DATA 2019 Paper #21

Distributed and scalable platform for collaborative analysis of massive time series data sets

Ed Duarte, Diogo Gomes, David Campos and Rui L. Aguiar

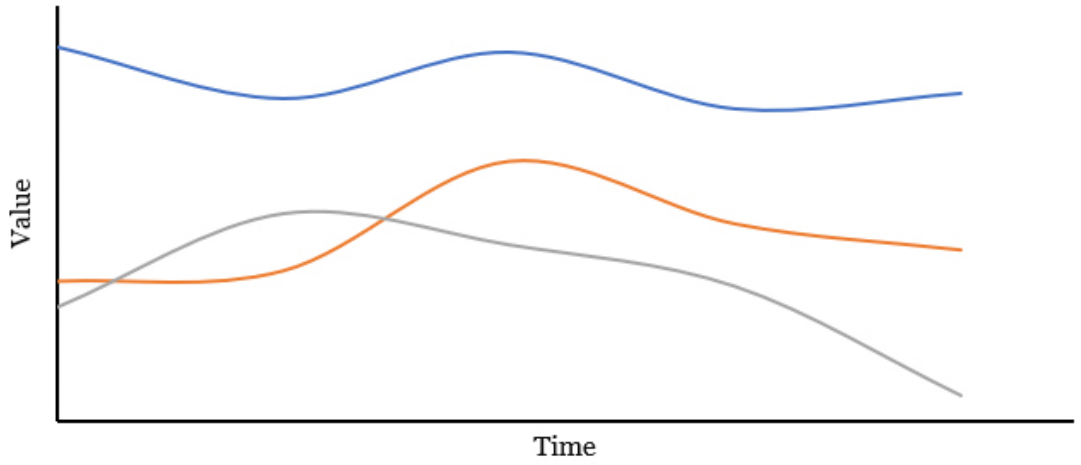
UNIÃO EUROPEIA
Fundo Europeu
de Desenvolvimento Regional

Introduction

- metrification of devices;
  - e.g. wearable gadgets, real-time IoT sensors, Smart Home devices
- annual data acquisition rate:
  - 2016 – 1.2 zb/y;
  - 2021 – 3.3 zb/y;
- requirements for digital data processing and storage are increasing exponentially;
- Volume, Variety and Velocity;
- Value and Veracity.

Introduction

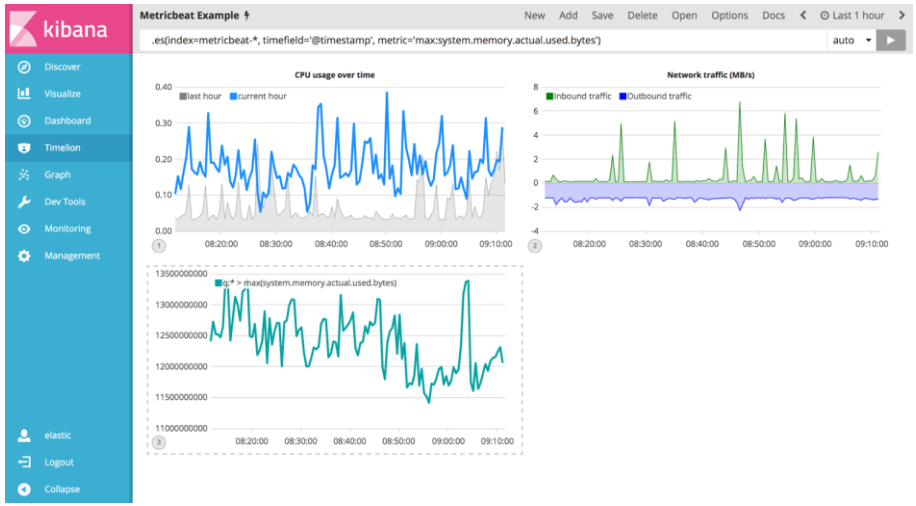
Time series analysis



- some metrics only have meaning when observed as a pattern over time;
- **time series** can be found in almost every aspect of human life;
- most domains produce massive amounts of series data;
- analysis is more agile when within a software solution.

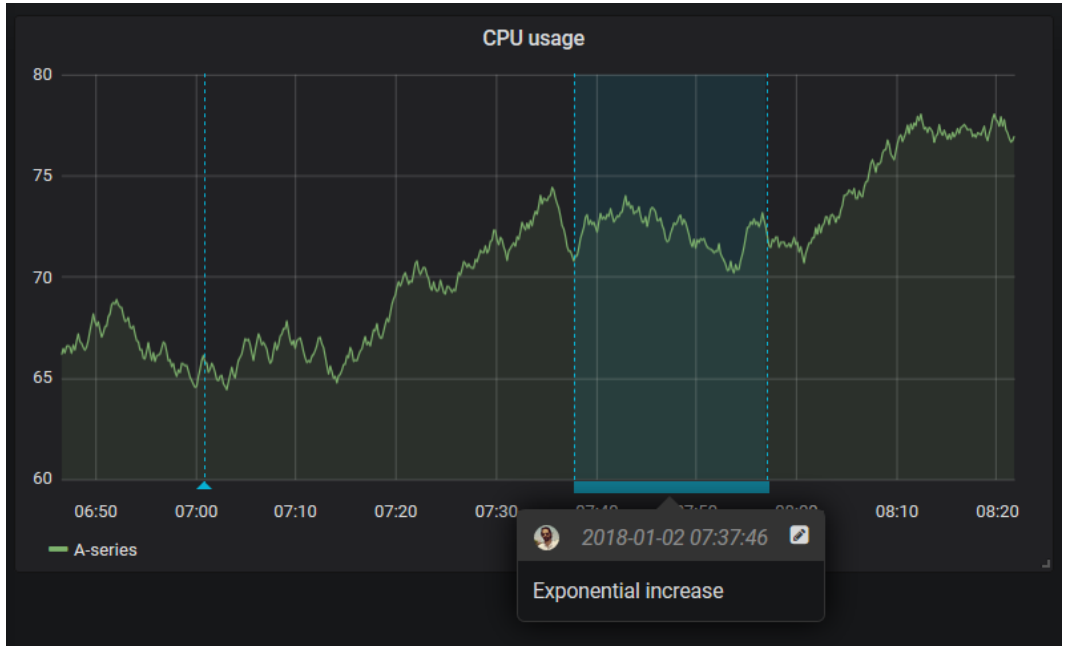
Introduction Time series visualization

- can be a very challenging task:
  - data sets commonly have high cardinality and complexity;
- comparative visualization tasks:
  - dashboard applications like Timelion, Grafana and Freeboard
- most analysis applications are built as web applications.



