DIONYSUS:
Towards Query-aware Distributed Processing of RDF Graph Streams

Syed Gillani, Gauthier Picard, Frederique Laforest

Laboratoire Hubert Curien & Institute Mines St-Etienne, France

GraphQ 2016

LABORATOIRE HUBERT CURIEN
UMR • CNRS • 5516 • SAINT-ETIENNE

MINES
Saint-Étienne

[Outline]

- Stream processing in general
- Semantic-enabled stream processing (RDF stream processing)
- Issues and challenges for RDF stream processing
- Expectations from DIONYSUS
- Functional Layers of DIONYSUS

[The Data Deluge]

- More than 3000 Exabytes (billions GBs) created in 2015 alone
  - Increased from 150 Exabytes in 2005

- Many new sources of data become available
  - Sensors, mobile devices
  - Web feeds, social networks
  - Surveillance video and audio
  - Knowledge Bases
  - ....

- **How can we make sense of all data**
  - Most of the data is not interesting
  - New data supersedes old data
  - Challenge is not only **storage** but **processing**

[Stream Processing to the Rescue!]

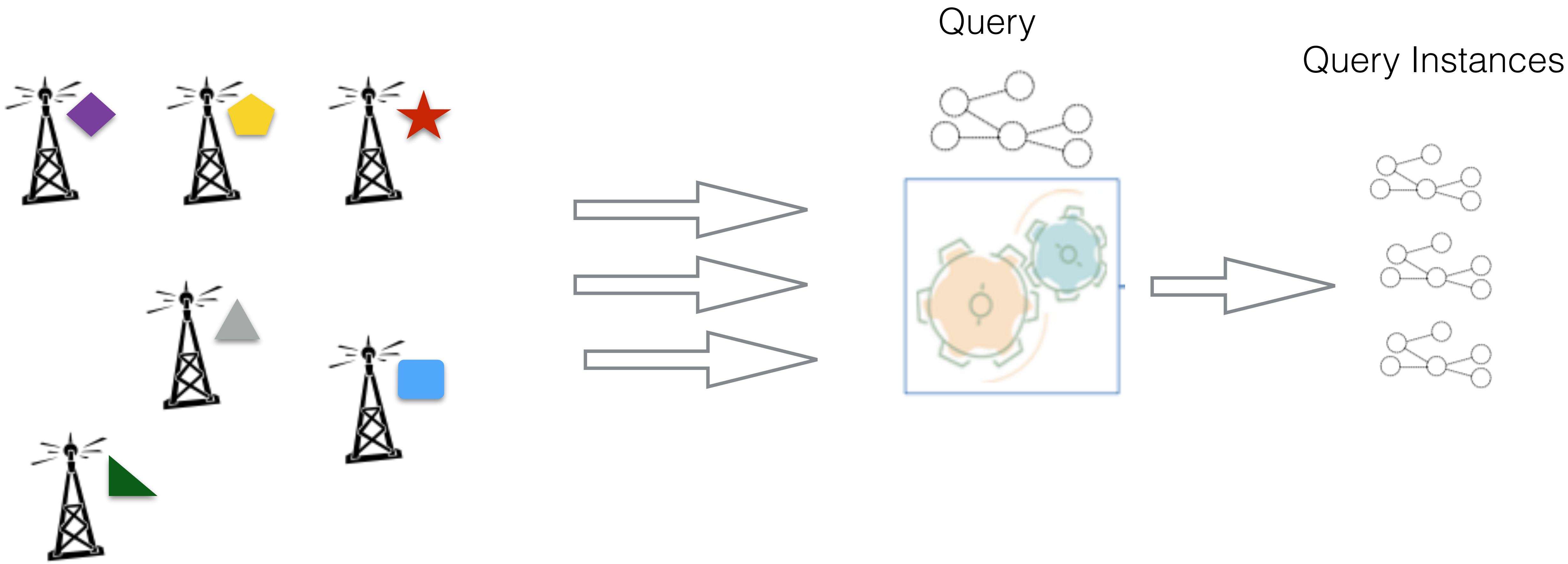
- **Process data streams on the fly without storage**

- Stream data rates can be high
  - Volume, type, frequency can vary
  - High resource requirement for processing

- Processing streams have real-time requirement
  - Latency of data processing matters
  - Limited amount of available memory
  - MUST be able to react to the events as they occur (Complex Event Processing)

- **Use cases**: Power management in Smart Grid, Traffic management, Social network analysis, fraud detection etc,.

[Stream Processing is it enough?]



