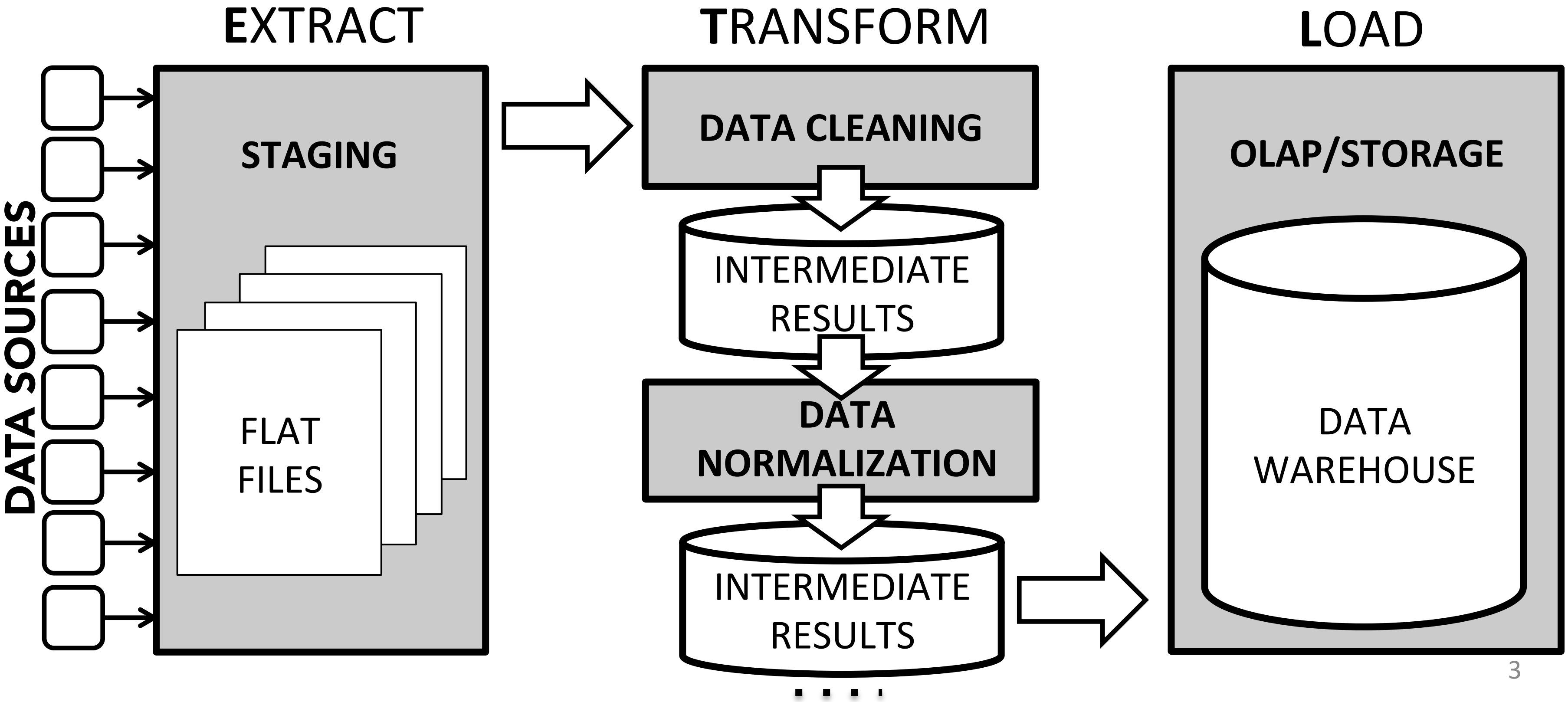
Data Ingestion for the Connected World

John Meehan, Cansu Aslantas, Stan Zdonik (Brown University)
Nesime Tatbul (Intel Labs & MIT)
Jiang Du (University of Toronto)

