aiven

From 🐛 to 🦋

Data Pipelines Evolution from Batch to Streaming

Francesco Tisiot - Staff Developer Advocate

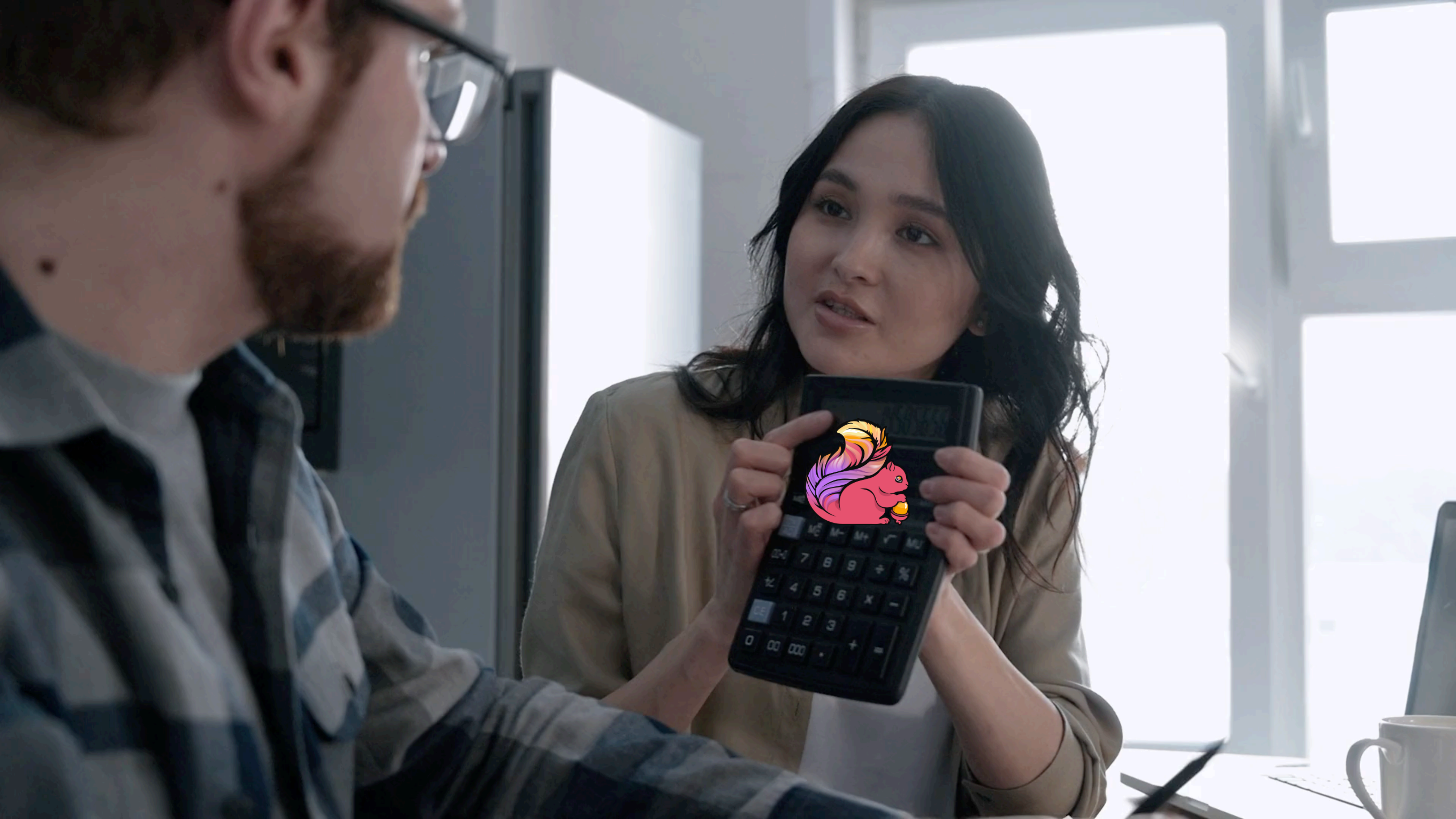
@ftisiot

Boss

We need to move from batch to streaming!

Boss

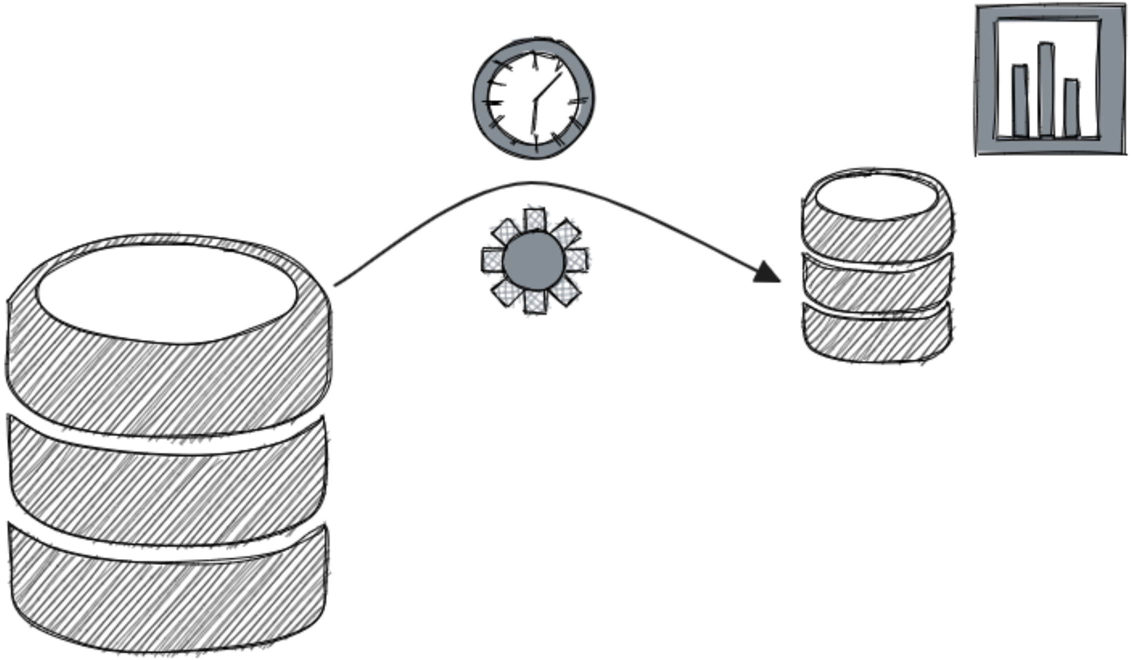
NOW!

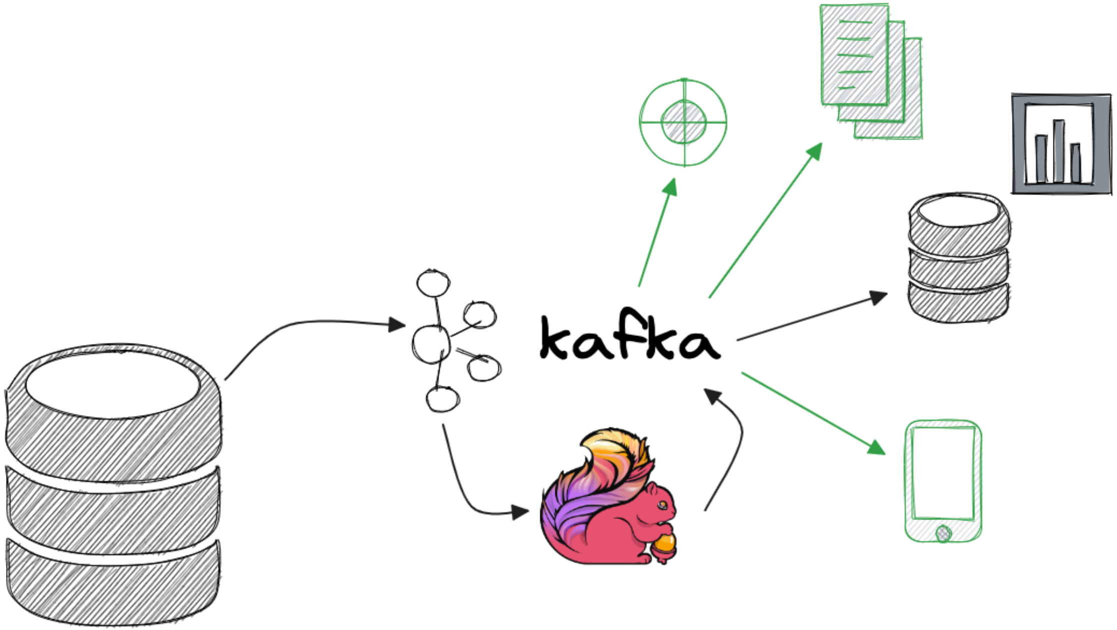


INSERT YOUR NAME

Boss, It's done!

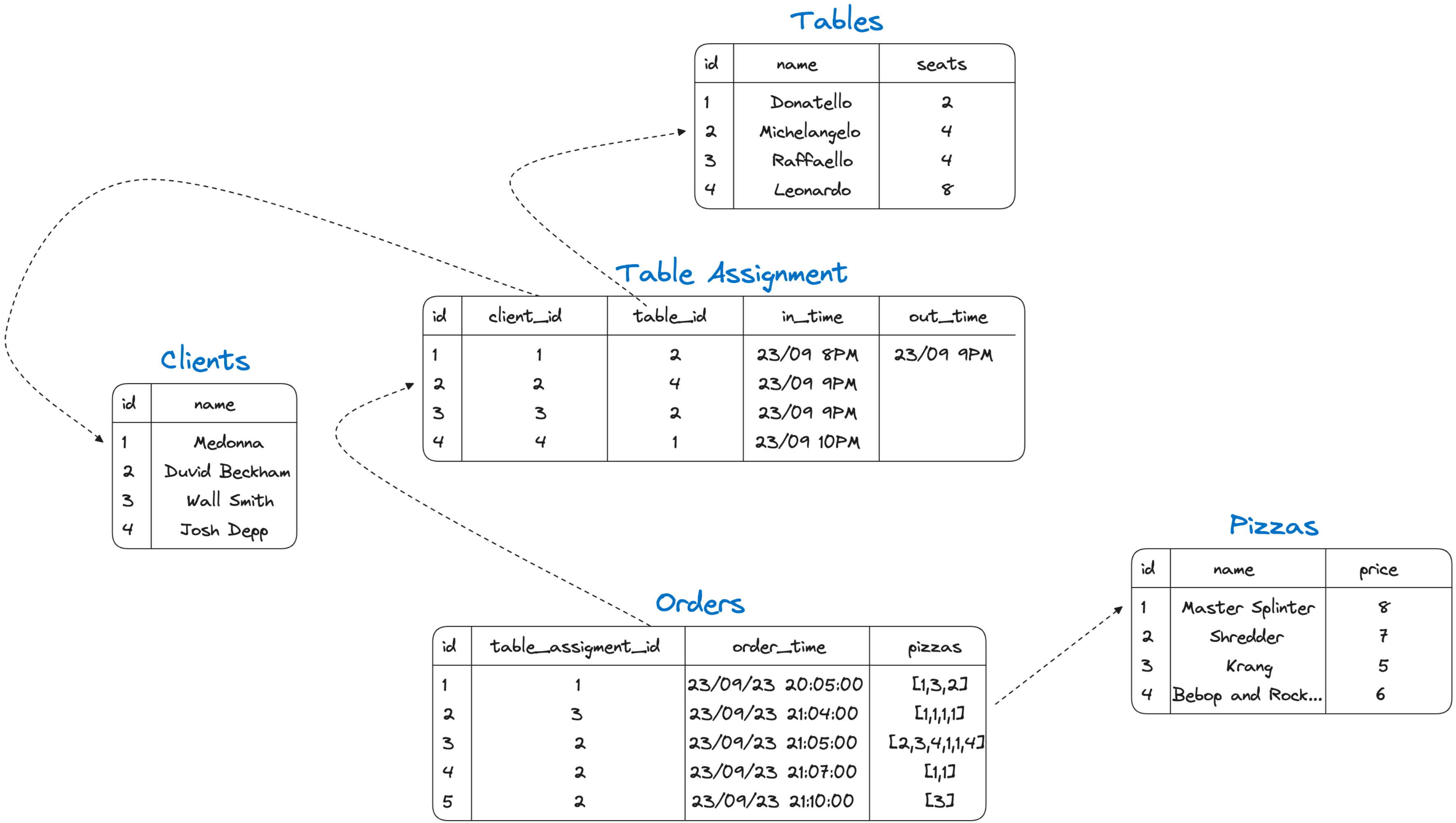
Stop the video... What's this talk about?




kafka

aiven

Let's talk serious business...



