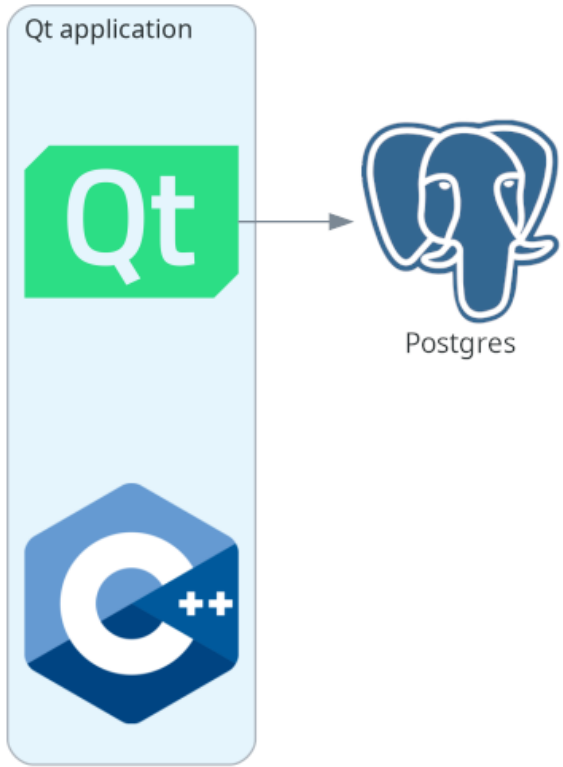
Let's start with the demo!

