SQL or NoSQL: Comparing Modern Databases

Ryan McArthur

Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA

19 November 2016
Morris, MN

Questions to consider

- What exactly are NoSQL databases?
- Why should we care about them?
- Are they really better than RDBMS?
- Why is there so much hype around them?

Why should you care?

- Interactions with databases happen everyday
- Data is growing at an exponential rate and projected to double every 2 years. In 2013 we had 4.4 zettabytes. (44 zettabytes projected by 2020) [1]
- One zettabyte is about one billion terabytes
- Affects how quick applications are
- Database selection can be crucial to a project's success

Outline

Outline

What are RDBMS?

- Relational Database Management Systems (RDBMS)
- Based on relational model introduced by E.F. Codd in 1971
- Structured Query Language (SQL) is based on this model
- SQL databases are currently used in almost everything
- A few SQL databases today are MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access

What is NoSQL

- Stands for Not Only SQL, Non SQL, or non relational
- Very broad term
- Databases of this form have existed since the 1960's
- The name is coined after SQL which is based off a model from 1971 but they have existed since 1960
- First noting of NoSQL is from 1998 but popularized in mid 2000's
- An alternative to Relational Database Management Systems (RDBMS)

What is NoSQL cont.

- There are many data models that all fall within NoSQL
  - Key-Value Store - Treat the data as a single opaque collection which may have different fields for every record, hash storage (Riak)
  - Document Databases - Similar to key-value, stored as a documents, embeds metadata allowing for query based on contents (MongoDB)
  - Column Databases - Stored data in grouped columns instead of rows like a RDBMS (Cassandra)
- These are three different approaches to handling data all with different benefits and downsides
- A different set of rules for databases, often seen as less strict

ACID - SQL

Information processing that is divided into individual, indivisible operations are called transactions

- **A**tomicity - Requires that each transaction be "all or nothing": if one part of the transaction fails, then the entire transaction fails, and the database state is left unchanged (ATM)
- **C**onsistency - Ensures that any transaction will bring the database from one valid state to another
- **I**solation - Ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed serially
- **D**urability - Ensures that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors

BASE - NoSQL

- **B**asic **A**vailability - Supporting partial failures without total system failure
- **S**oft-state - Data could change over time without any input
- **E**ventually Consistency - Consistency of the database will be fluctuating

Chart of Properties

- Here we can see some trade offs can be made by choice of database
- These are generalizations and not true for every database of each type

Data Model	Performance	Scalability	Complexity
NoSQL	High	High	Low
Relational Database	Variable	Variable	Moderate

Example of SQL Data Model

Preview of data to be generated (first 100 lines)

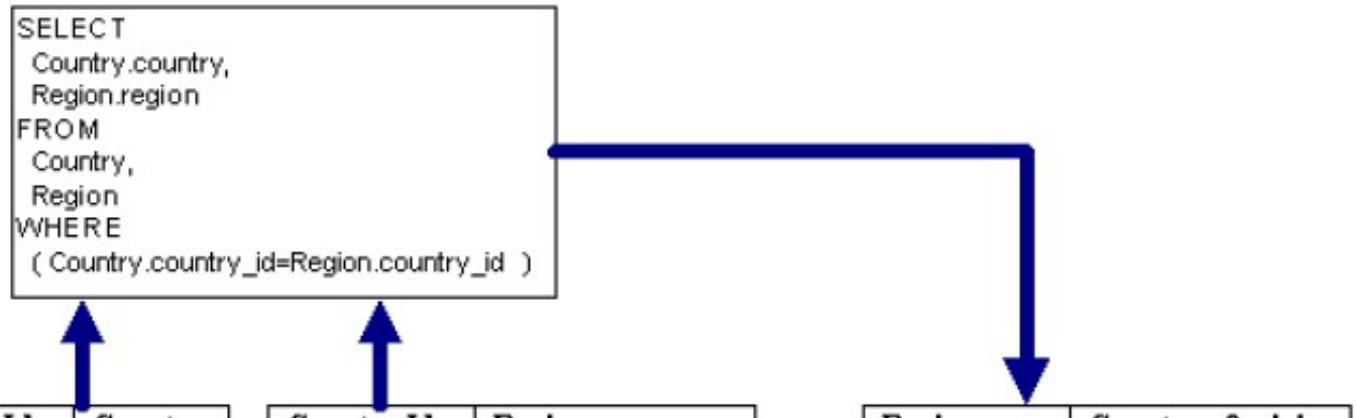
	Customer_ID *Server Assigned*	First_Name *First Name*	Last_Name *Last Name*	Social_Security_Nu *Social Security Nun*	Address_Street *Address Line (Stre*	Address_City *Address Line (Stre*	Address_State *US State Code*	Address_Zipcode *ZIP Code*
1		Trisha	Griffin	963-88-6299	225 Oak Road	758 South New...	HI	14213
2		Byron	Jacobson	390-67-2652	85 Old Freeway	994 West Rocky...	UT	01680
3		Joe	Beltran	671-89-4176	767 Rocky Old D...	19 South Fabien...	OK	47581
4		Edward	Mills	834-36-5332	655 Clarendon A...	96 Rocky Nobel...	AL	98626
5		Margarita	Ponce	713-14-3397	680 First Parkway	782 Oak Street	OK	11661
6		Karin	Bright	648-54-6243	97 South Nobel...	75 Clarendon Pa...	MO	40218
7		Sonia	Cross	348-17-2407	628 Fabien Aven...	293 North Rocky...	FL	50605
8		Sarah	Lindsey	592-73-8623	62 Green Cowle...	68 Green Fabien...	MO	96963
9		Suzanne	Franco	829-66-3800	562 North Oak...	23 White First St.	AR	26130
10		Jermaine	Arroyo	502-94-6616	927 Green New...	99 North Hague...	KS	18749
11		Hilary	Gardner	288-15-0191	255 Milton Avenue	752 Old Street	CT	93024
12		David	Hunt	715-14-5851	545 White New...	104 North Secon...	IA	23863
13		Gustavo	Jordan	059-56-7636	979 East Rocky...	223 White Nobel...	TX	19363
14		Tracy	Blevins	791-30-5838	446 West Green...	50 North Rocky...	WA	94813
15		Edwin	Reid	894-10-7618	861 Old Blvd.	60 White Hague...	AR	21000