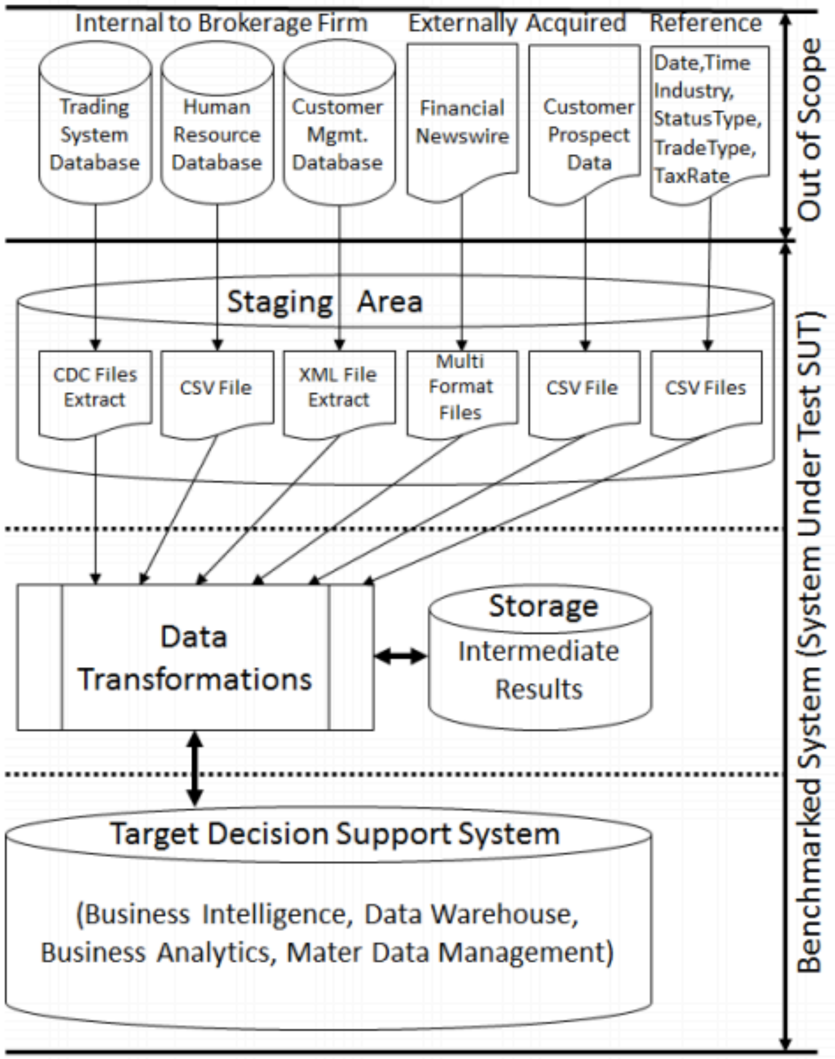
The IoT Era

Traditional Data Ingestion (ETL)



An Example: TPC-DI

- Brokerage firm
- 6 heterogeneous sources
- 3 key parts:
  1. Ingest raw data
  2. ETL transform
  3. Update warehouse

