LEIBNIZ-INFORMATIONSZENTRUM
TECHNIK UND NATURWISSENSCHAFTEN
UNIVERSITÄTSBIBLIOTHEK

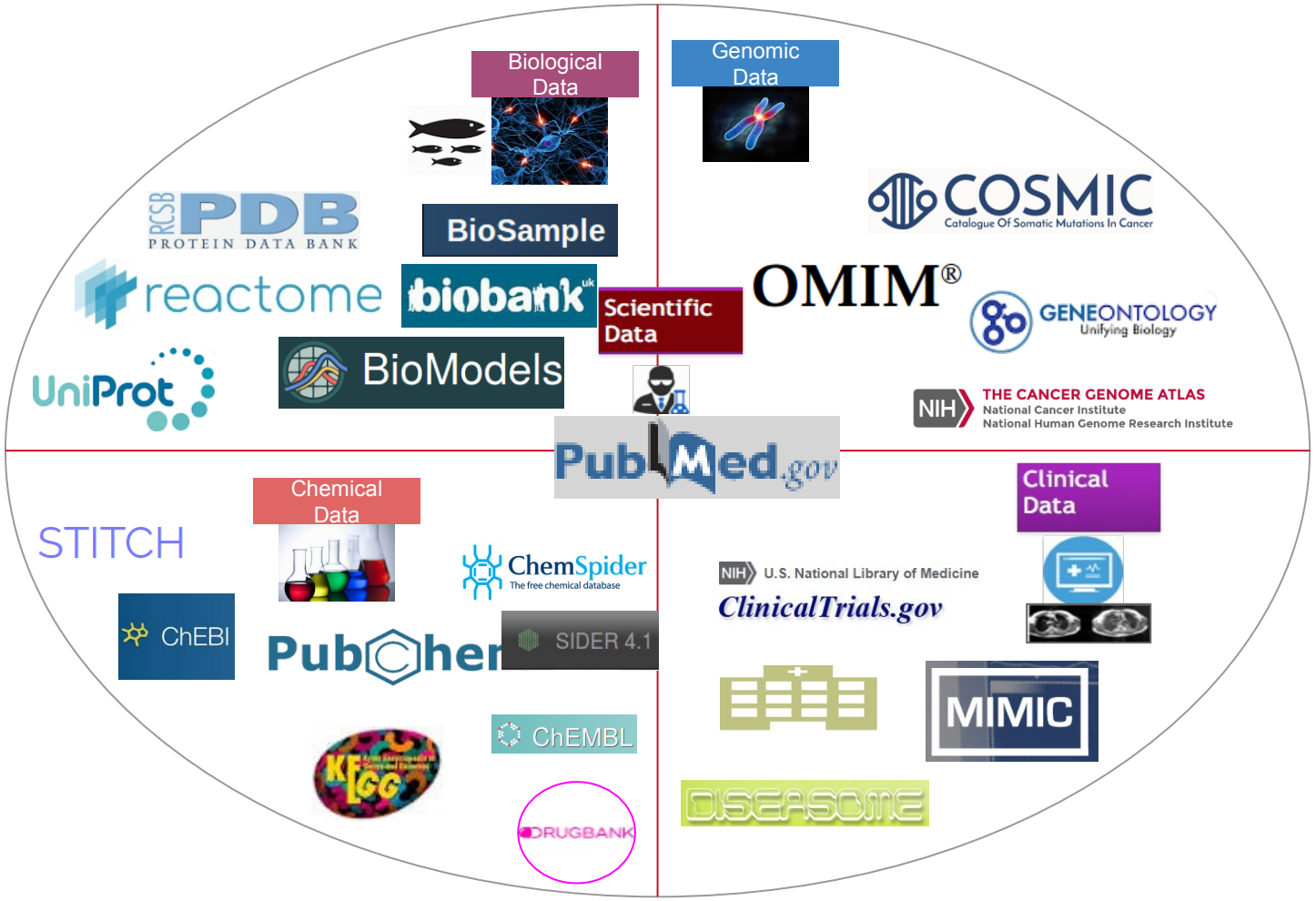
TIB

Federated Query Processing over RDF Graphs

Maria-Esther Vidal
Scientific Data Management Group TIB Leibniz Information Centre for Science and Technology University Library & L3S Research Centre Leibniz University of Hannover, Germany
Universidad Simón Bolívar, Venezuela