Example of NoSQL Data Model

```
{
    "empid": "SJ011MS",
    "personal": {
        "name":"Smith Jones",
        "gender": "Male",
        "age": 28,
        "address":
        {
            "streetaddress": "7 24th Street",
            "city": "New York",
            "state": "NY",
            "postalcode": "10038"
        }
    },
    "profile": {
        "designation": "Deputy General",
        "department": "Finance"
    }
}
```

www.kodingmadesimple.com

Join Example



Country Id	Country
1.00	US
2.00	France

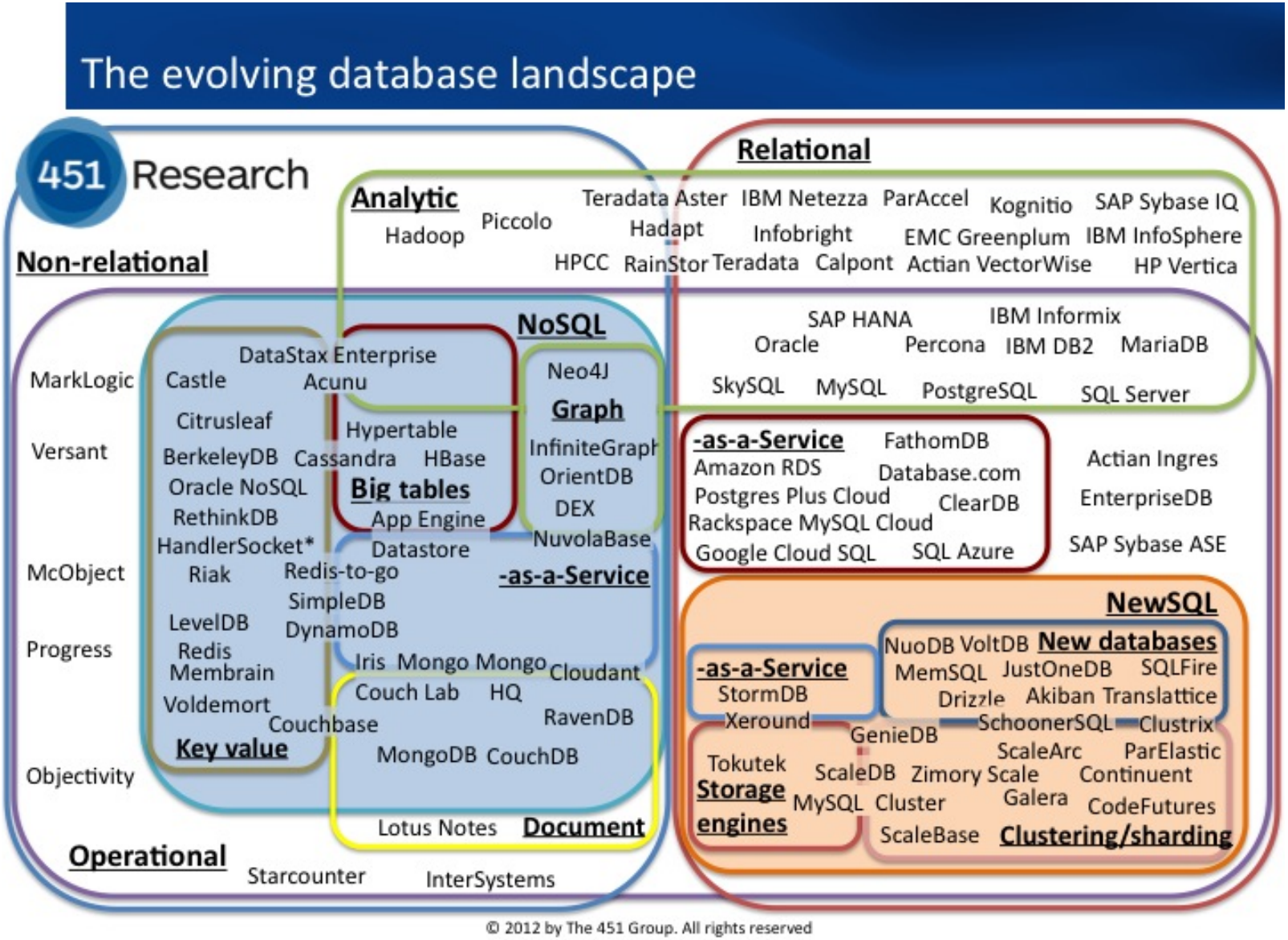
Country Id	Region
1.00	East Coast
1.00	Mid West
1.00	South
1.00	West
2.00	French Alps
2.00	Normandy
2.00	Paris
2.00	Provence

Region	Country of origin
East Coast	US
French Alps	France
Mid West	US
Normandy	France
Paris	France
Provence	France
South	US
West	US

Rapid change

- The database game is rapidly changing
- Especially NoSQL with many different databases and versions of those databases appearing
- MongoDB just recently replaced their entire storage engine in January of 2016

Rapid Change cont.