Introduction Annotation

- realistic analysis tasks involve collaboration and knowledge-sharing between human curators;
- **annotations** facilitate knowledge-building and decision-making in analysis processes.

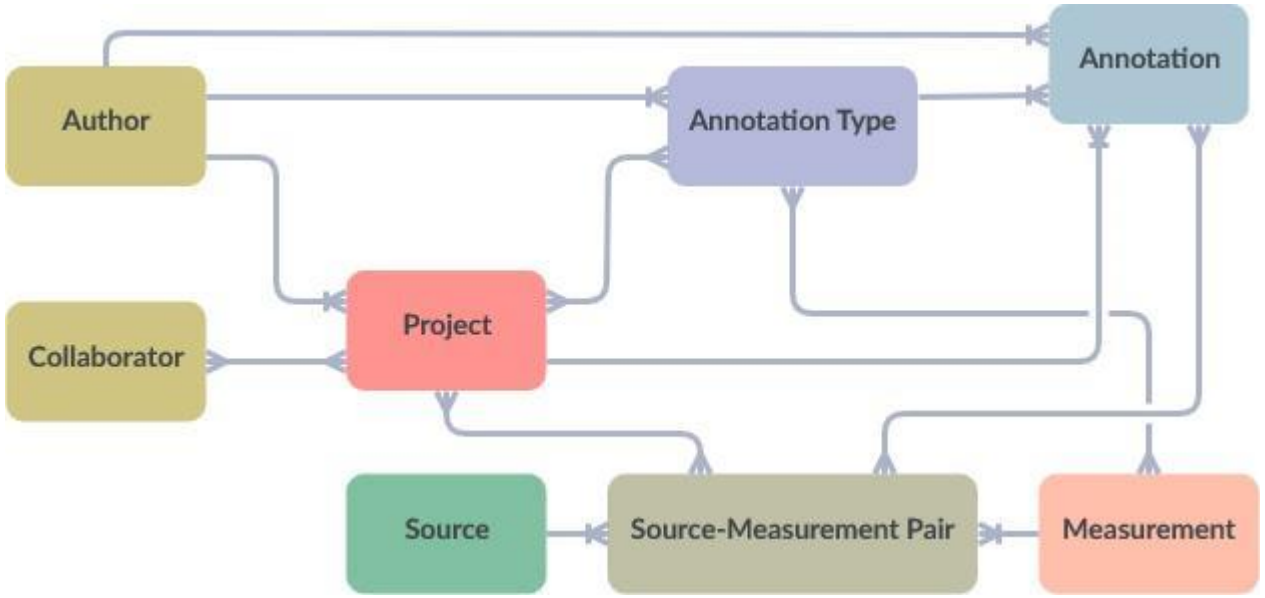


Proposal

- data-intensive architecture and web application for collaborative time series analysis;
- use most appropriate open-source tools for querying, storing and displaying time series and annotations;
- distributed architecture to handle high quantities of concurrent usage:
  - E+C for annotations, users and the knowledge base;
  - E+L for series.
- prototype tested with HVAC data set from 1000 boilers over 1.3 years.

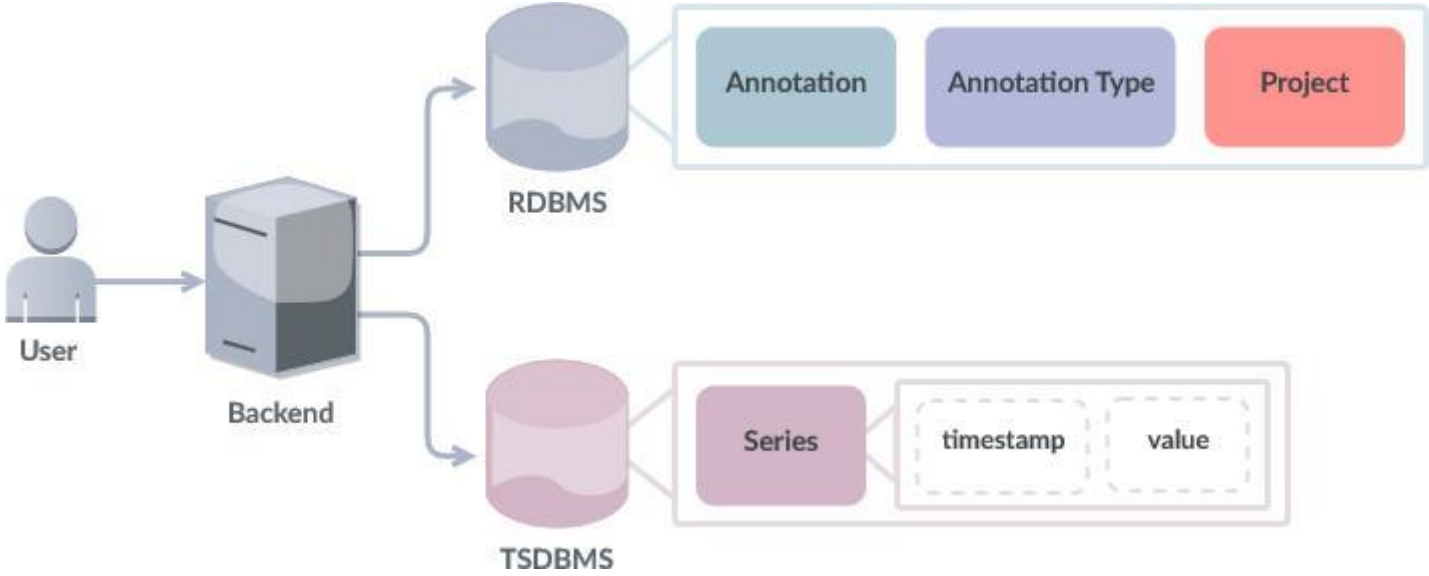
Proposal Data model

- time series has a **measurement** and a data **source**;
- annotations have a parent **type**, a point or ranged segment of time, and a set of affected series;
- **projects** restrict a set of collaborators to a segment of time, a set of series, and an annotation scope.