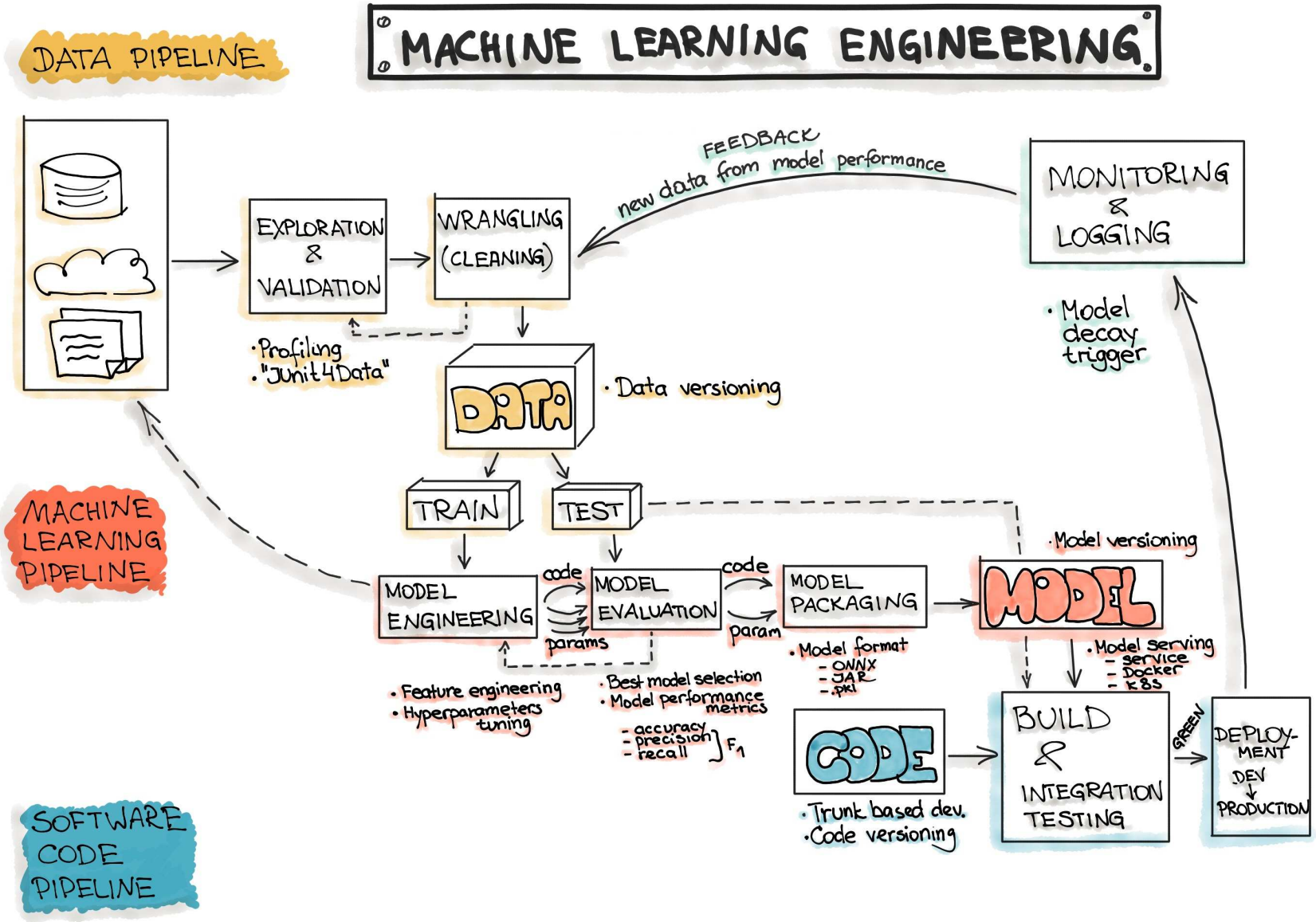


Feeding ML models with the data from the databases in real-time

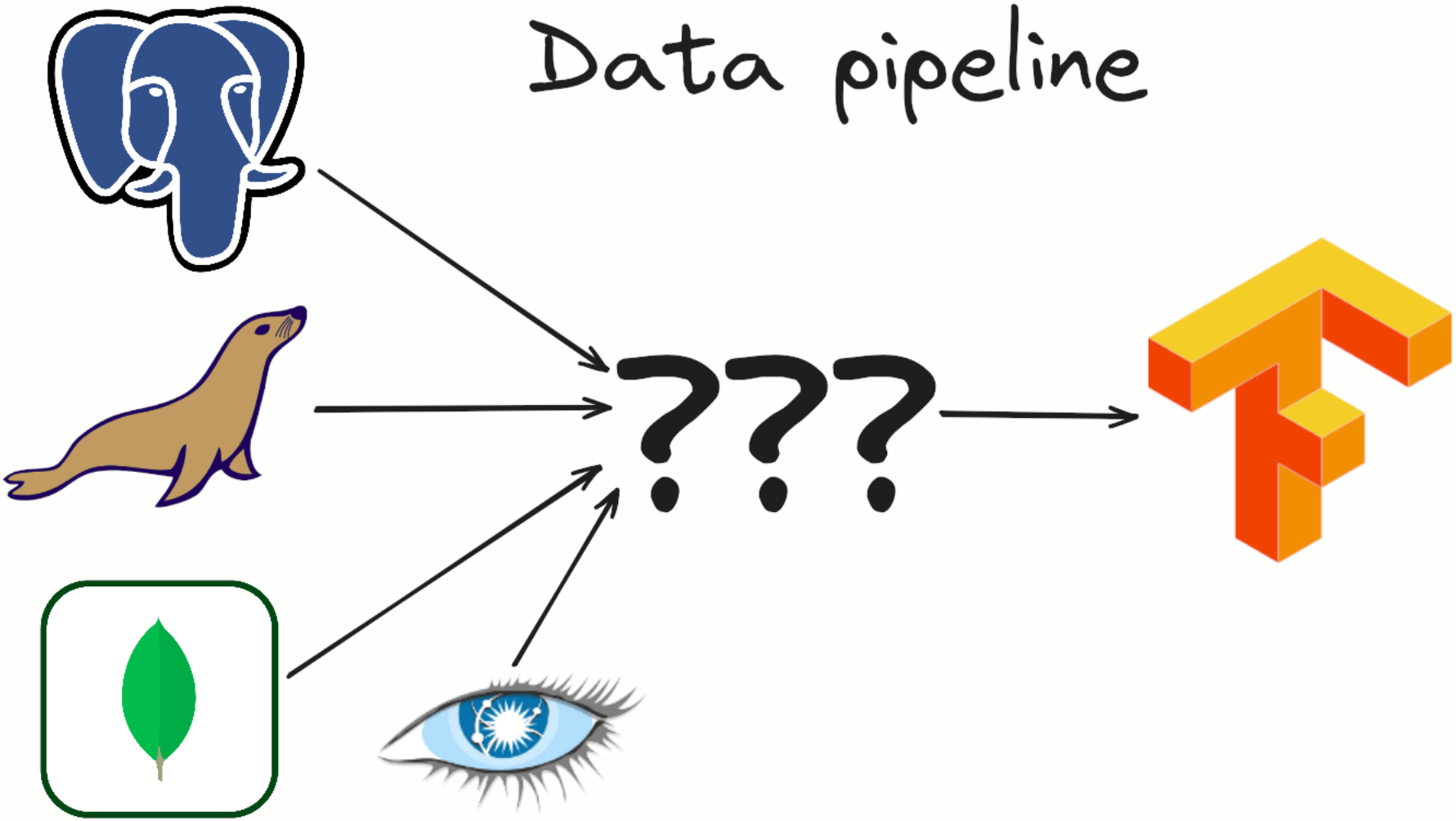
Vojtěch Juránek

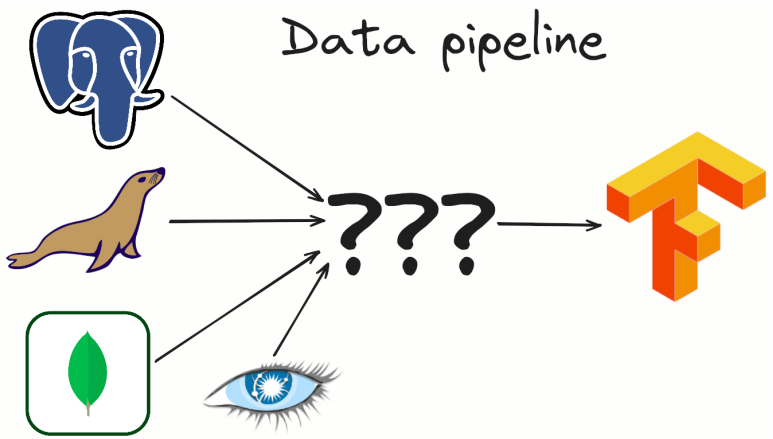
Red Hat

June 15th 2024, DevConf, Brno



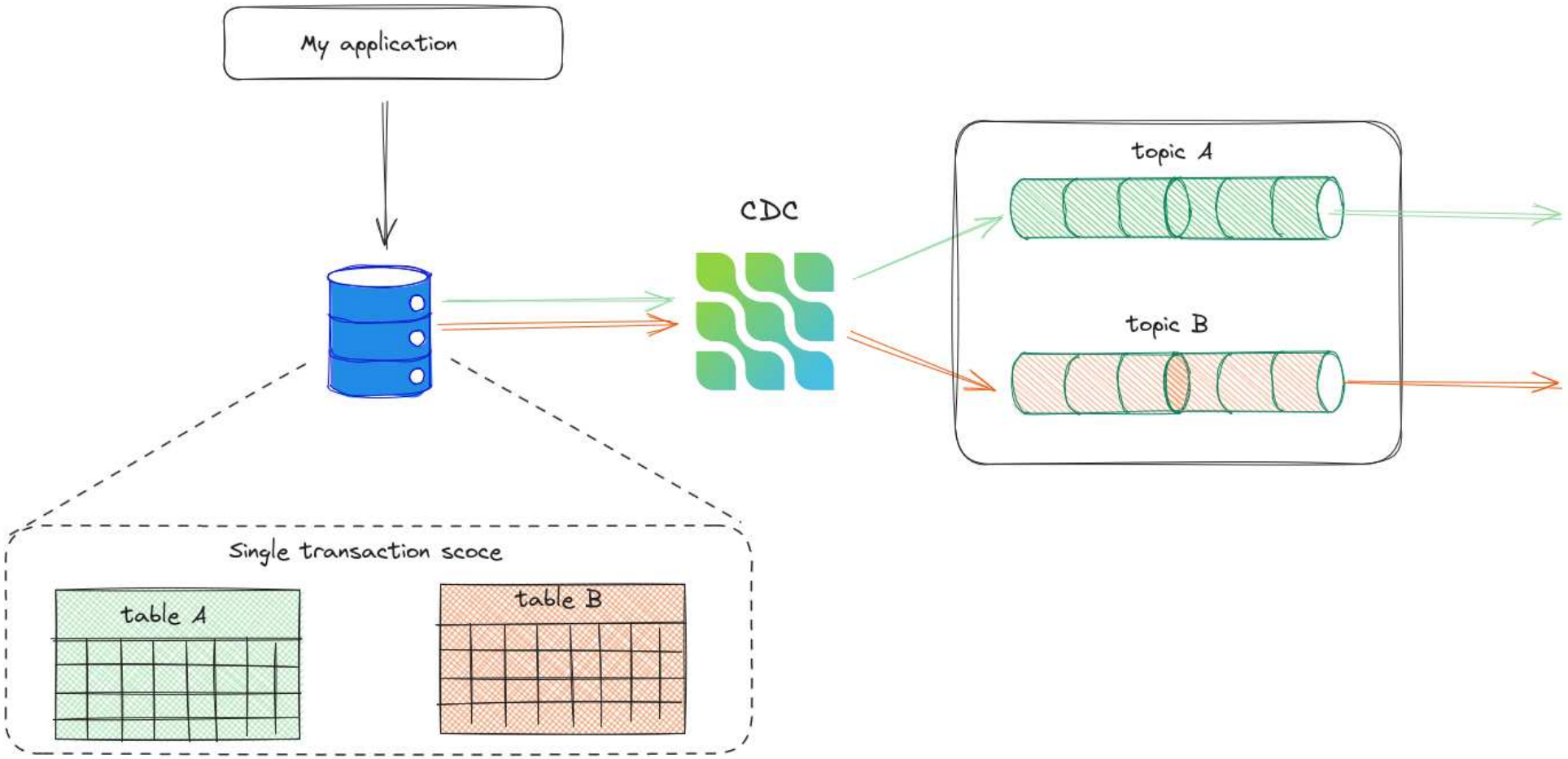
Source: `https://ml-ops.org/content/end-to-end-ml-workflow`


Data pipeline
???



- Consistent data, no data losses, no dual writes.
- Get all the changes without any delay in the real-time.
- Not overload the DB with the queries.

Change Data Capture (CDC)



Debezium

