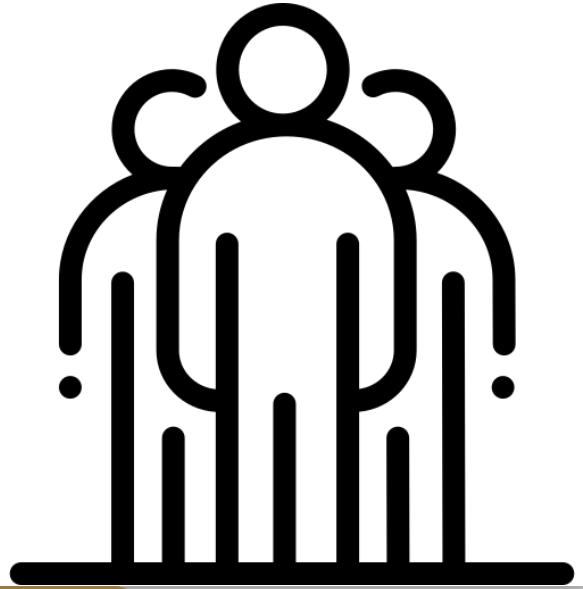
NoSQL Databases

PA195

Firebase Realtime Database

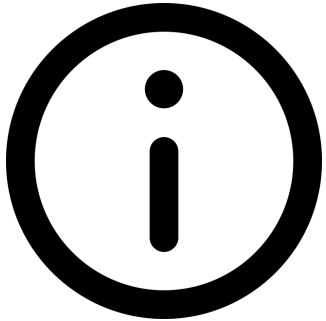
Team

- Adrián Tóth 491322
- Thành Đỗ Long 445402
- Michal Mrnuštík 487570
- Ondřej Novák 445494



Theory

About *Firebase*



- NoSQL database
- Developed by *Firebase, Inc.* in 2011
- Acquired by *Google, Inc.* in 2014
- *Firebase* platform has 18 products
  - including *Real-Time Database*

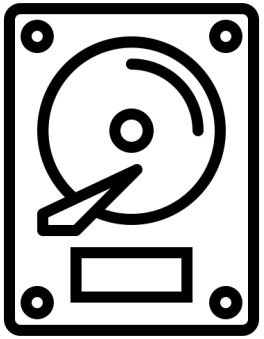
Source: https://firebase.google.com/docs/database

Real-Time Database Attributes

- *Cloud Firestore* is its Successor
- Cross-Platform
- Document-Oriented
  - Data is stored as JSON
- Persistency
- Real-Time Synchronization
- Cloud-Hosted Database
  - No need of Application Server
  - Accessible from Client Devices

Source: https://firebase.google.com/docs/database

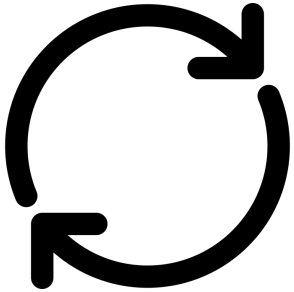
Persistency



- Firebase apps remain responsive even when offline because the Firebase Realtime Database SDK persists your data to disk.
- Once connectivity is reestablished, the client device receives any changes it missed, synchronizing it with the current server state.

Source: https://firebase.google.com/docs/database

Real-Time Synchronization



- Uses data synchronization — every time data changes, any connected device receives that update within milliseconds.
- Provide collaborative and immersive experiences without thinking about networking code.

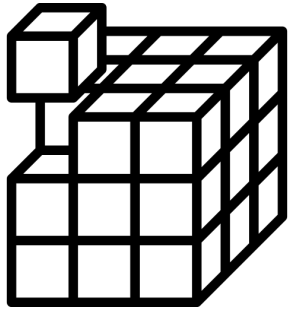
Source: https://firebase.google.com/docs/database

Cloud-Hosted Database

- The Firebase Realtime Database can be accessed directly from a mobile device or web browser; there’s no need for an application server.
- Security and data validation are available through the Firebase Realtime Database Security Rules, expression-based rules that are executed when data is read or written.