Tables

id	name	seats
1	Donatello	2
2	Michelangelo	4
3	Raffaello	4
4	Leonardo	8

Table Assignment

id	client_id	table_id	in_time	out_time
1	1	2	23/09 8PM	23/09 9PM
2	2	4	23/09 9PM	
3	3	2	23/09 9PM	
4	4	1	23/09 10PM	

Clients

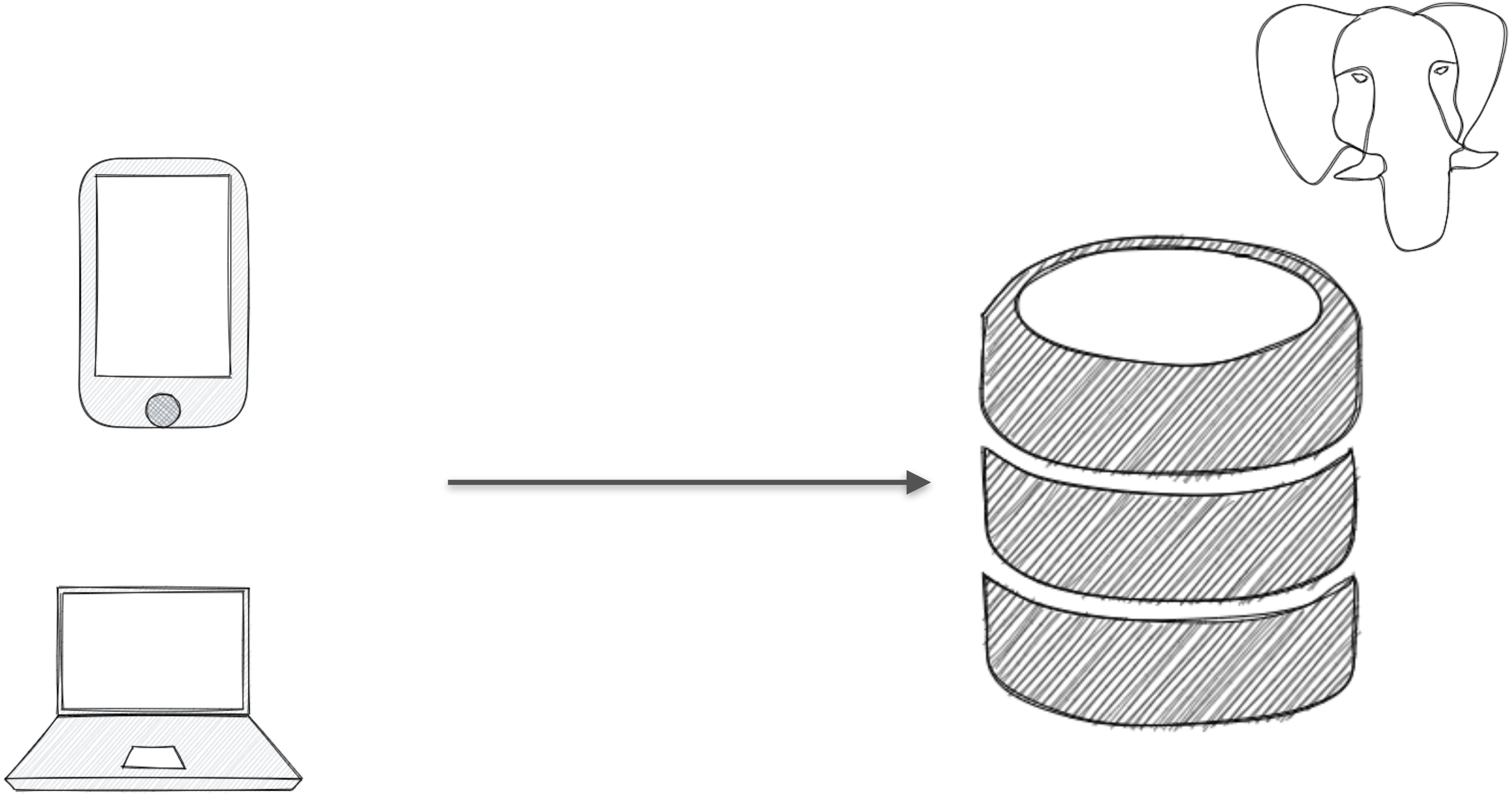
id	name
1	Medonna
2	Duvid Beckham
3	Wall Smith
4	Josh Depp

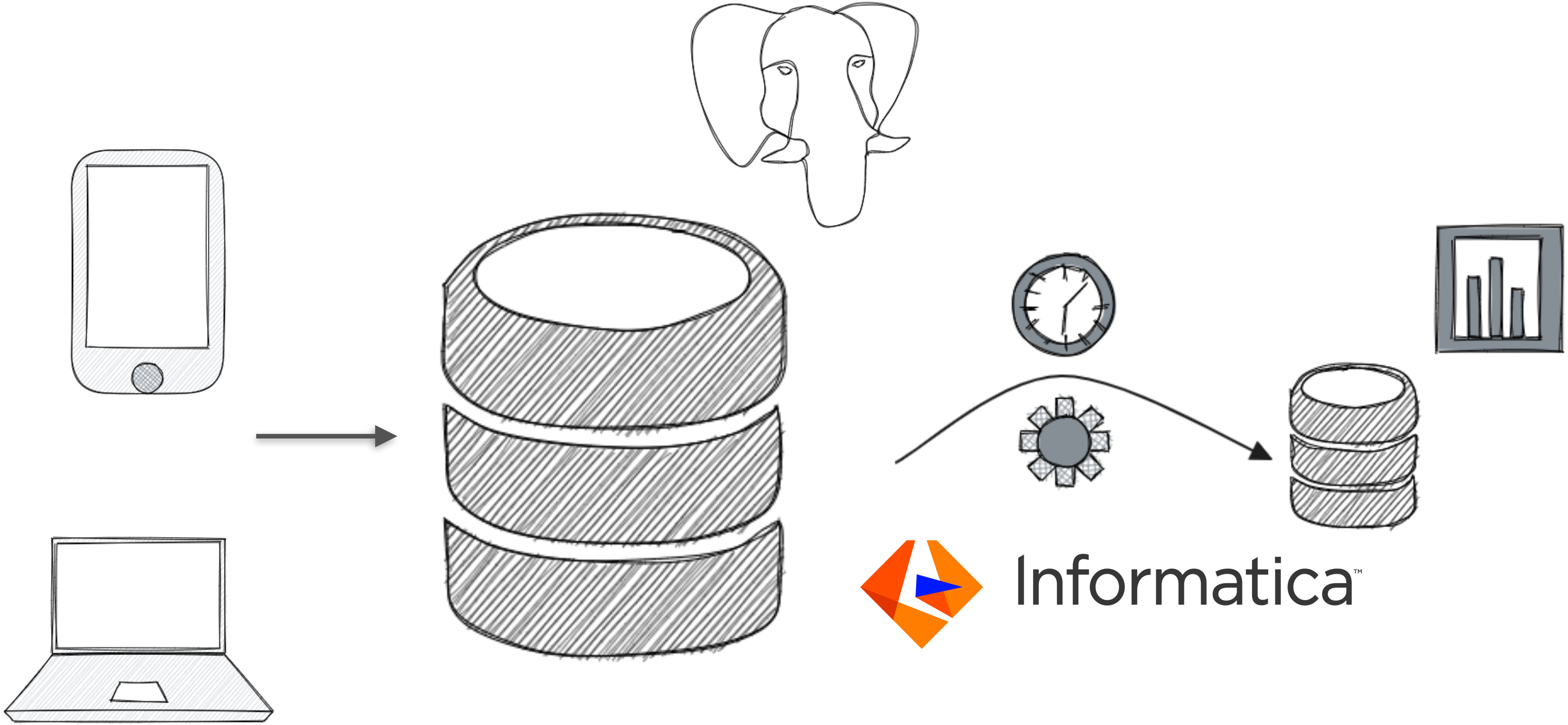
Orders

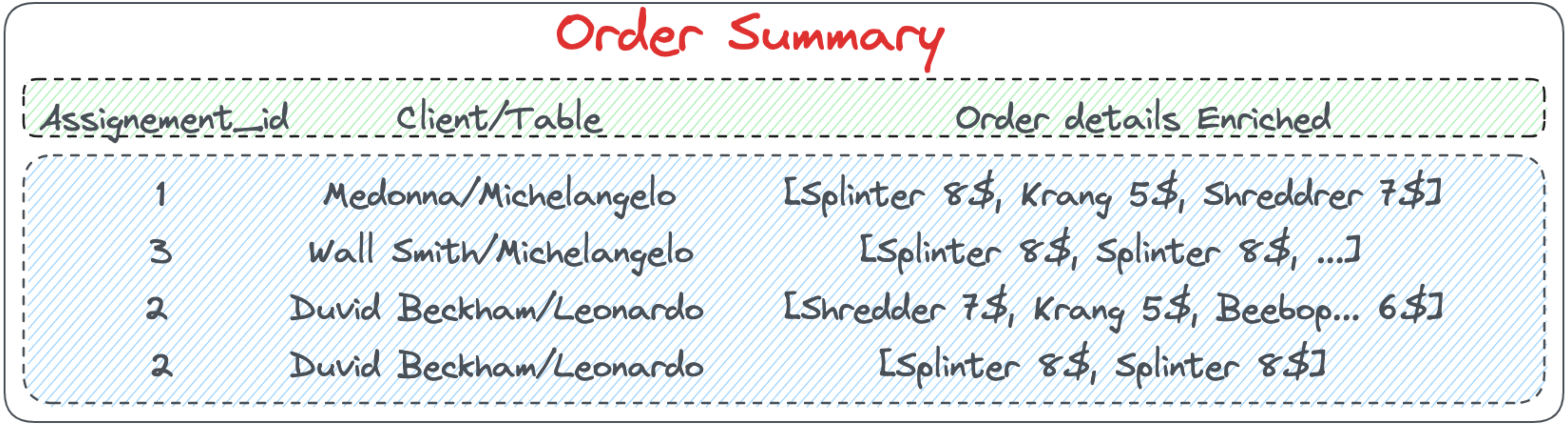
id	table_assigment_id	order_time	pizzas
1	1	23/09/23 20:05:00	[1,3,2]
2	3	23/09/23 21:04:00	[1,1,1,1]
3	2	23/09/23 21:05:00	[2,3,4,1,1,4]
4	2	23/09/23 21:07:00	[1,1]
5	2	23/09/23 21:10:00	[3]

