

BEAM  
SUMMIT

# Deduplicating And Analysing Time-Series Data With Apache Beam And QuestDB

# About me: I like databases & open source

**The SQL revenge/ the realtime database/ the embedded database**

2022- today. Developer relations at an open source database vendor

- QuestDB, PostgreSQL, MongoDB, Timescale, InfluxDB, Apache Flink

**The streaming era/ The database as a service singularity**

2019-2022. Data & Analytics specialist at a cloud provider

- Amazon Aurora, Neptune, Athena, Timestream, DynamoDB, DocumentDB, Kinesis Data Streams, Kinesis Data Analytics, Redshift, ElastiCache for Redis, QLDB, Elasticsearch, OpenSearch, Cassandra, Spark...

**The hadoop dark ages / The python hegemony/ The cloud database big migrations**

2013-2018. Data Engineer/Big Data & Analytics consultant

- PostgreSQL, Redis, Neo4j, Google BigQuery, BigTable, Google Cloud Spanner, Apache Spark, Apache BEAM, Apache Flink, HBase, MongoDB, Presto

**The libre and open SQL revolution / The NoSQL rise**

2006-2012 - Web developer

- MySQL, Redis, PostgreSQL, Sqlite, Elasticsearch

**The licensed SQL period**

late nineties to 2005. Desktop/CGI/Servlets/ EJBs/CORBA

- MS Access, MySQL, Oracle, Sybase, Informix

**The pre-SQL years**

As a student/hobbyist (late eighties - early nineties)

- Amsbase, DBase III, DBase IV, Foxpro, Microsoft Works, Informix



- The problem of data duplication
- The problem of data duplication
- The problem of data duplication
- The problem of data duplication
- Behold: a dashboard!
- The many challenges of time-series data
- QuestDB to the rescue
- Down the rabbit hole of writing a custom BEAM Sink
  - Finding several needles on a documentation haystack
  - When I sadly discovered Python streaming support is meh
  - The unsung hero saves the day (again): implementing the Sink in Java

WHY

HOW

Duplication

WHHAT



If you can use only one  
database for everything, go  
with PostgreSQL\*

\* Or any other major and well supported RDBMS



Imagine...

a factory floor with 500 machines, or

a fleet with 500 vehicles, or

50 trains, with 10 cars each, or

500 users with a mobile phone, or

500 financial instruments generating tick data

...sending data every second

## A conventional database's nightmare

43,200,000 rows a day.....

302,400,000 rows a week....

1,314,144,000 rows a month

Timestamps are hard



**Rashid Ashraf**

@rshidashrf



Elon Musk: I'm putting people on Mars!  
Developer: Fantastic, more timezone to support.  
[#ElonMusk](#)

4:57 AM · Mar 19, 2018



# Working with timestamped data in a database is tricky\*

\* specially working with analytics of data changing over time or at a high rate

master

25 branches

79 tags

Go to file

Code

## About

An open source time-series database for fast ingest and SQL queries

## questdb.io

[java](#)
[iot](#)
[postgres](#)
[sql](#)
[database](#)
  
[big-data](#)
[time-series](#)
[analytics](#)
[cpp](#)
  
[grafana](#)
[postgresql](#)
[simd](#)
  
[low-latency](#)
[financial-analysis](#)
[tsdb](#)
  
[hacktoberfest](#)
[time-series-database](#)
  
[questdb](#)