Source: https://firebase.google.com/docs/database

Data Model

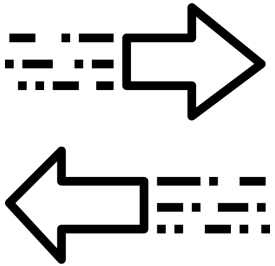


- Stores data as one large JSON tree.
- Added data become a new node in existing JSON tree.
- Complex, hierarchical data is harder to organize at scale.
- Simple data is very easy to store.
- Allows nesting data up to 32 levels deep.

```
{
  "users": {
    "alovelace": {
      "name": "Ada Lovelace",
      "contacts": { "ghopper": true },
    },
    "ghopper": { ... },
    "eclarke": { ... }
  }
}
```

Source: https://firebase.google.com/docs/database

Querying



- Queries can sort or filter on a property, but not both.
- Queries are deep by default: they always return the entire subtree.
- Queries can access data at any granularity, down to individual leaf-node values in the JSON tree.
- Queries do not require an index; however the performance of certain queries degrades as your data set grows.

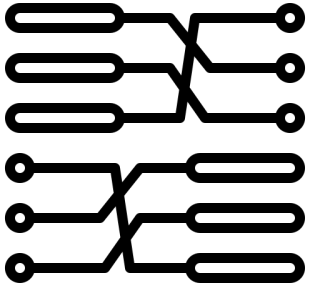
Source: https://firebase.google.com/docs/database

Rules

- Determine who has read and write access to your database, how your data is structured, and what indexes exist.
- Live on the Firebase servers and are enforced automatically at all times.
- Every read and write request will only be completed if your rules allow it.
- By default, your rules do not allow anyone access to your database. This is to protect your database from abuse until you have time to customize your rules or set up authentication.
- Custom authentication system can be used, but you have to provide unique user token to the Firebase side.

Source: https://firebase.google.com/docs/database

Indexes



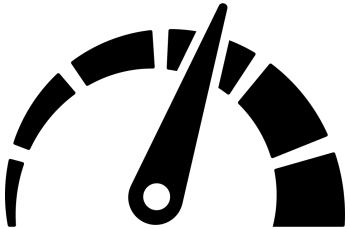
- For small data sizes, the database supports ad hoc querying, so indexes are generally not required during development.
- Indexes are specified using the *.indexOn* rule.

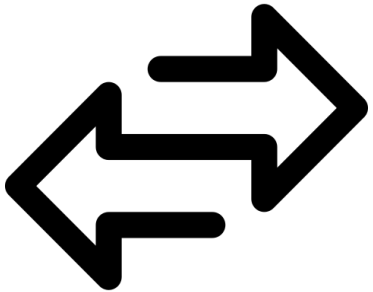
```
{
  "rules": {
    "dinosaurs": {
      ".indexOn": ["height", "length"]
    }
  }
}
```

Source: https://firebase.google.com/docs/database

Reliability, Performance & Transactions

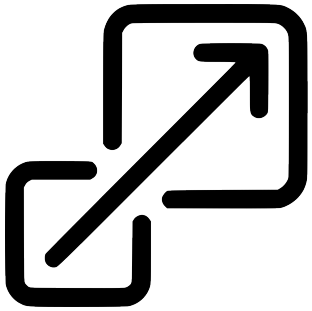
- Realtime Database is a single-region solution.
- Databases are limited to zonal availability in a single region.
- Extremely low latency, ideal option for frequent state-syncing.
- Transactions are atomic on a specific data subtree.





Source: https://firebase.google.com/docs/database

Scalability



- Scaling requires sharding.
- Scale to around 200,000 concurrent connections and 1,000 writes/second in a single database. Scaling beyond that requires sharding your data across multiple databases.
- No local limits on write rates to individual pieces of data.