Pizzas

id	name	price
1	Master Splinter	8
2	Shredder	7
3	Krang	5
4	Bebop and Rock...	6

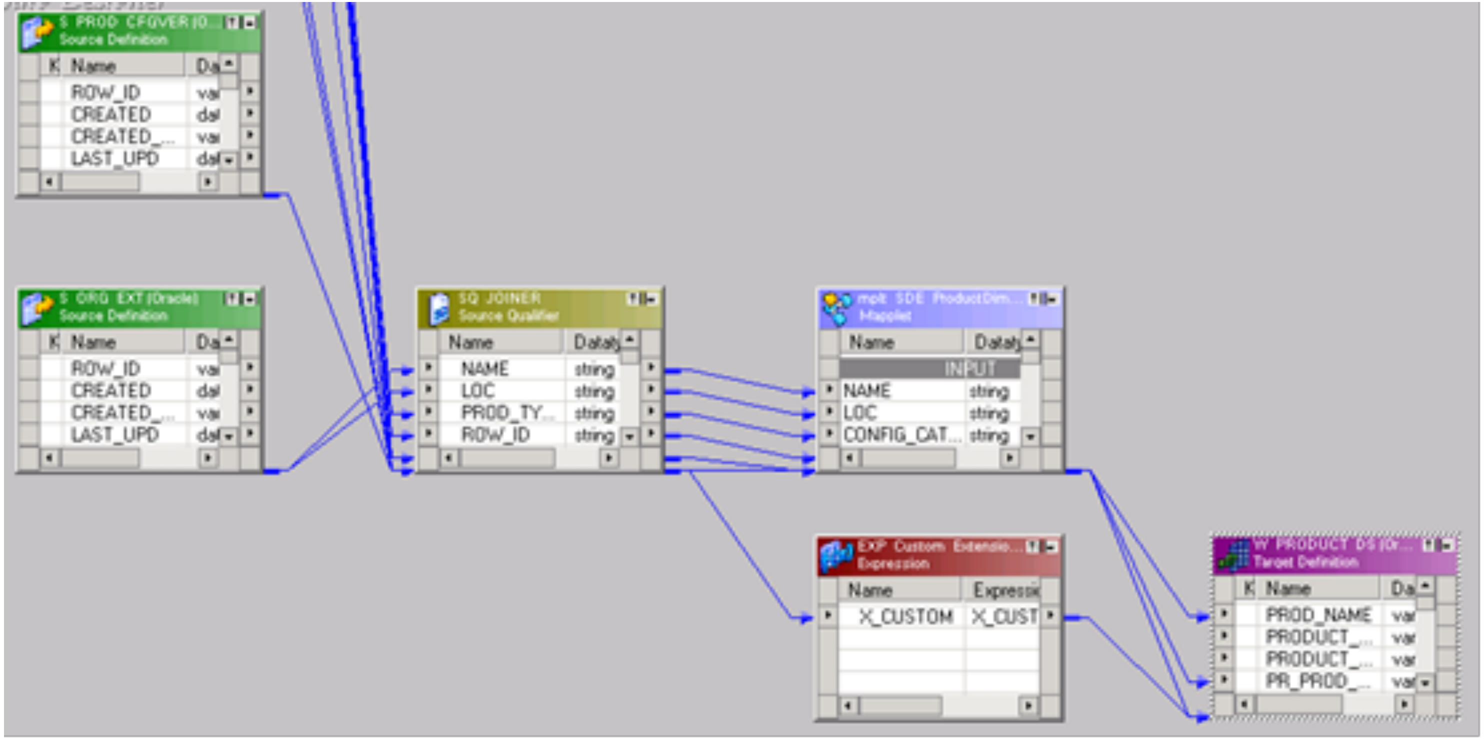



Informatica™



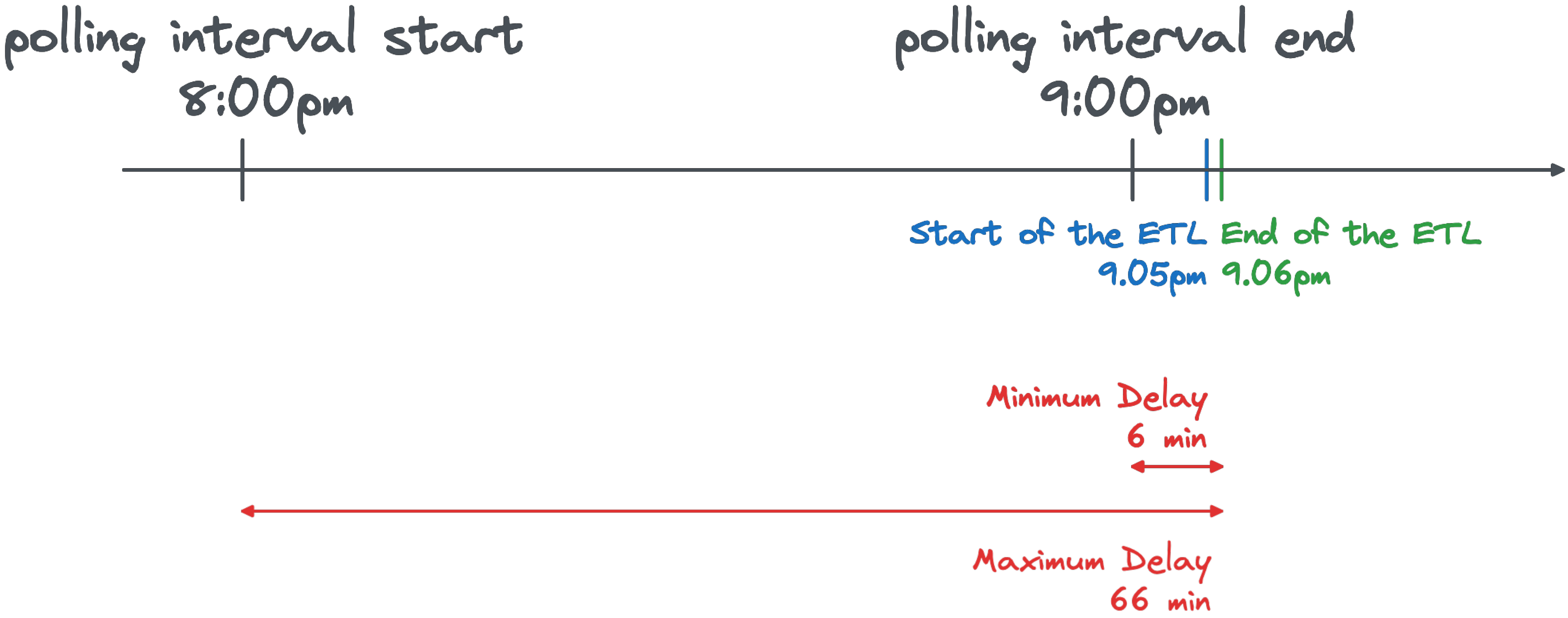
Order Summary

Assignement_id	Client/Table	Order details Enriched
1	Medonna/Michelangelo	[Splinter 8$, Krang 5$, Shreddrer 7$]
3	Wall Smith/Michelangelo	[Splinter 8$, Splinter 8$, ...]
2	Duvid Beckham/Leonardo	[Shredder 7$, Krang 5$, Beebop... 6$]
2	Duvid Beckham/Leonardo	[Splinter 8$, Splinter 8$]


S_PROD_CFGVER (O...
Source Definition
K Name Da
ROW_ID va
CREATED da
CREATED_... va
LAST_UPD da
S_ORG_EXT (Oracle)
Source Definition
K Name Da
ROW_ID va
CREATED da
CREATED_... va
LAST_UPD da
SQ_JOINER
Source Qualifier
Name Dataty
NAME string
LOC string
PROD_TY... string
ROW_ID string
mplt_SDE_ProductDim...
Mapplet
Name Dataty
INPUT
NAME string
LOC string
CONFIG_CAT... string
EXP_Custom_Extensio...
Expression
Name Expressi
X_CUSTOM X_CUST
W_PRODUCT_DS (Or...
Target Definition
K Name Da
PROD_NAME va
PRODUCT_... va
PRODUCT_... va
PR_PROD_... va

```
select
        orders.id order_id,
        clients.name client_name,
        tables.name table_name,
        JSON_AGG(
           JSON_BUILD_OBJECT('pizza', pizzas.name, 'price', pizzas.price))
from orders
        join table_assignment
             on orders.table_assignment_id = table_assignment.id
        join pizzas on pizzas.id = ANY (orders.pizzas)
        join clients on table_assignment.client_id = clients.id
        join tables on table_assignment.table_id = tables.id
where order_time > date_trunc('hour',current_timestamp) - interval '1' hour
and order_time <= date_trunc('hour',current_timestamp)
group by
    orders.id,
    clients.name,
    tables.name;
```

```
order_id |  client_name  |  table_name  |           json_agg
---------+---------------+--------------+--------------------------------------------------
       2 | Wall Smith    | Michelangelo | [{"pizza" : "Master Splinter", "price" : 8}]
       3 | Duvid Beckham | Leonardo     | [{"pizza" : "Master Splinter", "price" : 8},
         |               |              |  {"pizza" : "Shredder", "price" : 7},
         |               |              |  {"pizza" : "Krang", "price" : 5},
         |               |              |  {"pizza" : "Bebop and Rocksteady", "price" : 6}]
       4 | Duvid Beckham | Leonardo     | [{"pizza" : "Master Splinter", "price" : 8}]
(3 rows)
```