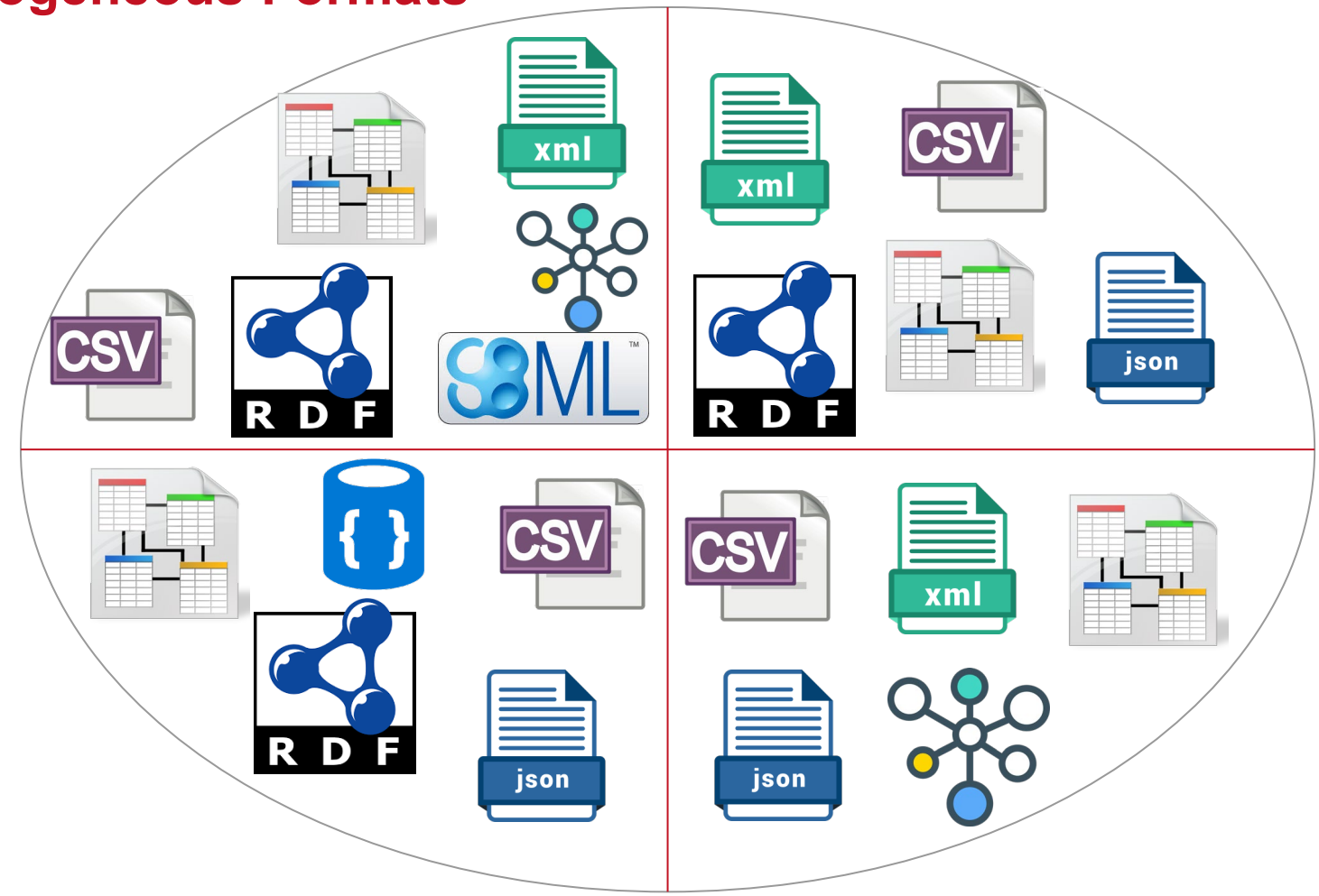
Leibniz Association

Motivating Example- Available Data Sources



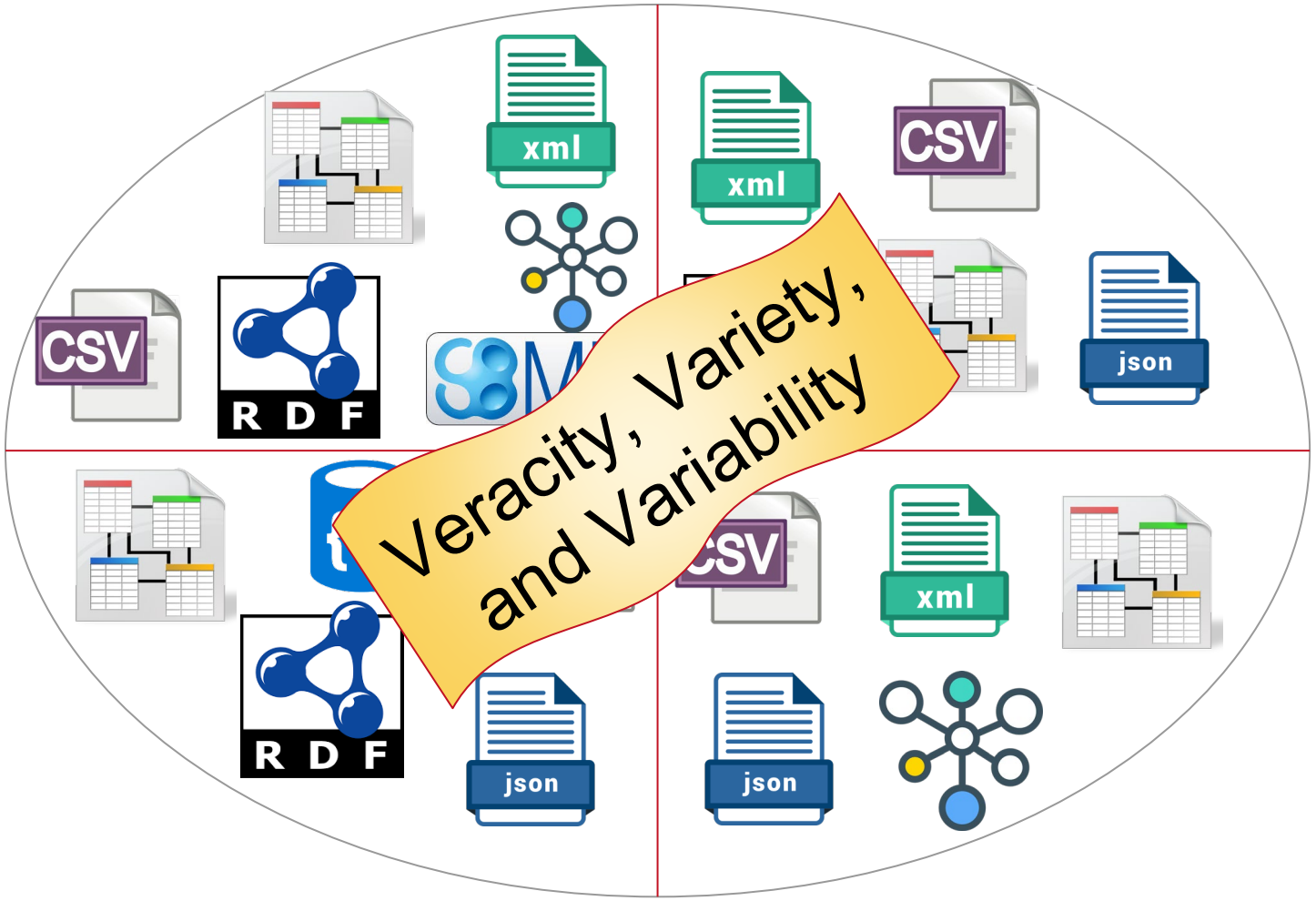
Diverse data sources potentially incomplete and noisy

Motivating Example- Data Sources in Heterogeneous Formats



Data sources is diverse formats, e.g., XML, CSV, JSON

Impacting Data Complexity Dimensions



Motivating Example

Query: Drugs with the active substance ***Simvastatin:***

- Name of possible drug targets,
- Chemical formula of a drug,
- Side effects, and
- Disease Name

```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
        ?drug       dailymed:activeIngredient          dailymed:Simvastatin .
        ?drug       dailymed:genericDrug               ?dbdrug .
        ?drug       dailymed:possibleDiseaseTarget     ?disease .
        ?drug       owl:sameAs                         ?sadrug .
        ?disease    rdfs:label                         ?disName
        ?sadrug     sider:sideEffect                   ?seffect .
        ?seffect    sider:sideEffectName               ?sename .
        ?dbdrug     drugbank:chemicalFormula           ?drugformula
}
```

Interoperability Issues During Query Processing

RDF DAILYMED

```
dailymed:798   rdf:type                        dailymed:drugs ;
               dailymed:activeIngredient       dming:Simvastatin .
               owl:sameAs                      sider:54454 .
               dailymed:genericDrug            drugbank:DB00641 ;
                    dailymed:possibleDiseaseTarget  diseasome:319,
                                                    diseasome:2839,
                                                    diseasome:2175 .
```

JSON DISEASOME

```
[{
  "diseaseID": "319",
  "name": "Diabetes_mellitus",
  "associatedGene": ["ACE", "ABCC8", "TCF1"]
  },{
  "diseaseID": "2839",
  "name": "Kaposi sarcoma, susceptibility to, 148000",
  "associatedGene": ["IL6", "IFNB2", "BSF2"]
}]
```

SQL DRUGBANK

Drug

accNum	DrugName	formula	pubChemId
DB00641	simvastatin	$C_{25}H_{38}O_5$	54454
DB00295	Morphine	$C_{17}H_{19}NO_3$	5288826

Drug_Target

Drug	Target
DB00641	631
DB00641	1882
DB00295	7683

Target

ID	Name	Gene	UniprotID
631	3-hydroxy-3-methylglutaryl-coenzyme A reductase	HMGCR	P04035
1882	Ras-related C3 botulinum toxin substrate 1	RAC1	P63000
7683	Mu-type opioid receptor	OPRM1	P35372