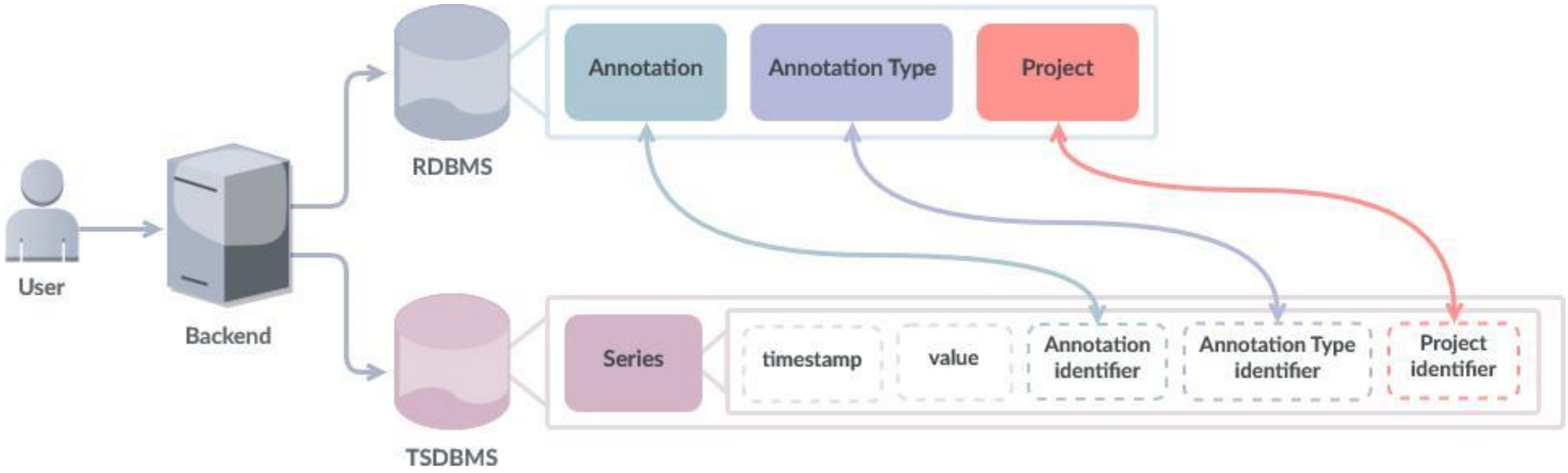
Proposal Data management

- polyglot persistence model:
  - time series are stored in InfluxDB, ontology is stored in PostgreSQL;
  - central backend enforces data access logic and conceals the real location of the data.