- Heterogeneity of sources, multiple Schemas, materialisation of implicit Knowledge

[RDF Stream Processing]

- Utilising Semantic Web Technologies

  - Facilitating data integration by using machine processable descriptions to reconcile heterogeneities (e.g., Semantic Sensor Web)

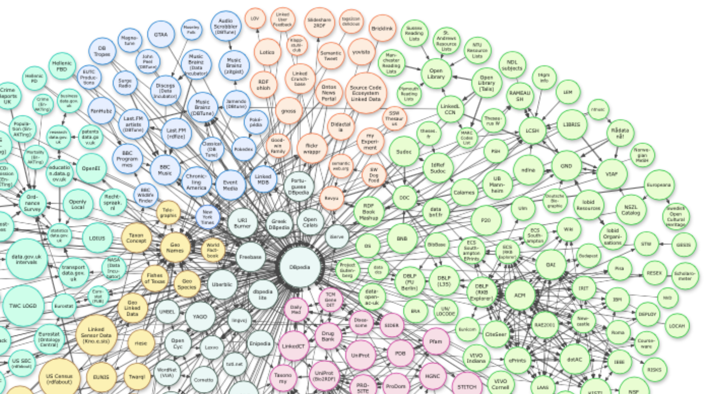
  - Handle diversity with **schema-less** Model

  - Graph-structured data model (RDF)

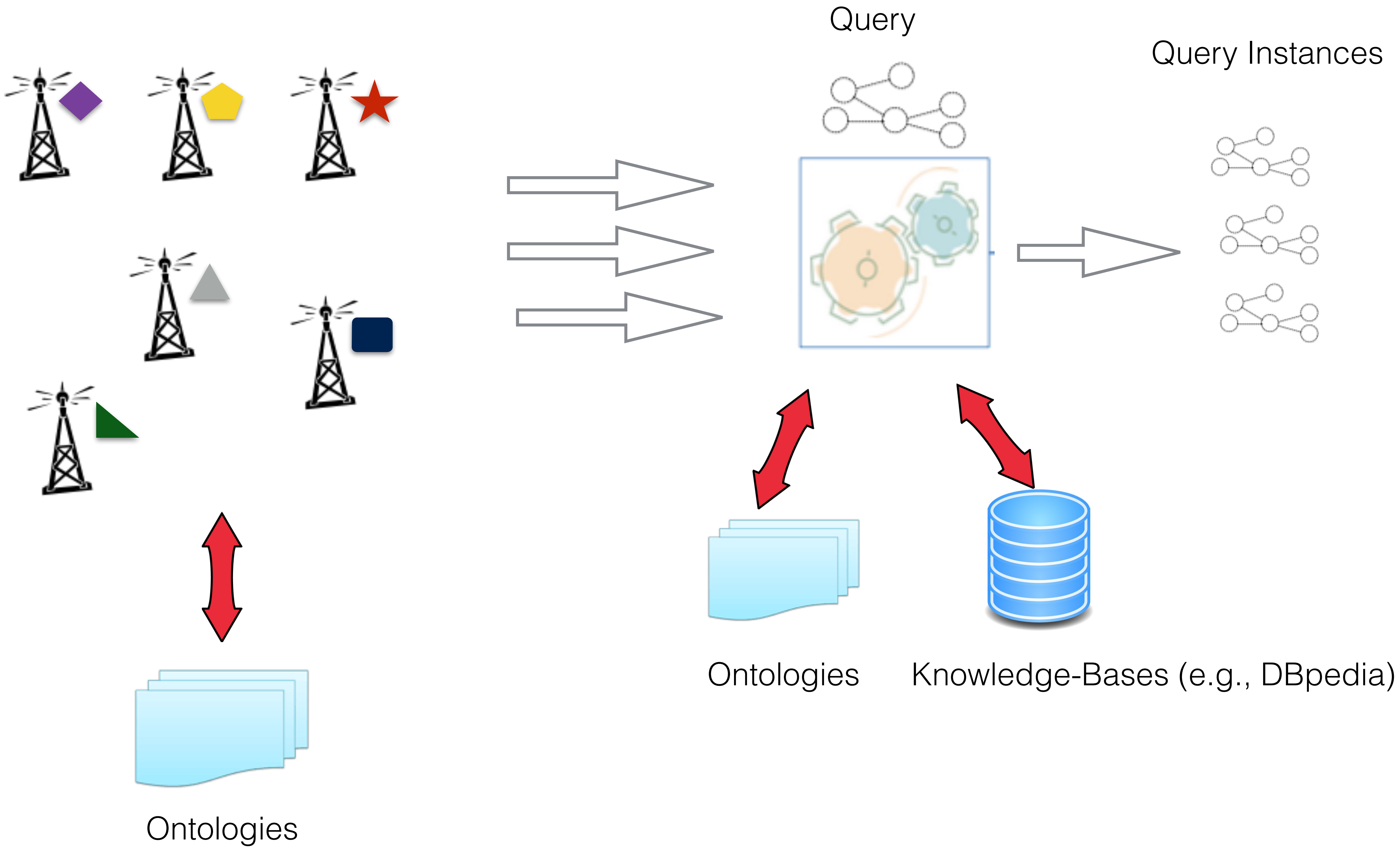
- Stream Reasoning

  - Performing materialisation of implicit knowledge (e.g., inference using ontologies)

  - Utilising static knowledge-bases (Linked-data Cloud) to extract contextual knowledge



[RDF Stream Processing]