CSV SIDER 4.1

side_effects.csv

DrugID,UMLS_ID,SideEffectName
54454,C0009806,Constipation
54454,C0236071,Throat tightness
54454,C0156404,Menstruation irregular
191,C0012833,Dizziness
191,C0232487, Abdominal discomfort
191,C1956346,Coronary artery disease

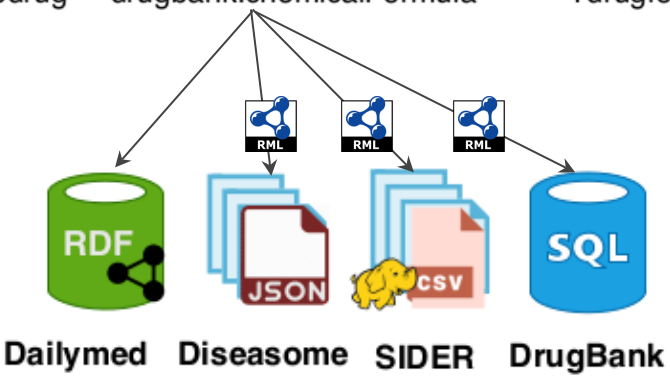
drug_names.csv

ID,DrugName
54454,simvastatin
191,adenosine

Query Over Heterogeneous Data Sources

- Query: Drugs with the active substance ***Simvastatin***:
  - Name of possible drug targets,
  - Chemical formula of a drug,
  - Side effects, and
  - Disease Name

```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
        ?drug       dailymed:activeIngredient          dailymed:Simvastatin .
        ?drug       dailymed:genericDrug               ?dbdrug .
        ?drug       dailymed:possibleDiseaseTarget     ?disease .
        ?drug       owl:sameAs                         ?sadrug .
        ?disease    rdfs:label                         ?disName
        ?sadrug     sider:sideEffect                   ?seffect .
        ?seffect    sider:sideEffectName               ?sename .
        ?dbdrug     drugbank:chemicalFormula           ?drugformula
}
```



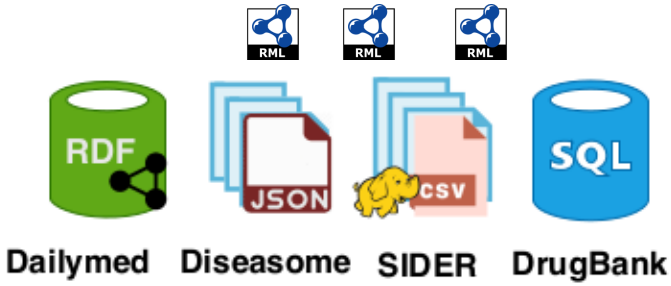
- Select the data sources required to execute a query, and
- Rewrite the query in terms of the selected data sources

Agenda

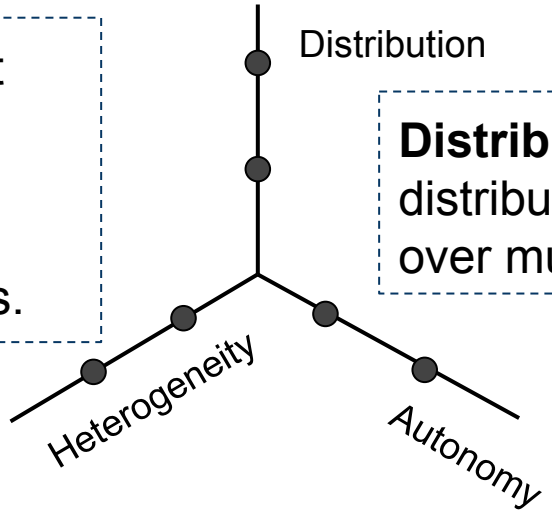
1. Distributed Data Management Systems
2. Data Integration Systems
3. Adaptive SPARQL Query Engines
4. Hybrid SPARQL Query Engines

Leibniz Association

Dimensions of Distributed Database Systems [*]



Heterogeneity: Different forms ranging from hardware, differences in network protocols, data models, query languages.



Distribution: Physical distribution of the data over multiple sites.

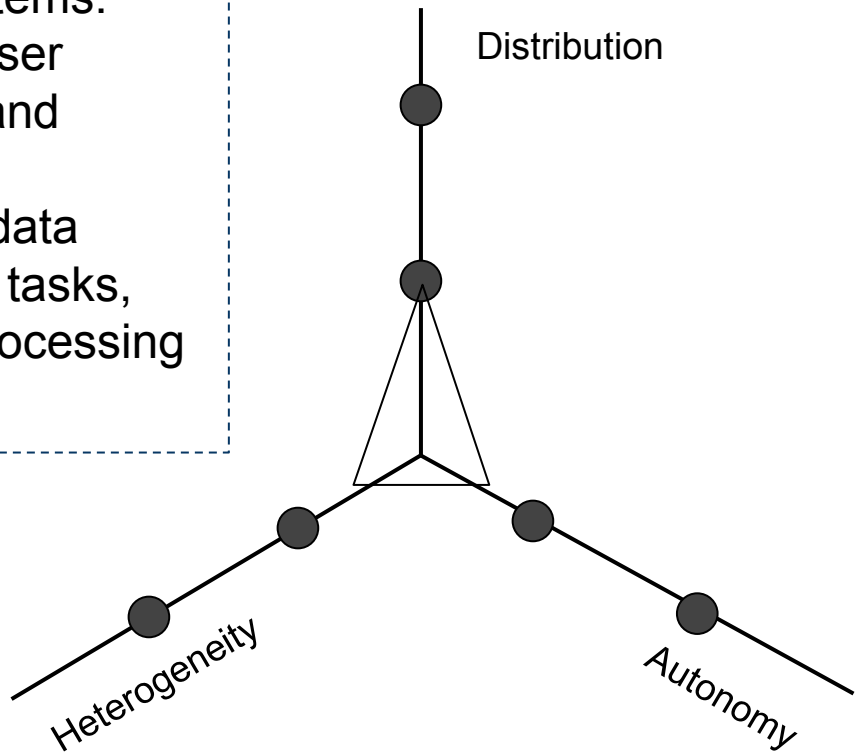
Autonomy: Distribution of the control; degree to which individual systems can operate independently.

[*] Tamer Ozsu and Patrick Valduriez. Principles of Distributed Database Systems (Third Edition). Springer, 2011.