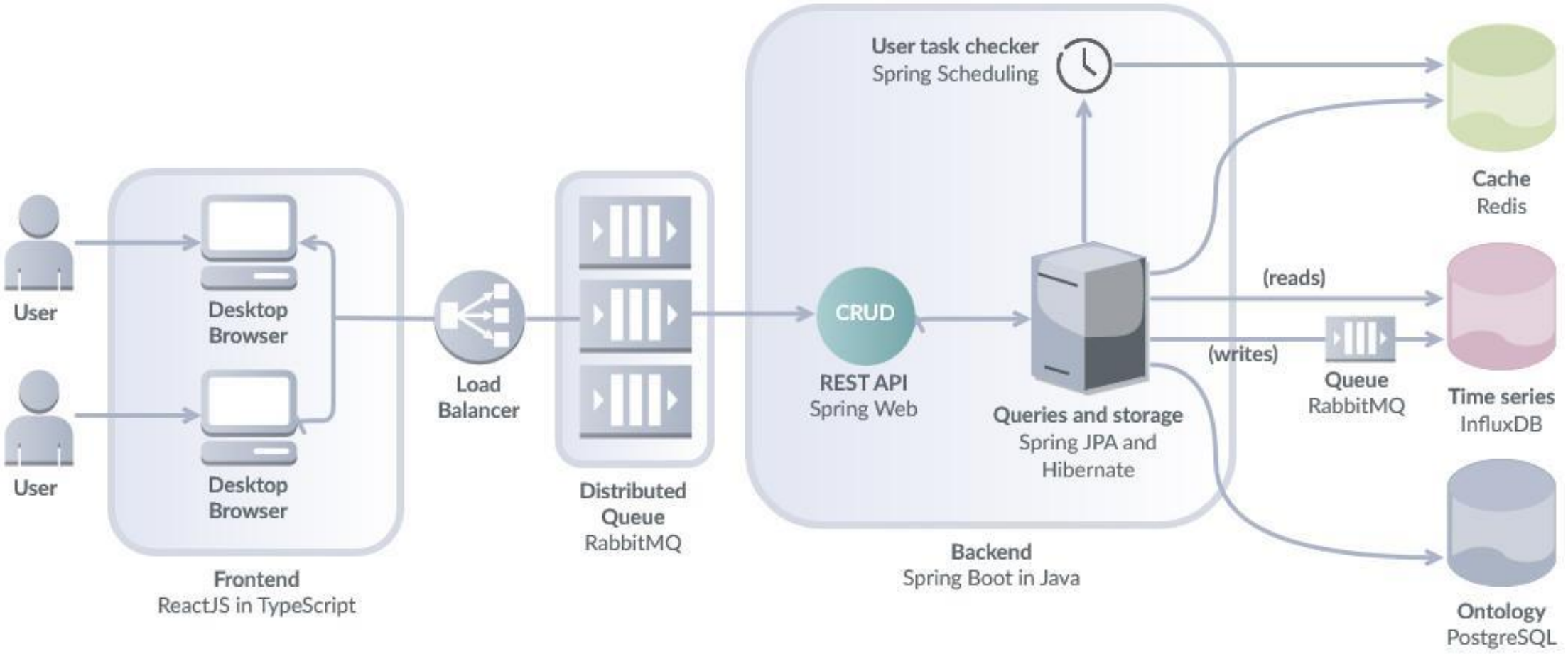


Proposal Data management

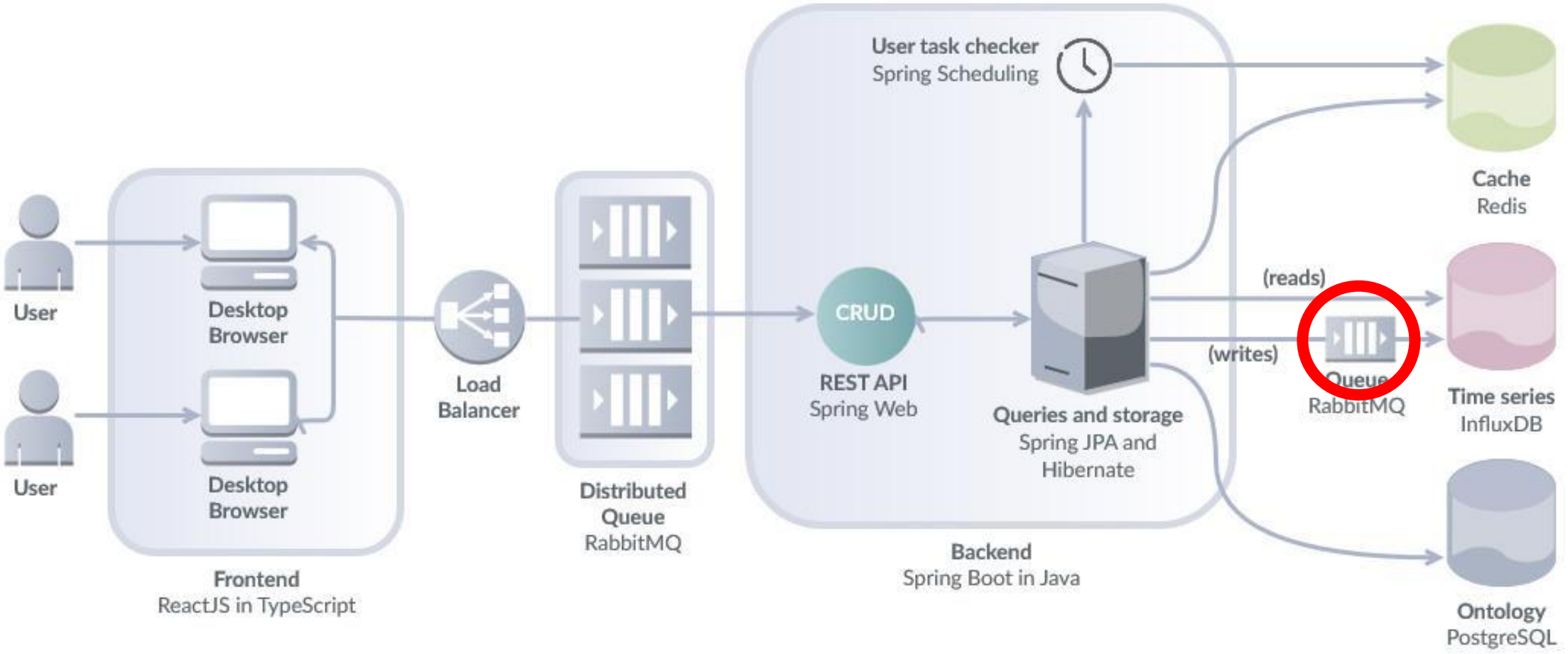
- overall traffic workload is distributed, but querying simultaneous data types can lead to bottlenecks;
- links are added on each data point and propagated to the TSDBMS on ontology updates.