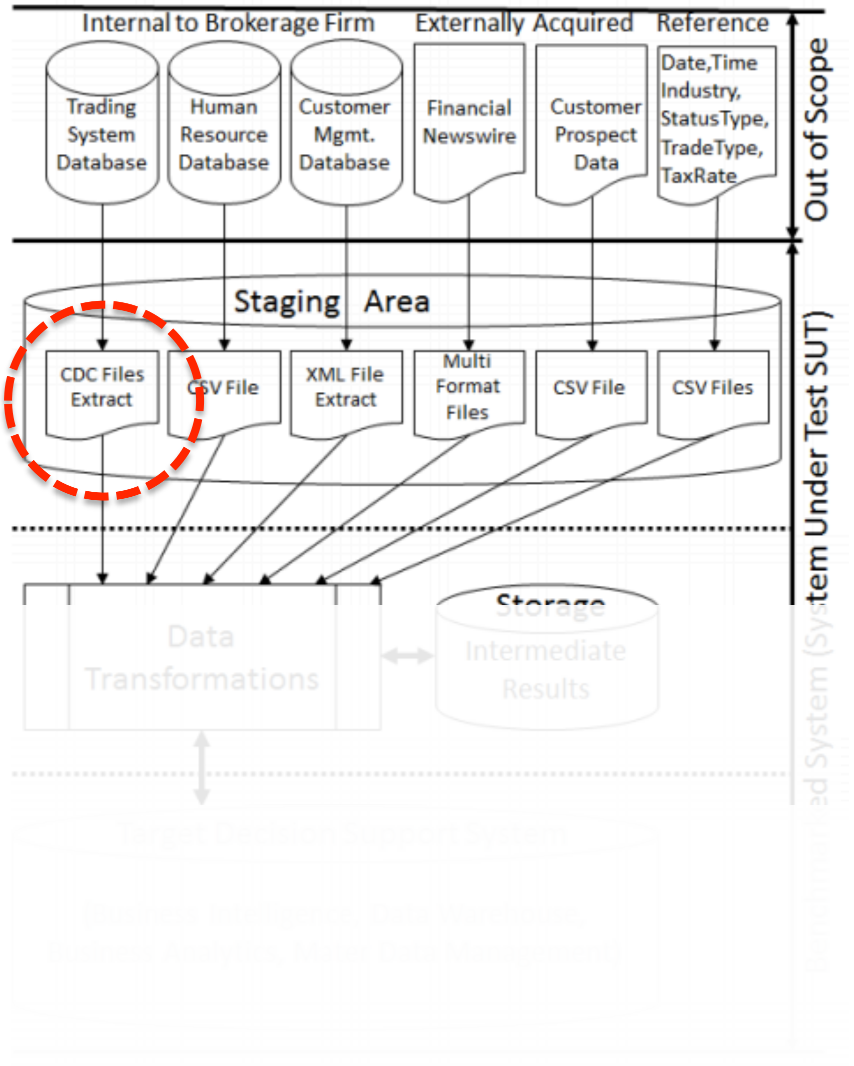


TPC Transaction Processing Performance Council

http://www.tpc.org/tpcdi/
Poess et al, VLDB 2014

An Example: TPC-DI

- Brokerage firm
- 6 heterogeneous sources
- 3 key parts:
  1. Ingest raw data



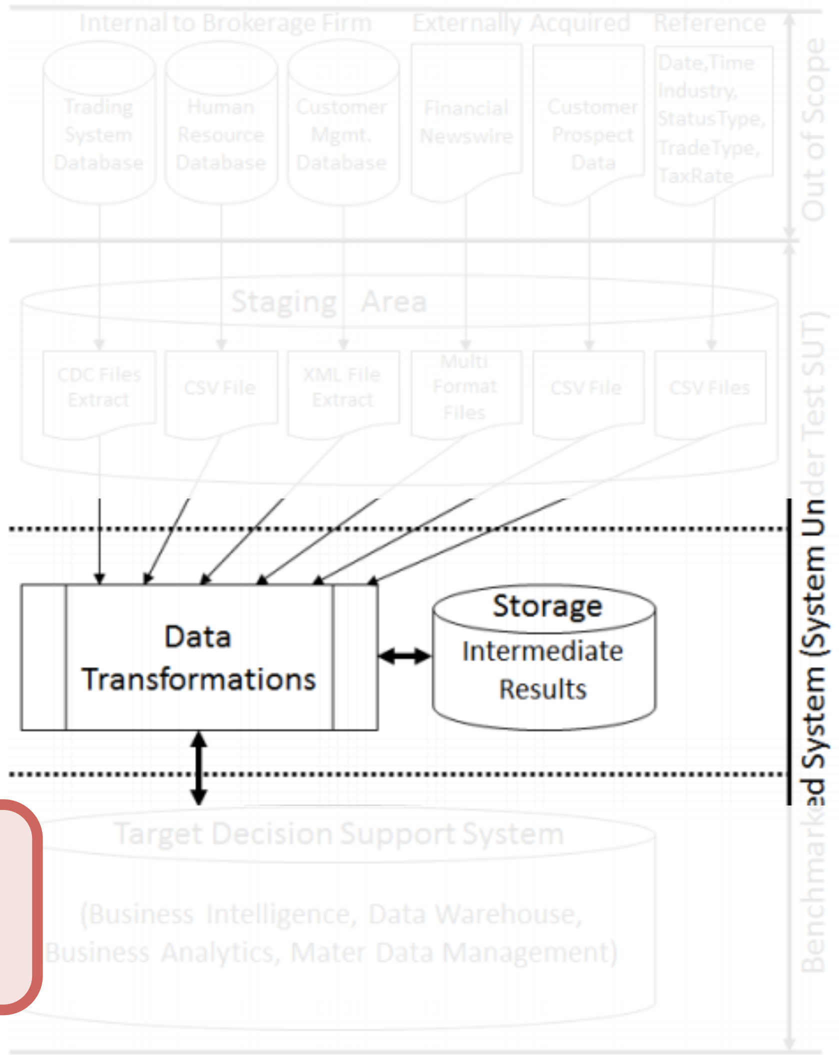
- ✓ Data collected into flat files
- ✓ Heterogeneous data types
- ✓ Incremental update from an OLTP source, once a day

