Change Data Capture With Flink SQL and Debezium

Marta Paes (@morsapaes)
Developer Advocate

About Ververica

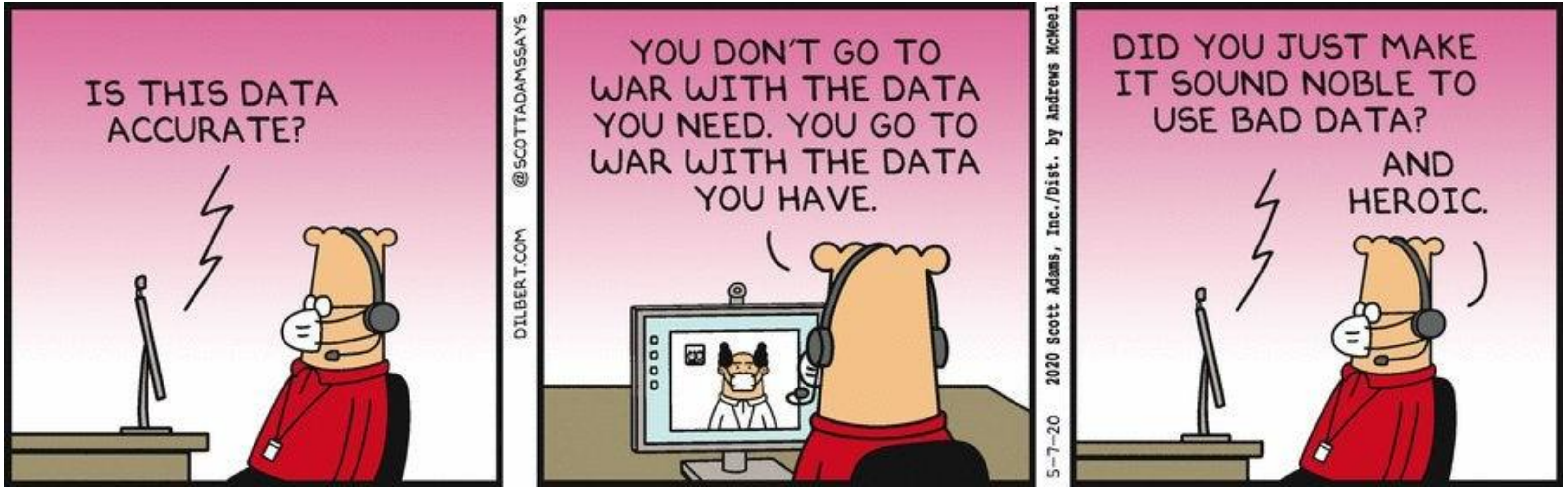
Original Creators of
Apache Flink®

Enterprise Stream Processing
With Ververica Platform

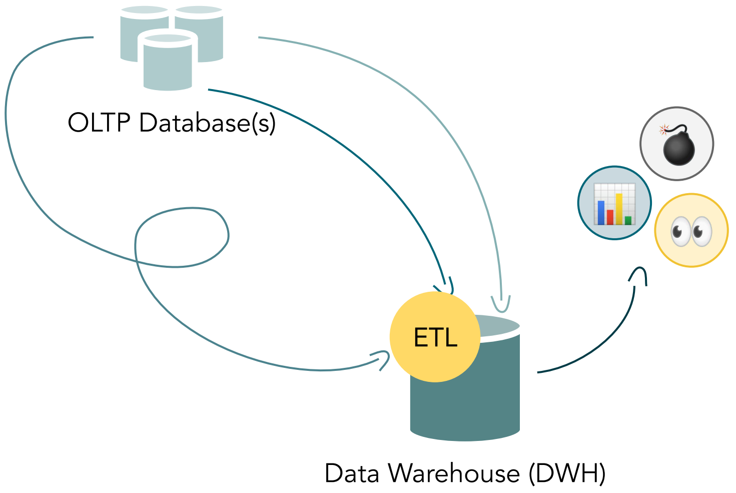
Part of
Alibaba Group

Try out Ververica Platform Community Edition (free forever!): https://www.ververica.com/platform

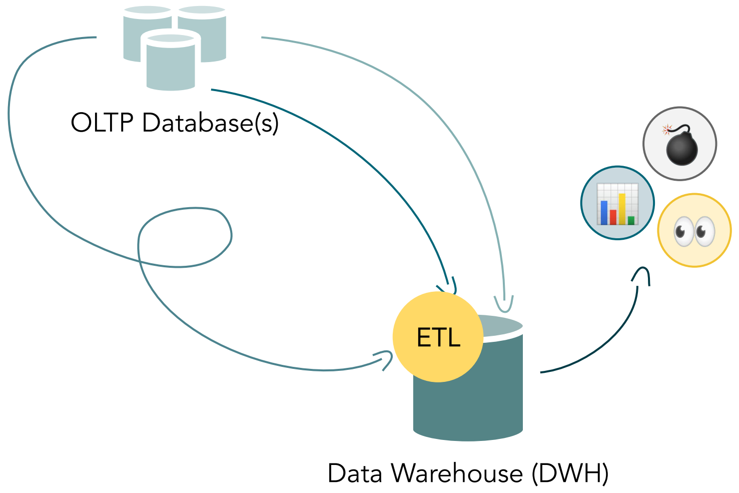
What's Wrong?


IS THIS DATA ACCURATE?
YOU DON'T GO TO WAR WITH THE DATA YOU NEED. YOU GO TO WAR WITH THE DATA YOU HAVE.
DID YOU JUST MAKE IT SOUND NOBLE TO USE BAD DATA?
AND HEROIC.
DILBERT.COM @SCOTTADAMSSAYS
5-7-20 © 2020 Scott Adams, Inc./Dist. by Andrews McMeel

The data in your DB is not dead…



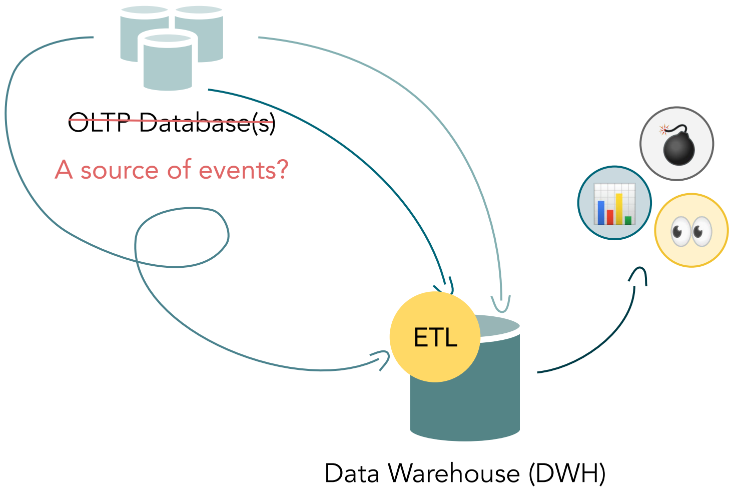
The data in your DB is not dead...