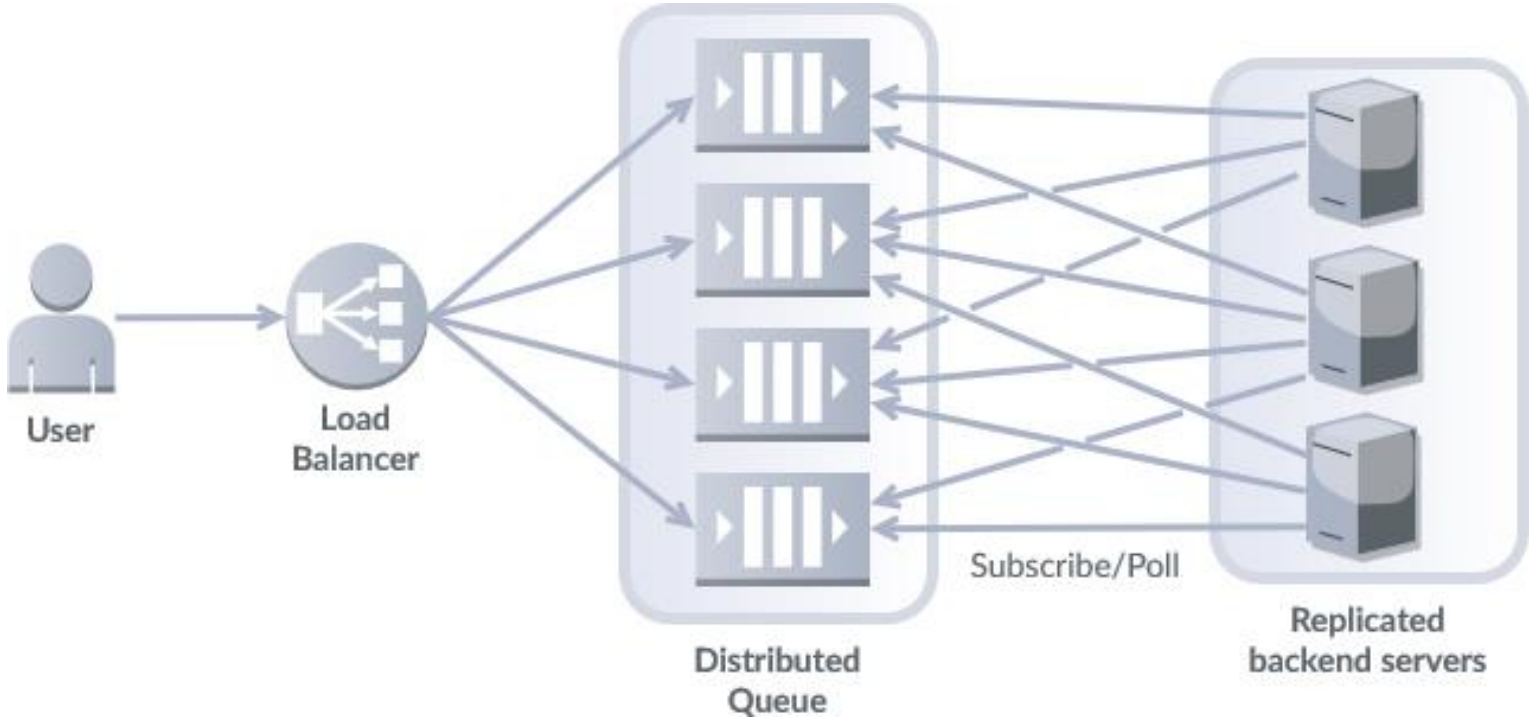
Proposal Architecture



Proposal Architecture

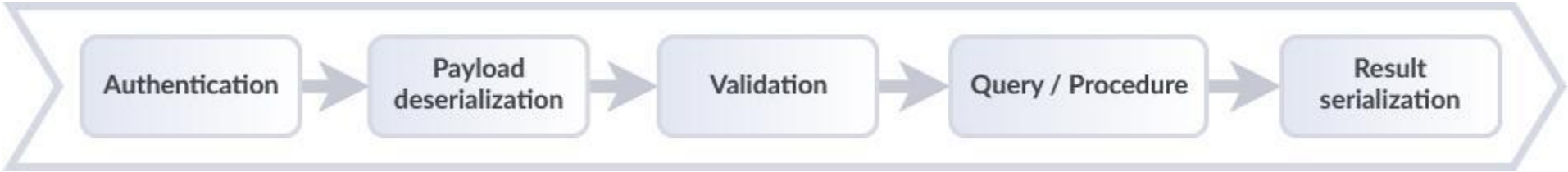


Proposal Architecture



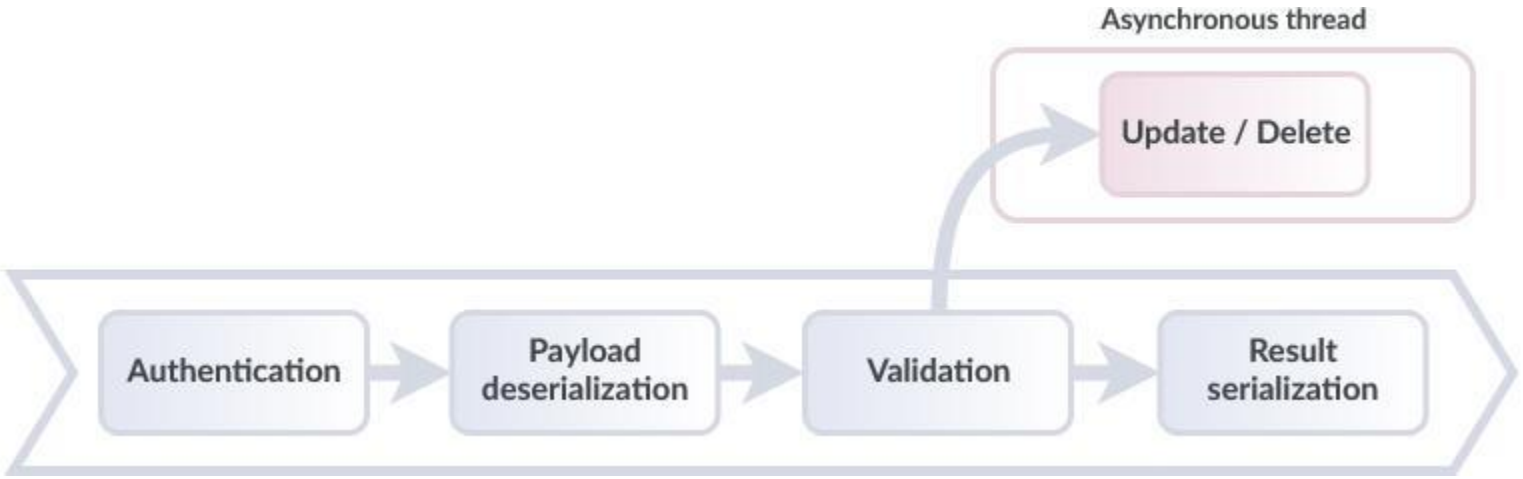
Proposal Architecture

- the backend opens processing pipelines for each request;
- authentication:
  - auth. session tokens are JWTs with an expiration date.
- validation stage checks for invalid contents or constraint violations



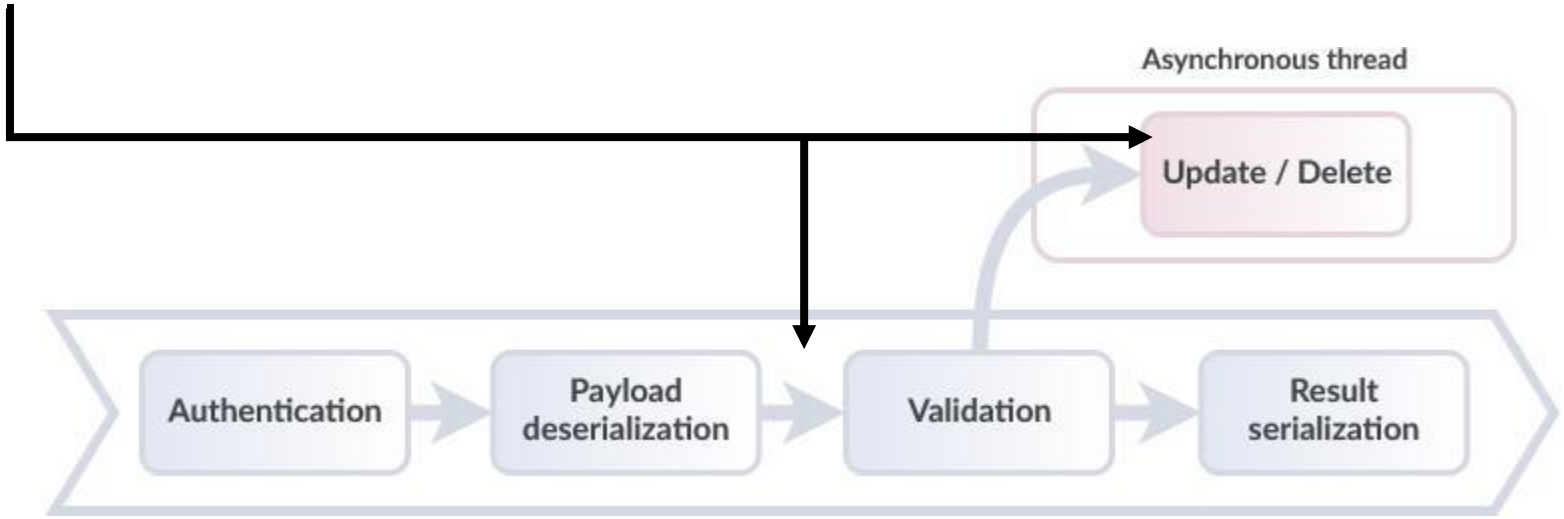
Proposal Architecture

- updates, deletions and rollbacks are made asynchronously:
  - user receives a simulated snapshot with proposed changes;
  - validation stage ensures that the update will likely be committed;
  - **caveat:** unexpected errors cannot be sent to the user.