- Leading CDC framework, de-facto industry standard.
- Fully open source: https://github.com/debezium/
- Supports all major databases, including non-relational databases.
- Integrations with many 3rd-party tools and frameworks.
- Large and active user community.
- Used by many companies in production (see Debezium public references).

Initial snapshot of the data

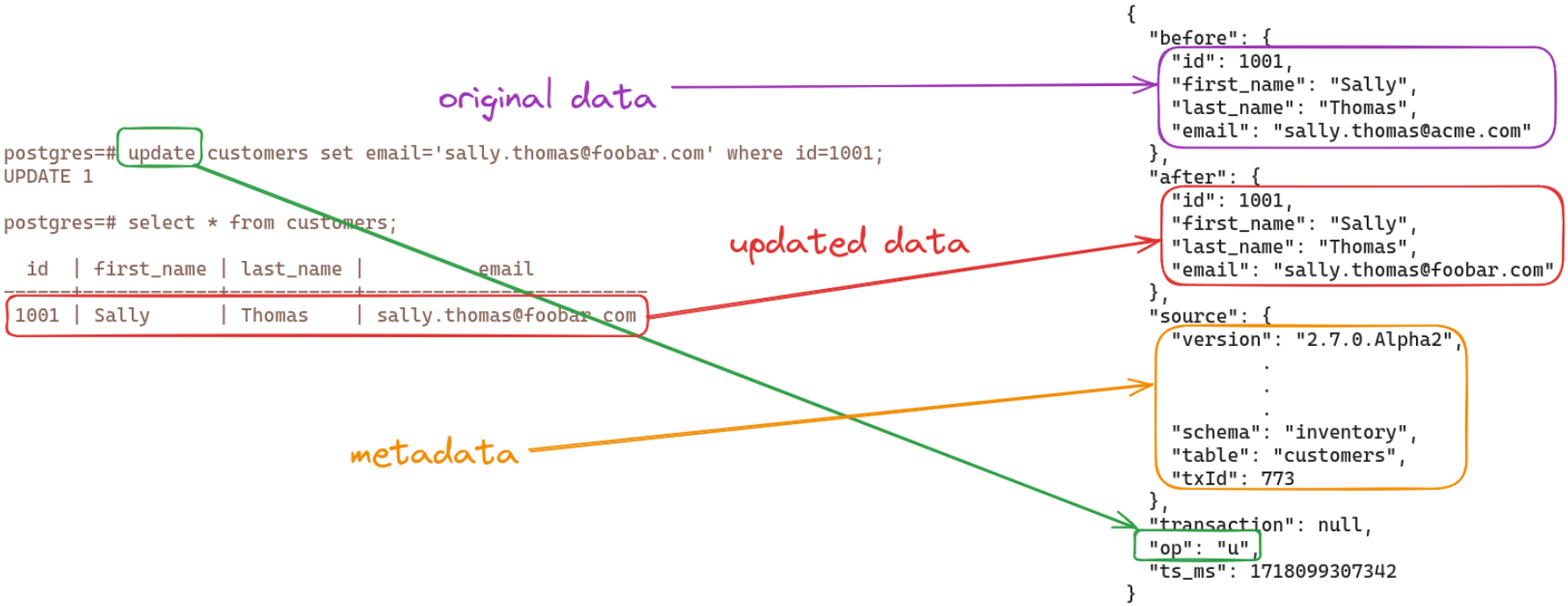
```
postgres=# select * from customers;
  id  | first_name | last_name |         email
------+------------+-----------+-----------------------
1001 | Sally      | Thomas    | sally.thomas@acme.com
```