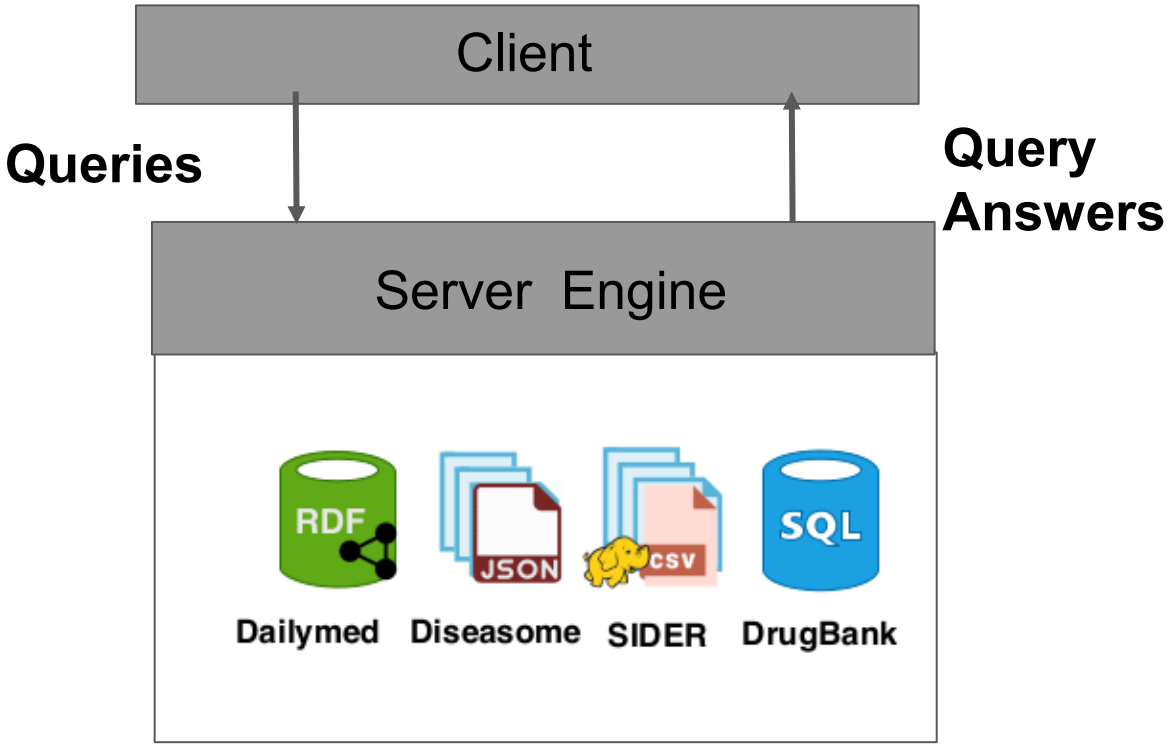
Client-Server Systems

Client-Server Systems:

- **Clients** run user applications and interfaces.
- **Servers** run data management tasks, e.g., query processing and storage.



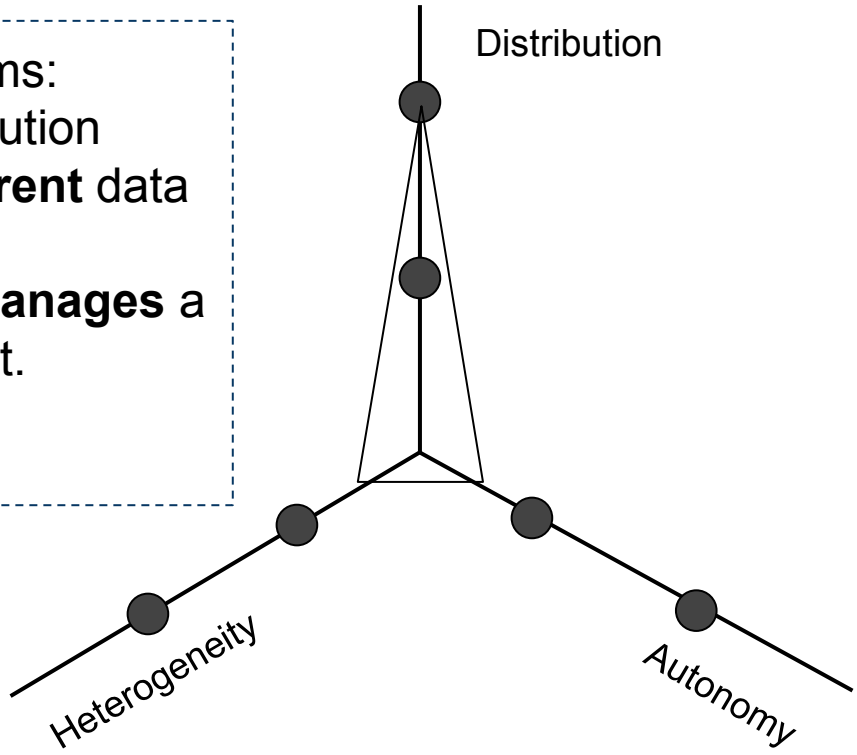
Client-Server Systems



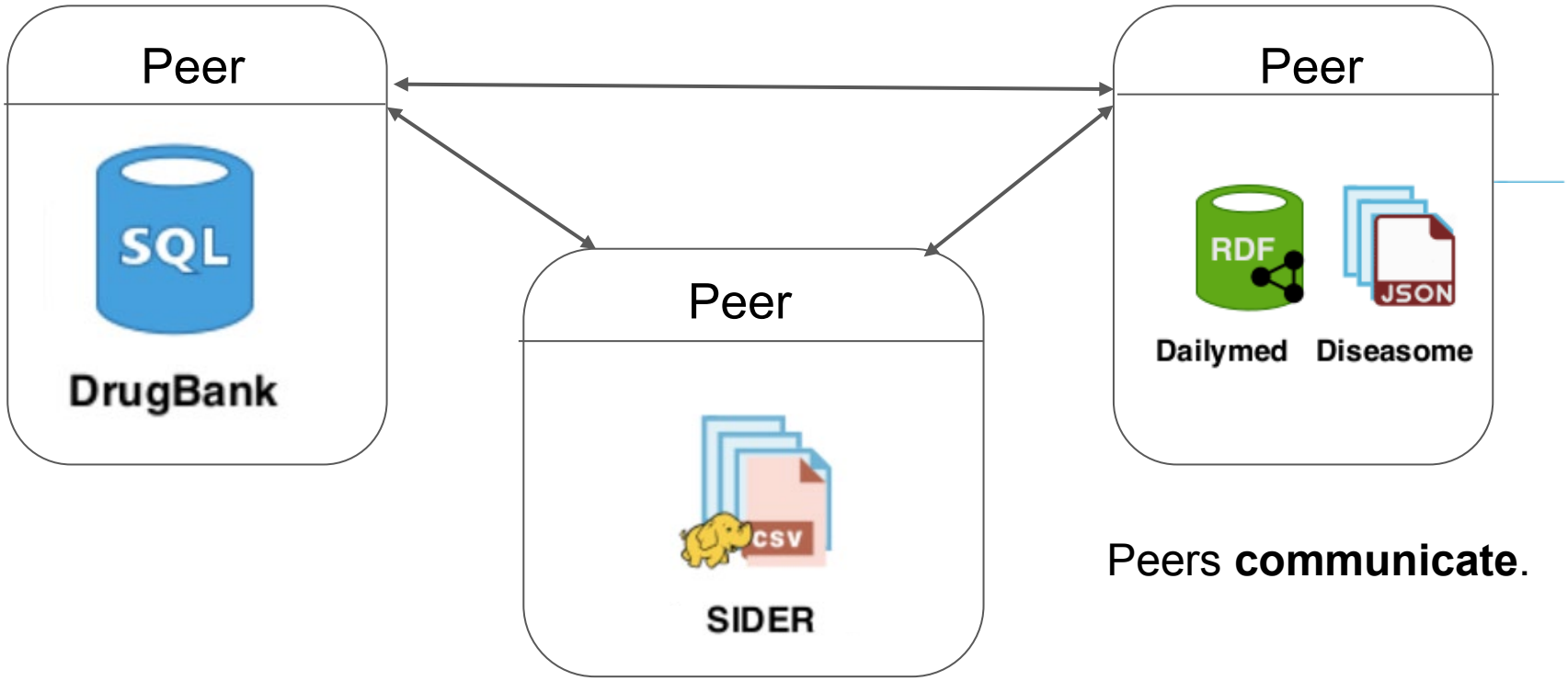
Peer-to-Peer Systems

Peer-to-Peer Systems:

- **Massive** distribution
- May used **different** data models.
- Each system **manages** a different dataset.
- Peers can **communicate**.