In the end:

- Most source data is continuously produced
- Most logic is not changing that frequently

The data in your DB is not dead...



In the end:

- Most source data is continuously produced
- Most logic is not changing that frequently

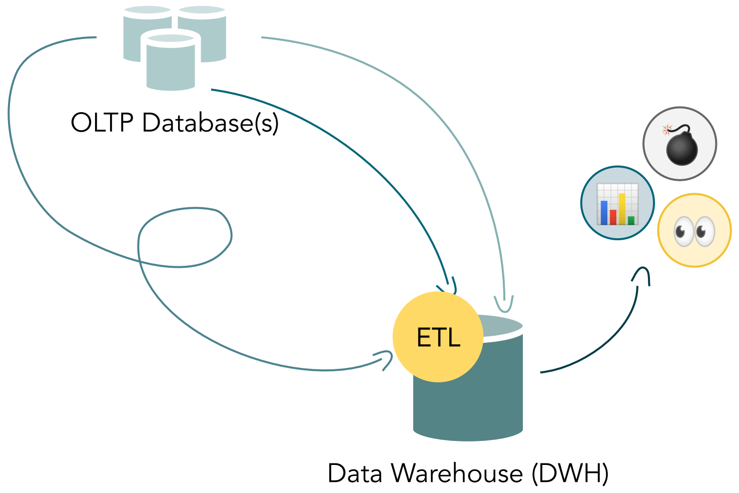
- Why are we looking at yesterday's data?
- Why are we not distributing the workload?
- Why are we letting the data go "stale"?

...but your integrations might be **killing** its value (and also some DBAs).

Can’t wait to scan a production database for changes using a 100-line query with 1000 business logic conditions.

— No one

Change Data Capture (CDC)

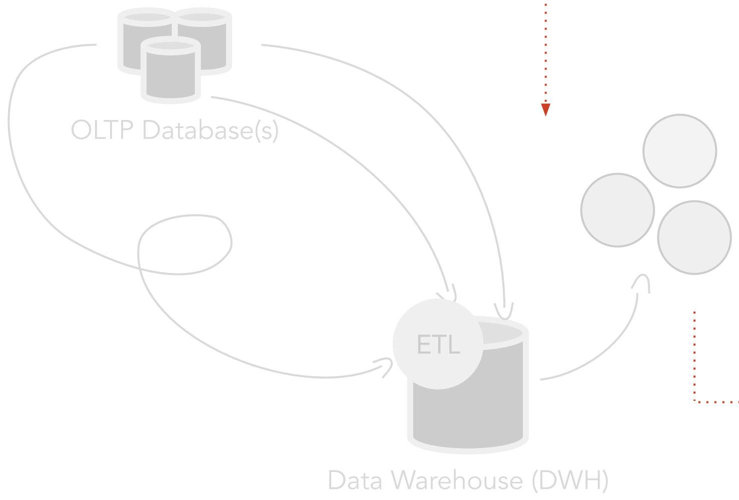


Tracking and propagating data changes in a database to downstream consumers.

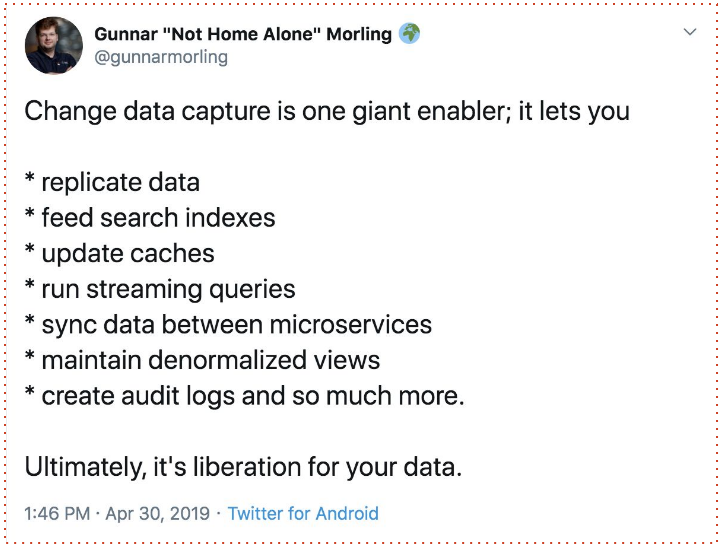
More on CDC use cases: https://speakerdeck.com/gunnarmorling/

Change Data Capture (CDC)

Can we do better?



Tracking and propagating data changes in a database to downstream consumers.



Gunnar "Not Home Alone" Morling
@gunnarmorling

Change data capture is one giant enabler; it lets you

* replicate data
* feed search indexes
* update caches
* run streaming queries
* sync data between microservices
* maintain denormalized views
* create audit logs and so much more.

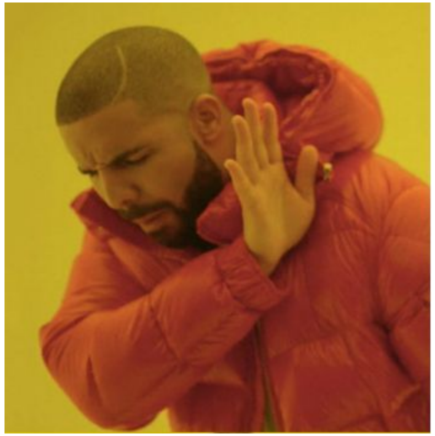
Ultimately, it's liberation for your data.

1:46 PM · Apr 30, 2019 · Twitter for Android

More on CDC use cases: https://speakerdeck.com/gunnarmorling/

Not all CDC is created equal

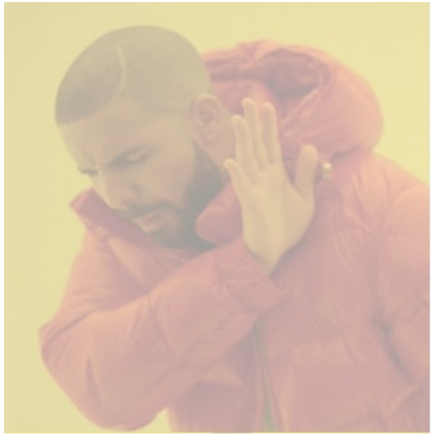
Query-based CDC

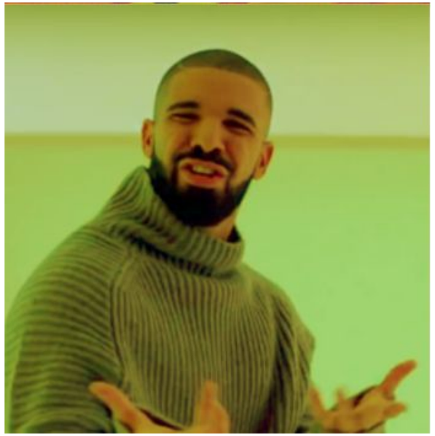


- ❌ Some data changes might get lost
- ❌ DELETE operations are not captured
- ❌ Trade-off: frequency vs. load on source DBs
- ❌ Can’t propagate schema changes