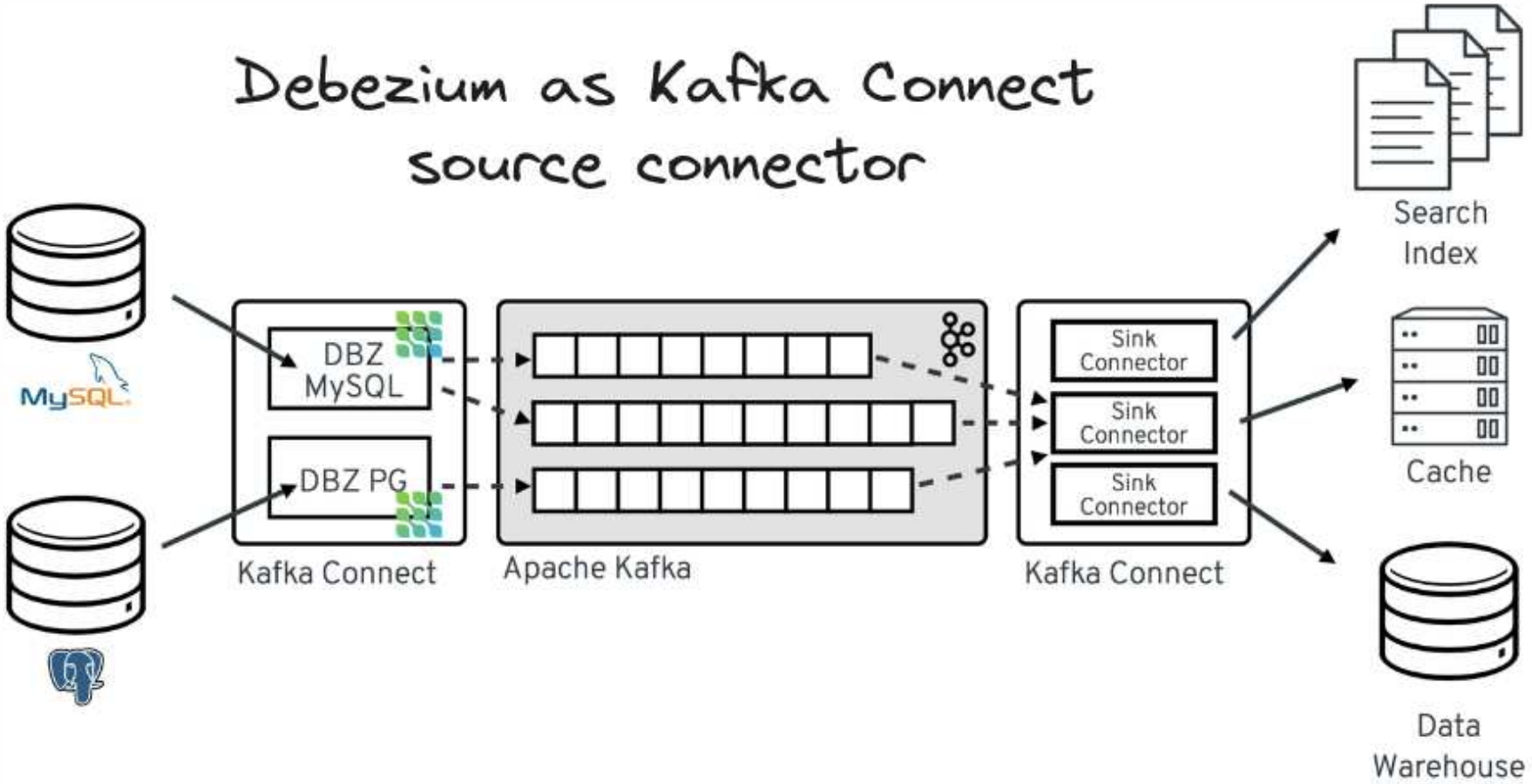
data

metadata

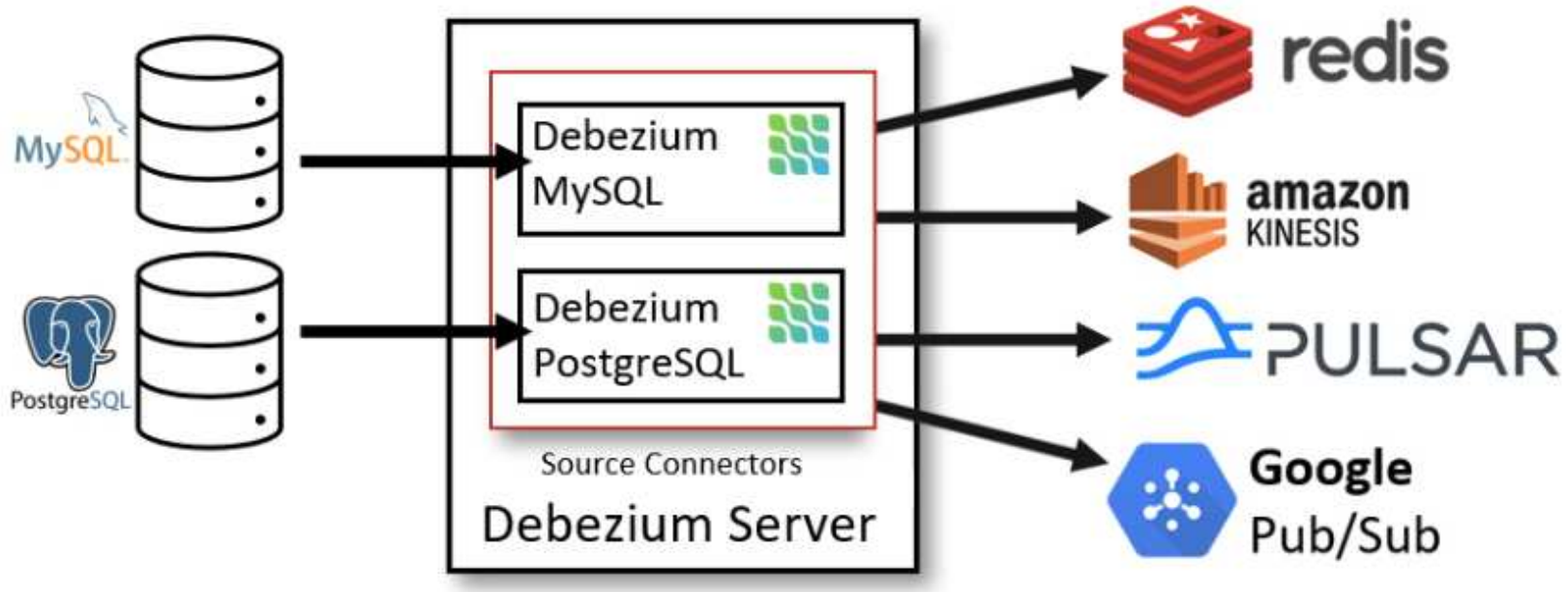

```
{
  "before": null,
  "after": {
    "id": 1001,
    "first_name": "Sally",
    "last_name": "Thomas",
    "email": "sally.thomas@acme.com"
  },
  "source": {
    "version": "2.7.0.Alpha2",
          .
          .
          .
    "schema": "inventory",
    "table": "customers",
    "txId": 758
  },
  "op": "r",
  "ts_ms": 1718098503663
}
```