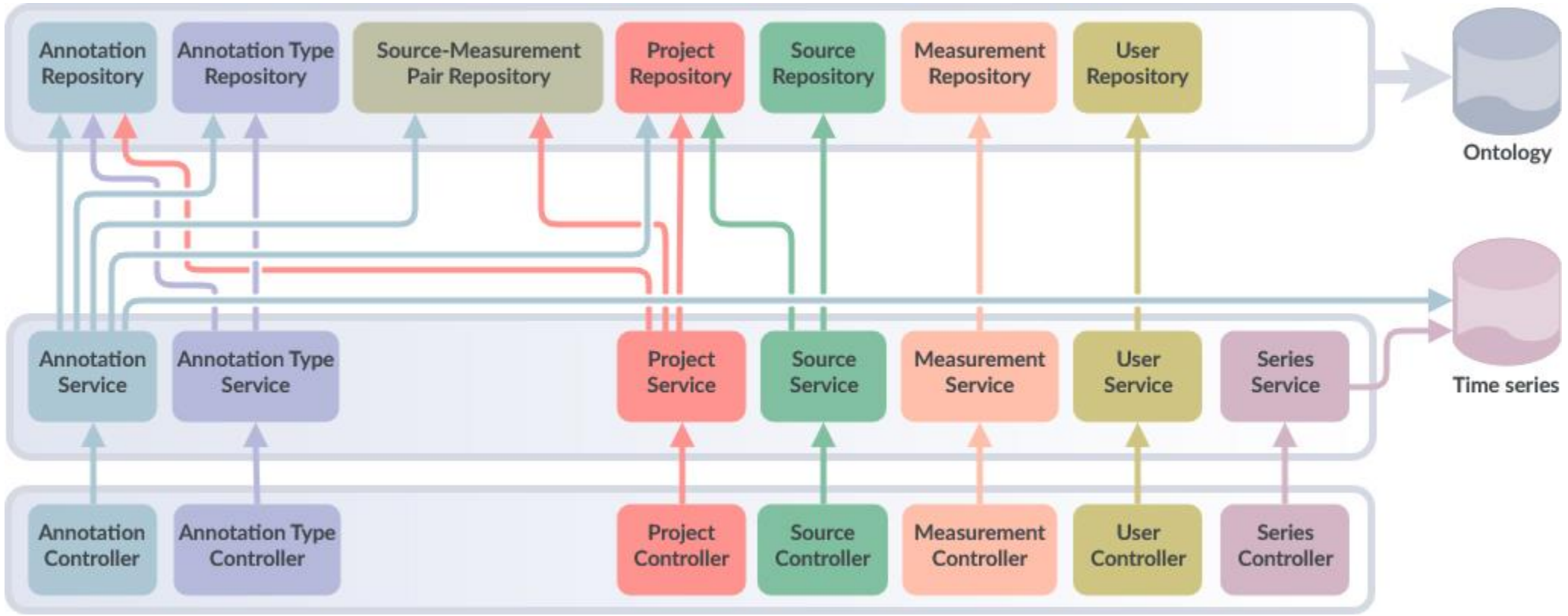
Proposal Architecture

- users make changes based on the observed data;
- if two users update the same record at the same time -> **race condition!!!**;
- optimistic-locking: last-modified dates checksum