Outline

Features of SQL

- Very structured
- Hard to beat for speed of simple operations such as create, read, update, and delete
- Little to no data duplication

Features of NoSQL

- Speed increases in certain aspects
- Data can be stored in interesting ways to improve performance
- Ability to work with arbitrarily large data sets
- Horizontal Scalability - The ability to distribute both the data and the load of these simple operations over many servers, with no RAM or disk shared among the servers
- Sharding - Breaking a large database into smaller pieces and having each piece on a different database server
- Vertical Scaling - Increasing allocated resources (RAM, CPU, storage capacity)

Why so much hype?

- Big Data
  - Tesla
    - 780 million miles of driving data as of early October, adding 1 million every 10 hours
  - Twitter
    - As of 1:27pm Wednesday November 16th 2016 approximately 205,266,529,758 Tweets had been sent since the start of the year
    - Roughly on average 7,411 tweets happen per second
    - If average tweet is 200KB about 128TB of tweets are produced each day
- Distributed Systems - Independent computers linked together (Horizontal Scaling)
- It is cheaper to scale horizontally than to scale vertically

Outline

Important Notes

- All performance are done under certain conditions
- There are applications were NoSQL excels but at the same time RDBMS blow NoSQL out of the water in some aspects
- Testing for your applications is very important in the selection of a database
- Yahoo Cloud Serving Benchmark (YCSB)
- Your Mileage May Vary

First Comparison

- Tests were conducted to see how well MongoDB performs against MySQL
- Tests were performed on 500,000 records each record has 28 columns or items
- Insertion and search tests were conducted
- The conference for this paper was held in 2013
- One note: indexing is done using one or more columns of a table to provide the basis for both rapid random lookups and efficient access of ordered records

MySQL vs MongoDB Insertion Times

- 1,606 seconds vs 18 seconds
- MongoDB writes to memory
- Chance for data loss, this is an example of less durability

Number of Entries	MySQL(ms)	MongoDB(ms)
500,000	16,064,999	17,860

Insertion Time

MySQL vs MongoDB Search Times

Searching on the data that was inserted with and average of four queries yielded these results

Searched on	No of Entries	Ind. Queries	Avg. Time (ms)
Columns w/o Index	500,000	4	1374.5
Columns With Index	500,000	4	621.75

Searching Time of Query in MySQL

Searched on	No of Entries	Ind. Queries	Avg. Time (ms)
Columns w/o Index	500,000	4	210.5
Columns With Index	500,000	4	26.25

Searching Time of Query in MongoDB

MySQL vs MongoDB conclusion

- MongoDB is a clear winner in this specific comparison of the two
- Insertion gains are larger than search gains
- This is on a pretty large data set

Second Comparison

- Four people from the Software Engineering Institute Carnegie Mellon University
- Two people from Telemedicine and Advanced Technology Research Center US Army Medical Research and Material Command
- Ran benchmark tests on different NoSQL databases for the development of a new electronic health record (EHR)
- Customer already had plenty of experience using SQL and wanted to see what NoSQL had to offer
- This paper was published in 2015

MongoDB vs Cassandra vs Riak

- MongoDB (Document Store)
- Cassandra (Column Store)
- Riak (Key-Value Store)