Query

Query Instances

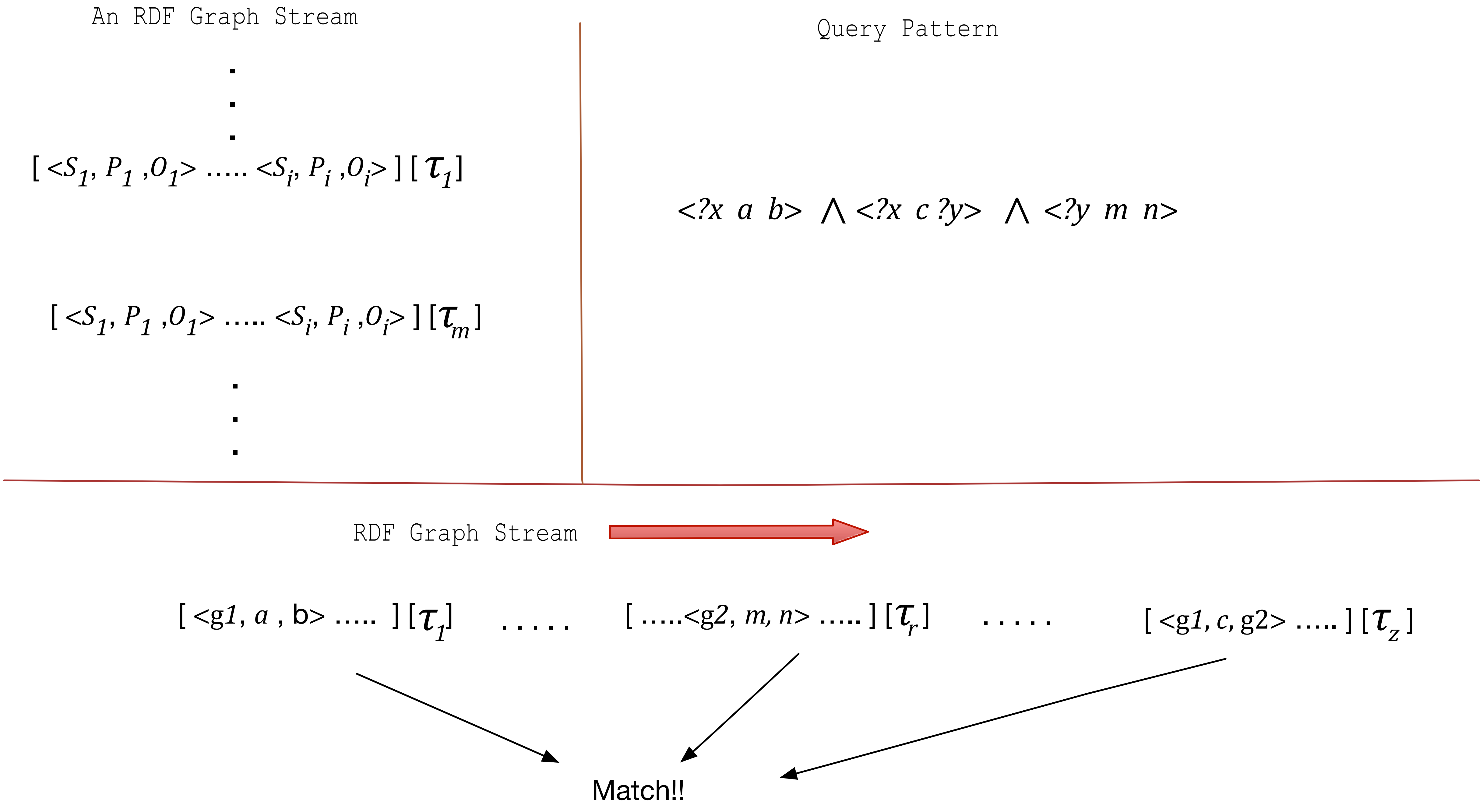
Ontologies

Knowledge-Bases (e.g., DBpedia)

Ontologies

[RDF Stream Processing]

- RDF Stream Processing is expensive
  - Graph Pattern Matching, an NP-complete problem



An RDF Graph Stream

⋮

$[\langle S_1, P_1, O_1\rangle \ldots\ldots \langle S_i, P_i, O_i\rangle]\,[\tau_1]$

$[\langle S_1, P_1, O_1\rangle \ldots\ldots \langle S_i, P_i, O_i\rangle]\,[\tau_m]$

⋮

Query Pattern

$\langle ?x\ a\ b\rangle \wedge \langle ?x\ c\ ?y\rangle \wedge \langle ?y\ m\ n\rangle$

RDF Graph Stream

$[\langle g1, a, \mathrm{b}\rangle \ldots\ldots]\,[\tau_1]$ . . . . . $[\ldots\ldots\langle g2, m, n\rangle \ldots\ldots]\,[\tau_r]$ . . . . . $[\langle g1, c, g2\rangle \ldots\ldots]\,[\tau_z]$

Match!!

[RDF Stream Processing]

- Existing Solutions

  - Rely-on centralised processing
  - Triple stream model, which ignores the graph nature of RDF
  - Extended for DSMS models, black-box approach
  - Re-evalution based query processing
  - No support to gather archives of streams

- Vision/Expectations (DIONYUSIS)

  - Scale-out solution
  - Handle the distributed nature of stream sources
  - Incremental indexing and incremental query processing
  - State-full operators to enable Semantic Complex Event Processing

[Data Distribution]

- Can we learn anything from static solutions

  - Hash-based clustering [Zeng et.al 2013], graph structure of RDF data?
  - Semantic Hash-based clustering [Lee et.al 2013], multivalued predicates?