Streaming the changes




Debezium as Kafka Connect source connector
MySQL
DBZ MySQL
DBZ PG
Kafka Connect
Apache Kafka
Sink Connector
Sink Connector
Sink Connector
Kafka Connect
Search Index
Cache
Data Warehouse

Debezium as standalone server



1:15 PM

35min

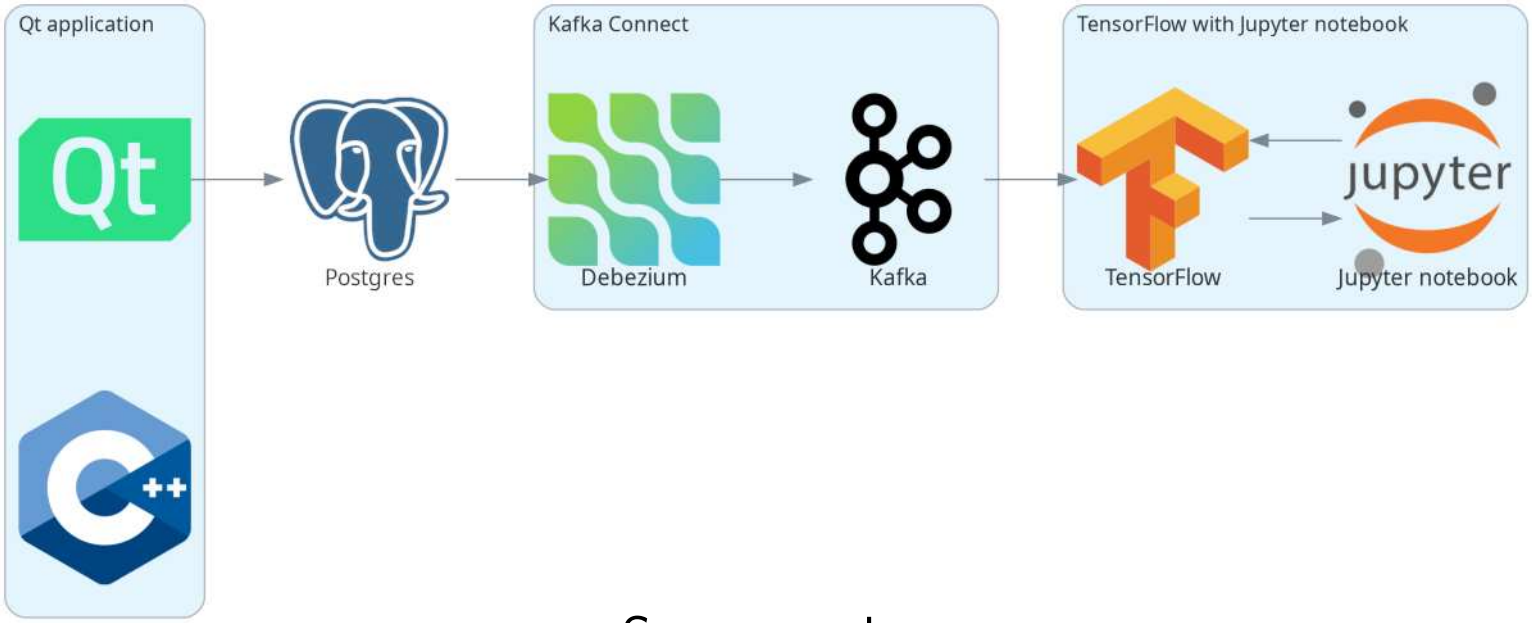
Effortless Change Data Capture with Debezium and Kubernetes

Jakub Čecháček, Ondrej Babec

1:30 PM

Cloud, Hybrid Cloud, and Hyper…

Back to the demo!