Not all CDC is created equal

Query-based CDC

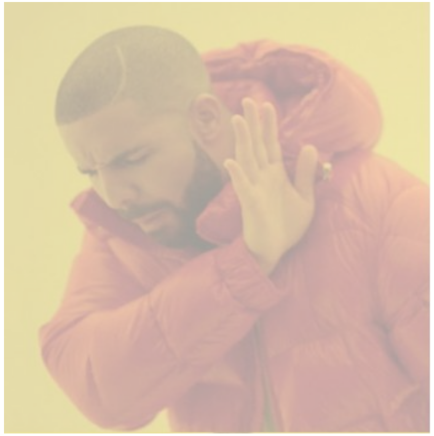


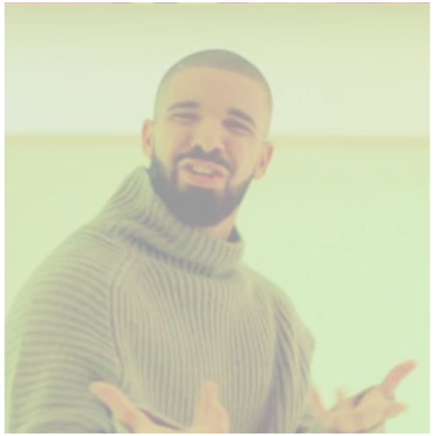


What if we tapped into the transaction log?

Not all CDC is created equal

Query-based CDC





Log-based CDC

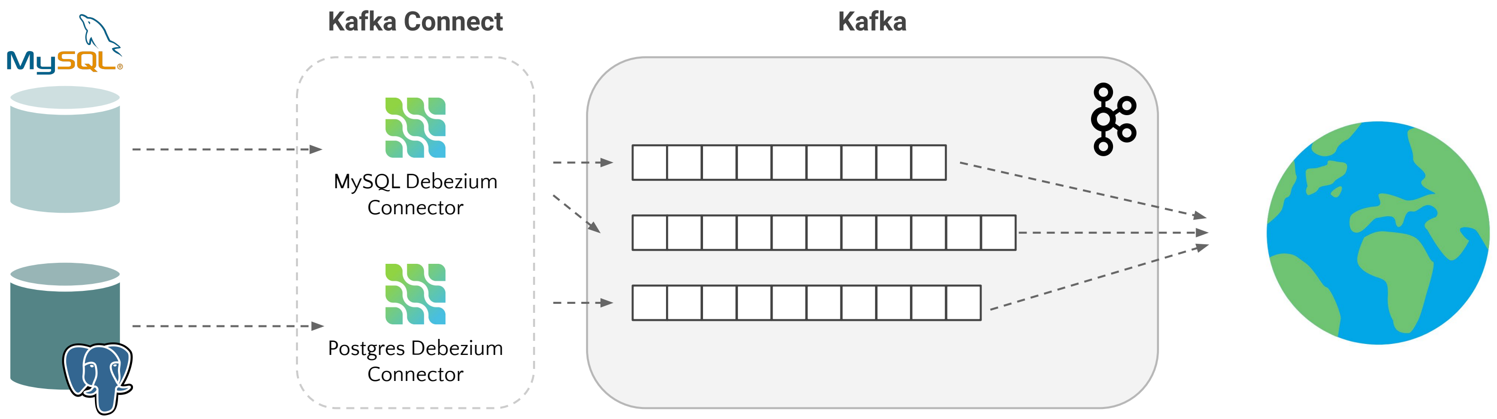
- ✓ All data changes are captured
- ✓ More context on the actual changes
- ✓ Low propagation delay (i.e. near real-time)
- ✓ Minimal impact on the source DBs

Learn more: https://debezium.io/blog/2018/07/19/advantages-of-log-based-change-data-capture/

Debezium

Debezium is an open source distributed platform for **log-based CDC**.

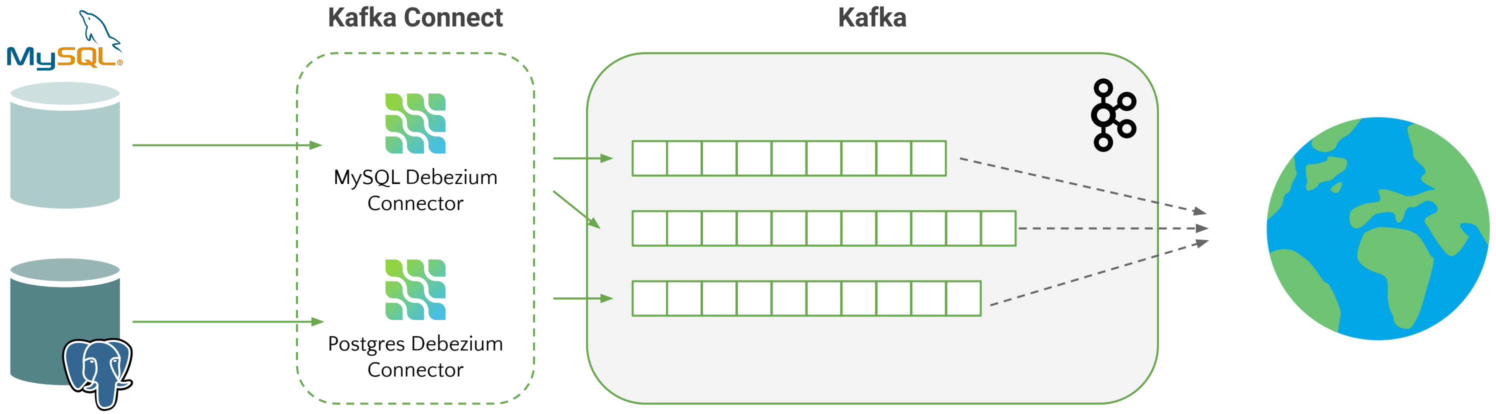
- Canonical format for change events → Different sources, same output
- Support for most common data sources (MySQL, Postgres, MongoDB, ...)