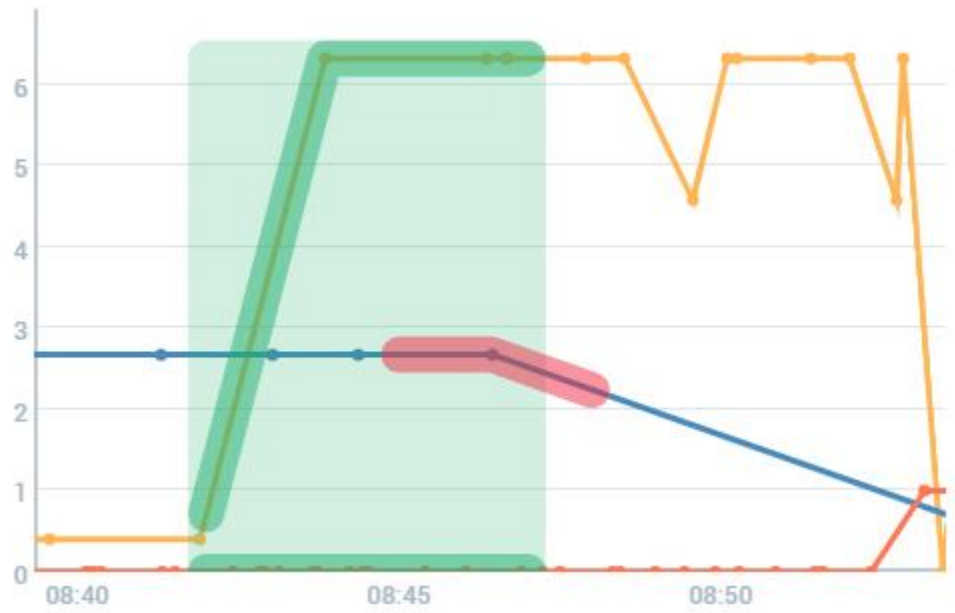
Proposal Architecture

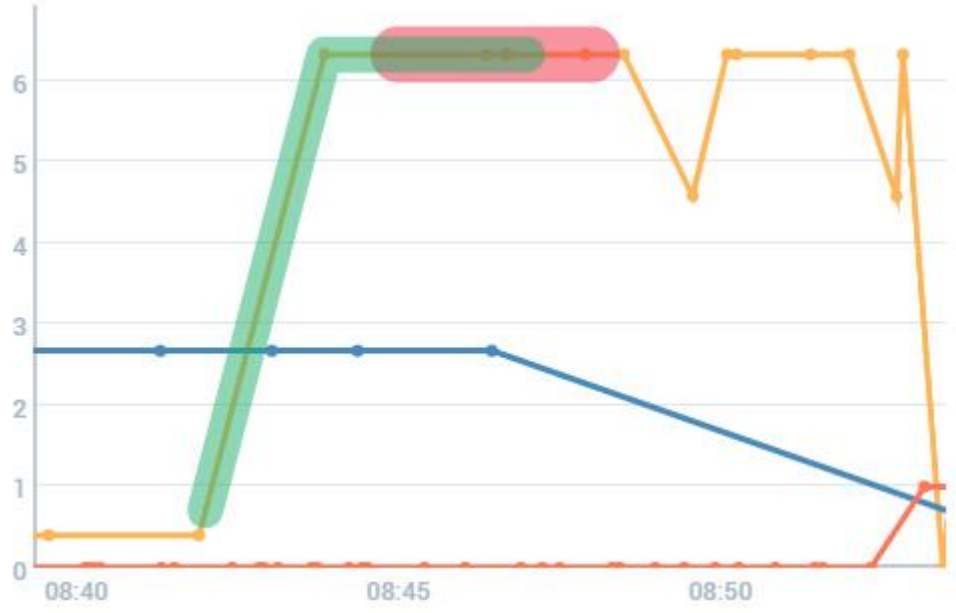
- Spring JPA provides abstraction layers for PostgreSQL queries (hot-swap)



Proposal **Annotations**

- snakes: arcs traced over series' curves;
- paint over existing points, interpolate when in-between;
- intersection handling (nesting).