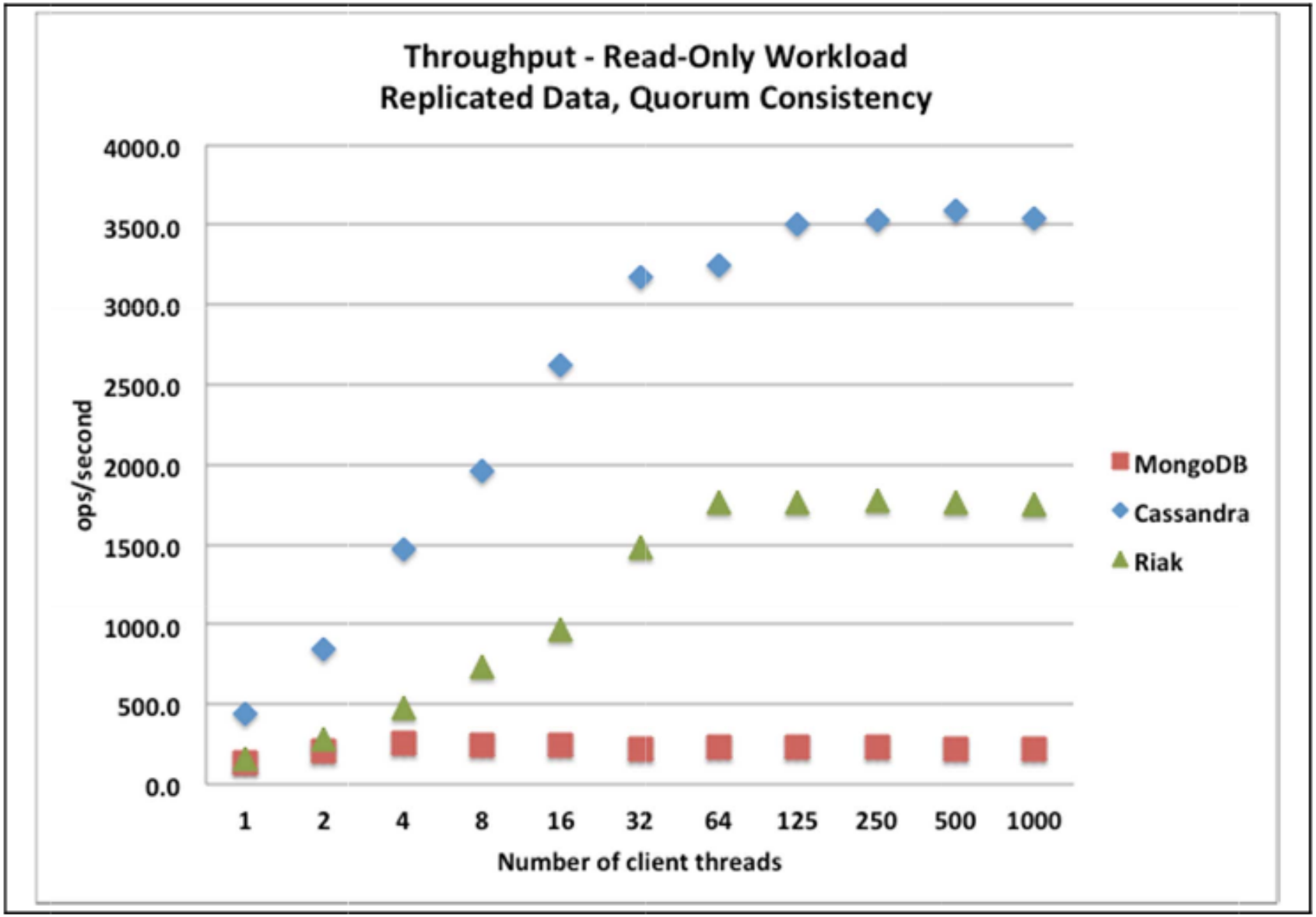
Mapping the Data Model

- Patient with information and test results
- Generated data set with one million patient records, 10 million lab results
- Each patient had between 0 and 20 test results
- The data were mapped into the data model for each database

