2022/01/kompensation-for-hoga-elpriser/ (Hämtad 2022-02-22)

NordPoolGroup. 2022. *About NordPoolGroup* https://www.nordpoolgroup.com/About-us/ (Hämtad 2022-02-22)

MongoDB docs PHP (2022) https://www.mongodb.com/docs/drivers/php/

MongoDB docs (2022) https://www.mongodb.com/docs/manual/core/document/

MySQL docs (2022) https://dev.mysql.com/doc/refman/8.0/en/

The PHP Group (2022) https://www.php.net/manual/en/set.mongodb.php