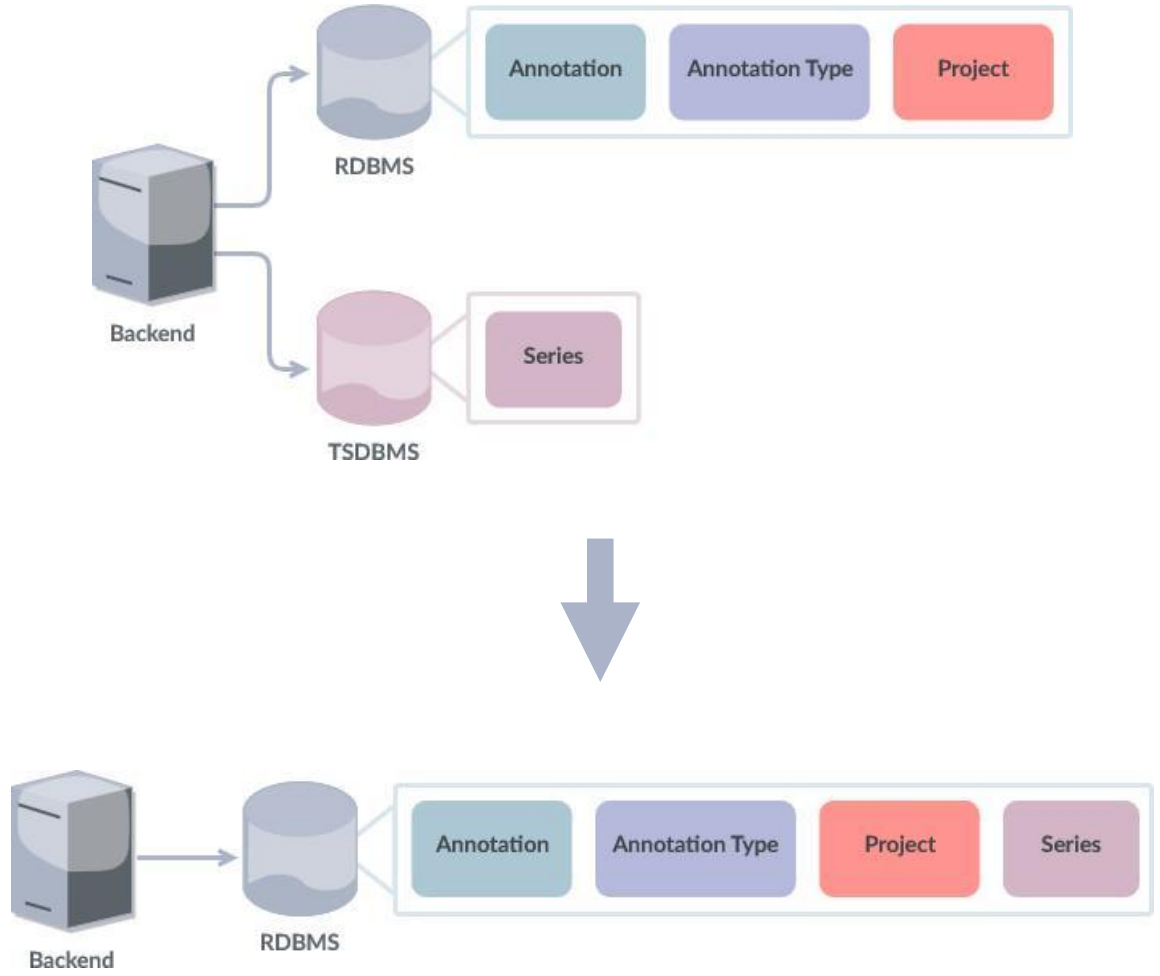




DEMO

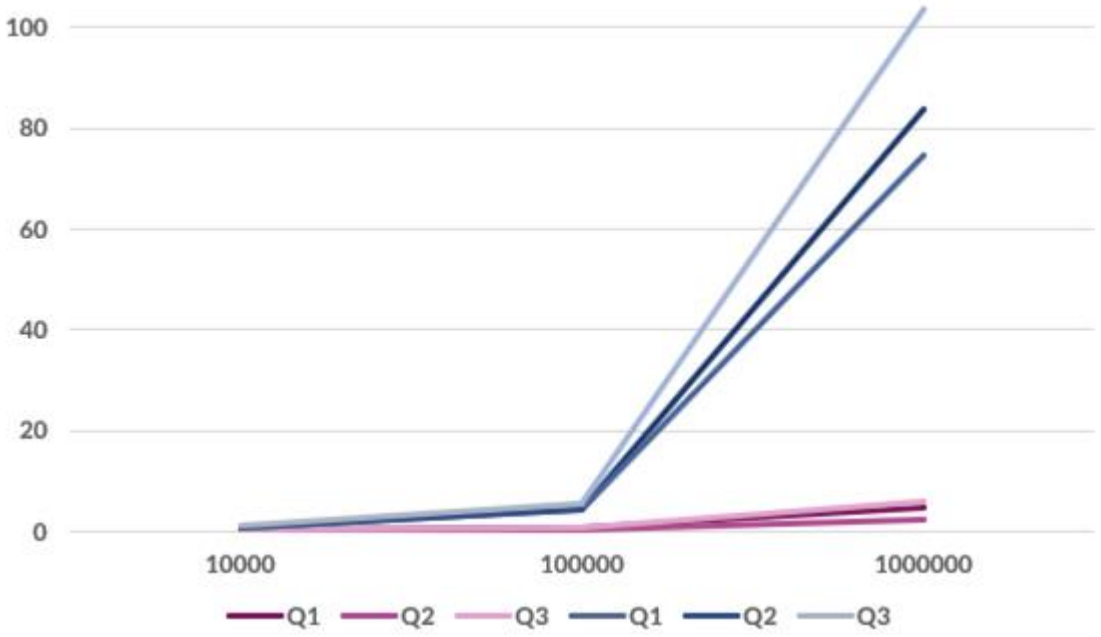
Evaluation Time series in PostgreSQL

- as granularity increases, Consistency is harder to attain;
- put all data in a single ACID-compliant RDBMS:
  - linking logic is built-in through the relational model;
  - better Consistency handling.
- benchmark read-write performance

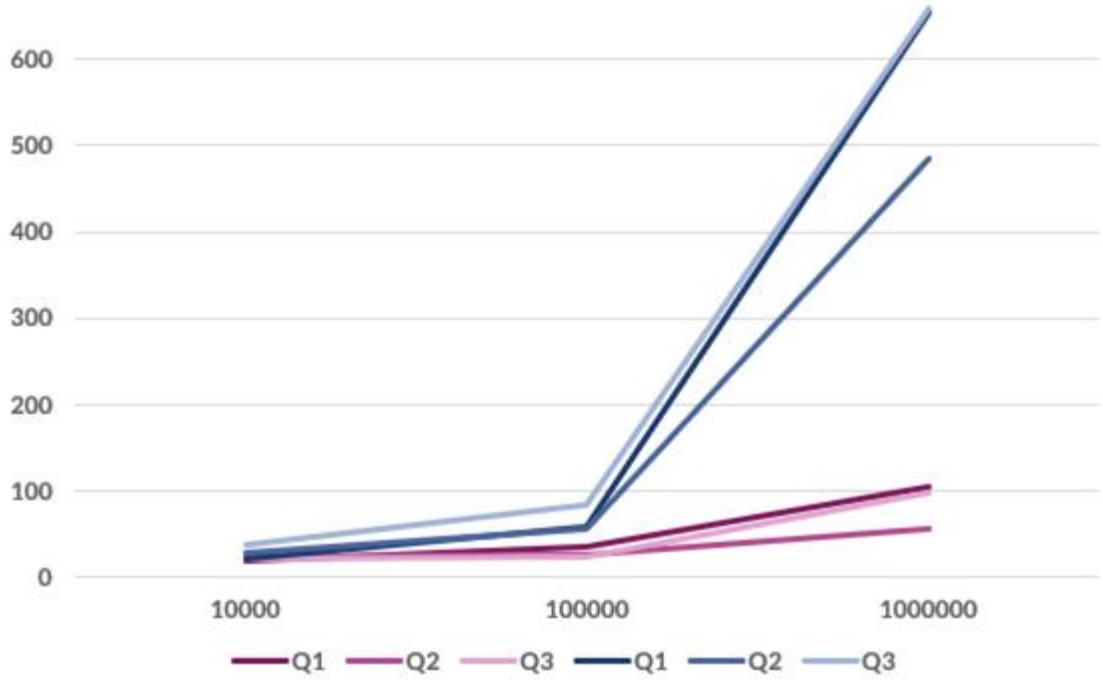


Evaluation Time series in PostgreSQL

READ PERFORMANCE

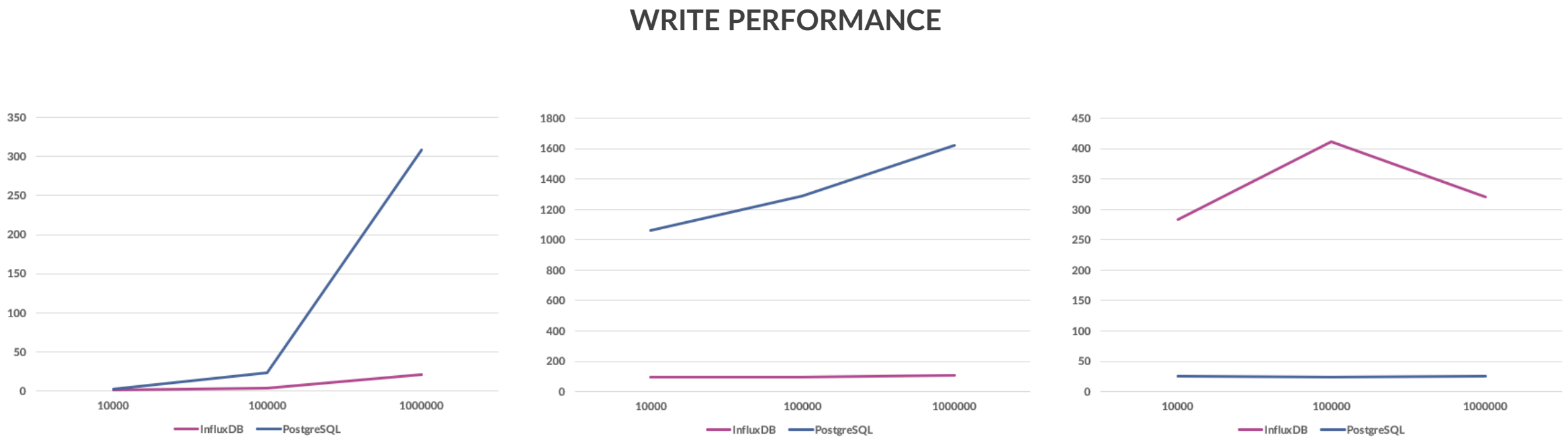


CPU (%)



Request time (ms)

Evaluation Time series in PostgreSQL



Write time (seconds)

Disk usage (MB)

RAM usage (MB)

Conclusion

- improved collaboration workflow:
  - enhanced model for building smaller scopes of analysis;
  - better visualization for comparison of data;
  - stronger annotation readability and flexibility of expression;
  - scalable architecture that adjusts to data set size and traffic amount;
  - linearizability and strongly validated contributions;
- the open REST API enables extensibility: more input and output modules can be added.

END

https://www.edduarte.com/time-series-analysis-platform/