polling interval start
8:00pm

polling interval end
9:00pm

Start of the ETL
9.05pm

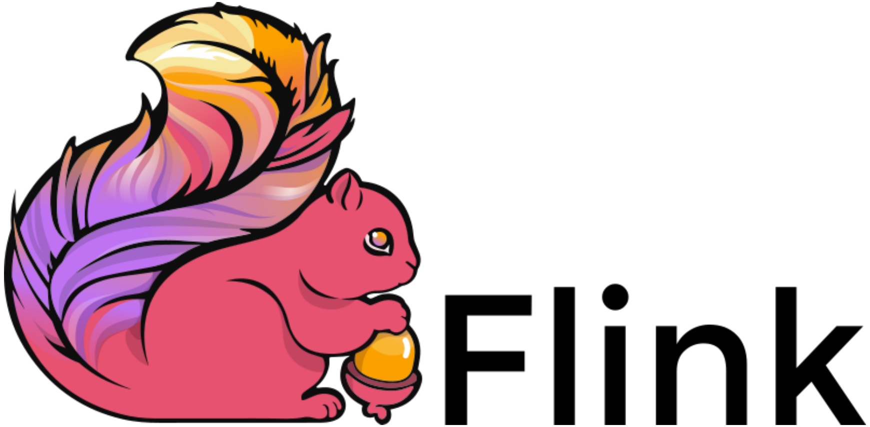
End of the ETL
9.06pm

Minimum Delay
6 min

Maximum Delay
66 min

Let's do Streaming

Flink


Flink

kafka

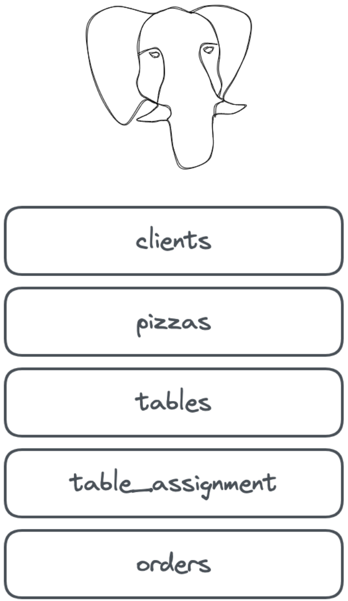
SQL

```
select
        orders.id order_id,
        clients.name client_name,
        tables.name table_name,
        JSON_AGG(
           JSON_BUILD_OBJECT('pizza', pizzas.name, 'price', pizzas.price))
from orders
        join table_assignment
             on orders.table_assignment_id = table_assignment.id
        join pizzas on pizzas.id = ANY (orders.pizzas)
        join clients on table_assignment.client_id = clients.id
        join tables on table_assignment.table_id = tables.id
where order_time > date_trunc('hour',current_timestamp) - interval '1' hour
and order_time <= date_trunc('hour',current_timestamp)
group by
    orders.id,
    clients.name,
    tables.name;
```

1st Attempt:

Basic - JDBC

Direct JDBC Query



clients

pizzas

tables

table_assignment

orders

```
CREATE TABLE src_tables (
    id int,
    name string,
    seats int
    )
WITH (
    'connector' = 'jdbc',
    'url' = 'jdbc:postgresql://',
    'table-name' = 'tables')",
    ...
    )
```

```
select  src_orders.id order_id,
        src_clients.name client_name,
        src_tables.name table_name,
        JSON_ARRAYAGG(JSON_OBJECT(
            'pizza' VALUE src_pizzas.name,
            'price' VALUE src_pizzas.price
            ))
from src_orders cross join unnest(src_orders.pizzas) as pizza_unnest(pizza_id)
  join src_pizzas on src_pizzas.id =  pizza_unnest.pizza_id
  join src_table_assignment
       on src_orders.table_assignment_id = src_table_assignment.id
  join src_clients on src_table_assignment.client_id = src_clients.id
  join src_tables on src_table_assignment.table_id = src_tables.id
where order_time > CEIL(LOCALTIMESTAMP to hour) - interval '1' hour
group by
    src_orders.id,
    src_clients.name,
    src_tables.name
```

✅

Use Apache Flink

SQL query is (almost) the same

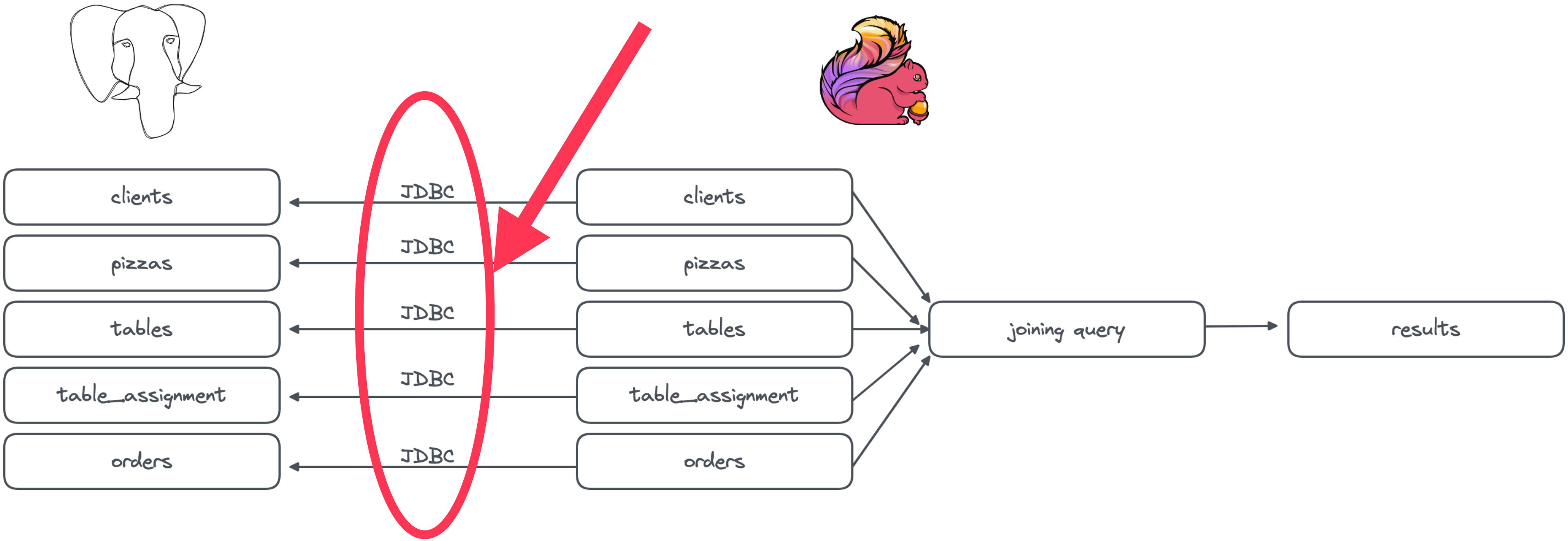
❌

Batch

Requires External Scheduler

Isolated queries - No Consistency

Lots of unfiltered data


clients
pizzas
tables
table_assignment
orders
JDBC
JDBC
JDBC
JDBC
JDBC
clients
pizzas
tables
table_assignment
orders
joining query
results

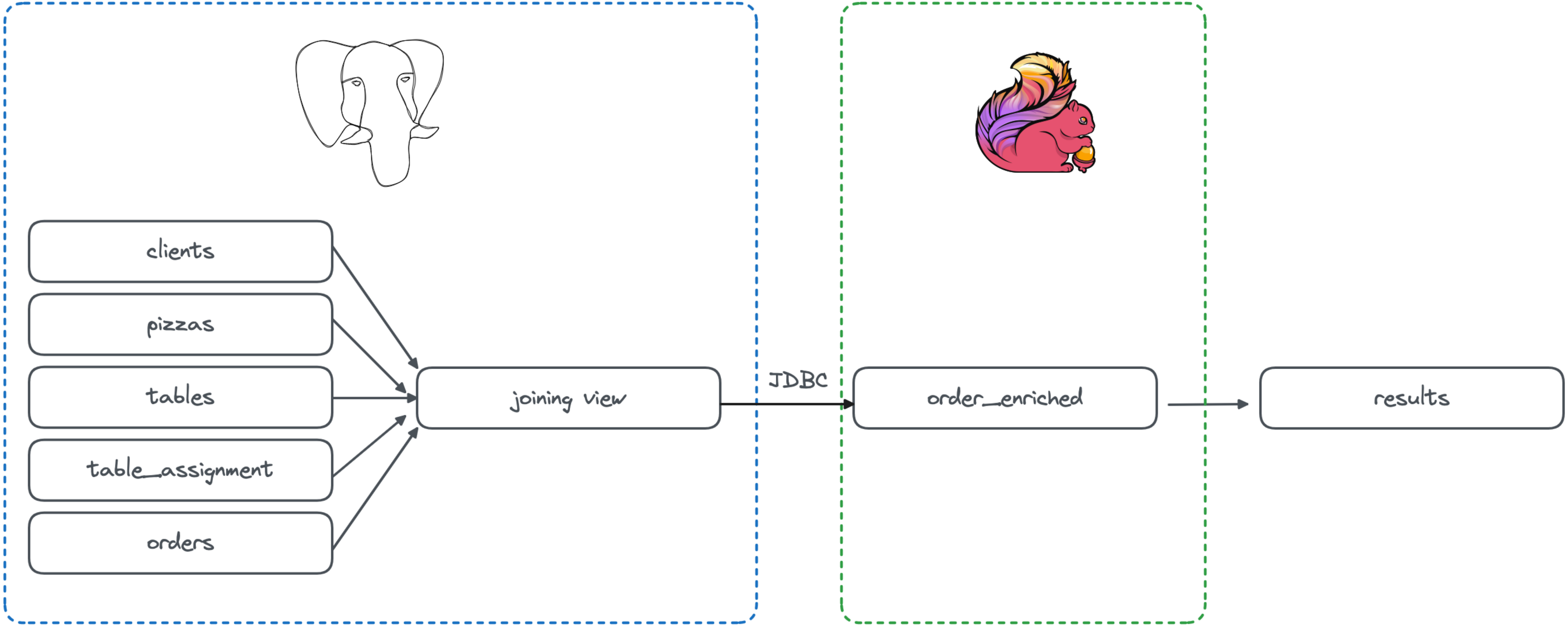
```
SELECT * FROM CLIENTS;

                SELECT * FROM ORDERS WHERE...;

 SELECT * FROM PIZZAS;

 SELECT * FROM TABLES;

                SELECT * FROM TABLE_ASSIGNMENTS;
```

```
[ERROR] Could not execute SQL statement.
Reason:
java.lang.OutOfMemoryError: Java heap space. A
heap space-related out-of-memory error has
occurred.
```

2nd Attempt:

Fix Consistency

JDBC View


clients
pizzas
tables
table_assignment
orders
joining view
JDBC
order_enriched
results

```
insert into order_output
select
  order_id,
  client_name,
  table_name,
  pizzas
from order_enriched_in
where
  order_time > CEIL(LOCALTIMESTAMP to hour)
               – interval '1' hour
  and order_time <= CEIL(LOCALTIMESTAMP to hour)
```