Source: https://firebase.google.com/docs/database

Pricing

- Free for a specific size and complexity.
- Charges only for bandwidth and storage, but at a higher rate.
- Till 1 GB stored and 10 GB transferred it’s free of charge, then about 5$ per 1 GB stored and 1$ per 1 GB transferred.
- https://firebase.google.com/pricing

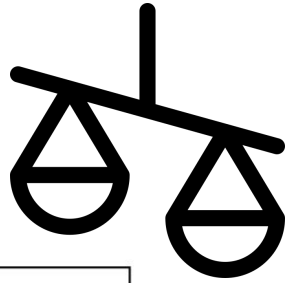
Source: https://firebase.google.com/docs/database

Libraries

- GO
- Java
- Python
- C++
- JavaScript
- Unity
- iOS
- Android
- ...

Source: https://firebase.google.com/docs/libraries

Comparison with other NoSQL DBs

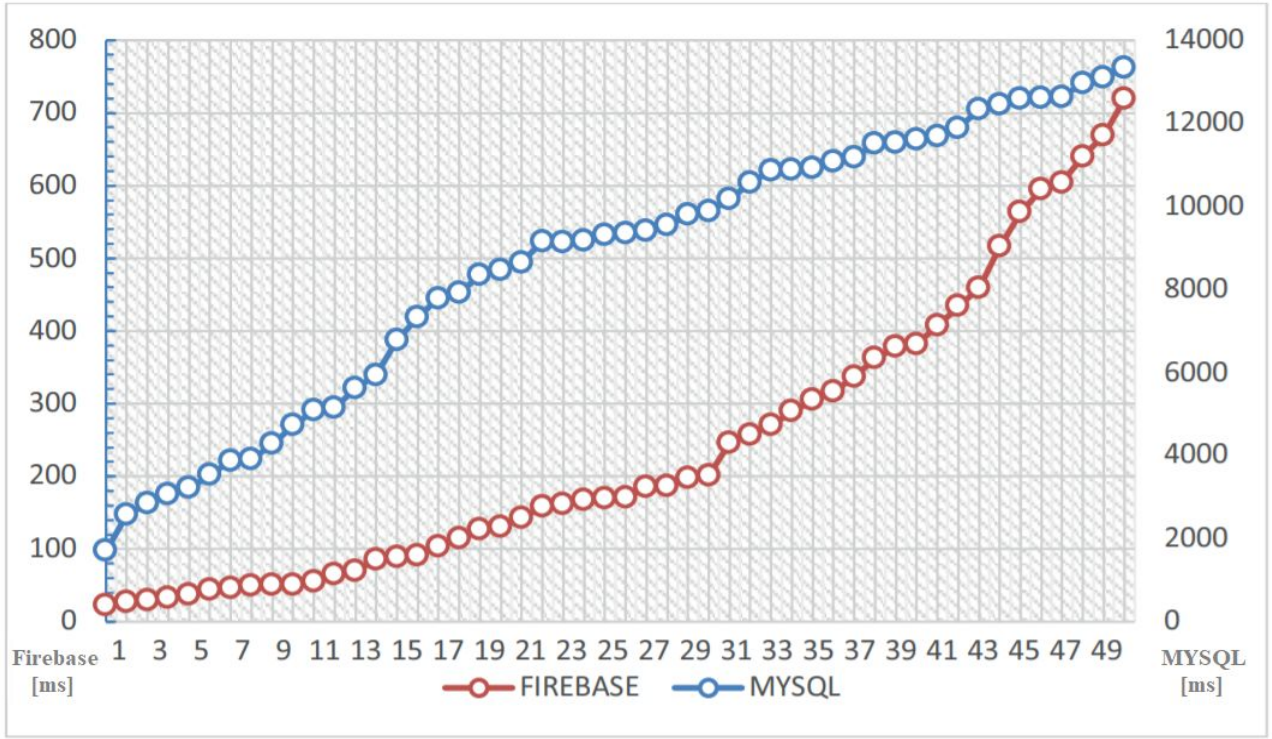


Test / DB	MongoDB	DynamoDB	CouchDB	Firebase
Upload Small Data	250	210	470	70
Upload large Data	1200	680	2800	500
Retrieve Small Data	160	150	366	55
Retrieve large Data	740	300	700	540
Update large Data	740	300	540	700
Update Small Data	250	210	520	40