Learn more: https://debezium.io/

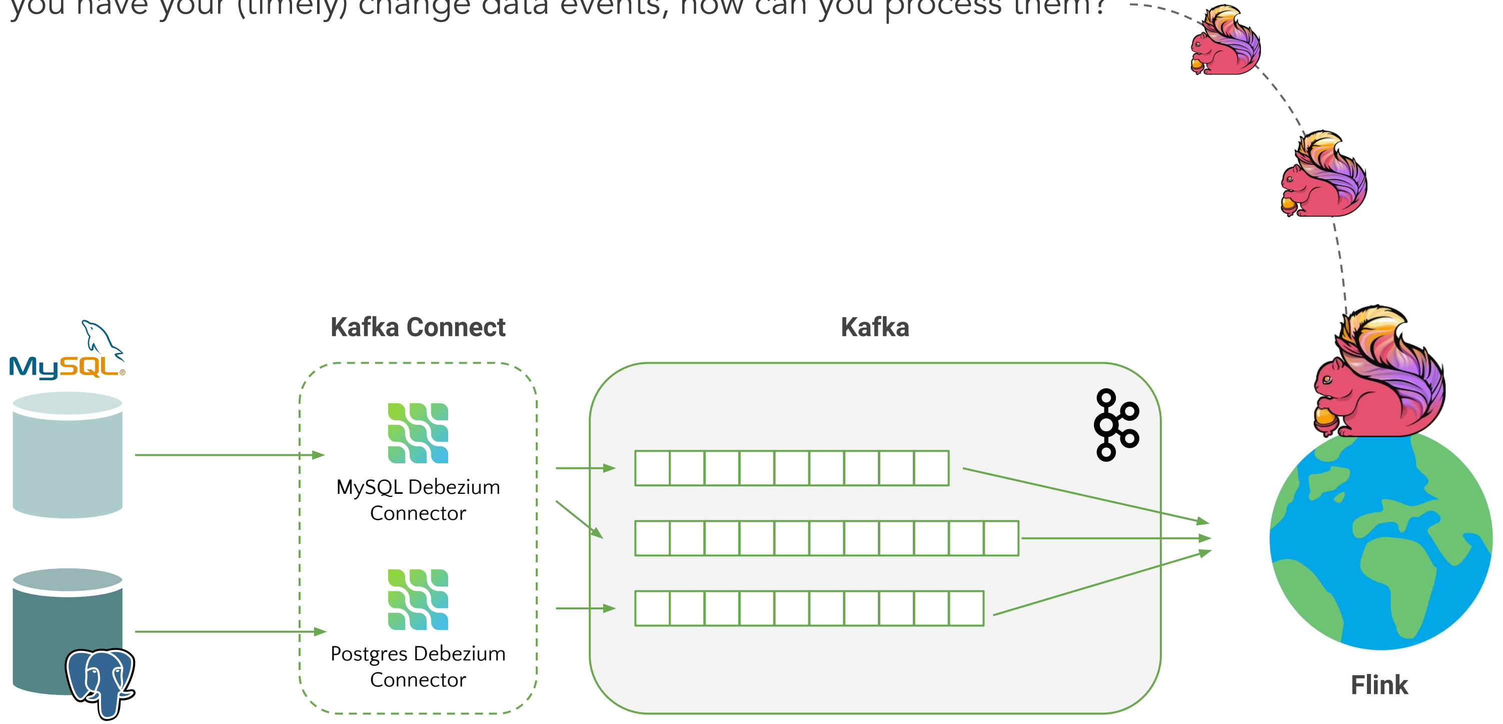
Change Data Captured. Now what?

Now that you have your (timely) change data events, how can you process them?



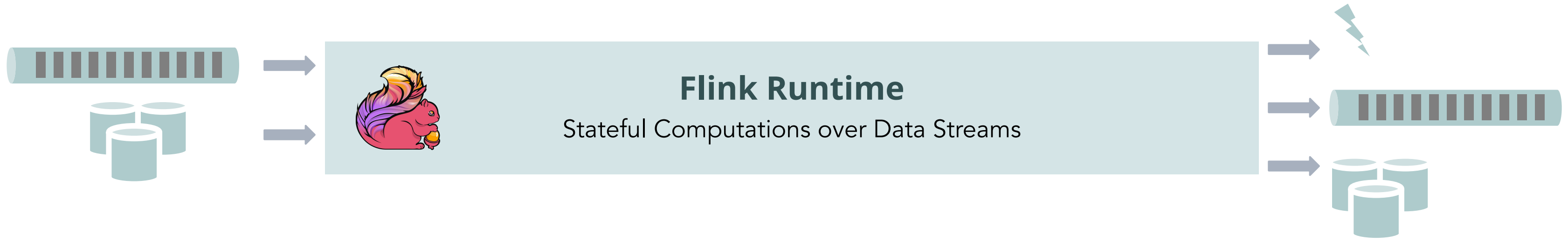
Change Data Captured. Now what?

Now that you have your (timely) change data events, how can you process them?



What is Apache Flink?

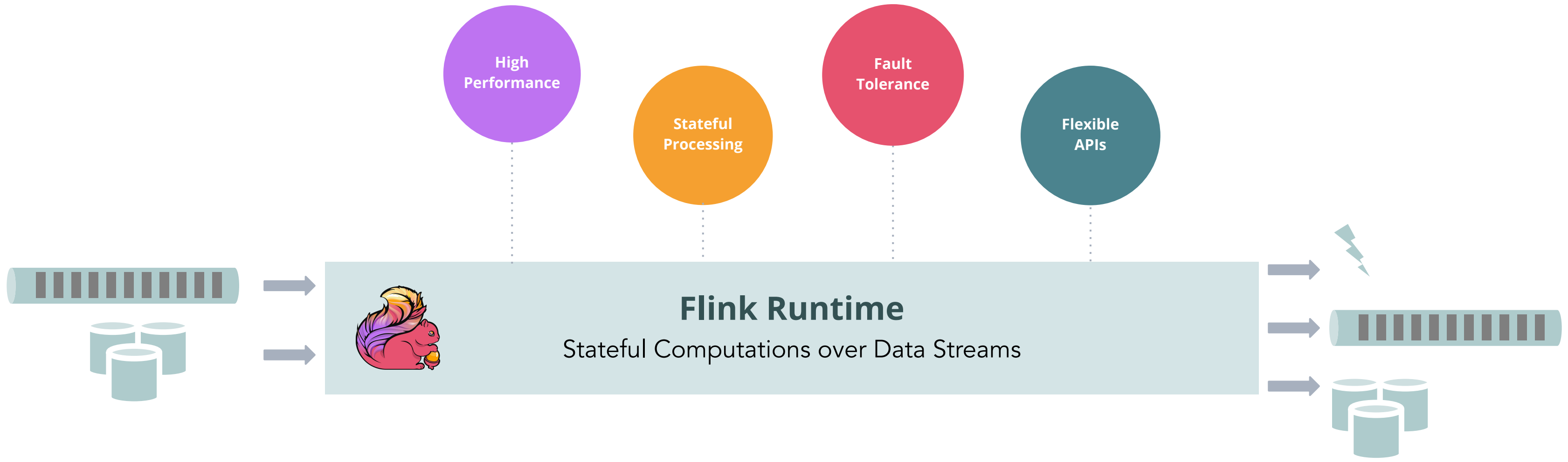
Flink is an **open source** framework and distributed engine for **stateful stream processing**.