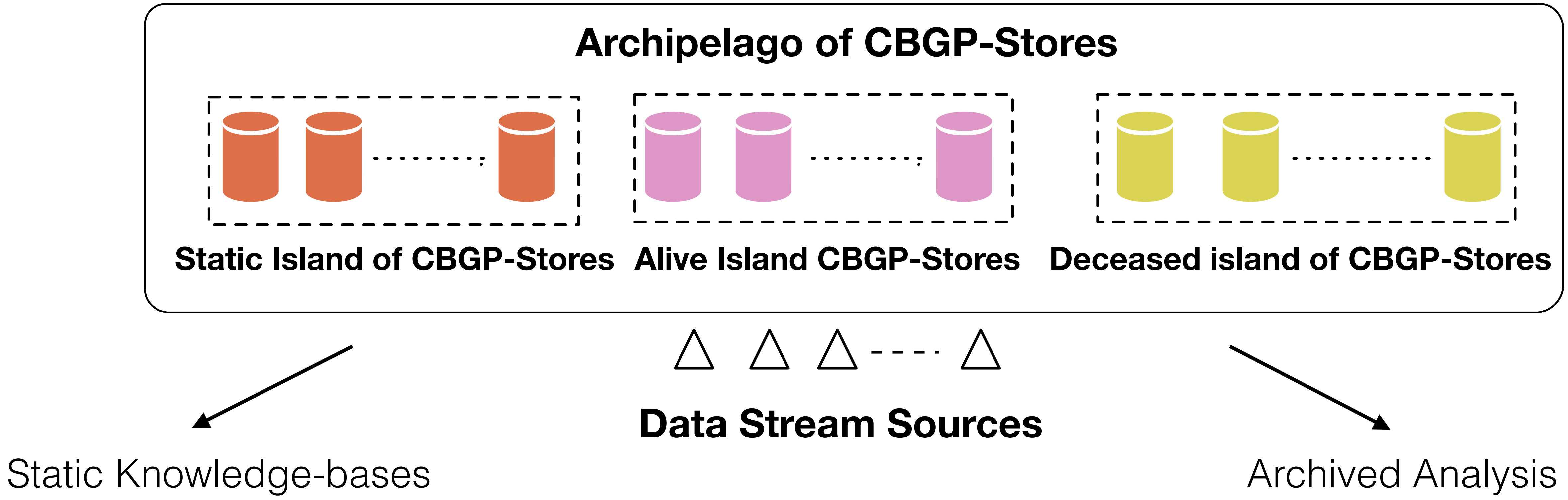
- **Distribution for RDF data is not trivial and requires extensive preprocessing**

- Reverse paradigm for stream data distribution

  - Data is not known in advance for distribution analysis
  - New sources are added dynamically and old sources provide data at variable velocities
  - Variable query loads

[Data Distribution for RDF Graph Streams]

- Relying on ontologies and use cases
  - Ontology modularisation [Aquin et al, 2011] [Bhatt et al, 2012] [Gillani et al, 2016]
  - Given a set of ontologies *O* defined on a set of streams *S* , produce a set of common basic graph patterns (CBGPs)
  - Each CBGP contains information about a coherent subtopic within an ontology



