



Multiple NoSQL Use Cases with Redis Modules

Kamran Yousaf

kamran@redislabs.com

About Redis



Open source. The leading **in-memory database platform**, supporting any high performance OLTP or OLAP use case.

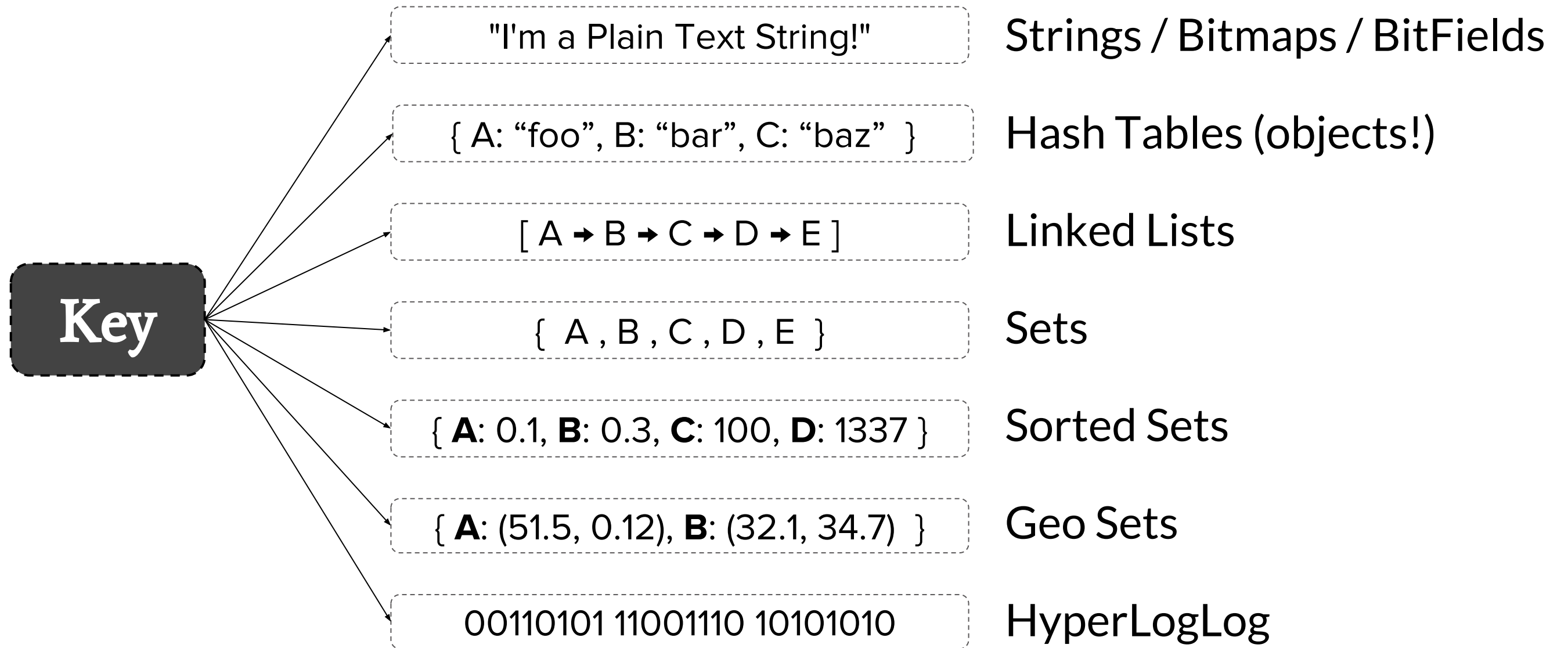


Founded in 2011

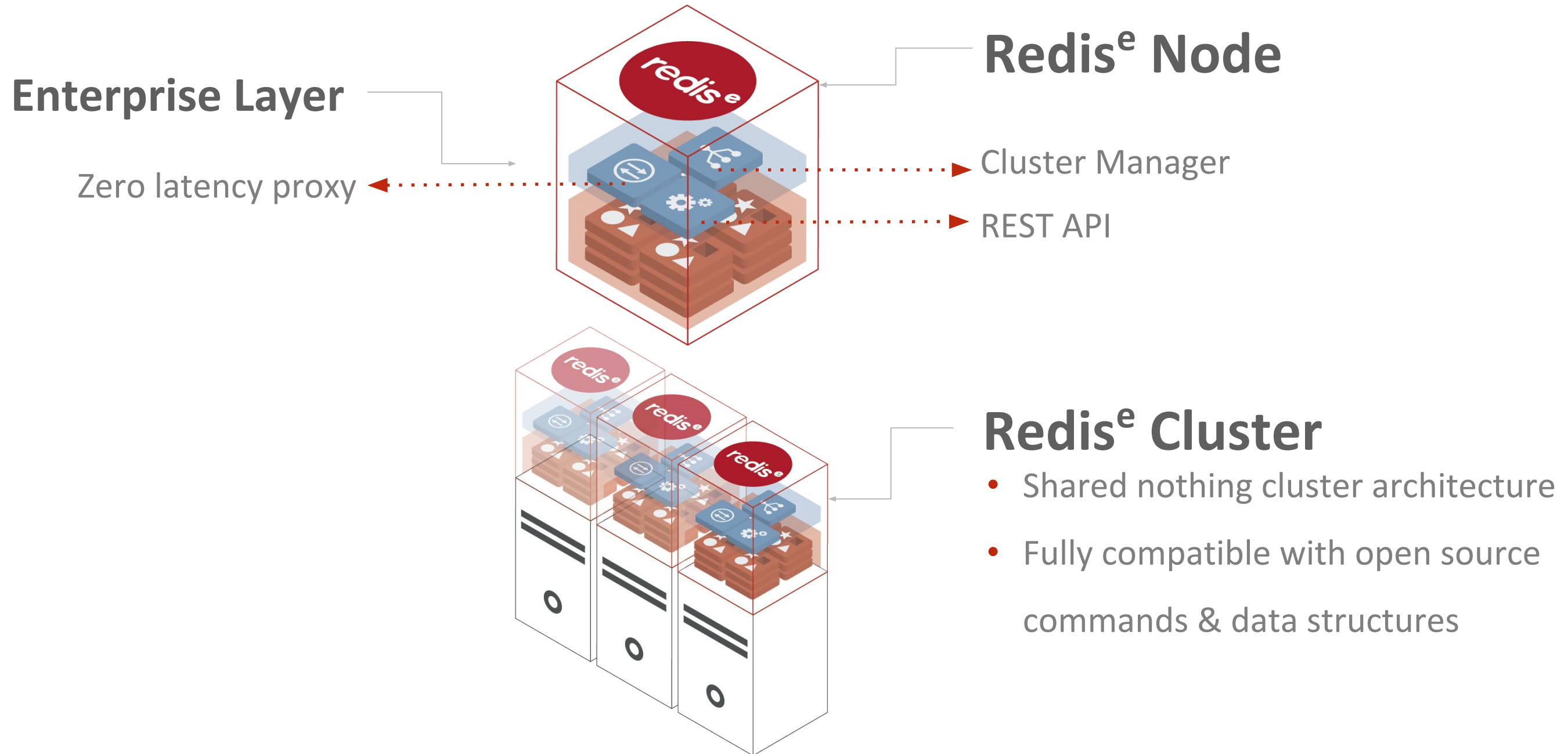
HQ in Mountain View CA, R&D center in Tel-Aviv IL.