## Readme

Apache-2.0 license

Code of conduct

Security policy

11.6k stars

123 watching

792 forks

Report repository

## Releases 67

7.2 Latest  
5 days ago

+ 66 releases

## Contributors 114



+ 103 contributors

## Languages

jerrinot fix(pgwire): gracefully handle a client disconnecting during aut...	... ✓ c67880e 11 hours ago	🕒 4,152 commits
📁 .github	docs(core): updating benchmark image in readme (#3141)	3 months ago
📁 .idea	fix(wal): fix wal table suspension on a race condition (#3019)	4 months ago
📁 artifacts	build: 7.1.3 (#3370)	3 weeks ago
📁 benchmarks	build: 7.2 (#3466)	4 days ago
📁 ci	ci(build): split windows griffin test run in CI into 2 (#3440)	2 weeks ago
📁 core	fix(pgwire): gracefully handle a client disconnecting during authenti...	11 hours ago
📁 examples	build: 7.2 (#3466)	4 days ago
📁 i18n	docs(core): update Chinese README (#3329)	last month
📁 pkg/ami/marketplace	feat(http): introduce configuration for health check pessimism (#32...	last month
📁 utils	build: 7.2 (#3466)	4 days ago
📁 win64svc	feat(core): deterministically deposit hs_err_pid files into db dire...	last year
📄 .all-contributorsrc	docs: add sucongou as a contributor for bug (#2383)	last year
📄 .git-blame-ignore-revs	chore(build): git blame to ignore the reformatting commit (#2880)	6 months ago
📄 .gitignore	feat(core): make partitions attached via soft link read-only, protect...	5 months ago
📄 CODEOWNERS	chore: switch to team-based codeowners (#1754)	last year
📄 CODE_OF_CONDUCT.md	chore(docs): add Prettier formatting to project files (#1720)	2 years ago
📄 CONTRIBUTING.md	docs(core): add code formatting info to contributing guide (#2784)	7 months ago
📄 LICENSE.txt	fix: license changed to Apache 2.0. Fixed #80	4 years ago
📄 README.md	docs(core): updating readme to include a link to the tsdb glossary ...	last week
📄 SECURITY.md	docs(core): add SECURITY policy (#2629)	8 months ago
📄 examples.manifest.yaml	docs(ilp): add an example with auth, but without TLS (#2455)	10 months ago
📄 pom.xml	build: 7.2 (#3466)	4 days ago

☰ README.md

## We'd like to be known for

- Performance
  - Better performance with smaller machines
- Developer Experience
- Proudly Open Source (Apache 2.0)

A quick overview of some  
interesting queries

Try it live on <https://demo.questdb.io>

## WHERE ... TIME RANGE

```
SELECT * from trips WHERE pickup_datetime in '2018';
```

```
SELECT * from trips WHERE pickup_datetime in '2018-06';
```

```
SELECT * from trips WHERE pickup_datetime in '2018-06-21T23:59';
```

---

```
SELECT * from trips WHERE pickup_datetime in '2018;2M' LIMIT -10;
```

```
SELECT * from trips WHERE pickup_datetime in '2018;10s' LIMIT -10;
```

```
SELECT * from trips WHERE pickup_datetime in '2018;-3d' LIMIT -10;
```

---

```
SELECT * from trips WHERE pickup_datetime in '2018-06-21T23:59:58;4s;1d;7'
```

```
SELECT * from trips WHERE pickup_datetime in '2018-06-21T23:59:58;4s;-1d;7'
```



## SAMPLE BY

Try it live on <https://demo.questdb.io>

Aggregates data in homogeneous time chunks

```
SELECT timestamp, min(tempF),  
max(tempF), avg(tempF)  
FROM weather SAMPLE BY 1M;
```

---

```
SELECT  
  timestamp,  
  sum(price * amount) / sum(amount) AS vwap_price,  
  sum(amount) AS volume  
FROM trades  
WHERE symbol = 'BTC-USD' AND timestamp > dateadd('d', -1, now())  
SAMPLE BY 15m ALIGN TO CALENDAR;
```

## SAMPLE BY ... FILL

Try it live on <https://demo.questdb.io>

Can fill missing time chunks using different strategies (NULL, constant, LINEAR, PREVIOUS value)

```
SELECT
  timestamp,
  sum(price * amount) / sum(amount) AS vwap_price,
  sum(amount) AS volume
FROM trades
WHERE symbol = 'BTC-USD' AND timestamp > dateadd('d', -1, now())
SAMPLE BY 1s FILL(NULL) ALIGN TO CALENDAR;
```

LATEST ON ...

Try it live on <https://demo.questdb.io>

PARTITION BY ...

Retrieves the latest entry by timestamp for a given key or combination of keys, for scenarios where multiple time series are stored in the same table.

```
SELECT * FROM trades
WHERE symbol in ('BTC-USD', 'ETH-USD')
LATEST ON timestamp PARTITION BY symbol, side;
```

## ASOF JOIN / LT JOIN

Try it live on <https://demo.questdb.io>

## SPLICE JOIN

ASOF JOIN joins two different time-series measured. For each row in the first time-series, the ASOF JOIN takes from the second time-series a timestamp that meets both of the following criteria:

- The timestamp is the closest to the first timestamp.
- The timestamp is strictly prior or equal to the first timestamp.

```
WITH trips2018 AS (  
  SELECT * from trips WHERE pickup_datetime in '2016'  
)  
SELECT pickup_datetime, fare_amount, tempF, windDir  
FROM trips2018  
ASOF JOIN weather;
```

QuestDB cannot do in-stream deduplications.