Source code:
https://github.com/vjuranek/debezium-mnist-demo

See also blog post:
Image classification with Debezium and TensorFlow blog post
https://debezium.io/blog/2023/05/02/tensorflow-mnist-classification

Possible issues

Deserialization issue in TensorFlow:

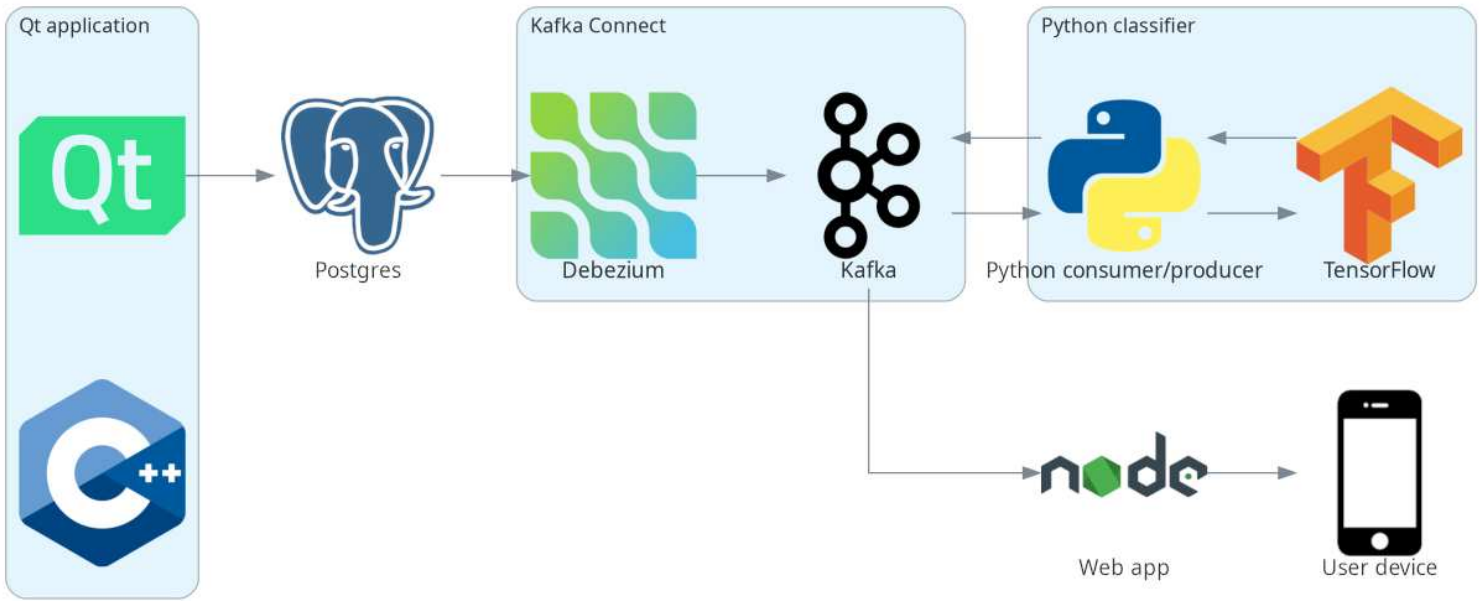
```
<tf.Tensor: shape=(64,), dtype=string, numpy=
array([b'[B@418b353d', b'[B@6aa28a4c',
       b'[B@3a3f1c1', b'[B@36e4ff1f',
       b'[B@765b7f70', b'[B@67567aa3',
       b'[B@58bc3a3a', b'[B@c6bc0ec',
       b'[B@64b48bac', b'[B@360a5b76',
       b'[B@3006c930', b'[B@54b3e5ad',
       b'[B@155dd43d', b'[B@5e88d5b6',
       b'[B@7dcdf024', b'[B@6570bf4e',
      dtype=object)>,
```

Single message transform (SMT) to rescue

- Transform inbound and/or outbound messages.
- Can be used also e.g. for filtering to save bandwidth on the early stage of the ML pipeline.
- Many SMTs available out-of-the-box.
- Very easy to write custom SMT.

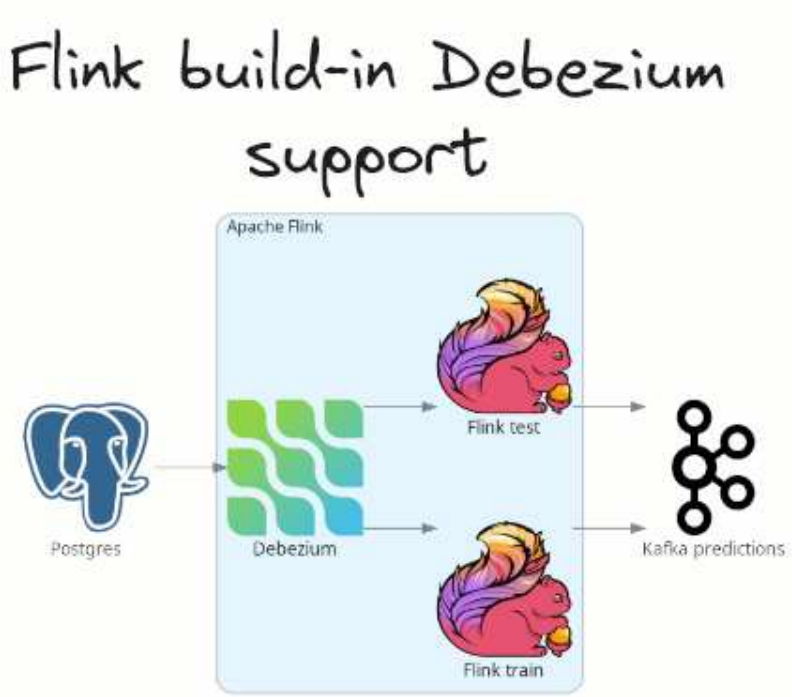
```
1  @Override
2  public R apply(R r) {
3      final Struct value = (Struct) r.value();
4      String key = value.getInt16(labelFieldName).toString();
5
6      StringBuilder builder = new StringBuilder();
7      for (byte pixel : value.getBytes(pixlesFieldName)) {
8          builder.append(pixel & 0xFF).append(",");
9      }
0      return r.newRecord(r.topic(), r.kafkaPartition(),
1          Schema.STRING_SCHEMA, key,
2          Schema.STRING_SCHEMA,
3          builder.toString(), r.timestamp());
4  }
```