✅

Easy SQL query

Consistent Data

❌

Batch

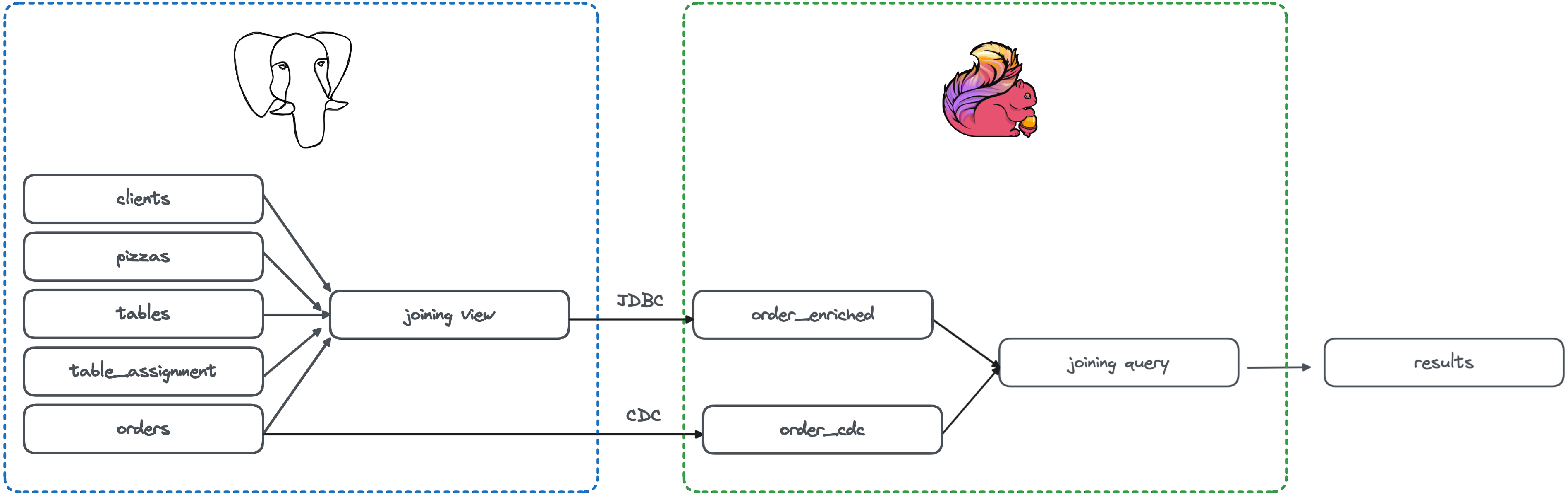
Requires External Scheduler

Lots of data movement

3rd Attempt:

Streaming

JDBC Lookup


clients
pizzas
tables
table_assignment
orders
joining view
JDBC
CDC
order_enriched
order_cdc
joining query
results

```
CREATE TABLE orders_cdc (
    id int,
    table_assignment_id int,
    order_time TIMESTAMP(3),
    PRIMARY KEY (id) not enforced,
    WATERMARK FOR order_time AS order_time
    )
WITH (
    'connector' = 'postgres-cdc',
    'database-name' = 'defaultdb',
    'hostname' = 'mydbhost',
    ...
    'schema-name' = 'public',
    'table-name' = 'orders'
    )
```

```
INSERT INTO order_output
select order_id,
    client_name,
    table_name,
    json_agg
from orders_cdc
join order_enriched_in
    FOR SYSTEM_TIME AS of order_enriched_in.proctime
on order_enriched_in.order_id=orders_cdc.id
```

Lookup join

Predicate Pushdown!

```
select * from orders_view
where order_id = 12345
```

✅

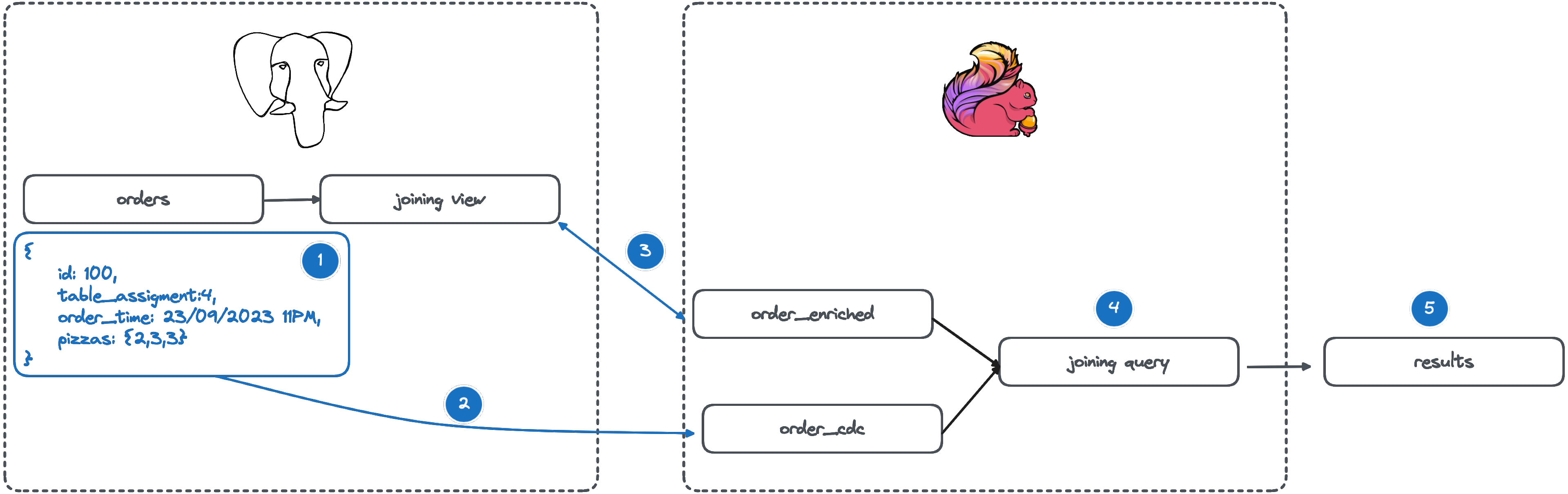
Streaming!

Consistent Data
(we read from the view)

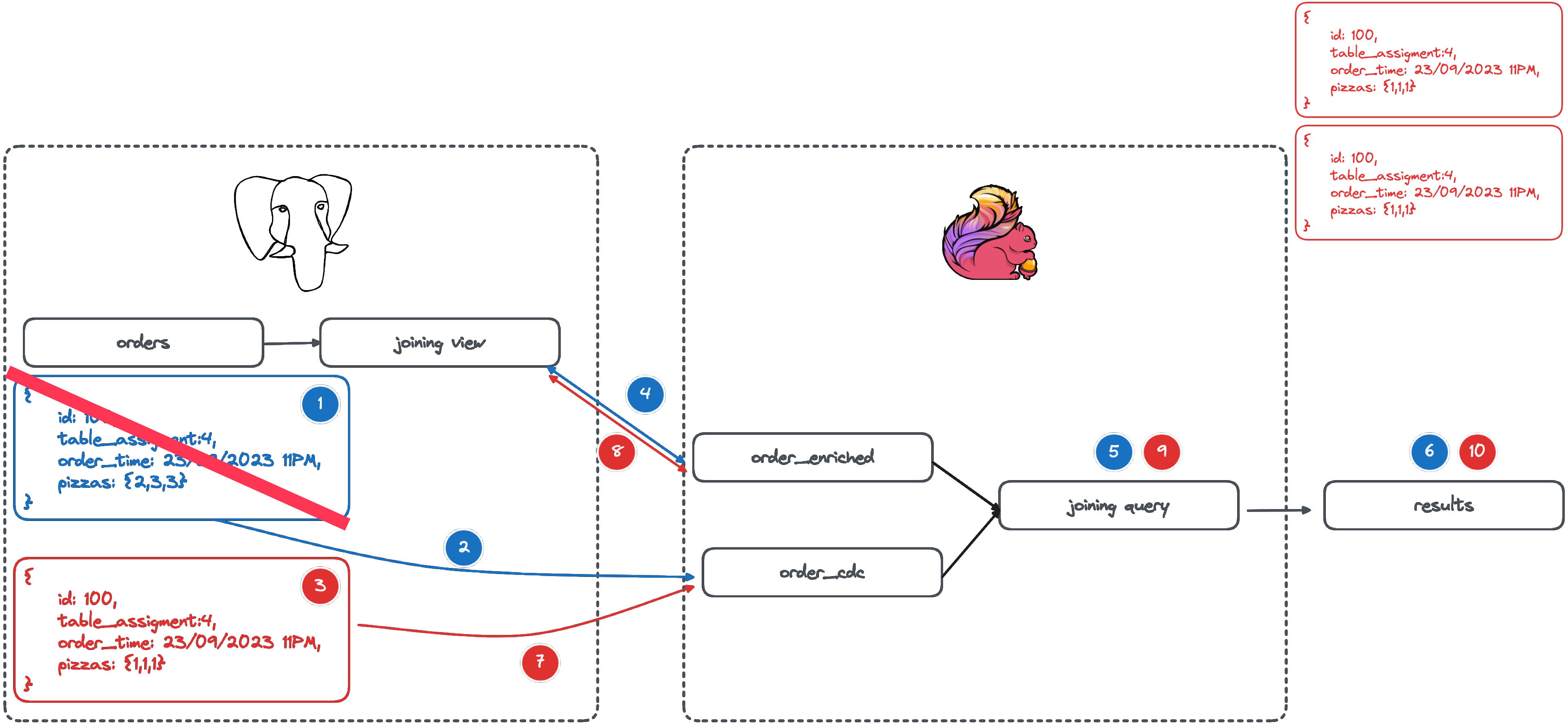
❌

Reading 2 times
from the DB

Is it really Consistent?



```
{
    id: 100,
    table_assigment:4,
    order_time: 23/09/2023 11PM,
    pizzas: {2,3,3}
}
```



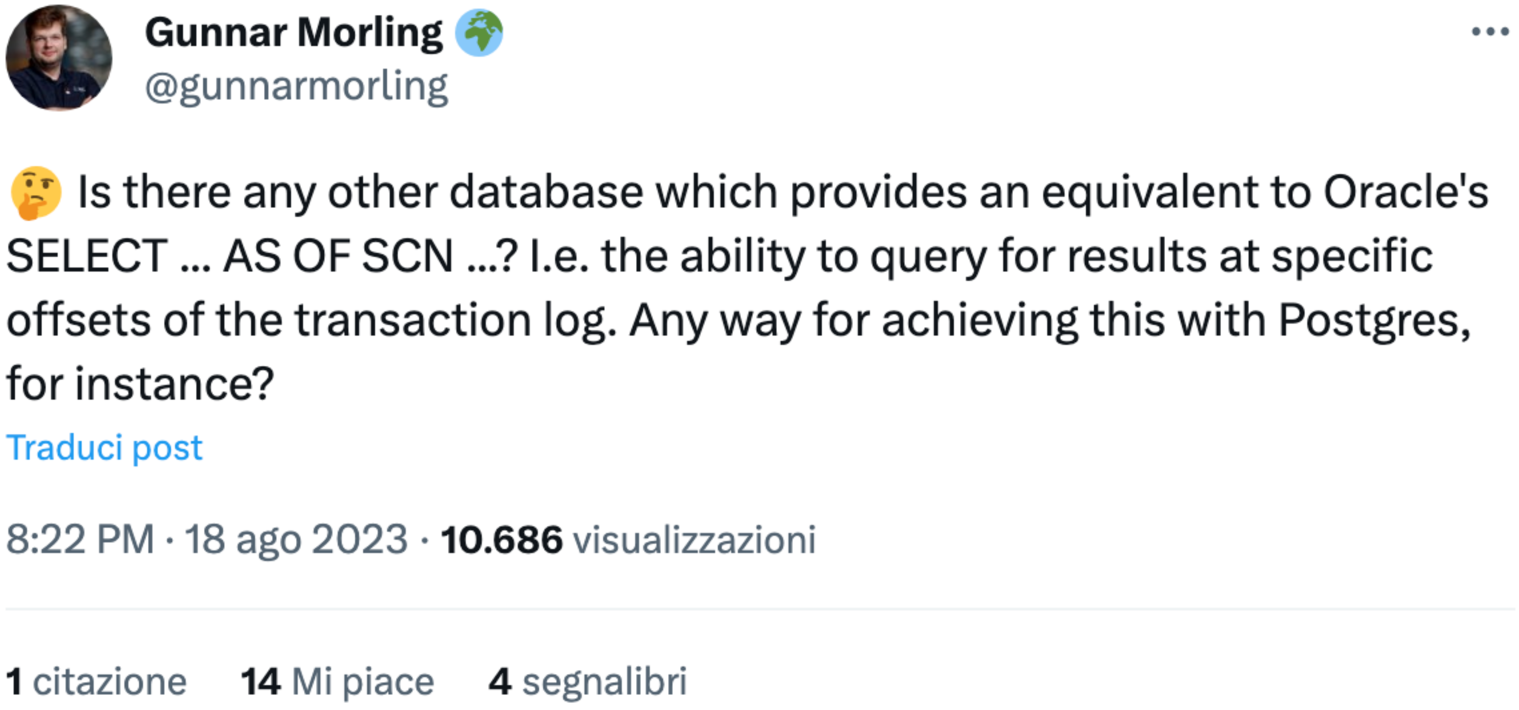
```
{
    id: 100,
    table_assigment:4,
    order_time: 23/09/2023 11PM,
    pizzas: {1,1,1}
}
```

```
{
    id: 100,
    table_assigment:4,
    order_time: 23/09/2023 11PM,
    pizzas: {1,1,1}
}
```