Apache BEAM can help

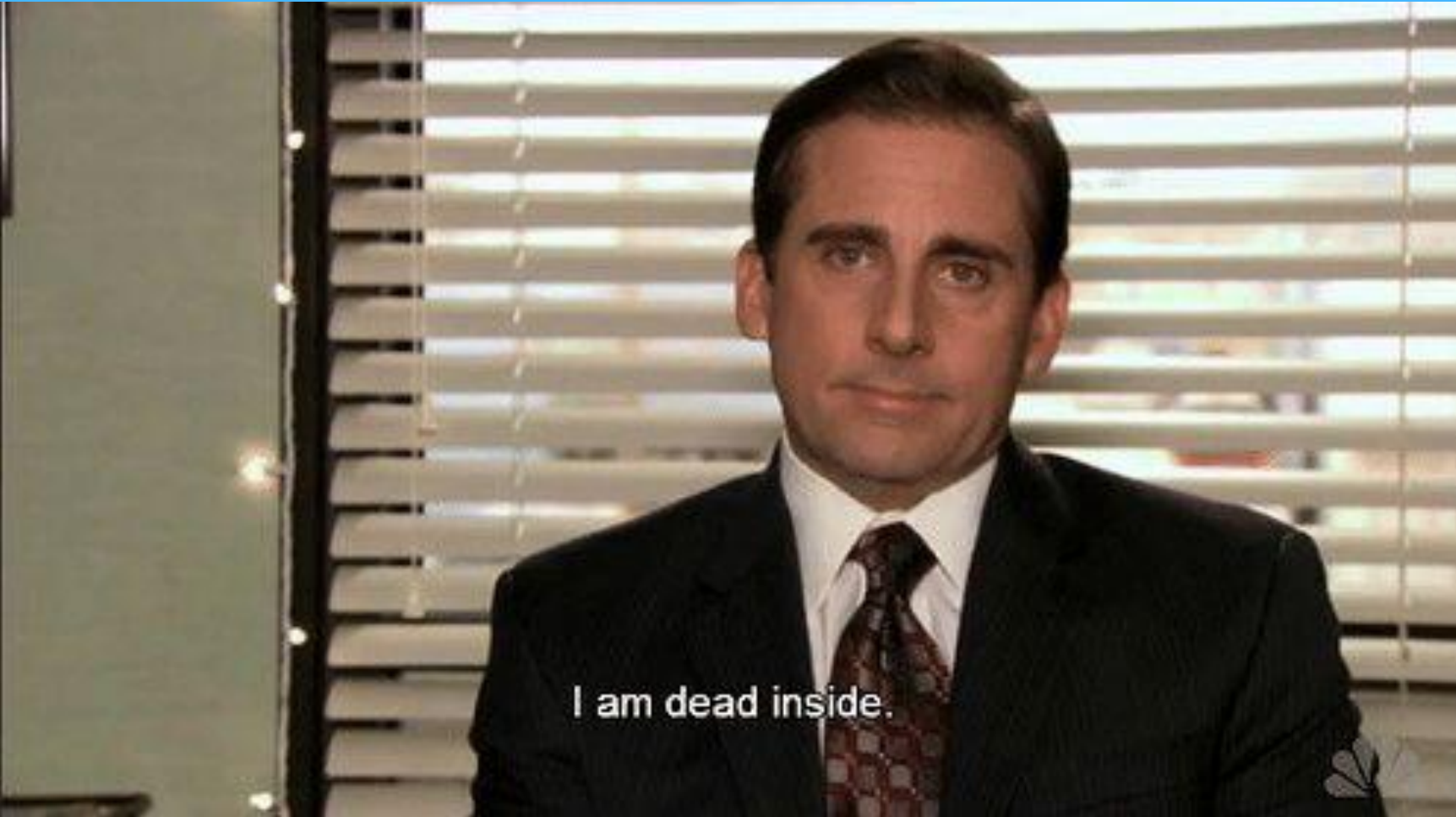
# The Python QuestDB Sink

- WriteToQuestDB(PTransform) class
  - Receives the args you need to pass to the sink
  - Implements the `expand` method, which receives the PCollection then invokes `ParDo` to `_WriteToQuestDBFn`
  -
- `_WriteToQuestDBFn(DoFn)` class
  - Instantiates `_QuestDBSink` on `start_bundle`
  - Flushes/releases `_QuestDBSink` on `finish_bundle`
  - Implements `display_data` to show info on the UI
  - Calls to `_QuestDBSink.write` on the `process` method
  -
- `_QuestDBSink` class
  - Deals with the QuestDB connection itself