Learn more: flink.apache.org

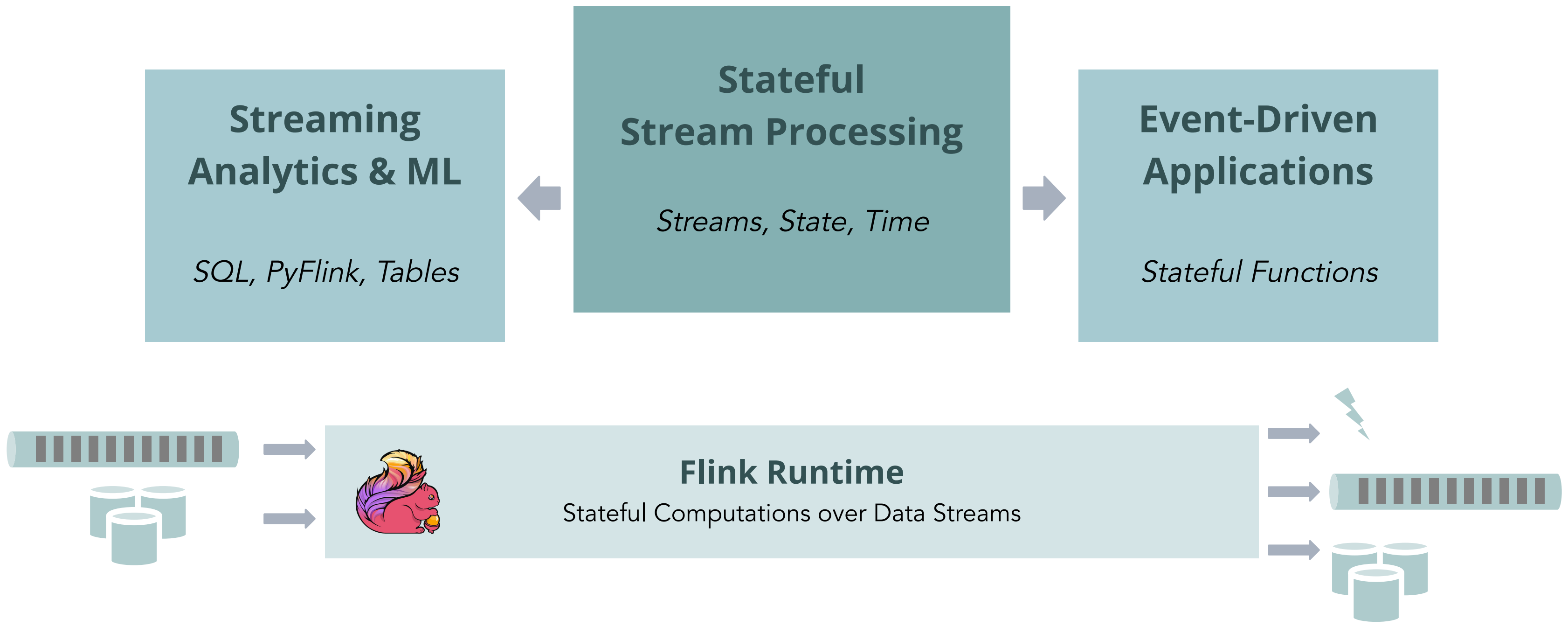
What is Apache Flink?

Flink is an **open source** framework and distributed engine for **stateful stream processing**.



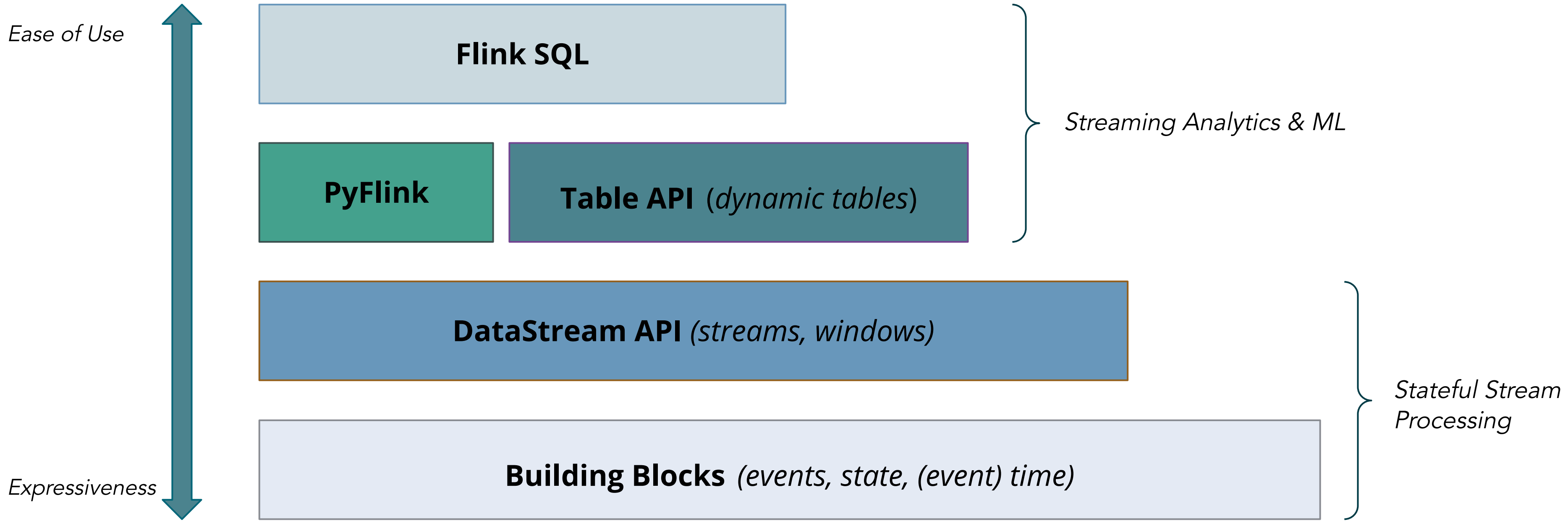
What is Apache Flink?

Flink is an **open source** framework and distributed engine for **stateful stream processing**.