TPC Transaction Processing Performance Council

http://www.tpc.org/tpcdi/
Poess et al, VLDB 2014.

An Example: TPC-DI

- Brokerage firm
- 6 heterogeneous sources
- 3 key parts:
  1. Ingest raw data
  2. ETL transform
  3. Update warehouse



✓ Storage for intermediate results
✓ Transactional state management

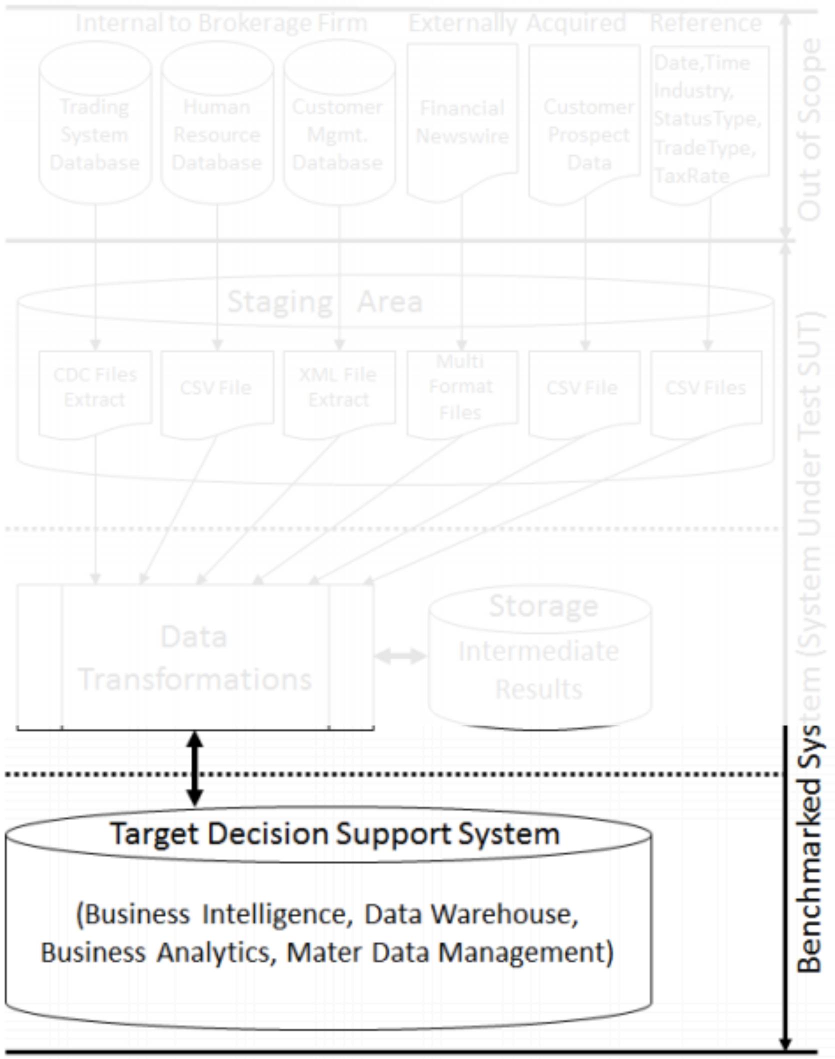
Performance Council

Poess et al, VLDB 2014.