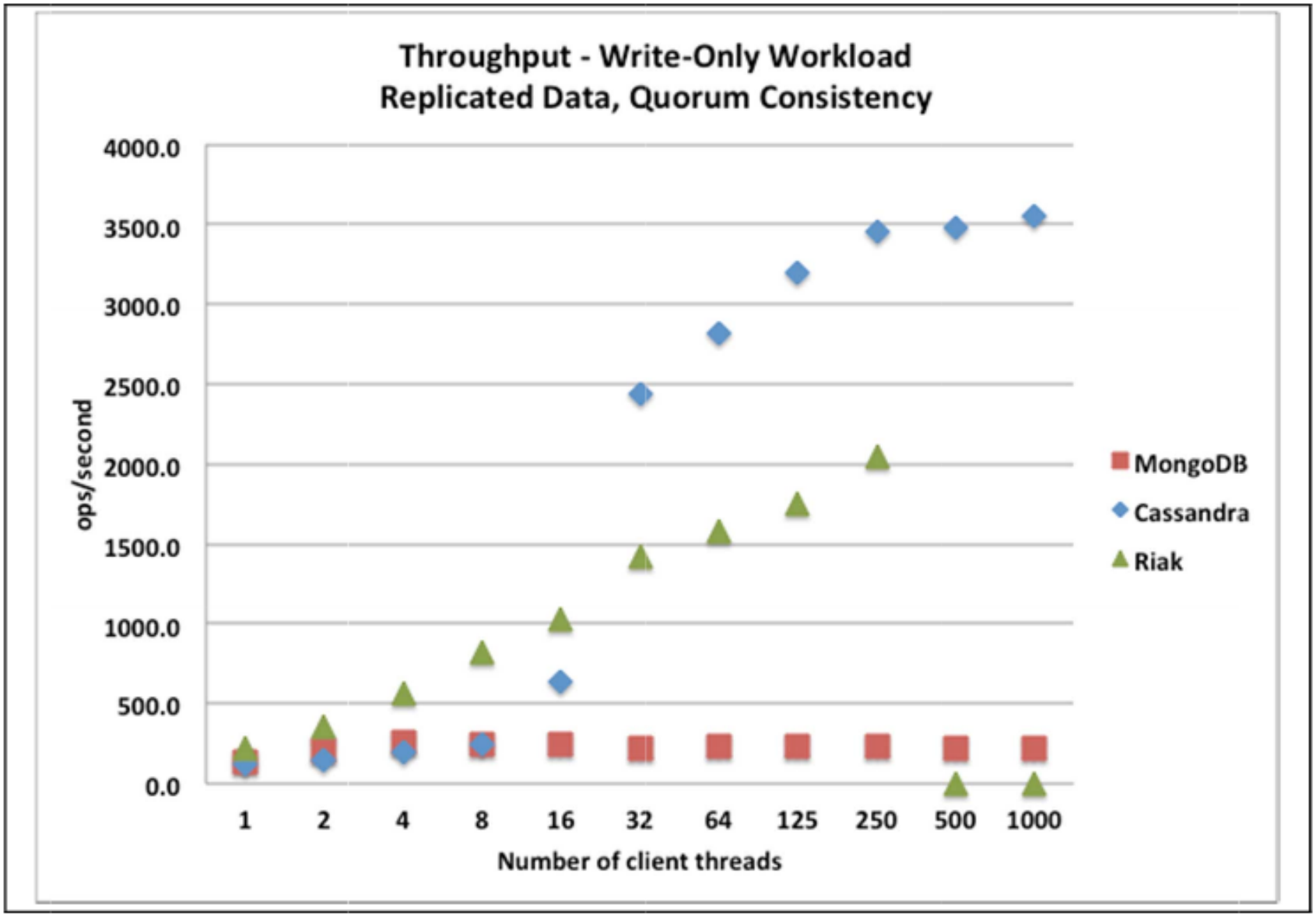
Testing Environment

- 80% read 20% write
- Read operation retrieves five most recent observations for a single patient
- Write operation inserts a single new observation for a single patient
- Three runs performed at each number of client threads 1, 2, 4, 8, 16, 32, 64, 125, 250, 500, and 1000
- Client threads are the number of client connections
- Post processed to average results across the three runs
- Tests conducted looked at throughput and latency
- Throughput - Operations per second
- Latency - Time between request and response

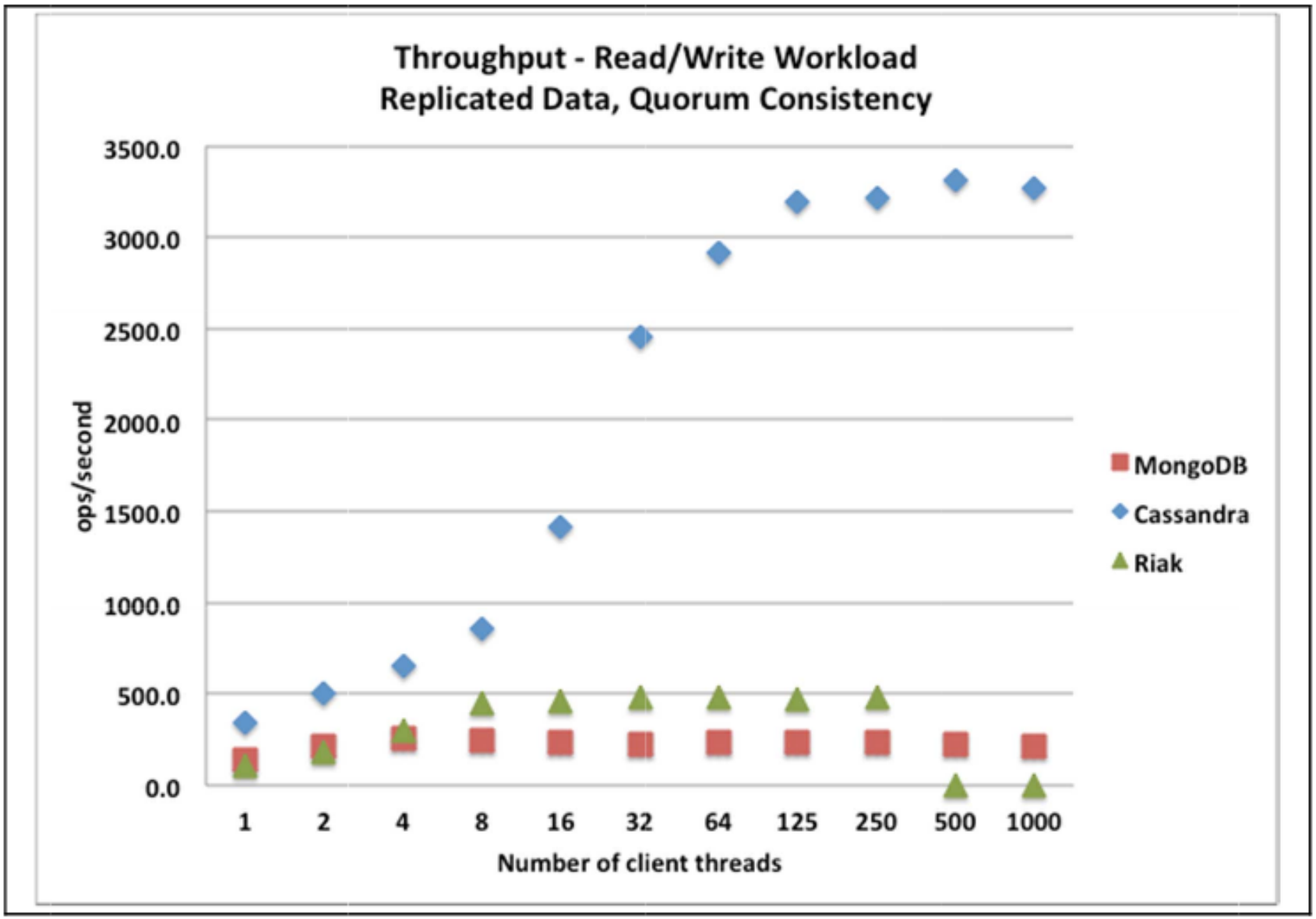
Throughput Results



Throughput Results Cont.



Throughput Results Cont.