The Flink API Stack

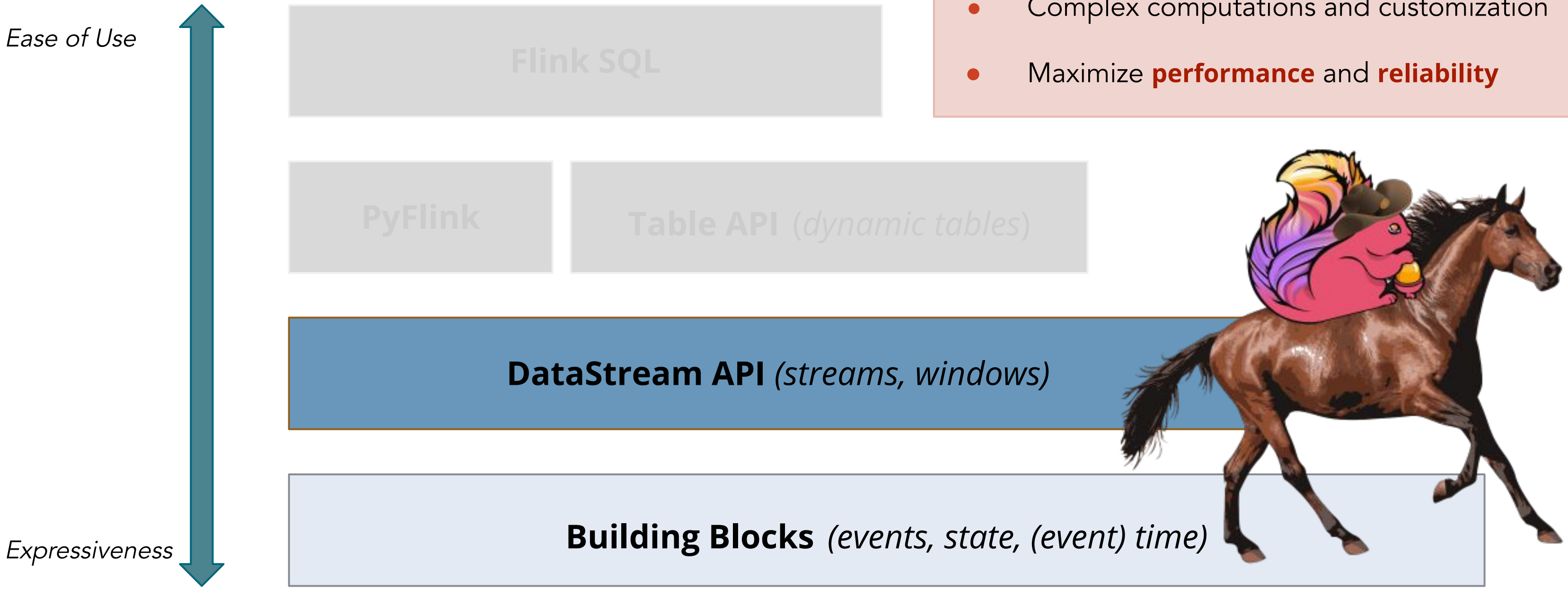
Flink has layered APIs with different tradeoffs for **expressiveness** and **ease of use**. You can mix and match all the APIs!


Ease of Use
Flink SQL
Streaming Analytics & ML
PyFlink
Table API *(dynamic tables)*
DataStream API *(streams, windows)*
Stateful Stream Processing
Building Blocks *(events, state, (event) time)*
Expressiveness

The Flink API Stack

For some use cases, you need Flink's full workhorse power.

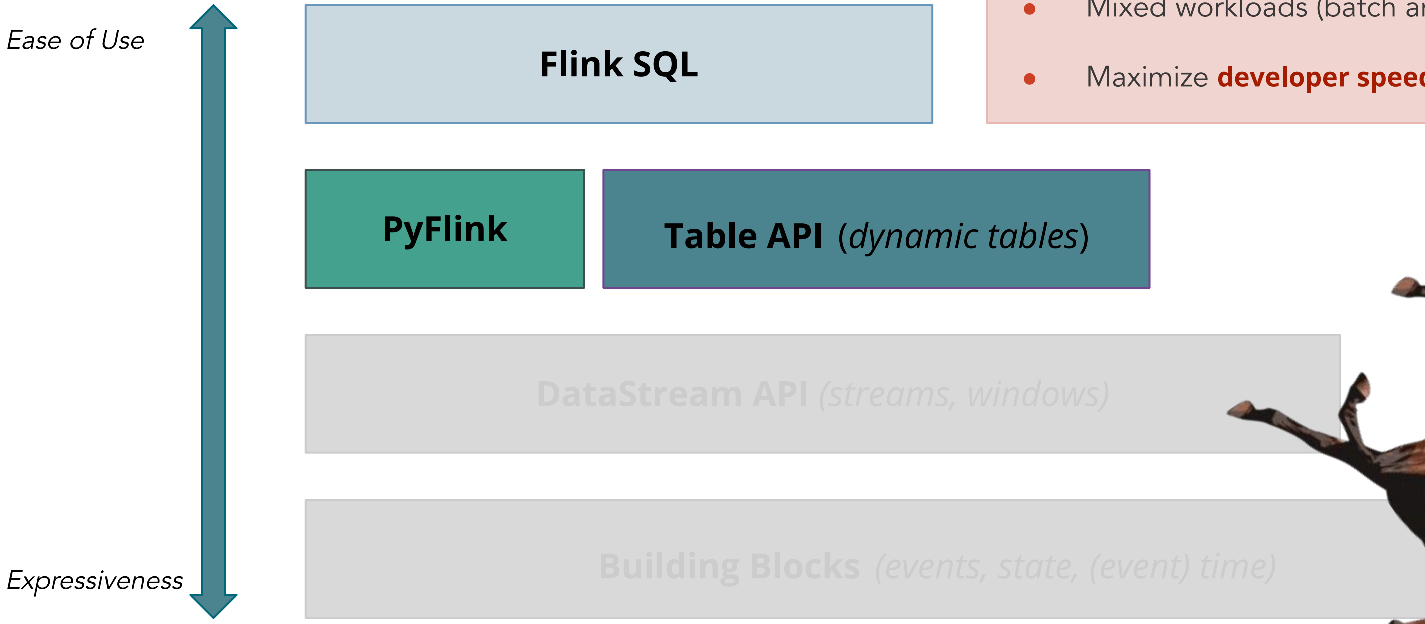
- Explicit control over core primitives (events, state, time)
- Complex computations and customization
- Maximize **performance** and **reliability**


Ease of Use
Expressiveness
Flink SQL
PyFlink
Table API *(dynamic tables)*
DataStream API *(streams, windows)*
Building Blocks *(events, state, (event) time)*

The Flink API Stack

But for a lot of others, you don't.

- Focus on logic, not implementation
- Mixed workloads (batch and streaming)
- Maximize **developer speed** and **autonomy**