An Example: TPC-DI

- Brokerage firm
- 6 heterogeneous sources
- 3 key parts:
  1. Ingest raw data
  2. ETL transform
  3. Update warehouse

✓ Bulk loading



TPC Transaction Processing Performance Council

http://www.tpc.org/tpcdi/
Poess et al, VLDB 2014.

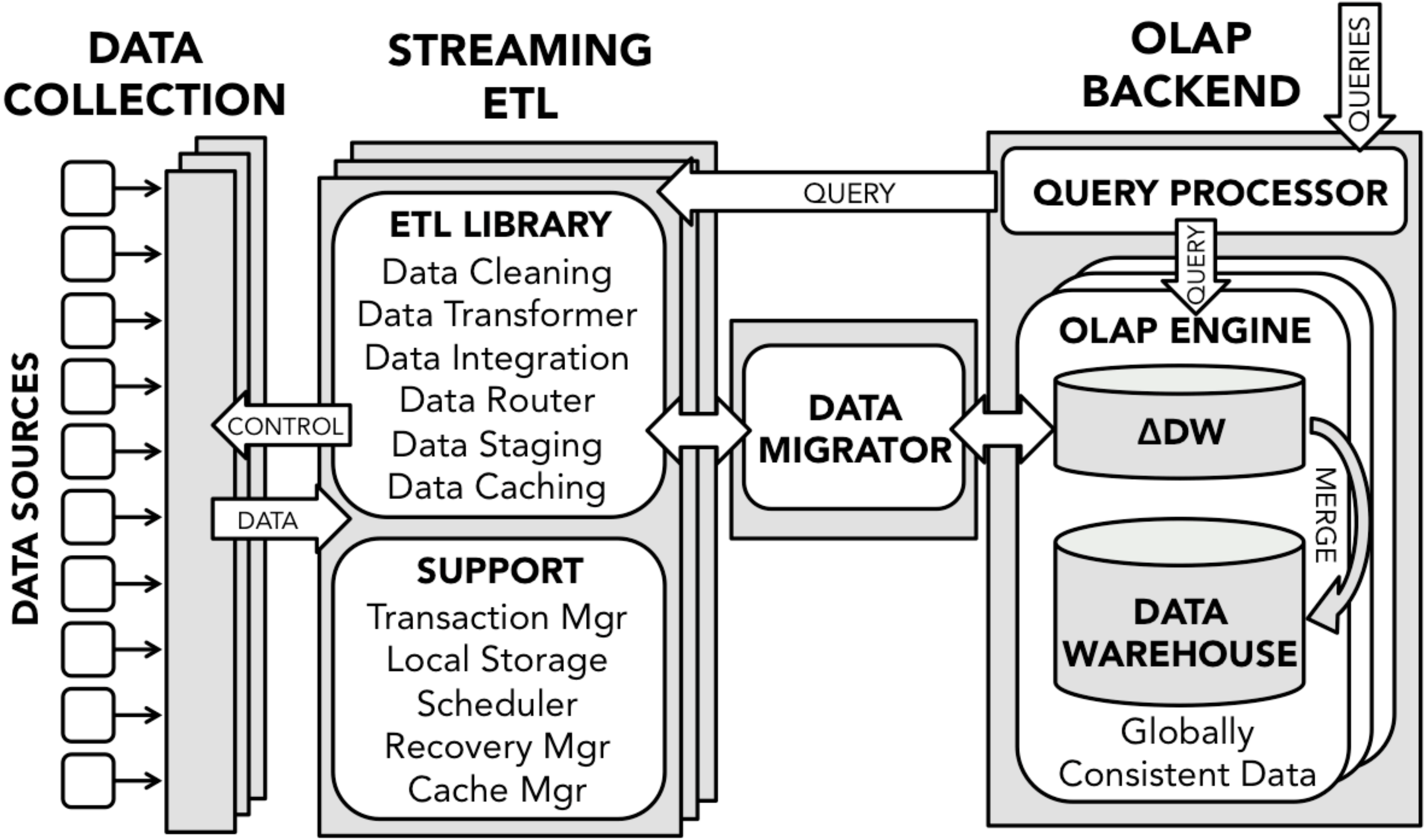
Streaming Data Ingestion

- In modern apps such as IoT:
  - real-time streams of data from a large number of sources
  - majority of these sources report in the form of time-series
  - data currency & low latency is key for real-time decision making & control