Archipelago of CBGP-Stores

Static Island of CBGP-Stores

Alive Island CBGP-Stores

Deceased island of CBGP-Stores

Data Stream Sources

Static Knowledge-bases

Archived Analysis

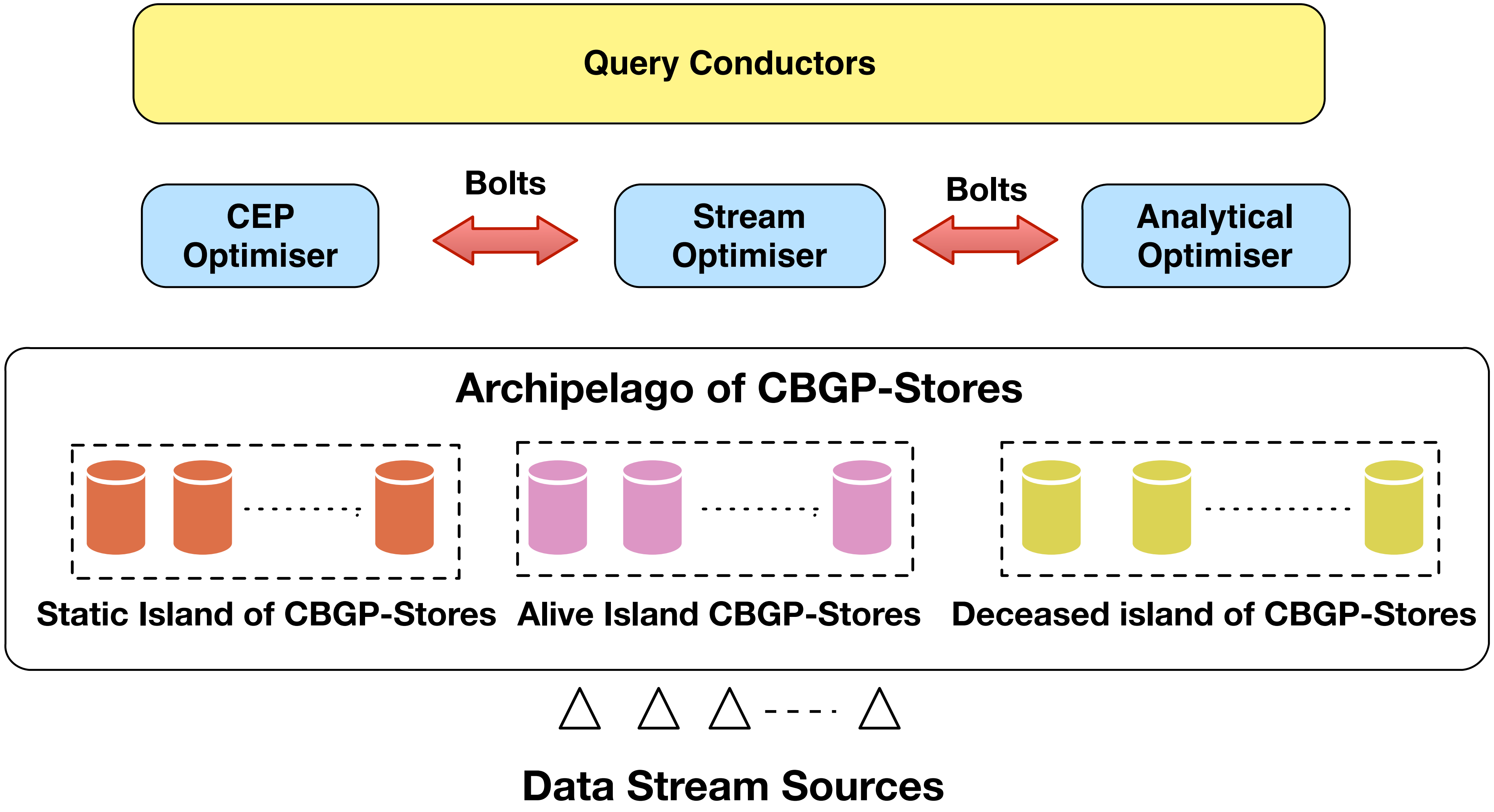
[CBGP Stores and Query Processing]

- CBGP Stores
  - Aggregating common linked-concepts within a single CBGP store
  - Each CBGP stores has customised optimisation, light-weight adaptive indexing
  - Reducing the network traffic and load at federation level
  - Acts as data filter, only relevant data is stored from a set of sources
  - Divided into three flavours: static, alive and deceased

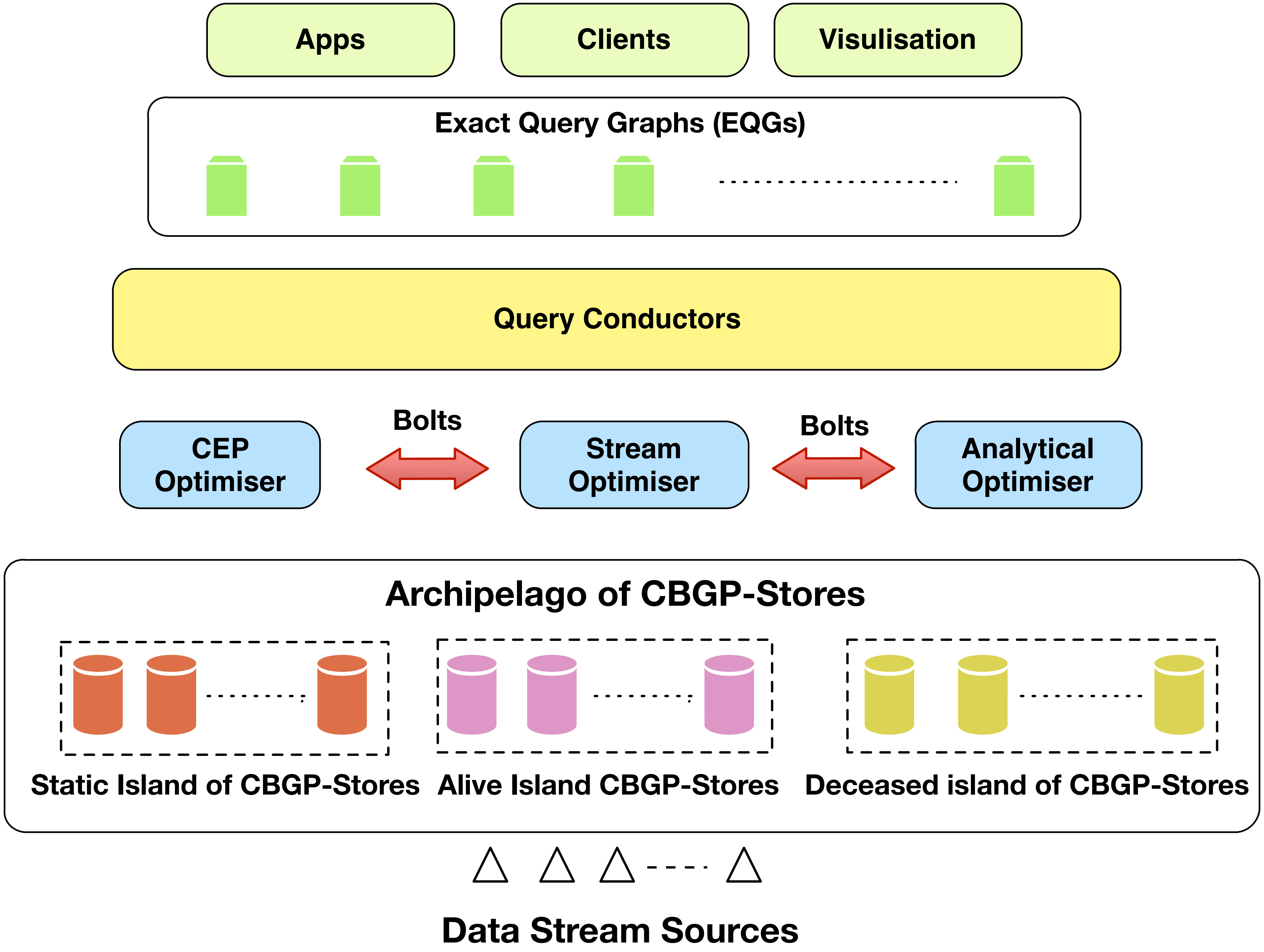
- A collection of CBGP stores is abstracted under an Island, each island is assigned to a set of **query-conductors**