Average time in ms taken to perform each test

Source: https://pdfs.semanticscholar.org/e846/d6ba2cd2338c9ec207a0699d9b6b39d3ebc0.pdf

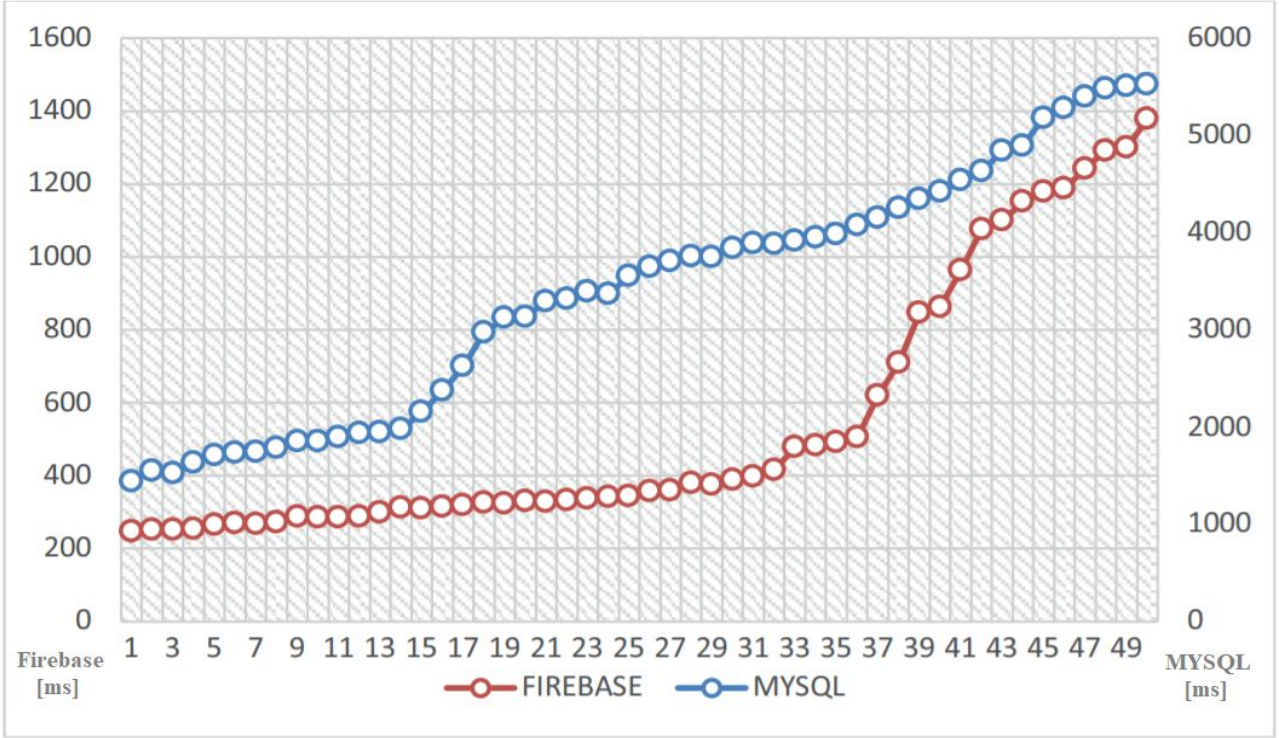
Comparison with MySQL on CREATE



Mean performance time in milliseconds for the *CREATE* operation

Source: https://www.sciencedirect.com/science/article/pii/S1877050919311500

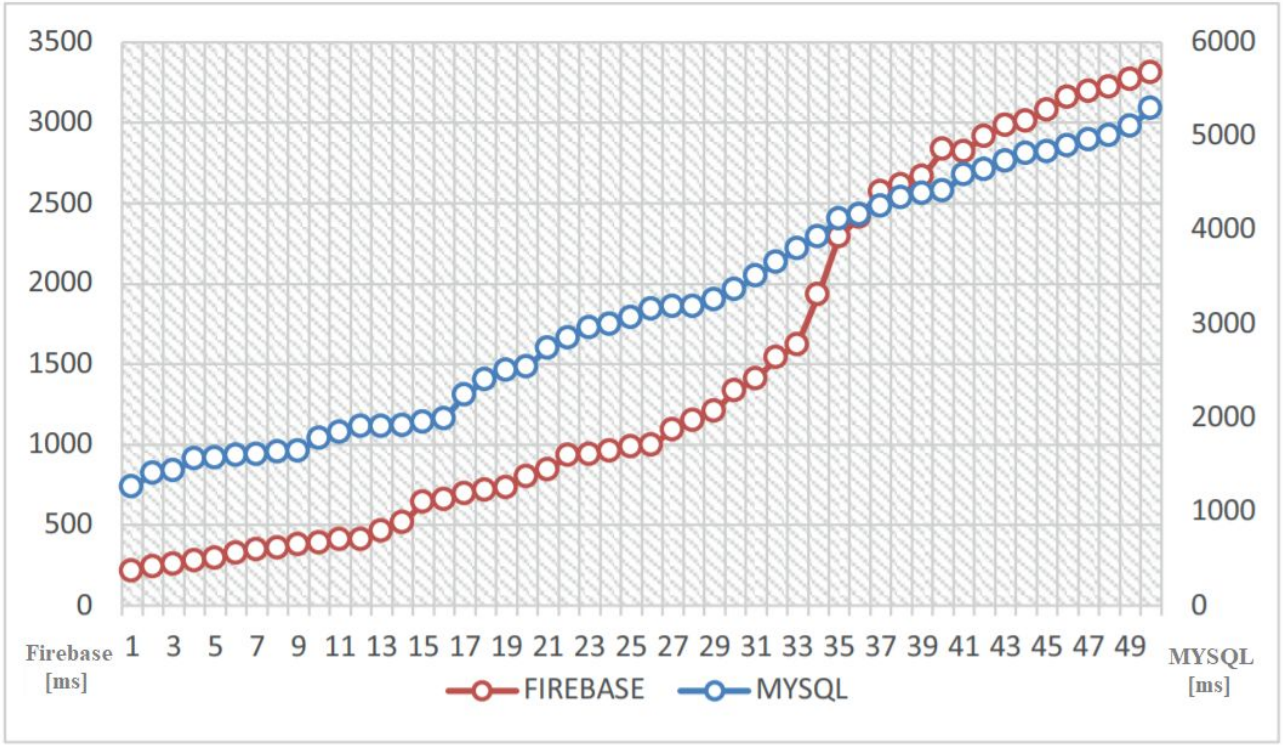
Comparison with MySQL on READ



Mean performance time in milliseconds for the *READ* operation

Source: https://www.sciencedirect.com/science/article/pii/S1877050919311500