~~{~~
~~id: 100,~~
~~table_assigment:4,~~
~~order_time: 23/09/2023 11PM,~~
~~pizzas: {2,3,3}~~
~~}~~

```
{
    id: 100,
    table_assigment:4,
    order_time: 23/09/2023 11PM,
    pizzas: {1,1,1}
}
```

https://twitter.com/gunnarmorling/status/1692602893033934902



Gunnar Morling 🌍
@gunnarmorling

🤔 Is there any other database which provides an equivalent to Oracle's SELECT ... AS OF SCN ...? I.e. the ability to query for results at specific offsets of the transaction log. Any way for achieving this with Postgres, for instance?

Traduci post

8:22 PM · 18 ago 2023 · **10.686** visualizzazioni

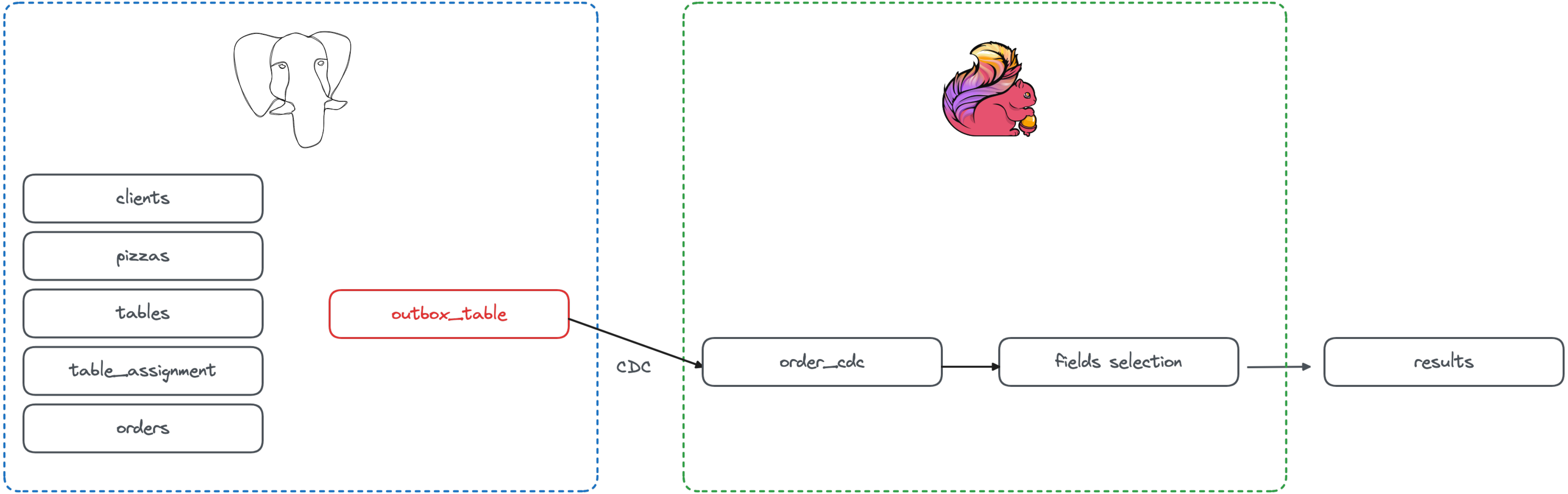
1 citazione **14** Mi piace **4** segnalibri

4th Attempt:

Streaming + Consistency

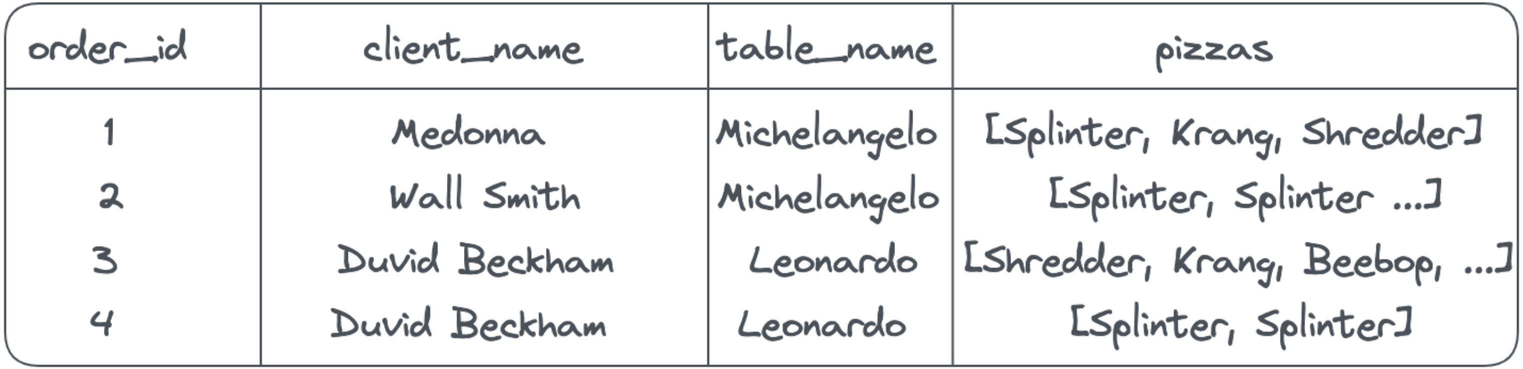
The Outbox pattern

Outbox Table



```
create table orders_outbox (
  order_id int,
  client_name text,
  table_name text,
  pizzas json
);
```

Order Outbox

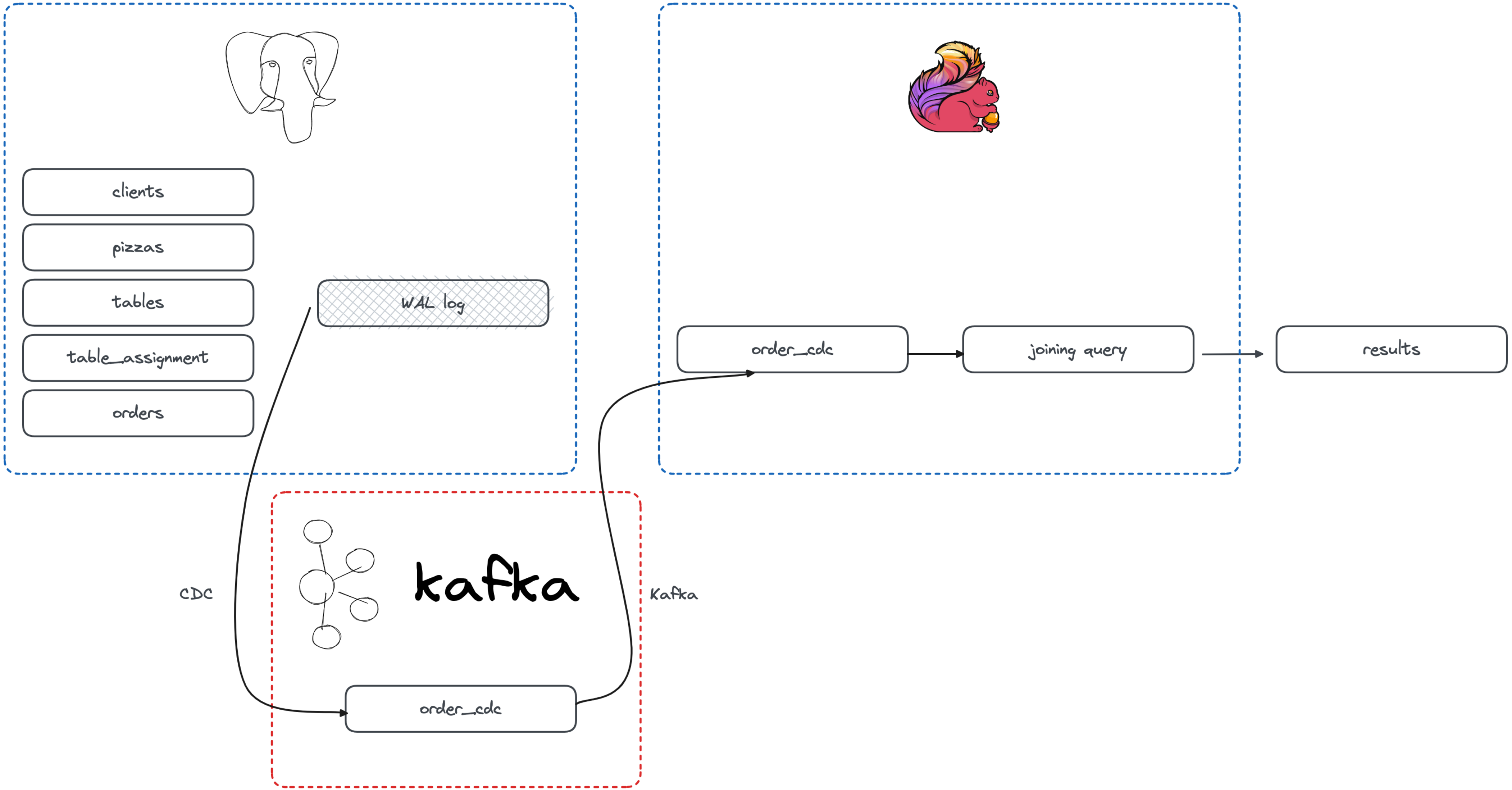


order_id	client_name	table_name	pizzas
1	Medonna	Michelangelo	[Splinter, Krang, Shredder]
2	Wall Smith	Michelangelo	[Splinter, Splinter ...]
3	Duvid Beckham	Leonardo	[Shredder, Krang, Beebop, ...]
4	Duvid Beckham	Leonardo	[Splinter, Splinter]

✅	❌
Streaming!	
Consistent Data	
Decouple representations	

✅	❌
Streaming!	DML Change needed
Consistent Data	Data is stored twice
Decouple representations	

PostgreSQL Logical Decoding Messages



francesco@aiven.io

@ftisiot | @aiven_io

Part of the transaction

```
SELECT *
FROM
pg_logical_emit_message(true,'myprefix',JSON_ORDER);
```

Decode

JSON extraction

```
INSERT into order_output
select
    JSON_VALUE(FROM_BASE64(message.content), '$.order_id'  RETURNING INT),
    JSON_VALUE(FROM_BASE64(message.content), '$.client_name'),
    JSON_VALUE(FROM_BASE64(message.content), '$.table_name'),
    JSON_QUERY(FROM_BASE64(message.content), '$.pizzas[*]')
from pg_messages
```

✅

Streaming!