[Query Processing via Query Conductors]

- Query Conductors
  - Determines the type of the registered query graphs: analytical query over archived data, streaming query, sequence-based query (CEP)
  - Divide the query graph into a set of subgraph queries, share loads, and orchestrates the query execution

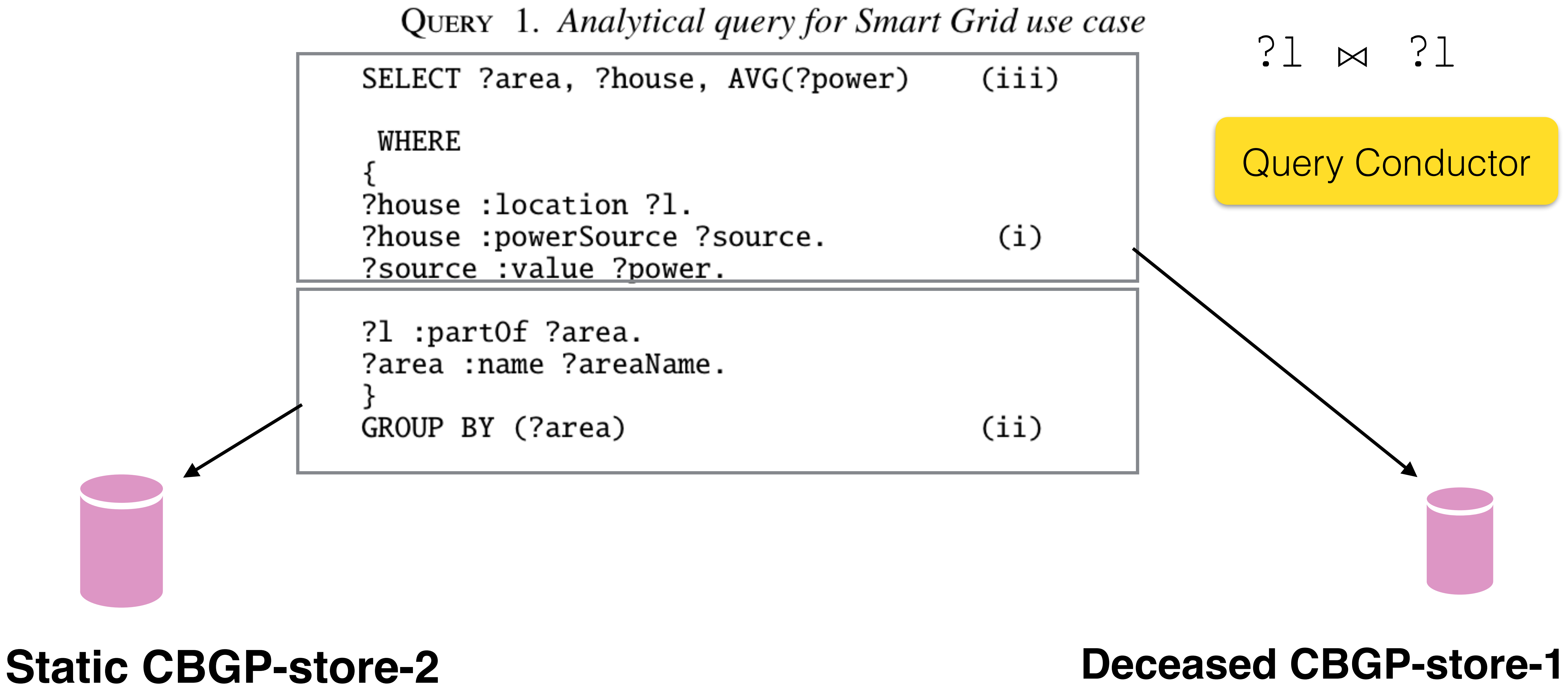


[Functional Layers of DIONYUSIS]