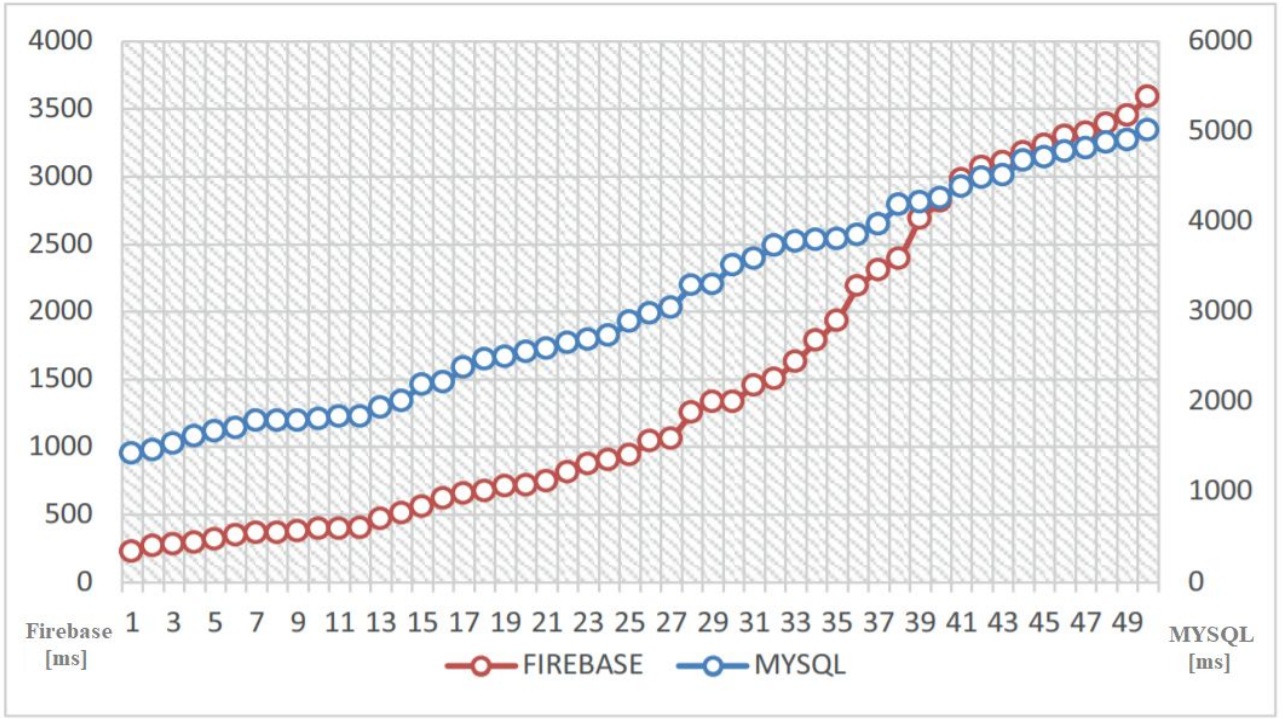
Comparison with MySQL on UPDATE



Mean performance time in milliseconds for the UPDATE operation

Source: https://www.sciencedirect.com/science/article/pii/S1877050919311500

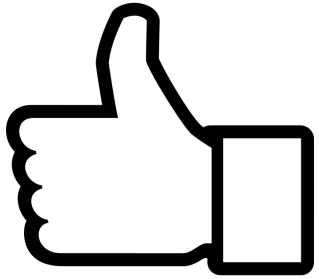
Comparison with MySQL on DELETE



Mean performance time in milliseconds for the DELETE operation

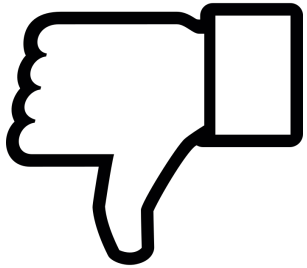
Source: https://www.sciencedirect.com/science/article/pii/S1877050919311500

Advantages



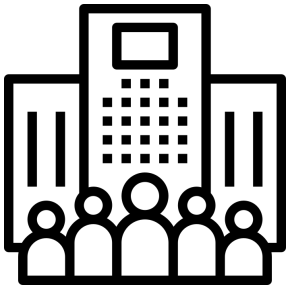
- Robust client libraries.
- Full support for offline mode.
- Automatic synchronization.
- Low latency.
- Comprehensive set of security rules.
- Easy-to-use data browsing tool.
- Fast executing thanks to its efficiency platform.
- No need of server side.
    - Is provided by Google Inc.
- Complex tool package.

Disadvantages



- Pricing.
- Limited query abilities.
  - Due to Firebase's data stream model.
- Bad for big data.
  - Better for many smaller chunks of data.

Companies using Firebase



- Venmo
- Lyft
- Duolingo
- The New York Times
- Alibaba
- Shazam

Demonstration

Thank you for your attention.