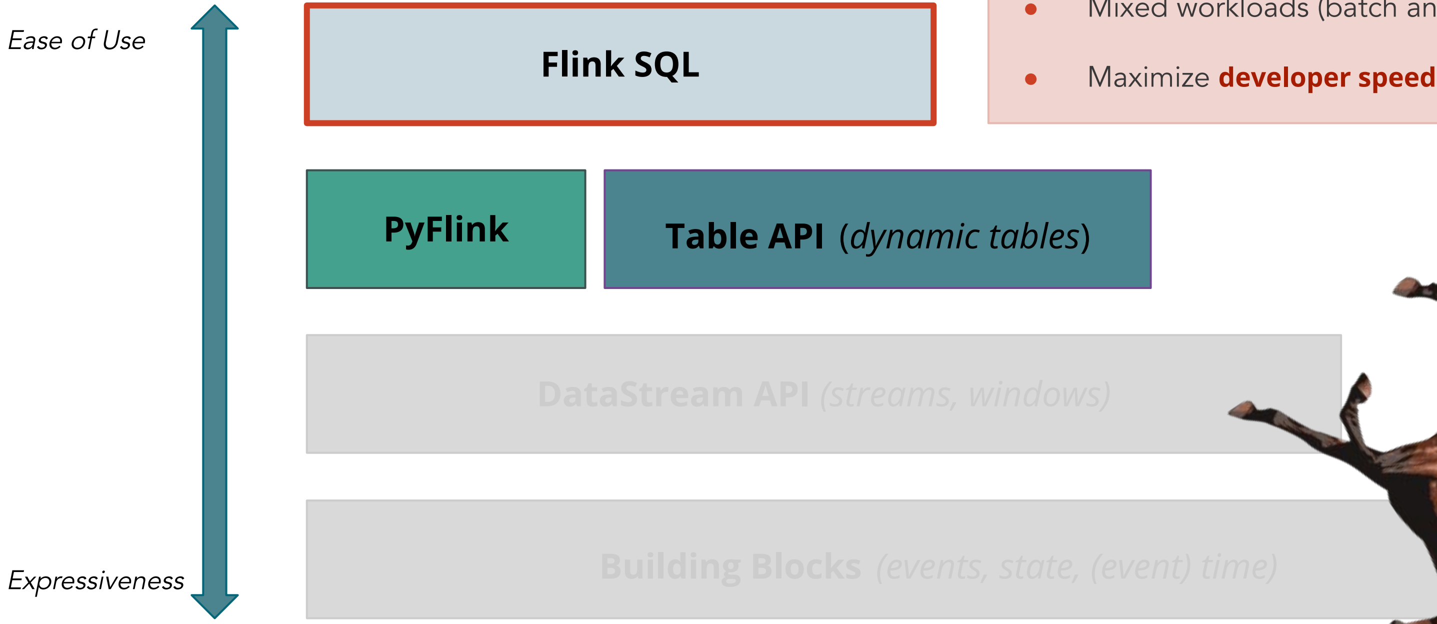


Ease of Use
Expressiveness
Flink SQL
PyFlink
Table API *(dynamic tables)*
DataStream API *(streams, windows)*
Building Blocks *(events, state, (event) time)*

The Flink API Stack

But for a lot of others, you don't.

- Focus on logic, not implementation
- Mixed workloads (batch and streaming)
- Maximize **developer speed** and **autonomy**


Ease of Use
Expressiveness
Flink SQL
PyFlink
Table API *(dynamic tables)*
DataStream API *(streams, windows)*
Building Blocks *(events, state, (event) time)*

Flink SQL

"Everyone knows SQL, right?"

```
SELECT user_id, COUNT(url) AS cnt
FROM clicks
GROUP BY user_id;
```

This is standard SQL (ANSI SQL)

Flink SQL

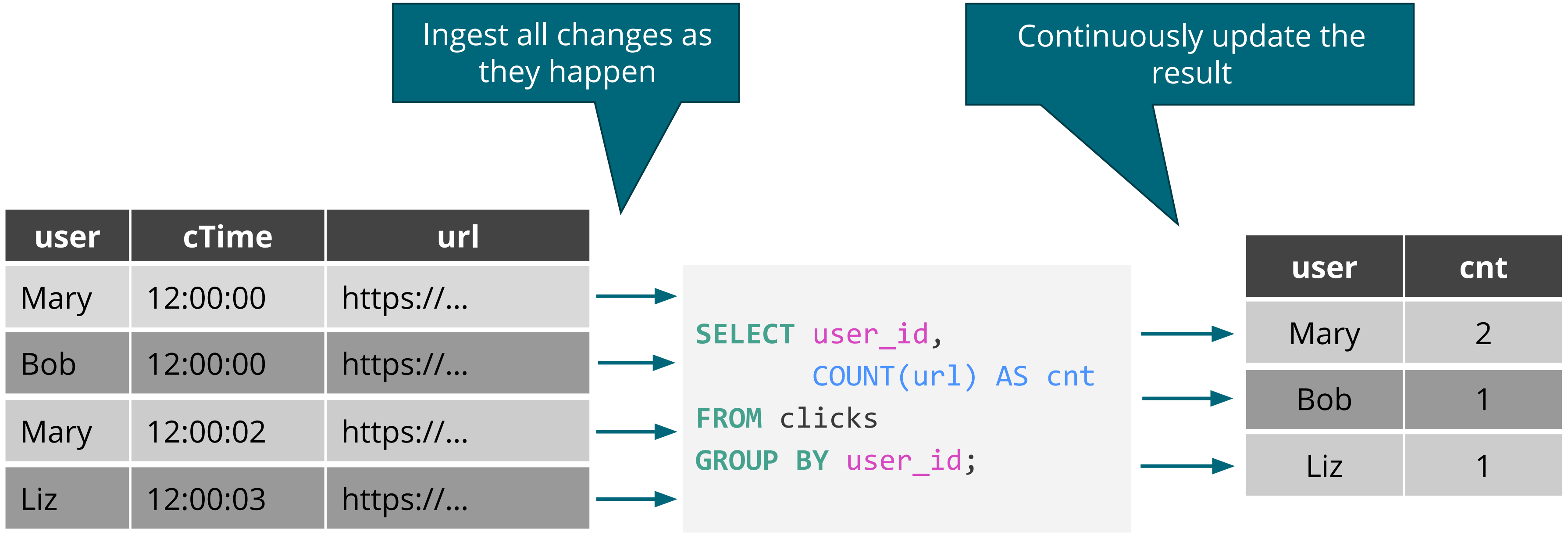
"Everyone knows SQL, right?"

```
SELECT user_id, COUNT(url) AS cnt
FROM clicks
GROUP BY user_id;
```

This is ~~standard SQL (ANSI SQL)~~

also Flink SQL

A Streaming SQL Engine


Ingest all changes as they happen
Continuously update the result

user	cTime	url
Mary	12:00:00	https://...
Bob	12:00:00	https://...
Mary	12:00:02	https://...
Liz	12:00:03	https://...

```sql
SELECT user_id,
       COUNT(url) AS cnt
FROM clicks
GROUP BY user_id;
```

user	cnt
Mary	2
Bob	1
Liz	1

Flink SQL in a Nutshell

- Standard SQL syntax and semantics (i.e. not a “SQL-flavor”)
- Unified APIs for batch and streaming
- Support for **advanced operations** (e.g. temporal joins, pattern matching/CEP)

Execution

Streaming | Batch

TPC-DS Coverage

UDF Support

Java | Python

Scala

Native Connectors

Apache Kafka

FileSystems

JDBC

HBase

Elasticsearch

+

Data Catalogs

Metastore

Postgres (JDBC)