[Query Optimisations for DIONYUSIS]

- Analytical Queries: Traditionally
  - Compute each pattern against all the available data stores and the results are joined at the server
  - Evaluating each pattern in nested-loop-join fashion: substituting the results from one pattern to another



QUERY 1. *Analytical query for Smart Grid use case*

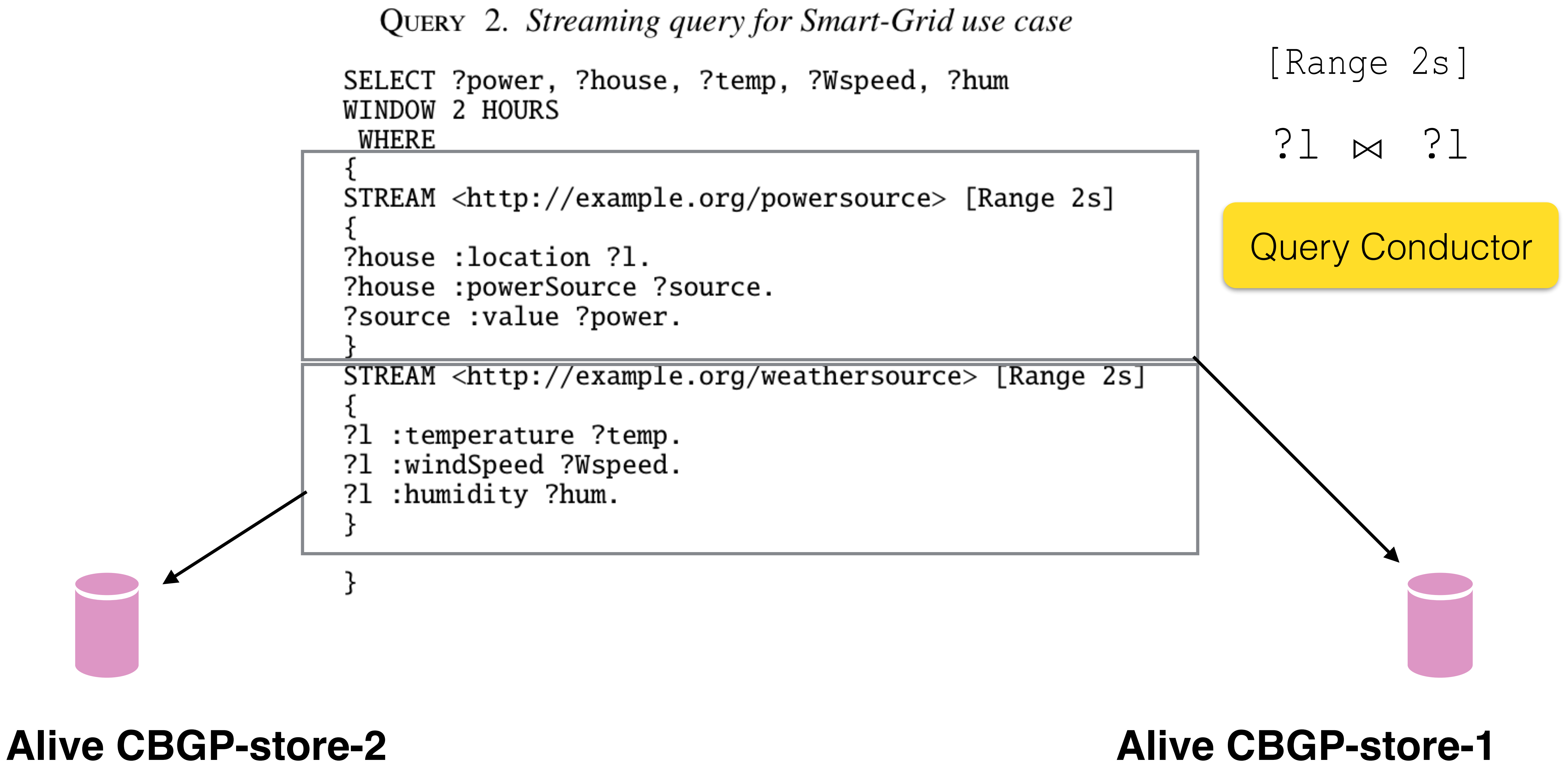
```
SELECT ?area, ?house, AVG(?power)     (iii)

 WHERE
{
?house :location ?l.
?house :powerSource ?source.            (i)
?source :value ?power.

?l :partOf ?area.
?area :name ?areaName.
}
GROUP BY (?area)                        (ii)
```

[Query Optimisations for DIONYUSIS]

- Streaming Queries
  - Query is distributed into set of subquery graphs each is hosted by a CBGP store
  - The matches are computed locally and window operators is executed at query conductor level



Query 2. *Streaming query for Smart-Grid use case*

```
SELECT ?power, ?house, ?temp, ?Wspeed, ?hum
WINDOW 2 HOURS
 WHERE
{
STREAM <http://example.org/powersource> [Range 2s]
{
?house :location ?l.
?house :powerSource ?source.
?source :value ?power.
}
STREAM <http://example.org/weathersource> [Range 2s]
{
?l :temperature ?temp.
?l :windSpeed ?Wspeed.
?l :humidity ?hum.
}

}
```