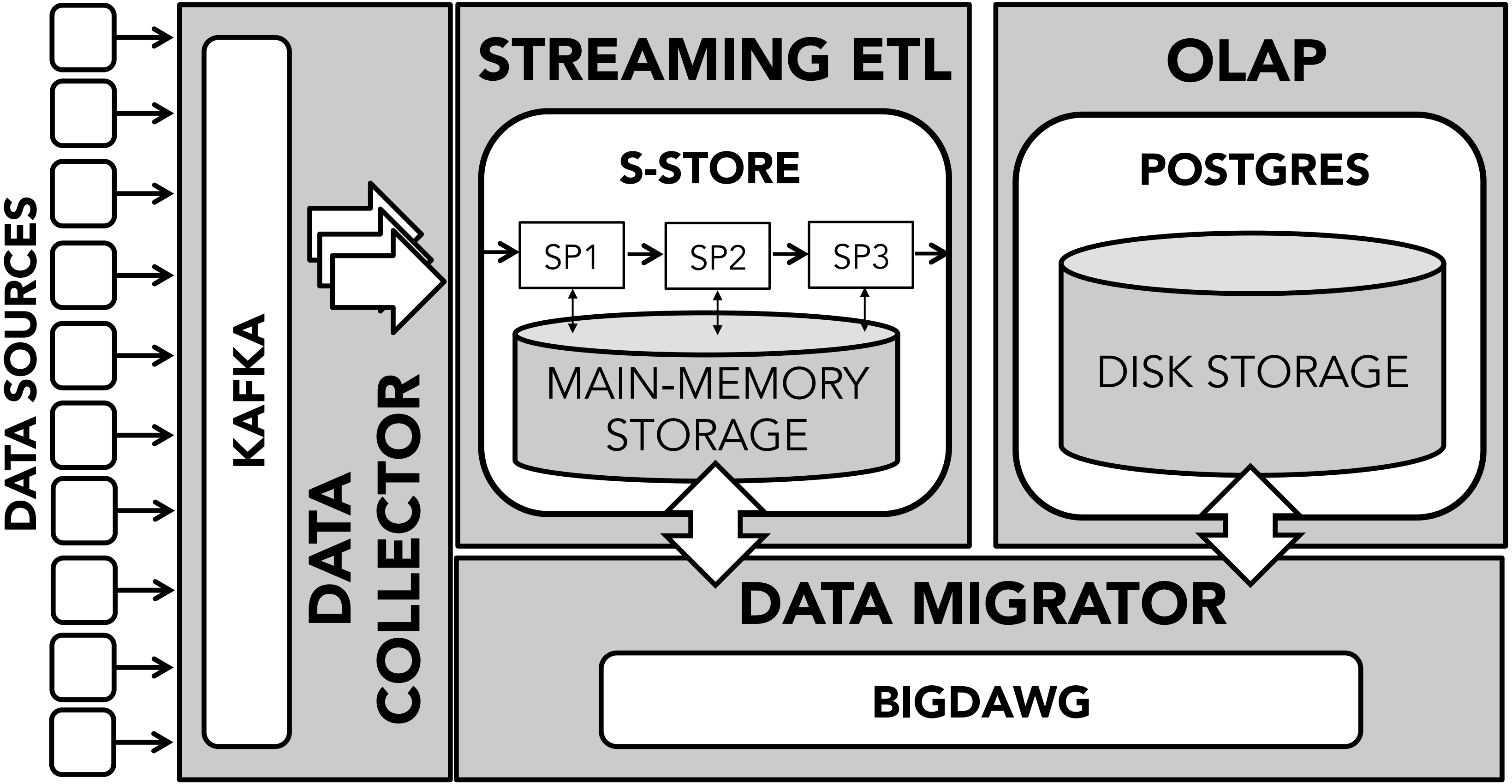
✓ Need a stream-based ingestion architecture

✓ Must pay attention to time-series data type and operations (both during ingestion & analytics)

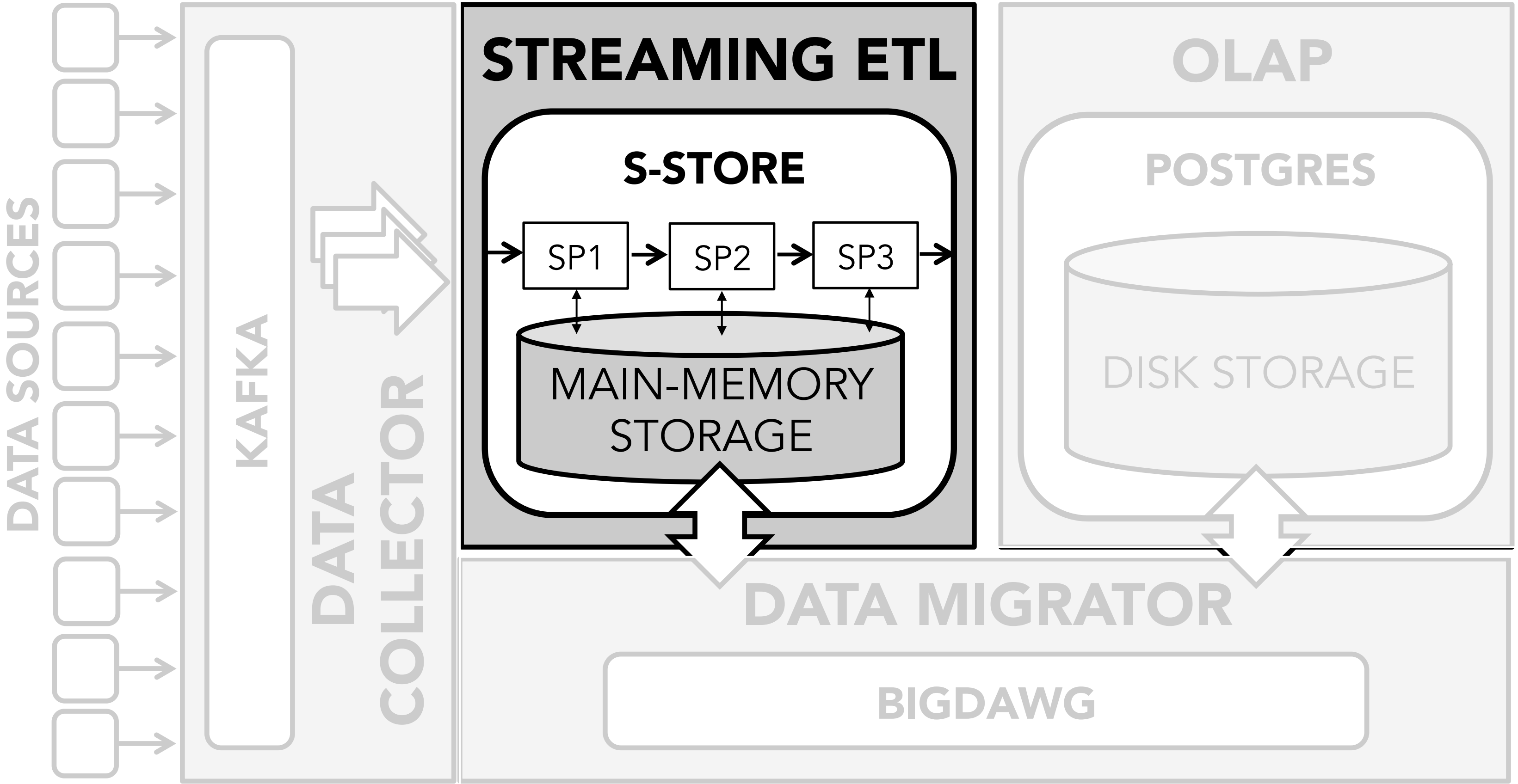
An Architecture for Streaming Data Ingestion