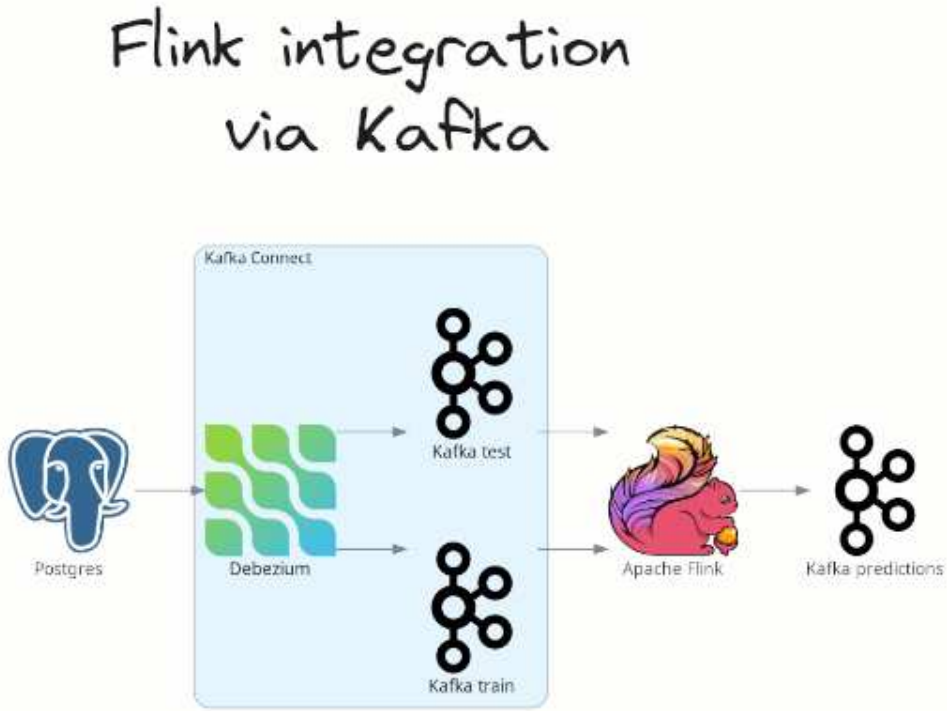
Works seamlessly with Python Kafka client



No SMT needed.

Flink and Spark





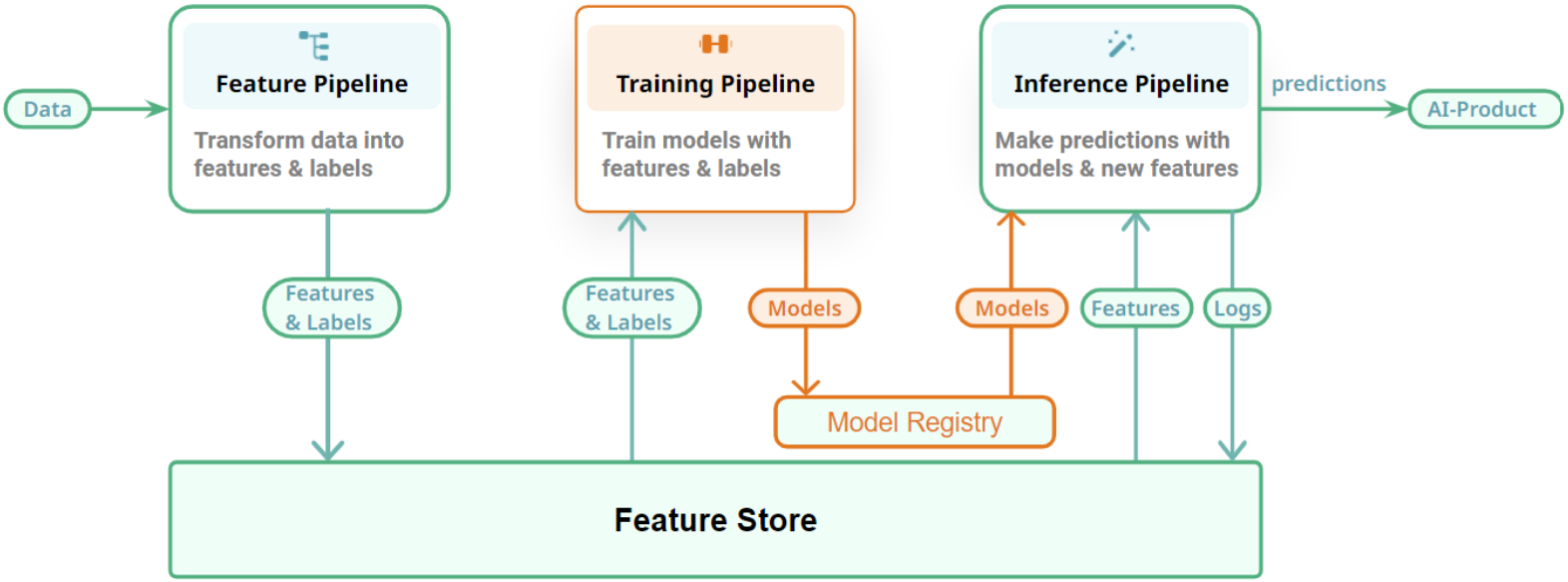
Similar for Apache Spark.

For more details see

- https://debezium.io/blog/2023/09/23/flink-spark-online-learning
- https://github.com/debezium/debezium-examples/tree/main/machine-learning/flink-