[Range 2s]

$?l \bowtie ?l$

Query Conductor

Alive CBGP-store-2

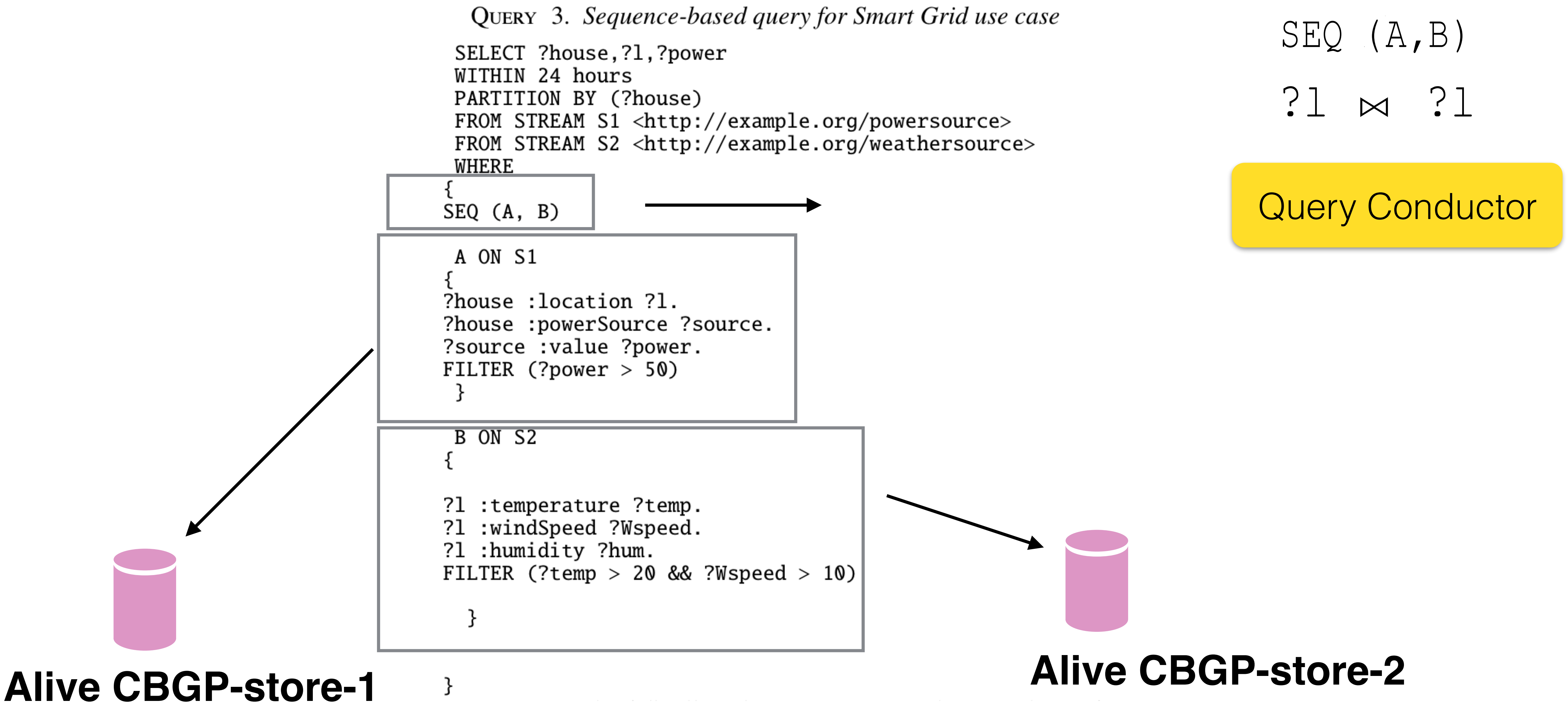
Alive CBGP-store-1

[Query Optimisations for DIONYUSIS]

- Sequence-based Query

  - Query is distributed into set of subquery graphs each is hosted by a CBGP store

  - The matches are computed locally and window operators is executed at query conductor level



QUERY 3. *Sequence-based query for Smart Grid use case*

```
 SELECT ?house,?l,?power
 WITHIN 24 hours
 PARTITION BY (?house)
 FROM STREAM S1 <http://example.org/powersource>
 FROM STREAM S2 <http://example.org/weathersource>
 WHERE
{
SEQ (A, B)
 A ON S1
{
?house :location ?l.
?house :powerSource ?source.
?source :value ?power.
FILTER (?power > 50)
 }
 B ON S2
{

?l :temperature ?temp.
?l :windSpeed ?Wspeed.
?l :humidity ?hum.
FILTER (?temp > 20 && ?Wspeed > 10)

  }

}
```

[Conclusion]

- **Addressing the requirements of RDF graph streams:**
  - Scalability, state management, distribution of data sources

- **One query interface to support:**
  - Continuous and distributed streaming queries
  - Queries over archived streams
  - Temporal sequential queries

- **Future Work:**
  - Integration of separate layers of the system.
  - Benchmarking distributed RDF graph streams

[Questions?]