Consistent Data

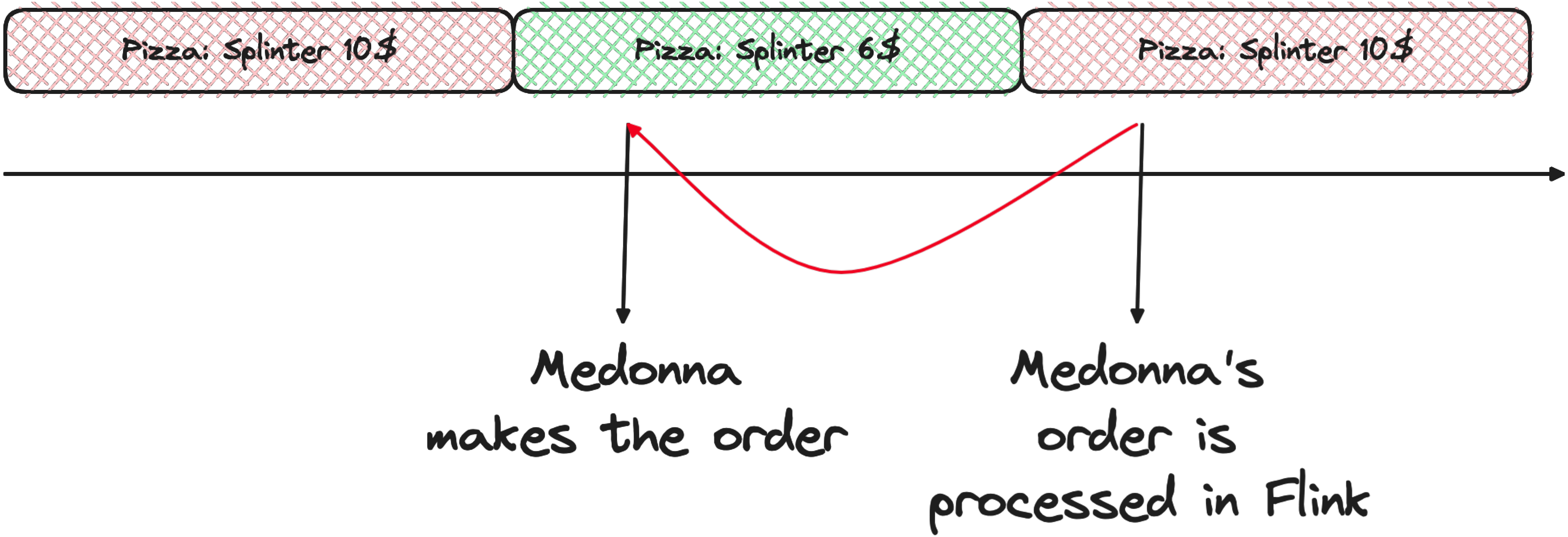
❌

DML Change needed

~~Data is stored twice~~

5th Attempt:

All in Flink - Temporal joins



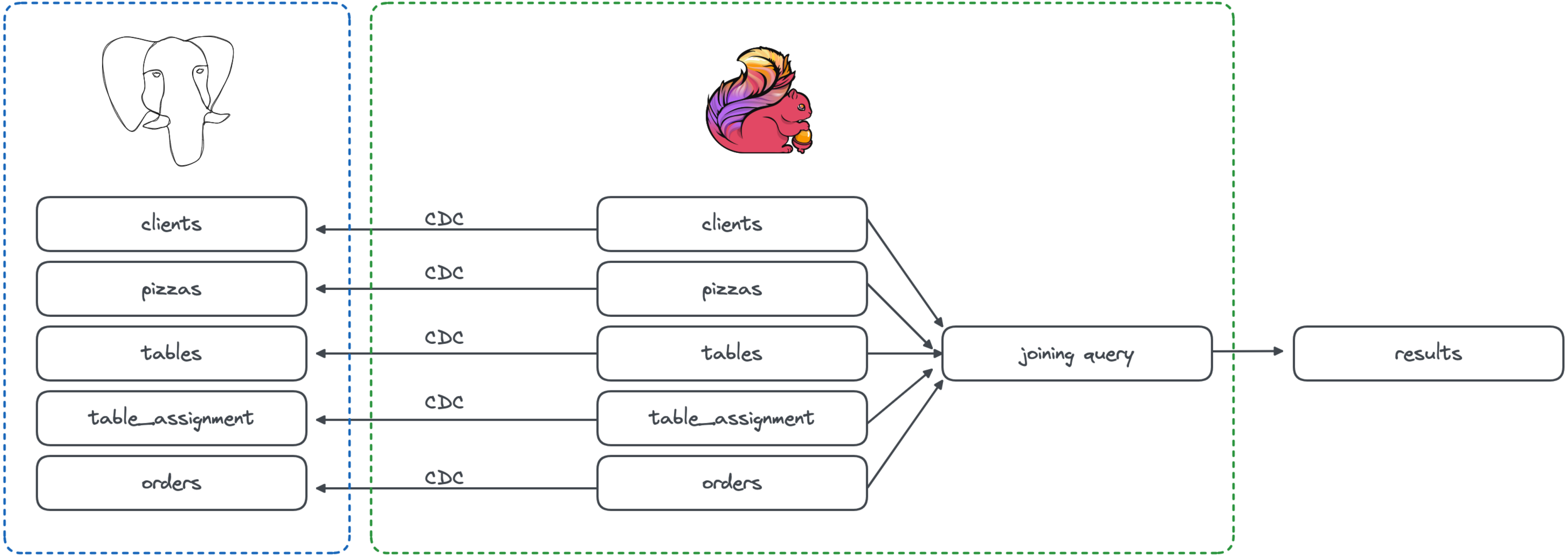
Pizza: Splinter 10$

Pizza: Splinter 6$

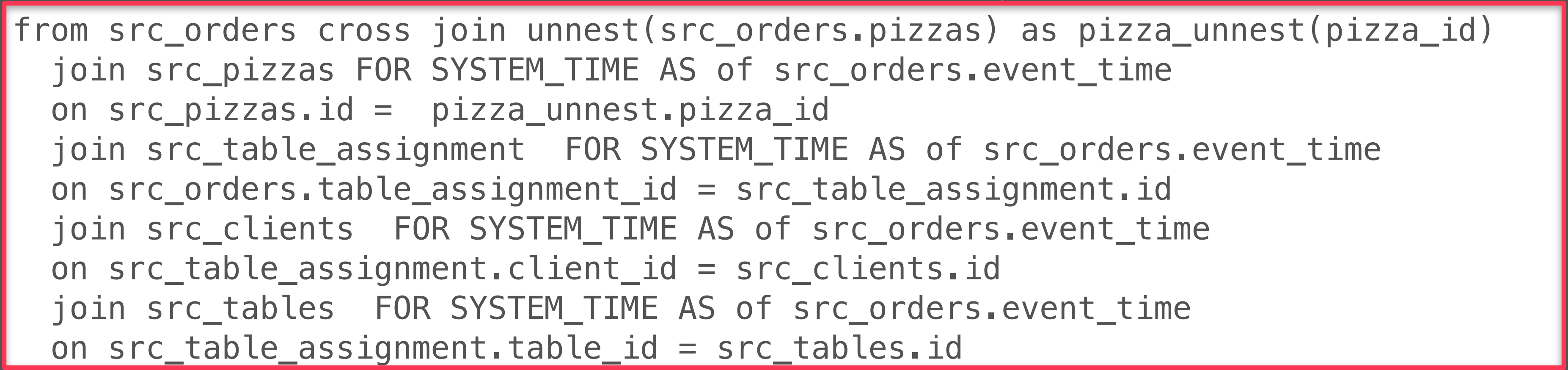
Pizza: Splinter 10$

Medonna makes the order

Medonna's order is processed in Flink


clients
pizzas
tables
table_assignment
orders
CDC
CDC
CDC
CDC
CDC
clients
pizzas
tables
table_assignment
orders
joining query
results

Temporal Joins



```
from src_orders cross join unnest(src_orders.pizzas) as pizza_unnest(pizza_id)
  join src_pizzas FOR SYSTEM_TIME AS of src_orders.event_time
  on src_pizzas.id =  pizza_unnest.pizza_id
  join src_table_assignment  FOR SYSTEM_TIME AS of src_orders.event_time
  on src_orders.table_assignment_id = src_table_assignment.id
  join src_clients  FOR SYSTEM_TIME AS of src_orders.event_time
  on src_table_assignment.client_id = src_clients.id
  join src_tables  FOR SYSTEM_TIME AS of src_orders.event_time
  on src_table_assignment.table_id = src_tables.id
```

✅

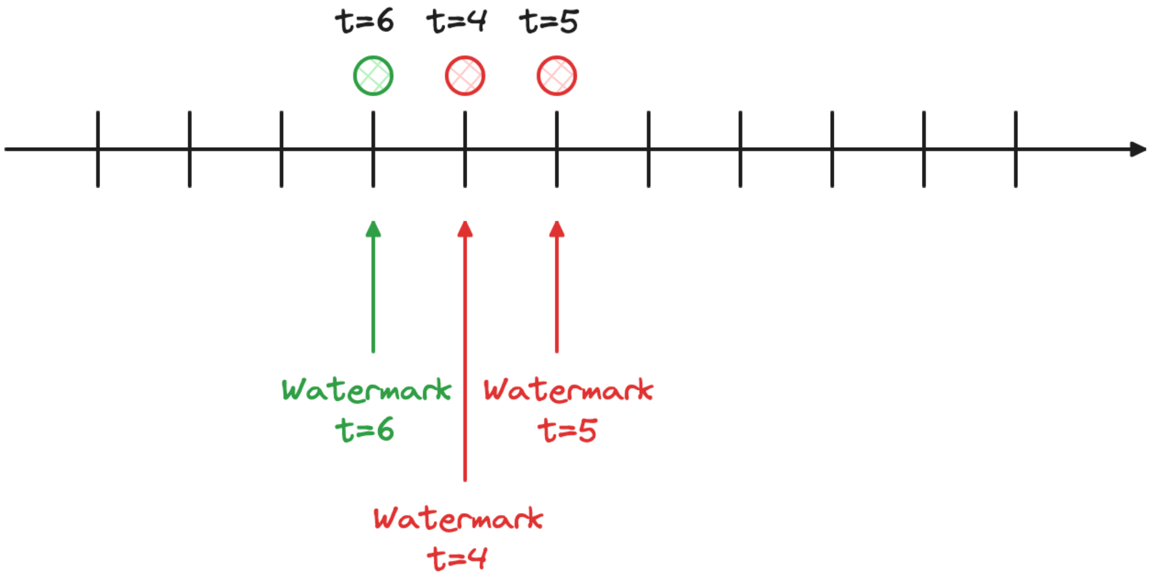
Streaming!