Implementation



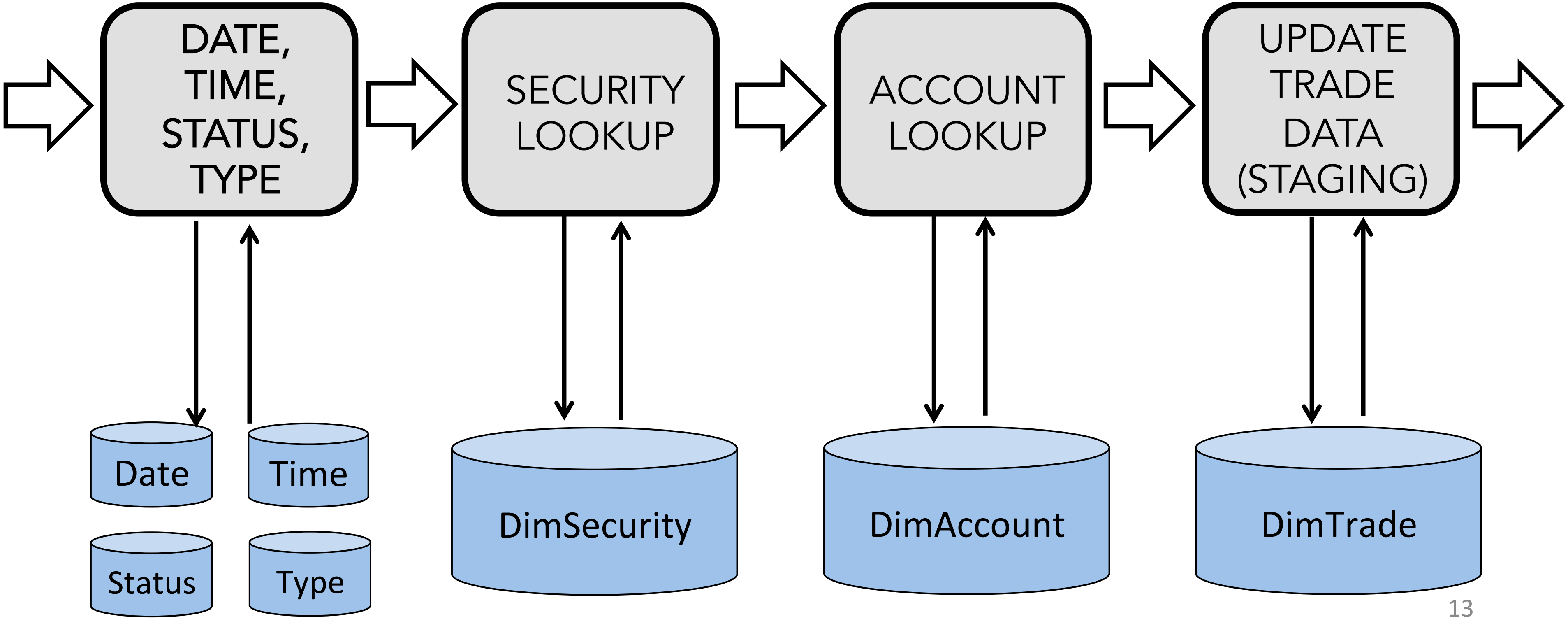
Implementation



S-Store: Shared Mutable State in Streaming

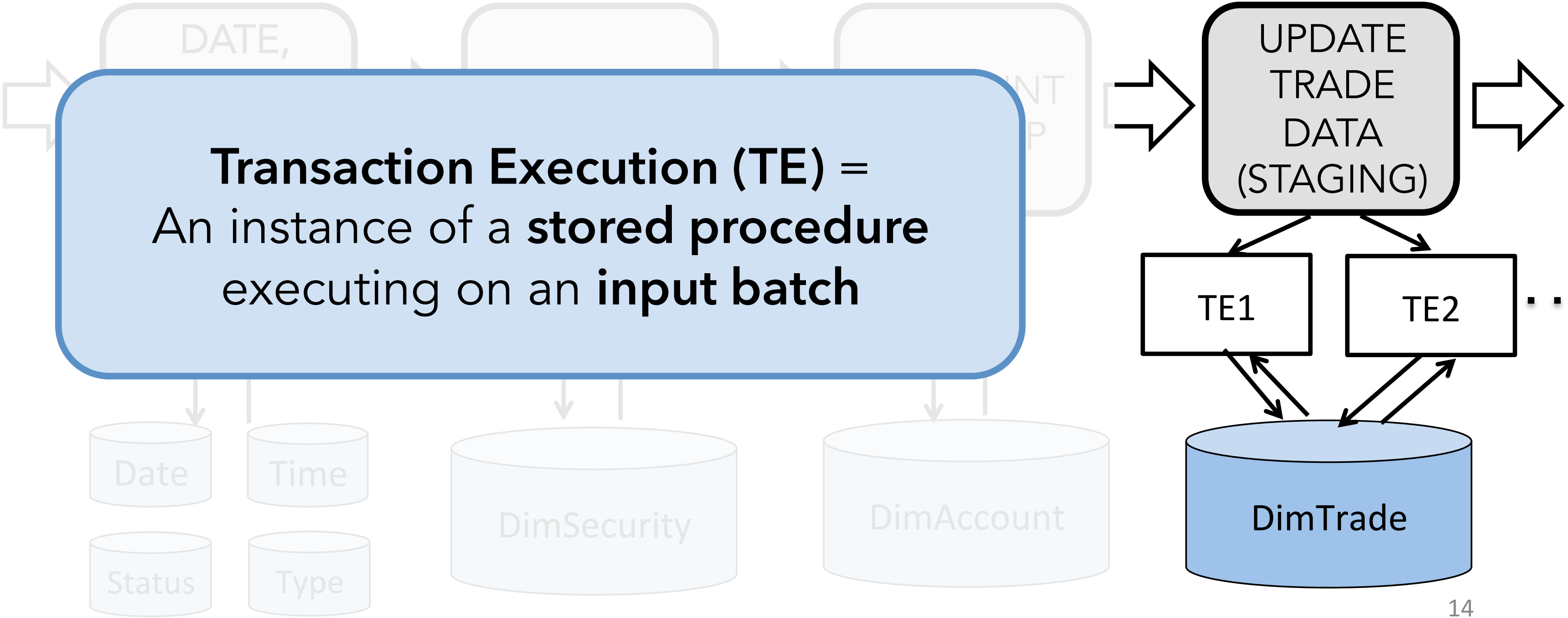
- A hybrid system for transaction & stream processing
  - combines main-memory OLTP with streaming constructs (windowing, triggers, dataflow graphs)
- Transactions as user-defined stored procedures (Java + SQL)
- Three complementary correctness guarantees
  - **ACID**, for individual transactions
  - **Ordered execution**, for streams and dataflow graphs
  - **Exactly-once processing**, for streams (no loss or duplicates due to failures/recovery)