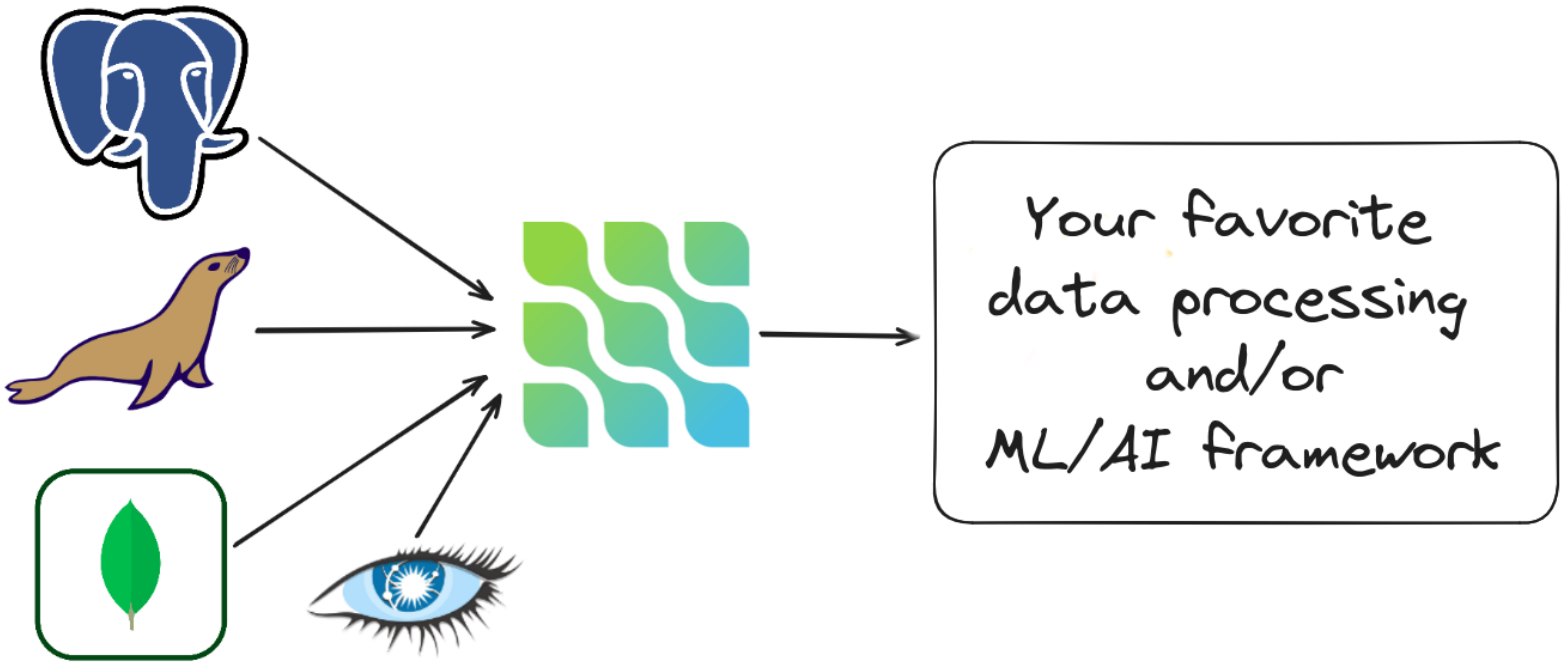
Feature stores

- A centralized repository of features.
- Consistency across training and inference.
- Collaboration and reusability.
- Precomputing features.
- Monitoring for feature drift.
- Versioning.
- Integration with other MLOps tools, unified access to the features.



Source: `https://www.hopsworks.ai/dictionary/feature-store`

Data processing pipeline



Thank you!

debezium

https://debezium.io
https://github.com/debezium
https://debezium.zulipchat.com
https://groups.google.com/g/debezium

Backup slides

Debezium configuration

```
{
  "name": "mnist-connector",
  "config": {
    "connector.class":
        "io.debezium.connector.postgresql.PostgresConnector",
    "tasks.max": "1",
    "database.hostname": "postgres",
    "database.port": "5432",
    "database.user": "postgres",
    "database.password": "postgres",
    "database.dbname" : "postgres",
    "topic.prefix": "tf",
    "table.include.list":  "public.mnist_.*",
    "key.converter":
        "org.apache.kafka.connect.storage.StringConverter",
    "value.converter":
        "org.apache.kafka.connect.storage.StringConverter",
    "transforms":  "unwrap, mnist",
    "transforms.unwrap.type":
        "io.debezium.transforms.ExtractNewRecordState",
    "transforms.mnist.type":  "io.debezium.transforms.MnistToCsv"
  }
}
```

Reading data from Kafka in TensorFlow

TensorFlow I/O provides KafkaGroupIODataset:

```
import tensorflow_io as tfio

# define Kafka data stream
test_ds = tfio.experimental.streaming.KafkaGroupIODataset(
    topics=[KAFKA_TEST_TOPIC],
    group_id=KAFKA_CONSUMER_GROUP,
    servers=KAFKA_SERVERS,
    stream_timeout=KAFKA_STREAM_TIMEOUT,
    configuration=[
        "session.timeout.ms=10000",
        "max.poll.interval.ms=10000",
        "auto.offset.reset=earliest"
    ],
)
```

Reading data from Kafka in TensorFlow

```
# define function for decoding Kafka records
def decode_kafka_stream_record(message, key):
    img_int = tf.io.decode_csv(message, [[0.0] for i in range(
                                        NUM_COLUMNS)])
    img_norm = tf.cast(img_int, tf.float32) / 255.
    label_int = tf.strings.to_number(key, out_type=tf.dtypes.int32)
    return (img_norm, label_int)
# define Kafka data stream
test_ds = tfio.experimental.streaming.KafkaGroupIODataset(
    topics=[KAFKA_TEST_TOPIC],
    group_id=KAFKA_CONSUMER_GROUP,
    servers=KAFKA_SERVERS,
    stream_timeout=KAFKA_STREAM_TIMEOUT,
    configuration=[
        "session.timeout.ms=10000",
        "max.poll.interval.ms=10000",
        "auto.offset.reset=earliest"
    ],
)
# read batches of Kafka records
test_ds = test_ds.map(decode_kafka_stream_record)
test_ds = test_ds.batch(BATCH_SIZE)
# make predictions on the data samples
model.evaluate(test_ds)
```

Reading data from Kafka using Python consumer

```
def classify_stream(model):
    # create Kafka consumer
    consumer = kafka.KafkaConsumer(
        "tf.public.mnist_test",
        bootstrap_servers=["localhost:9092"],
        auto_offset_reset="earliest",
        consumer_timeout_ms=100000,
        enable_auto_commit=True,
        group_id="mnist_classifier"
    )

    for msg in consumer:
        # extract image from Debezium CDC message
        val_json = json.loads(msg.value)
        pixels = val_json['payload']['after']['pixels']
        pixels_decoded = base64.b64decode(pixels)

        # pass the image to the model for classification
        image  = [int(n) for n in pixels_decoded]
        number = plot_and_predict(model, image)
```