Peer-to-Peer Systems

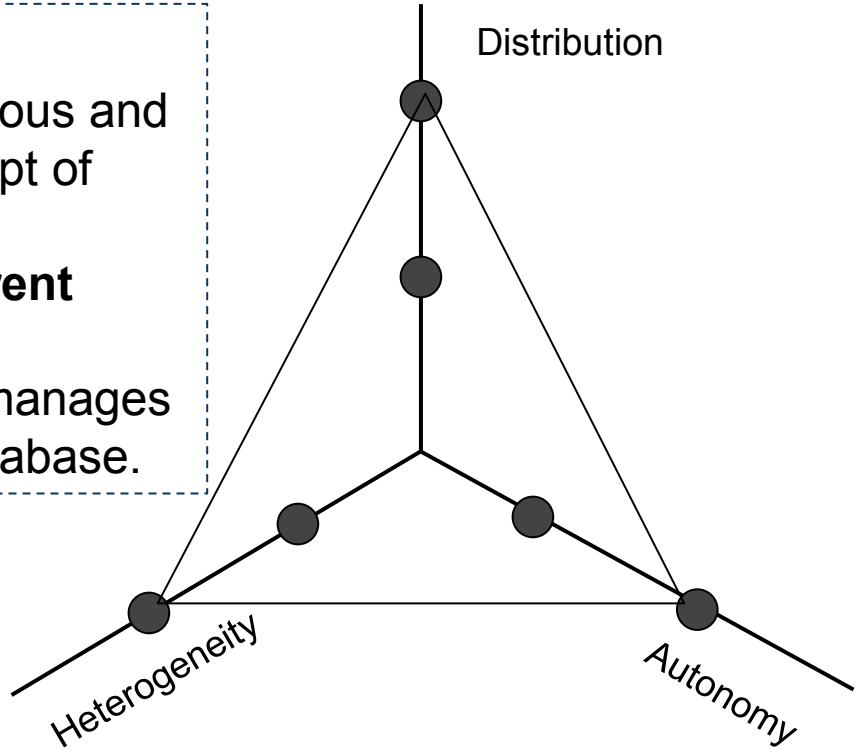


Peers **communicate**.

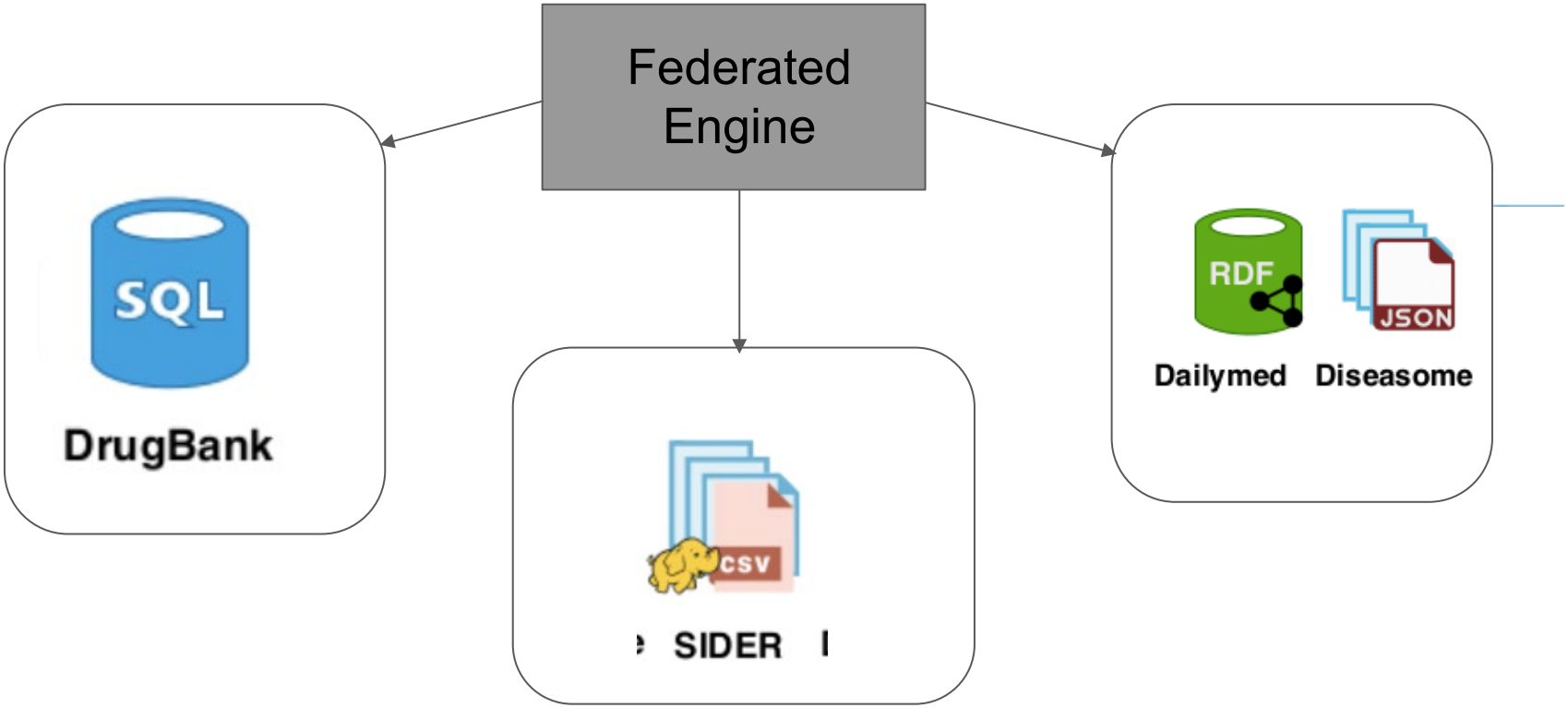
Federated Query Systems

FDM Systems:

- **Fully** autonomous and have no concept of cooperation.
- May use **different** data models.
- Each system manages a **different** database.