Schema Registry

FLIP-125

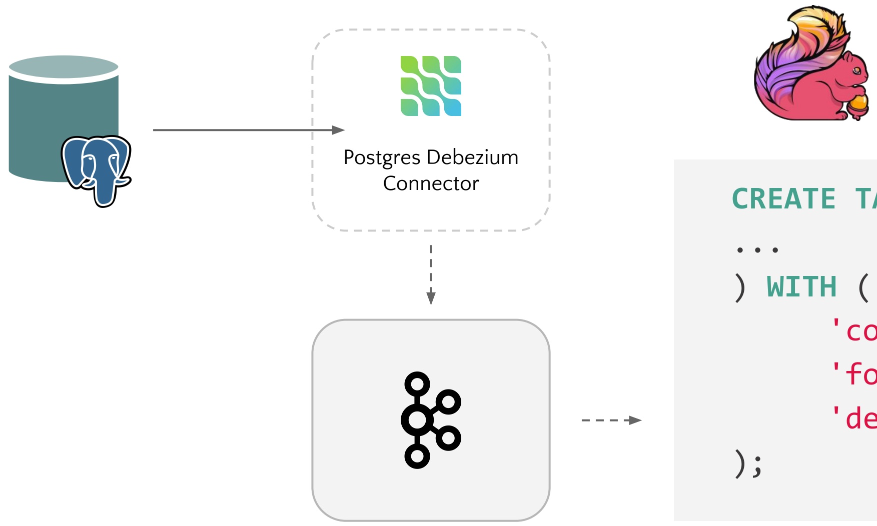
Notebooks

Apache Zeppelin

For an overview of supported operations, check the Flink documentation: Table API&SQL / SQL / Queries

Flink SQL + CDC

- Available from **Flink 1.11** * (released Jul. 2020)
- Initial implementation:
  - **JSON-encoded** changelogs;
  - **Kafka** as a changelog source.



```
SELECT user_id, COUNT(url) AS cnt
FROM clicks
GROUP BY user_id;
```

...

```
CREATE TABLE clicks (
...
) WITH (
      'connector'='kafka',
      'format'='debezium-json',
      'debezium-json.schema-include'='false'
);
```

* We recommend using Flink 1.11.1 for full-blown CDC support.

Demo

What are we doing?

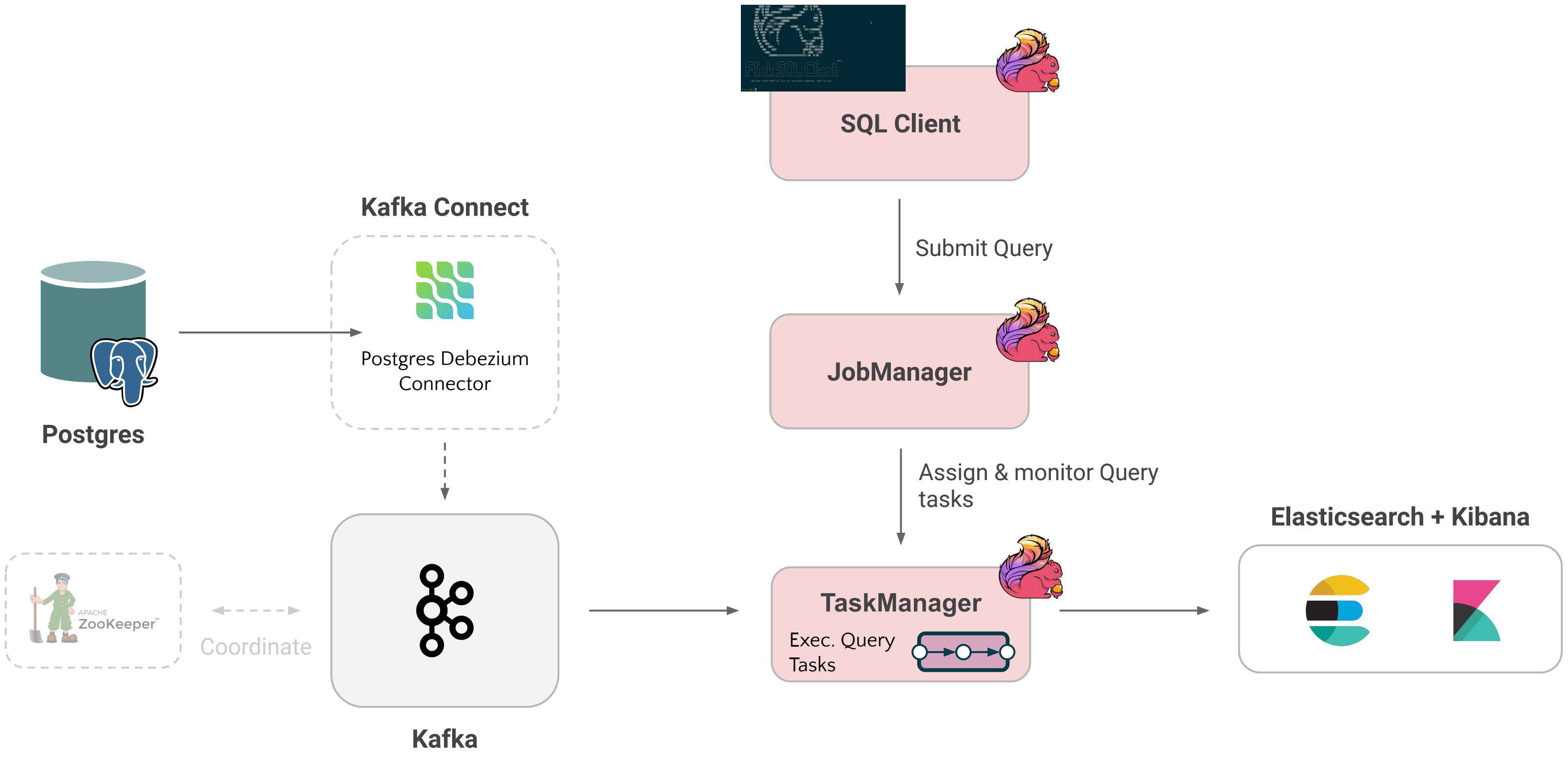
A Cassowary aka the world's most dangerous bird.

Processing (fake) insurance claim data related to animal attacks in Australia.

- Get Debezium up and running with Kafka
- Register a Postgres catalog to access external table metadata
- Create a changelog source to consume Debezium CDC data from Kafka
- See CDC in action!
- Maintain a Materialized View (MV) in Elasticsearch
- Visualize the results in Kibana

Try out the demo: https://github.com/morsapaes/flink-sql-CDC

The Demo Environment



Try out the demo: https://github.com/morsapaes/flink-sql-CDC

Demo

To wrap it up...

- Flink SQL is used at **massive scale** in companies like Alibaba, Uber, Yelp and Lyft.

	Infrastructure	Data Size	Throughput (Peak)	Latency	State Size (Biggest)
Alibaba Group Single's Day 2019	>5K nodes, >500K CPU cores	985PB	2.5B events/sec	Sub-sec	100TB

- Flink SQL is **standard**, provides **unified APIs** and has a **growing ecosystem** of integrations around it.

Upcoming CDC Improvements

- Umbrella ticket (FLINK-18822):
  - **Avro-encoded** changelogs;
  - **Temporal joins** with changelog sources;
  - **Batch** support.

Check out these open-sourced Flink CDC connectors:
https://github.com/ververica/flink-cdc-connectors

Thank you, DataEngBytes!

Follow me on Twitter: @morsapaes

Learn more about Flink: https://flink.apache.org/

ververica