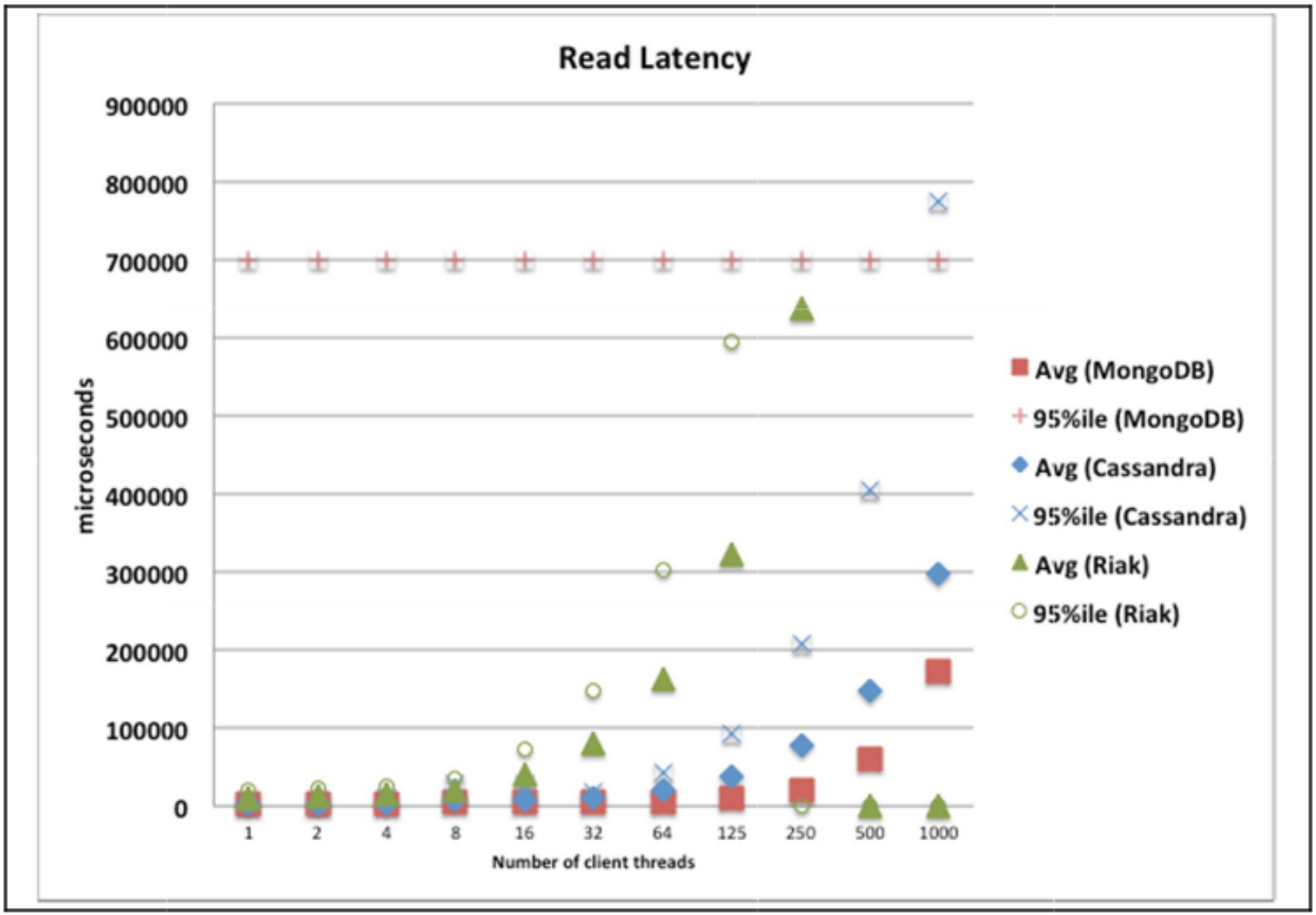
Multiple Nodes vs Single Node

- Cassandra was clearly the best for throughput
- About same for single node on read only, slight improvement for write and read/write
- Performance increases due to decreased contention for disk, and other per node resources is greater than additional work of coordinating work over multiple nodes
- Riak saw a performance increase of about 4x compared to single node configuration
- MongoDB multiple node configuration was less than 10% of the single node configuration