- The open source home and commercial provider of Redis
- Providing Data Analytics at the Speed of business
- #1 Top Databases Popularity Rankings, #1 Leader and Challenger in Analyst Quadrants
- Multi-model, can support all popular database models and modern use cases

A Quick Recap of Redis



Redis^e Platform



Redis Labs Products



Redis^e Cloud

Fully managed Redis^e service on hosted servers within AWS, MS Azure, GCP, IBM Softlayer, Heroku, CF & OpenShift

RAM



Redis^e Cloud Private

Fully managed Redis^e service in VPCs within AWS, MS Azure, GCP & IBM Softlayer

RAM

or

Flash



Redis^e Pack

Downloadable Redis^e software for any enterprise datacenter or cloud environment

RAM

or

Flash



Redis^e Pack Managed

Fully managed Redis^e Pack in private data centers

RAM

or

Flash

Secondary Index?

Full Text Search?

SQL?

Machine Learning?

But Can Redis Do X?

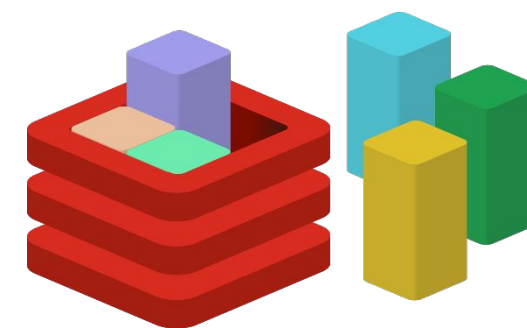
AutoComplete?

Graph?

Time Series?

Redis Modules

- C extensions to Redis
- Work at native speed
- Add commands
- Can implement their own native data types
- Can extend redis or just use it as a “Server Framework”
 - Isolated API
 - ABI Backwards Compatibility



Adapt your database to your data

Neural Redis

Simple Neural Network
Native to Redis

Redis-ML

Machine Learning Model
Serving

RediSearch

Full Text Search Engine in
Redis

ReJSON

JSON Engine on Redis.
Pre-released

Time Series

Time series values
aggregation in Redis

Graph

Graph database on Redis
based on Cypher language

Rate Limiter

Based on Generic Cell Rate
Algorithm (GCRA)

Crypto Engine Wrpper

Secure way to store data in
Redis via encrypt/decrypt
with
various Themis primitives

Secondary Index/RQL

Indexing + SQL -like syntax
for querying indexes.
Pre-released

Redisearch=Text Search

Redisearch V1

- From-Scratch search index over redis
- Uses String DMA for holding compressed index data
- This allows efficient compression of data
- Includes stemming, exact phrase match, etc.
- Up to X5 faster than Elastic / Solr

Search in Action

```
> FT.CREATE products SCHEMA title TEXT price NUMERIC
OK
> FT.ADD products id1 1.0 FIELDS title "LG 42'' LCD TV" price 500
OK
> FT.ADD products id2 1.0 FIELDS title "Toshiba 40'' LCD TV" price 400
OK
> FT.SEARCH products "lcd tv" FILTER price 450 +inf
1) (integer) 1
2) "id1"
3) 1) "title"
   2) "LG 42'' LCD TV"
   3) "price"
   4) "500"
```

Where

Source code:

<https://github.com/RedisLabsModules/RediSearch>

Documentation:

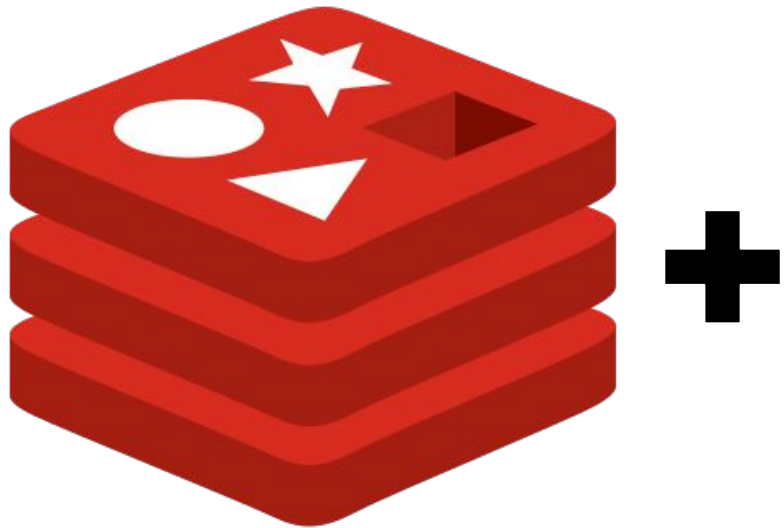
<http://redisearch.io/>

redis-ml +neural-redis=Machine Learning

Redis-ML

- Database for ML models
- Train wherever you want
- Store the result in redis
- Query models in real time
- Scale your models beyond a single machine

The Redis-ML Module



Redis Module

Tree Ensembles

Linear Regression

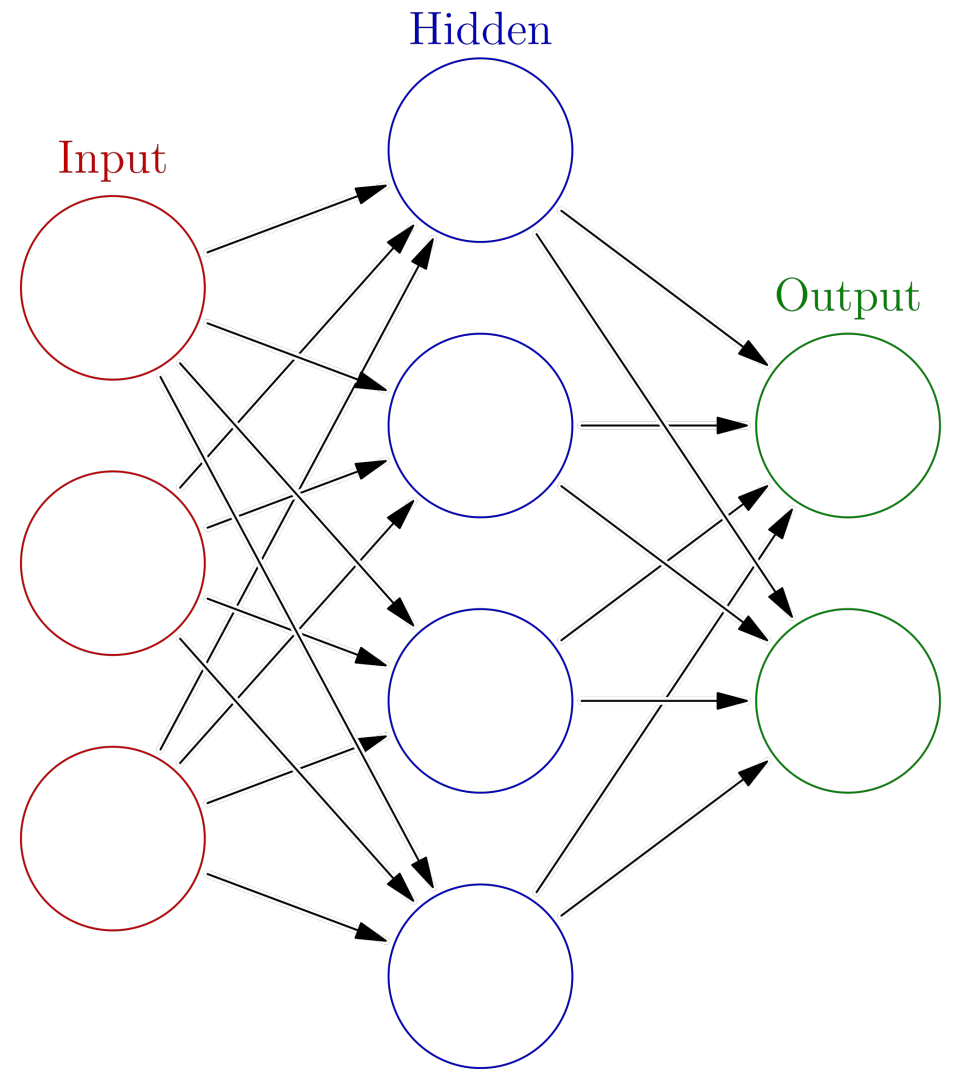
Logistic Regression

Matrix + Vector Operations

More to come...