Federated Query Systems



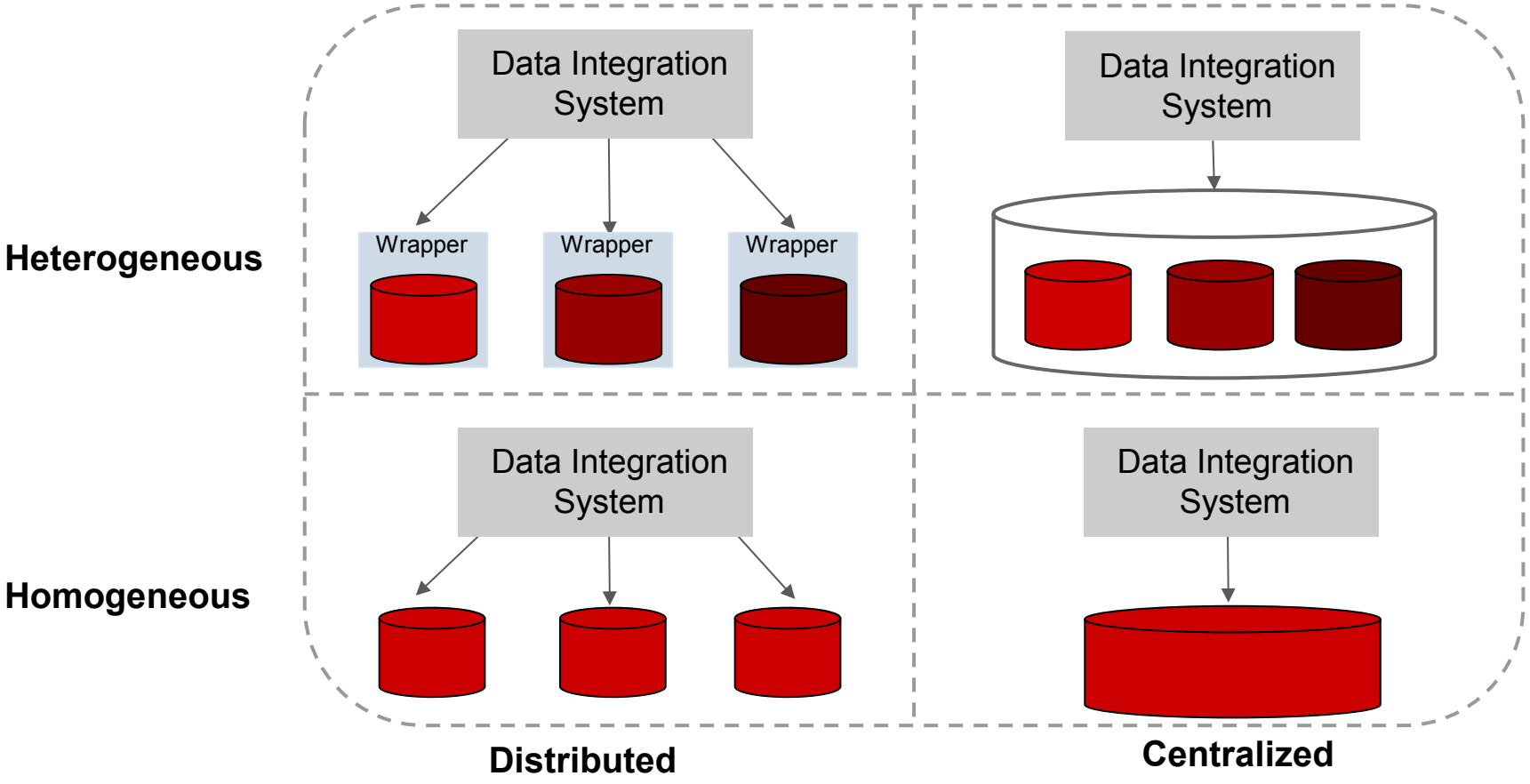
Data Integration Systems

A data integration system **DIS**=<**O**,**S**,**M**>:

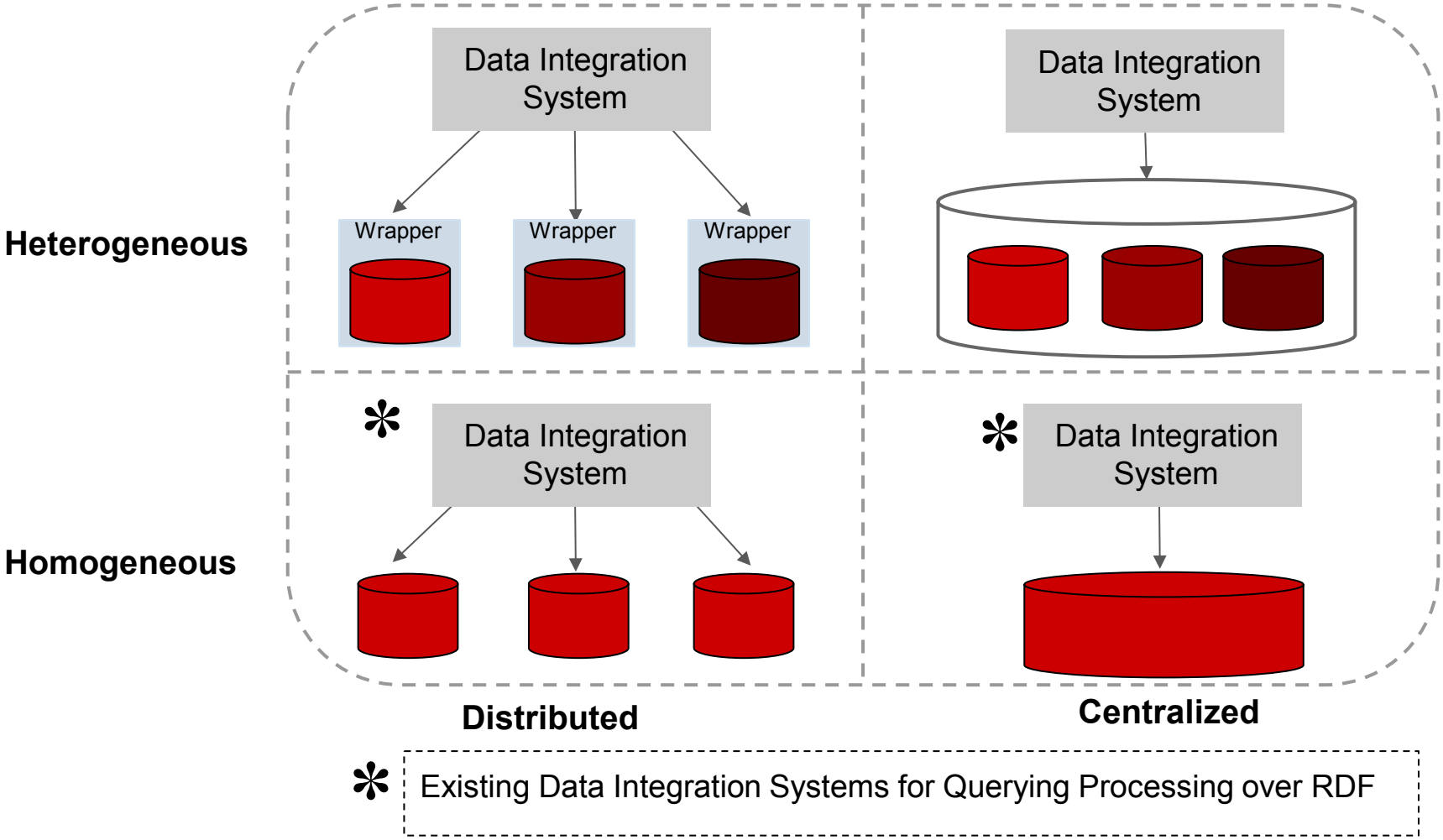
- **O** is a set of general concepts in a general schema (virtual)
- **S** is a set of {**S1**,..,**Sn**} of data sources
- **M** is a set of mappings between sources in **S** and general concepts in **O**