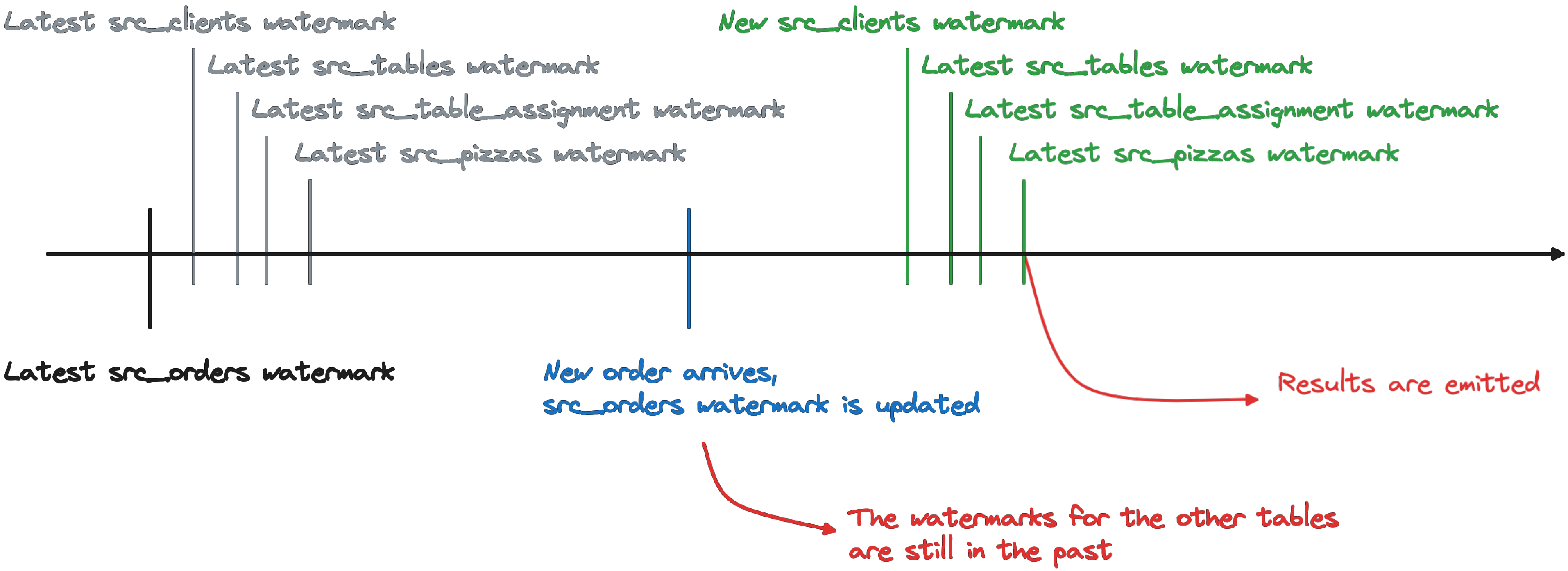
Consistent Data

❌

~~DML Change needed~~

Watermarks





Latest src_clients watermark

Latest src_tables watermark

Latest src_table_assignment watermark

Latest src_pizzas watermark

New src_clients watermark

Latest src_tables watermark

Latest src_table_assignment watermark

Latest src_pizzas watermark

Latest src_orders watermark

New order arrives,
src_orders watermark is updated

Results are emitted

The watermarks for the other tables
are still in the past

How to Advance Watermarks

`table.exec.source.idle-timeout`

Database **triggers**

Debezium **heartbeat action** query & **interval**

✅

Streaming!

Consistent Data

❌

~~DML Change needed~~

Additional load in the database

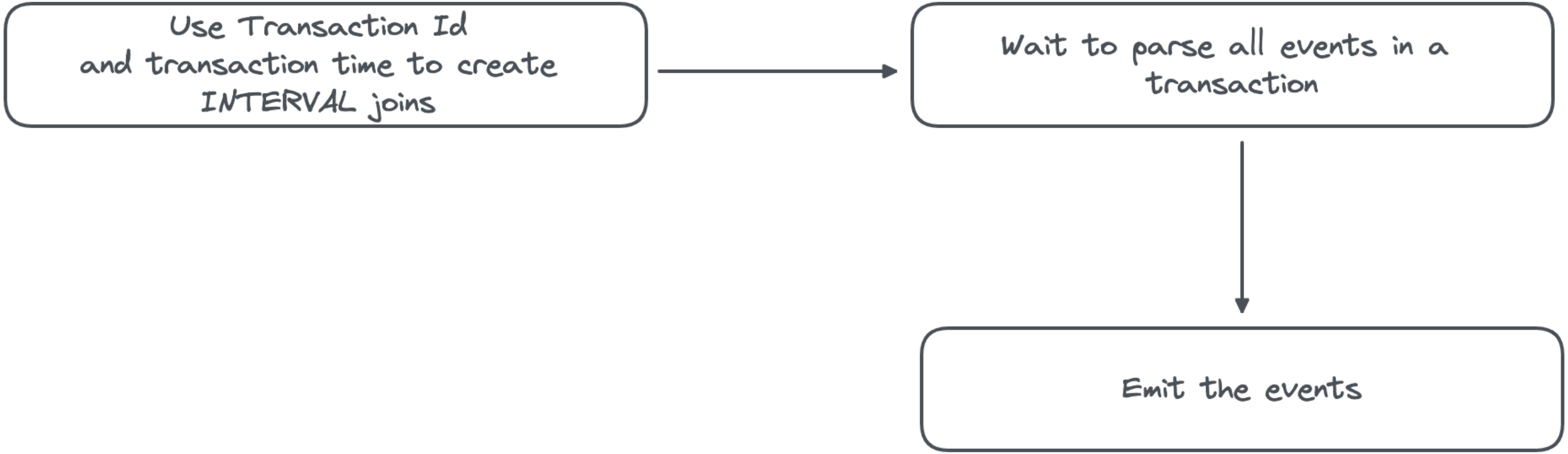
Trigger interval -> Batch?

6th Attempt:

All in Flink - Transaction boundaries

https://debezium.io/documentation/reference/stable/connectors/postgresql.html#postgresql-transaction-metadata

```
{
  "before": null,
  "after": {"pk": "2","aa": "1"},
  "source": {...},
  "op": "c",
  "ts_ms": "1580390884335",
  "transaction": {
    "id": "53195832",
    "total_order": "1",
    "data_collection_order": "1"
  }
```



Use Transaction Id and transaction time to create INTERVAL joins

Wait to parse all events in a transaction

Emit the events

✅

Streaming!

Consistent Data

❌

~~DML Change needed~~

~~Additional load in the database~~

✅

Streaming!

Consistent Data

❌

~~DML Change needed~~

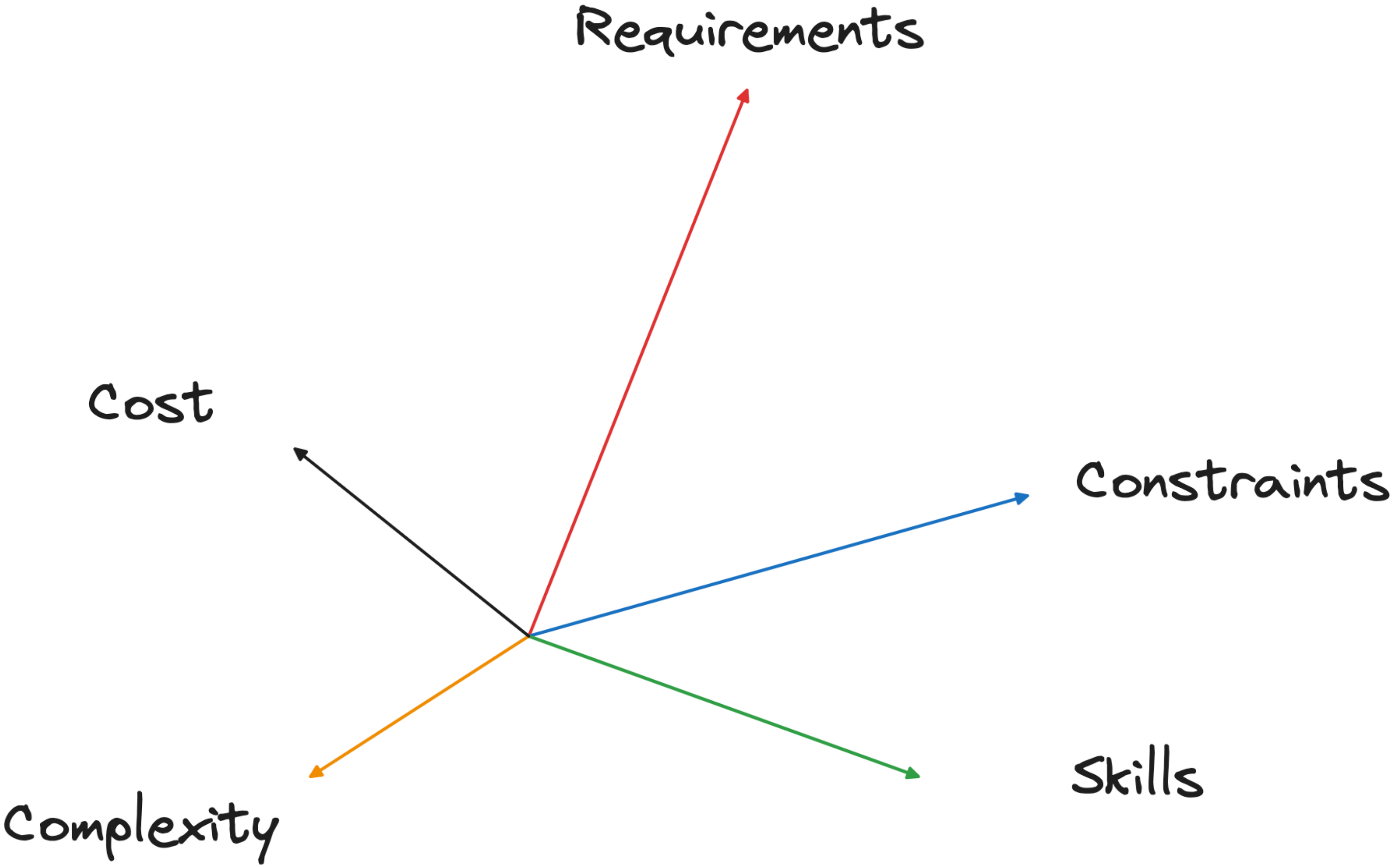
~~Additional load in the database~~

Complex queries

Are we at the summary already?

	Streaming	Consistency	DML Changes	Additional DB Load	Simple Join
Direct JDBC	❌	❌	✅	✅	✅
JDBC with View	❌	✅	✅	✅	✅
CDC with JDBC Lookup	✅	⚠️	✅	❌	✅
Outbox pattern	✅	✅	❌	❌	✅
Flink Temporal Joins	⚠️	✅	✅	❌	✅
Flink Transactional Joins	✅	✅	✅	✅	❌

It's SQL!



Requirements

Cost

Constraints

Complexity

Skills

INSERT YOUR NAME

Boss, It's done!

Slides

https://go.aiven.io/ft-current-23

Try in Aiven

300$

Extra
100$

https://go.aiven.io/ft-current-23-signup