Neural Redis

- Developed by Salvatore
- Training is done inside Redis
- Online continuous training process
- Builds Fully Connected NNs



Neural Redis In Action

```
>NR.CREATE net REGRESSOR 2 3 -> 1 NORMALIZE DATASET 50 TEST 10
```

```
>NR.OBSERVE net 4 5 -> 9
```

```
>NR.OBSERVE net 1 1 -> 2
```

```
....
```

```
>NR.OBSERVE net 5 6 -> 11
```

```
>NR.TRAIN net AUTOSTOP
```

```
>NR.RUN net 1 1
```

```
1) "2.1406970024108887"
```

```
>NR.RUN net 3 5
```

```
1) "8.3342075347900391"
```

Where

Redis ML

Source code:

<https://github.com/redislabsmodules/redis-ml>

Neural Redis

Source code:

<https://github.com/antirez/neural-redis>

ReJSON = Redis + JSON

ReJSON [Preview Release]

- A custom JSON data type for Redis
- Keys can contain any valid JSON value
 - Scalars, objects or arrays
 - Nested or not
- Data is stored decoded in binary format
- JSONPath-like syntax for direct access to elements
- Strongly-typed atomic commands

ReJSON In Action

```
> JSON.SET doc . '{ "foo": "bar", "baz": [1,2] }'  
> JSON.SET doc .goo true  
> JSON.NUMINCRBY doc .baz[0] 9  
> JSON.GET doc .baz  
"[10,2]"  
> JSON.GET doc .  
"{ \"foo\": \"bar\", \"baz\": [10,2], \"goo\": true }"
```

ReJSON commands

General	JSON.DEL, JSON.GET, JSON.MGET, JSON.SET & JSON.TYPE
Numbers	JSON.NUMINCRBY & JSON.NUMMULTBY
Strings	JSON.STRAPPEND & JSON.STRLEN
Objects	JSON.OBJKEYS & JSON.OBJLEN
Arrays	JSON.ARRAPPEND, JSON.ARRINDEX, JSON.ARRINSERT, JSON.ARRLEN, JSON.ARRPOP & JSON.ARRTRIM
Other	JSON.RESP

Where

Source code:

<https://github.com/RedisLabsModules/rejson>

Documentation:

<https://redislabsmodules.github.io/rejson>

redis-secondary=Secondary Indexes

Redis-Secondary

- Schema aware secondary indexes
 - Integers, floats, booleans, strings, etc
- Either a raw data type or automatic index
- Treat HASH keys as objects or table rows
- SQL-like WHERE clauses as proxies to redis commands

Secondary Indexing In Action

```
> IDX.CREATE users_name_age TYPE HASH SCHEMA name STRING age INT32

> IDX.INTO users_name_age HMSET user1 name "alice" age 30
> IDX.INTO users_name_age HMSET user2 name "bob" age 25

> IDX.FROM users_name_age WHERE "name LIKE 'b%'" HGET $ name
    1) user1
    2) "bob"

> IDX.FROM users_name_age WHERE "name >= 'alice' AND age < 31" HGETALL $
    1) user1
    ...
```

Where

Source code:

<https://github.com/RedisLabsModules/secondary>

Modules For Probabilistic Data Analysis

redabloom

Bloom filter:Used for set membership

topk

Top k most frequent:Counts the top most frequent elements in a stream of data

countminsketch

Counts of observations:Tracks the frequency of samples in a stream of observations

T-digest

Rank based stats estimator:Useful for quantiles, cumulative distributions for very large datasets