cf. Lenzerini 2002

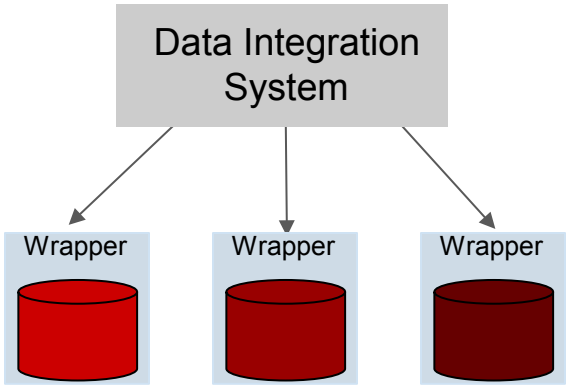
Data Integration Systems



Data Integration Systems



Query Rewriting Problem



Query Rewriting Problem (QRP):

- A query **Q** is a conjunctive query over predicates in **O**
- Find a conjunctive query **Q'** expressed in sources in **S** based on rules in **M**, such that
  - Evaluation of **Q'** produces only answers of **Q**
  - Evaluation of **Q'** produces all the answers of **Q** given the sources in **S**

Theorem [Levy et al. 1995]
To check if there is a valid rewriting **Q'** of **Q** with at most the same number of goals as **Q** is an **NP-complete problem.**

Challenges for Query Processing

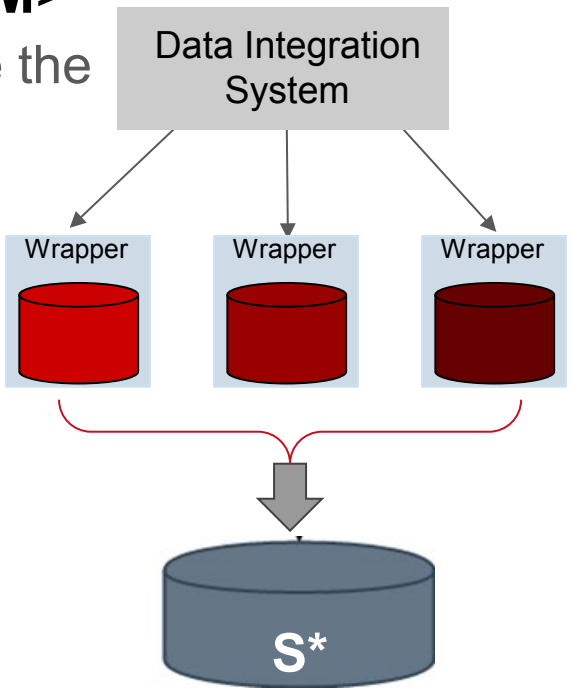
Given a query Q in a formal language, i.e., SPARQL

- **Identify** the **relevant** data sources for Q (**Source Selection)**
- **Decompose** Q into subqueries on **relevant** data sources (**Query Decomposition**)
- **Plan** evaluation of **subqueries** against **relevant** data sources (**Query Planning**)
- **Merge** data collected from **relevant** data sources (**Query Execution**)

Relevant data sources for **Q**: **minimal set** of sources **S** from a federation of source **F** such that the answer of evaluating **Q** in **S is the same than** evaluating **Q** in **F**

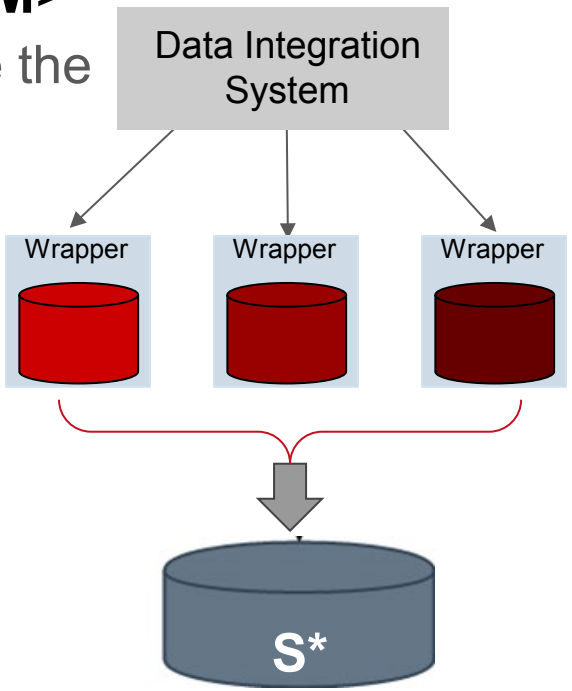
Federated Query Processing Problem

- Given a Data Integration System **DIS=<O,S,M>** and a **query**, Q, expressed over **O.** Let S* be the virtual dataset of S