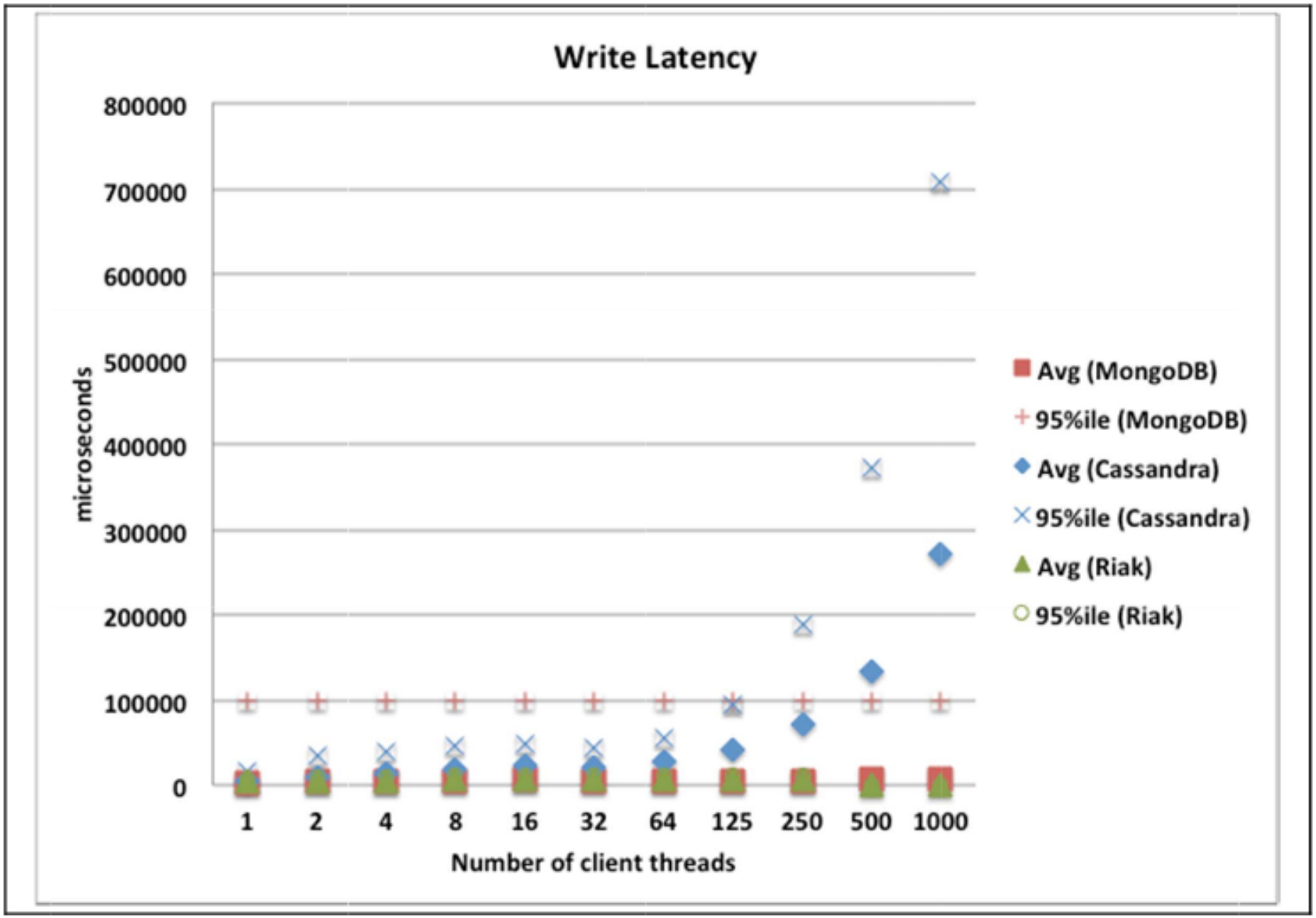
Throughput Results Cont.

- Riak cuts off at 250 mark due to resource exhaustion with the thread pool.
- MongoDB has poor performance due to sharding scheme causing all writes to go to the same shard. After tests were already finished MongoDB introduced hash-based sharding in v2.4 (tests were done on 2.2)
- Sharding - Splitting the database into shards to spread the workload (horizontal scaling)

Latency Results



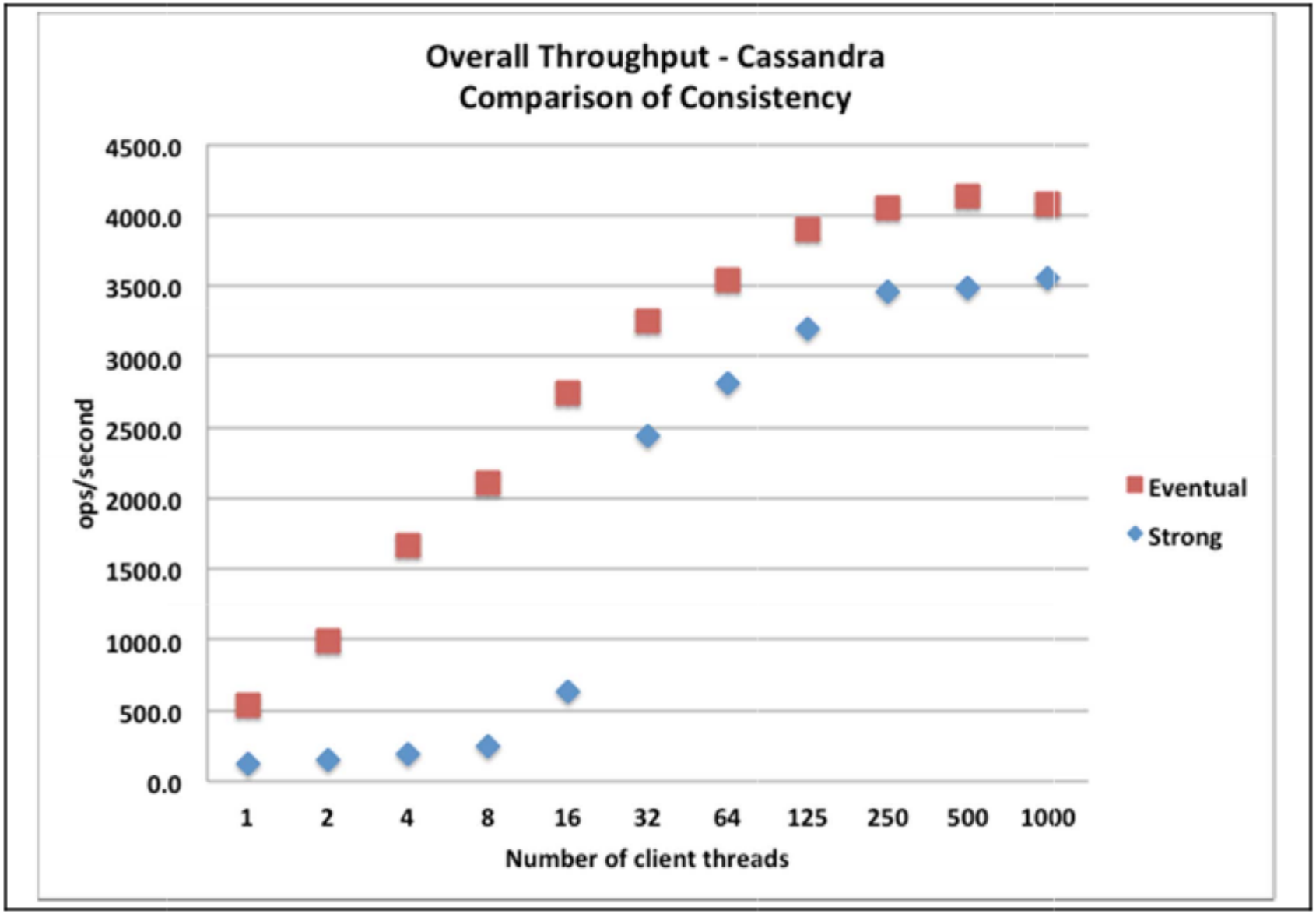
Latency Results Cont.