- Optimized for memory space efficiency
- Ultra-fast query response times even with millions of elements
- Does not require storing of the actual streaming data elements

Redabloom=Scalable, counting bloom filters

- Membership Query. Test whether an element is a member of a set.
- False positive matches are possible, but false negatives are not
- Provides scalable, counting bloom filters
- Port of the dablooms library
- Provides automatic ID generation using a sequence or the server's clock

Source: <https://github.com/RedisLabsModules/redablooms>

```
> CBF.ADD users Jeff
> CBF.ADD users Bob
> CBF.ADD users Itamar
> CBF.ADD users Dvir
> CBF.REM users Itamar

> CBF.CHECK users Dvir
(integer) 1
> CBF.CHECK users kamran
(integer) 0
> CBF.CHECK users Itamar
(integer) 0
>
```

t-digest=quantiles and cumulative distribution

- Accurate on-line accumulation of rank-based statistics
- Quantiles and cumulative distribution

Source:


<https://github.com/usmanm/redis-tdigest>

```
> TDIGEST.NEW latency.disk1
> TDIGEST.ADD latency.disk1 7
> TDIGEST.ADD latency.disk1 30
....
> TDIGEST.QUANTILE latency.disk1
0.95
1) "32.12499999999999993"
>
```

The Start of a Module Ecosystem

- Modules were launched May 2016 at RedisConf
- Initial modules started to appear, mostly toy projects
- We launched Modules-Hub

Module Hub ^{Beta}




Redis Modules are add-ons to Redis which extend Redis to cover most of the popular use cases for any industry. They seamlessly plug into open source Redis or enterprise-class Redis, are processed in-memory and enjoy Redis' simplicity, super high performance, infinite scalability and high availability.


Modules can be created by anyone. The Module Hub marketplace includes Modules created by Redis Labs as well as by others.

All modules in this marketplace (open source or commercial) are certified by Redis Labs for use with open source Redis, Redis Labs Enterprise Cluster (RLEC) or Redis Cloud.

[Submit Your Own Module](#)



redisearch
High performance full text search engine
By: Redis Labs
OSS - GNU AGPLv3



graphicsmagick
Swiss army knife of image processing
By: Redis Labs
OSS - GNU AGPLv3

Redis Modules Hack 2016

- A global, distributed hackathon
- With on-site events in TLV and SF
- Some very interesting projects
- We gave them cash!

First Place - Redis-Cell

- Author: Brandur Leach
- Advanced rate limiting algorithm
- Actually written in Rust, not C

```
CL.THROTTLE user123 15 30 60 1
```

```
      ▲      ▲      ▲      ▲      ▲
```

```
      |      |      |      |      |_____ apply 1 token (default if omitted)
```

```
      |      |      |_____ 30 tokens / 60 seconds
```

```
      |      |_____ 15 max_burst
```

```
      |_____ key "user123"
```

Notable Mention: Redis-Graph

- Author: Roi Lipman
- A fast graph database implemented in Redis
- Subset of Cypher (Neo4J QL) implementation
- Uses Redis Hash keys as nodes

```
GRAPH.QUERY presidents
  "MATCH (president)-[born]->(state:Hawaii)
  RETURN president.name, president.age"
```

Other Cool Modules

- Time-series aggregation engine
- Pyrecks - Python function execution
- Statsd reporting of Redis metrics
- Zstd value compression
- Time decaying popularity ranking
- Shared memory support
- And many more!

When

Redis 4.0 status

- RC2 released 2 months ago
- RC3 soon to be released
- Surprisingly little serious issues reported so far
- However... a few things to fix / add before 4.0-final
- ETA: about 2 months

Coming In Redis^e

- To be released soon in Redis Labs' enterprise cluster
- Pre-bundled modules:
 - RediSearch
 - ReJSON
 - Secondary
 - Redis-ML
- Use your own modules at their own risk

Thank You!