Example: A TPC-DI Dataflow Graph in S-Store

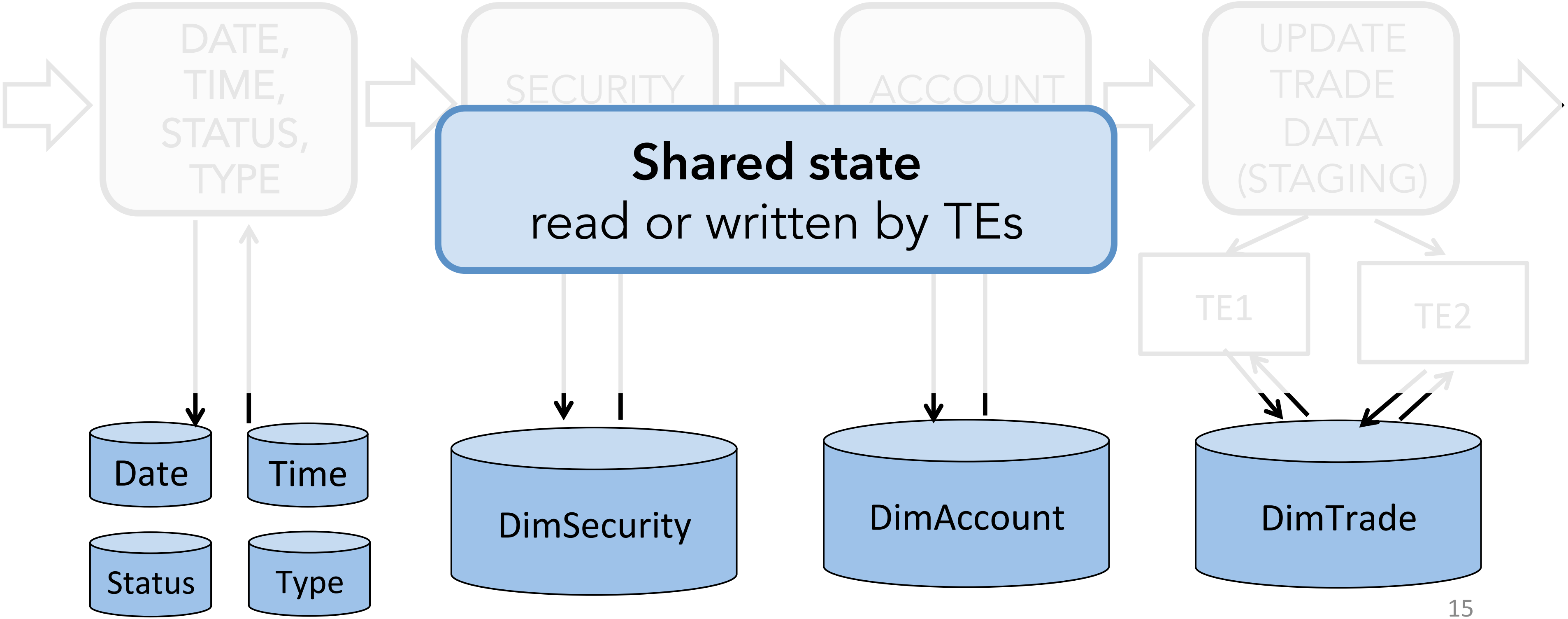


Example: A TPC-DI Dataflow Graph in S-Store

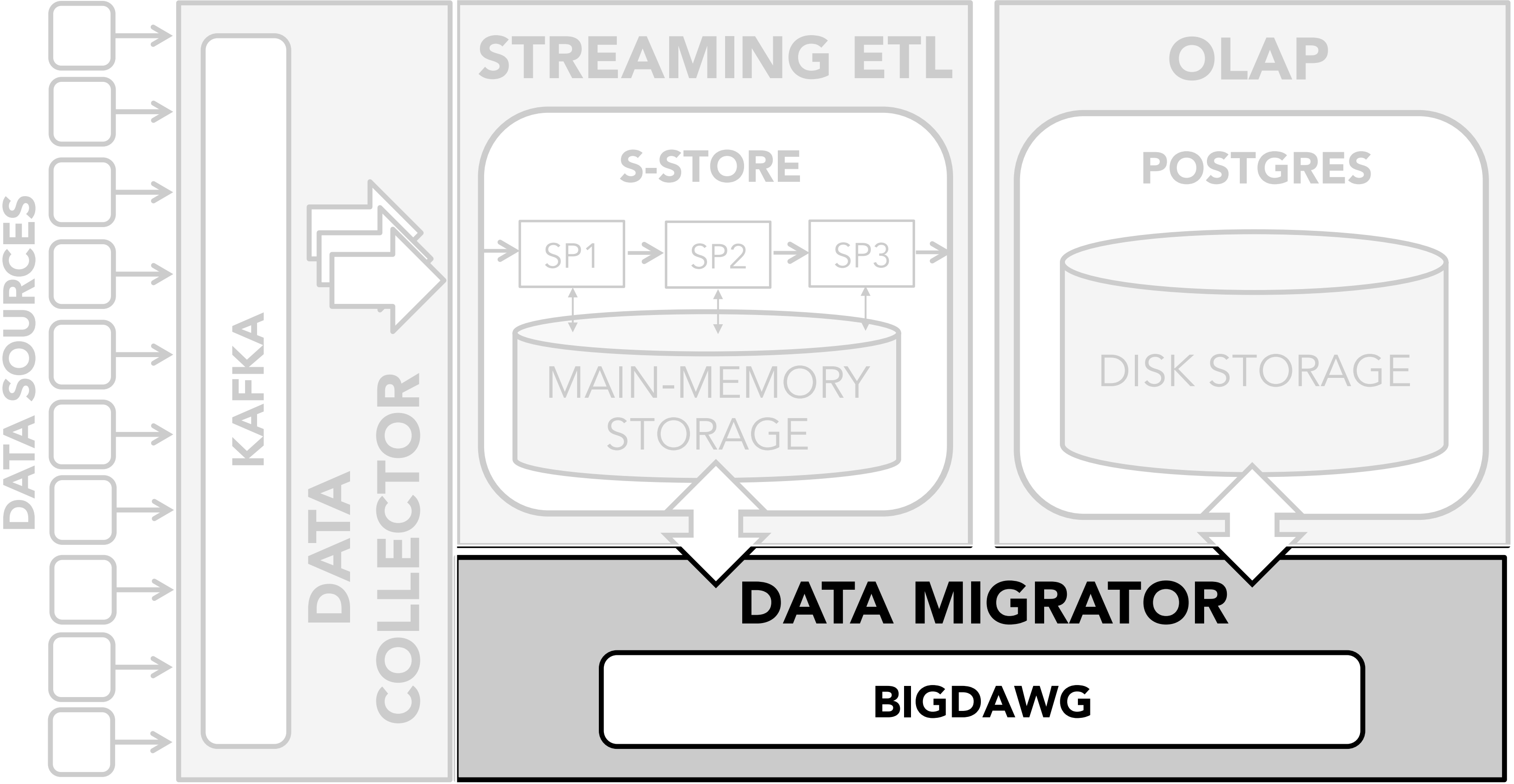
Transaction Execution (TE) =
An instance of a **stored procedure**
executing on an **input batch**



Example: A TPC-DI Dataflow Graph in S-Store



Implementation



Data Migrator

- Provides durable migration into the data warehouse using an ack mechanism that simulates 2PC
- Leverages the BigDAWG polystore middleware (*see Session 4*)
  - can support a variety of destination warehouses
  - can participate in federated querying
- Supports both "push" and "pull" modes

TPC-DI Experiment: Push vs. Pull Tradeoffs

- How often to migrate? Push or pull?
- Impacts:
  - Maximum ingest latency in S-Store
  - Query execution time in Postgres
  - Staleness of the query results in Postgres
- Result summary: Push in small batches, every 1-5 seconds. Fine-grained ingestion performs well.

Ongoing Work

- **Time-series** data management (ingestion & beyond)
  – New ingestion challenges and opportunities (e.g., synchronization/alignment of time-series, using predictive techniques for dealing with missing/delayed values)
  – Append-based updates, window-based reads
  – Need to support complex analytics operations (forecasting/prediction, pattern matching, anomaly detection, signal processing)
  – Exploit the resources on edge devices