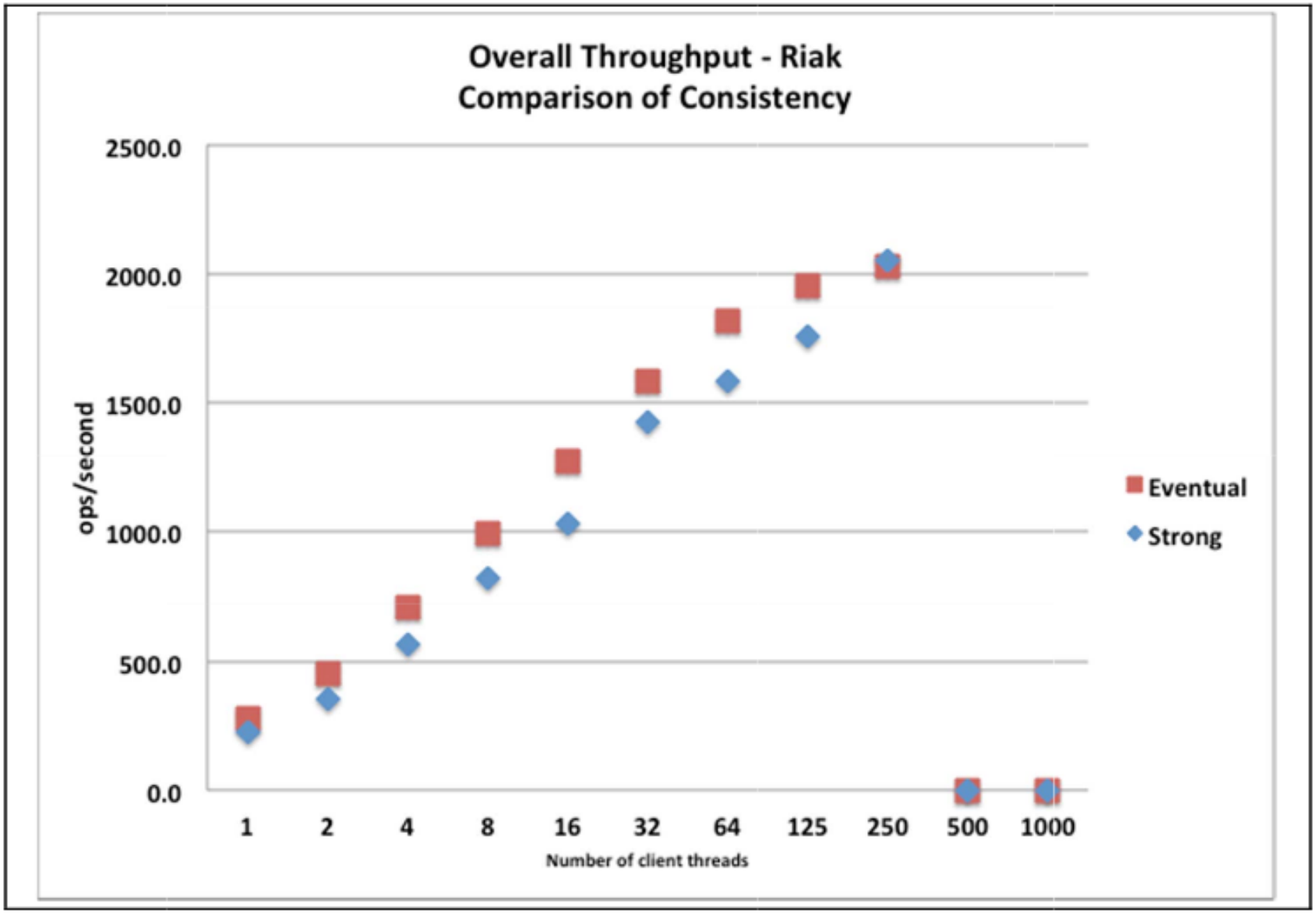
Consistency Comparison

- MongoDB ruled out as an option
- Due to the sharding errors causing all writes to go to one shard
- Comparing strong consistency and eventual consistency of Riak and Cassandra only on throughput
- There are settings within Cassandra and Riak to make strong consistency an option, as we will see performance does take a hit.

Consistency Results



Consistency Results Cont.



Outline

Conclusion

- Testing and benchmarks are very important in making decisions
- RDBMS and NoSQL each have their own place
- I do not see either going extinct any time soon
- I also cannot see an optimal best choice emerging for all applications within each category