## The Python QuestDB Sink

<https://github.com/javier/questdb-beam/tree/main/python>

```
pcoll | WriteToQuestDB(table,  
    symbols=[list_of_symbols],  
    columns=[list_of_columns],  
    host=host,  
    port=port,  
    batch_size=optionalSizeOfBatch,  
    tls=optionalBoolean,  
    auth=optionalAuthDict)
```

A medium shot of Steve Carell as Michael Scott from the TV show 'The Office'. He is wearing a dark suit, a white shirt, and a patterned tie. He has a deadpan, somewhat weary expression. The background consists of horizontal window blinds. The entire image is set against a solid blue background.

I am dead inside.



- QuestDbIO.Write class, extends PTransform
  - Receives the args you need to pass to the sink
  - Uses @AutoValue to generate classes “magically”
  - Implements the expand method, which receives the PCollection then invokes ParDo to QuestDbIO.Write.WriteFn (with optional deduplication)
  - Implements populateDisplayData
  
- QuestDbIO.Write.WriteFn class, extends DoFn
  - Instantiates QuestDBSender on start\_bundle
  - Flushes/closes QuestDBSender on finish\_bundle
  - Parses/sends the QuestDbRow to QuestDB on the process method

# Where the magic happens

<https://github.com/javier/questdb-beam/blob/main/java/src/main/java/org/apache/beam/sdk/io/questdb/QuestDbIO.java>

```
keydAndWindowed = (PCollection) input.apply(WithKeys.of(new SerializableFunction<QuestDbRow, String>() {
    @Override
    public String apply(QuestDbRow r) {
        return String.valueOf(r.hashCode());
    }
}));

PCollection windowedItems = (PCollection)
    keydAndWindowed.apply(
        Window.
            <KV<String, String>>into(
                Sessions.
                    withGapDuration(
                        Duration.standardSeconds(deduplicationDurationMillis())
                    )
            )
    );

PCollection<QuestDbRow> uniqueRows = (PCollection<QuestDbRow>)
    ((PCollection) keydAndWindowed.apply(
        Deduplicate.keyedValues()
    )
    ).apply(Values.create());
```

<https://github.com/javier/questdb-beam/tree/main/java>

```
// pcoll needs to be of type QuestDbRow

pcoll.apply(ParDo.of(new LineToMapFn()));
  parsedLines.apply(QuestDbIO.write()
    .withUri("your-instance-host.questdb.com:YOUR_PORT")
    .withTable("beam_demo")
    .withDeduplicationEnabled(true)
    .withDeduplicationByValue(false)
    .withDeduplicationDurationMillis(5L)
    .withSSLEnabled(true)
    .withAuthEnabled(true)
    .withAuthUser("admin")
    .withAuthToken("verySecretToken"))
```



**SELF-HOSTED OR FULLY MANAGED?**

JAKE-CLARK.TUMBLR

imgflip.com

<https://github.com/questdb/questdb>

<https://cloud.questdb.com>

A promotional graphic for QuestDB. It features a dark background with stylized clouds in shades of blue and purple. A pink paper airplane is flying towards the right. Below it, a pink arrow points to a pink cloud shape containing the text "\$200 Free Credits". At the bottom, the QuestDB logo is displayed, consisting of a stylized 'Q' icon and the text "QuestDB".

<https://cloud.questdb.com/>

**Sign up and claim**

**\$200 Free Credits**

 **QuestDB**

Thanks!

# QUESTIONS?

Javier Ramirez  
@supercoco9

<https://github.com/javier/questdb-beam>

<https://github.com/javier/questdb-quickstart>

<https://github.com/questdb/questdb>

<https://demo.questdb.io>

<https://cloud.questdb.com>