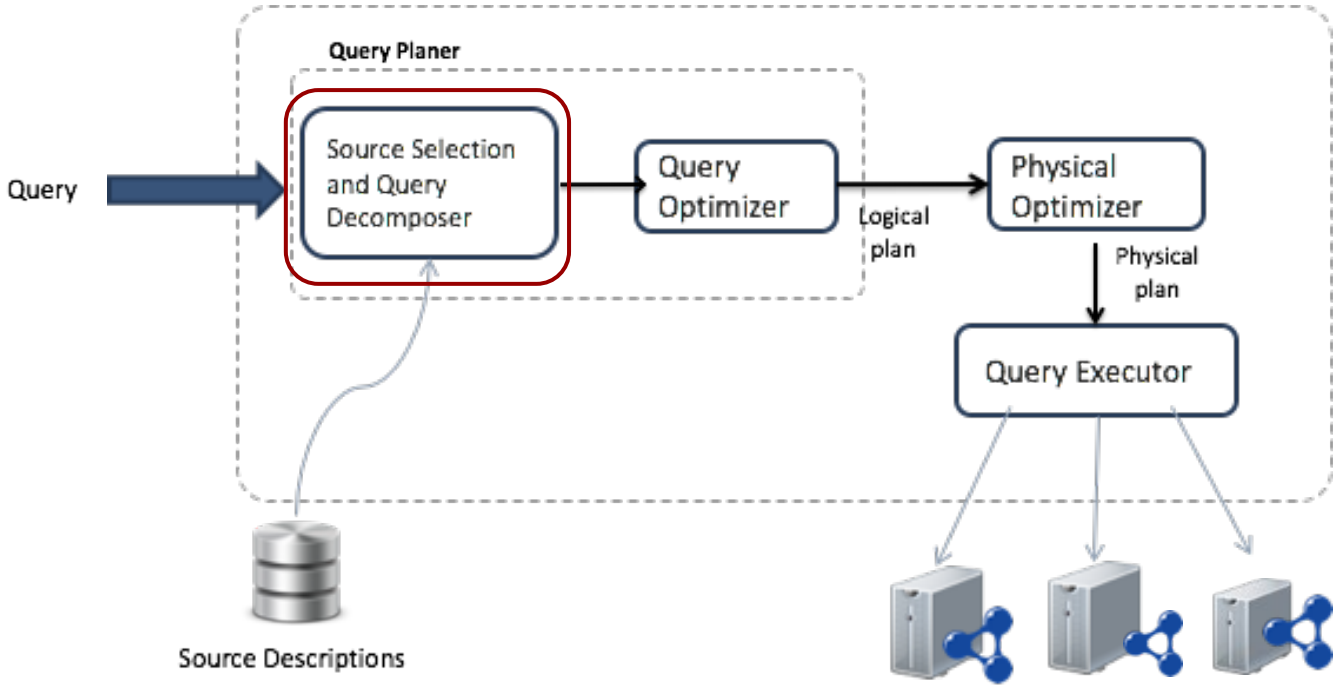


Federated Query Processing Problem

- Given a Data Integration System **DIS=<O,S,M>** and a **query**, Q, expressed over **O.** Let S* be the virtual dataset of S
- Find a query rewriting, **Q'** over **S**, that:
  - **Maximize** answer completeness,
    $[[Q]]_{S^*} = \text{argmax}_{Q' \in RW(Q)} [[Q']]_S$
  - **Minimize** execution time,
    cost = argmin$_{Q' \in RW(Q)}$ cost(Q')



Federated Engine Architecture



Our Running Example

Query: Drugs with the active substance ***Simvastatin:***

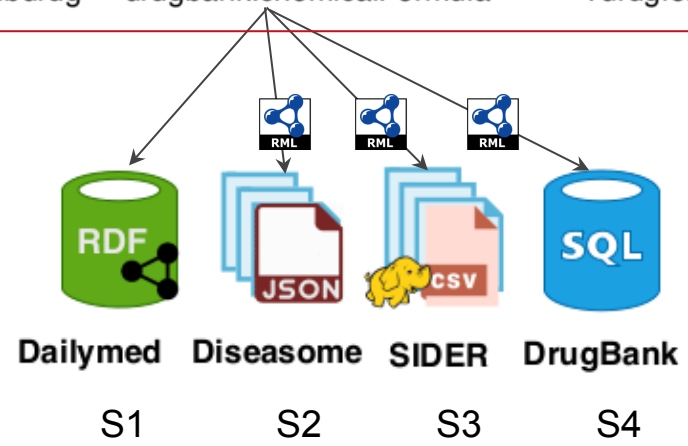
- Name of possible drug targets,
- Chemical formula of a drug,
- Side effects, and
- Disease Name

```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
        ?drug       dailymed:activeIngredient          dailymed:Simvastatin .
        ?drug       dailymed:genericDrug               ?dbdrug .
        ?drug       dailymed:possibleDiseaseTarget     ?disease .
        ?drug       owl:sameAs                         ?sadrug .
        ?disease    rdfs:label                         ?disName
        ?sadrug     sider:sideEffect                   ?seffect .
        ?seffect    sider:sideEffectName               ?sename .
        ?dbdrug     drugbank:chemicalFormula           ?drugformula
}
```

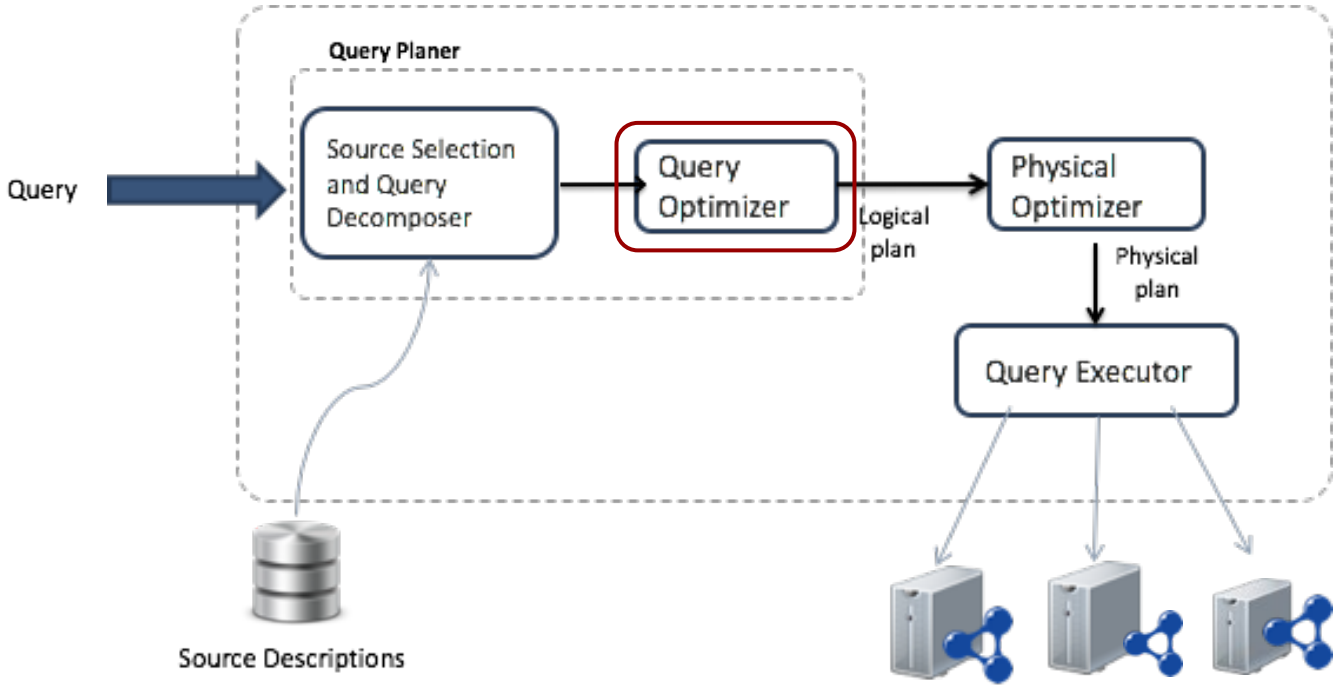
Source Selection & Decomposition

- Query: Drugs with the active substance ***Simvastatin***:
  - Name of possible drug targets,
  - Chemical formula of a drug,
  - Side effects, and
  - Disease Name

```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
S1  t1  ?drug     dailymed:activeIngredient      dailymed:Simvastatin .
    t2  ?drug     dailymed:genericDrug           ?dbdrug .
    t3  ?drug     dailymed:possibleDiseaseTarget ?disease .
    t4  ?drug     owl:sameAs                     ?sadrug .
S2  t5  ?disease  rdfs:label                     ?disName
S3  t6  ?sadrug   sider:sideEffect               ?seffect .
    t7  ?seffect  sider:sideEffectName           ?sename .
S4  t8  ?dbdrug   drugbank:chemicalFormula       ?drugformula
}
```

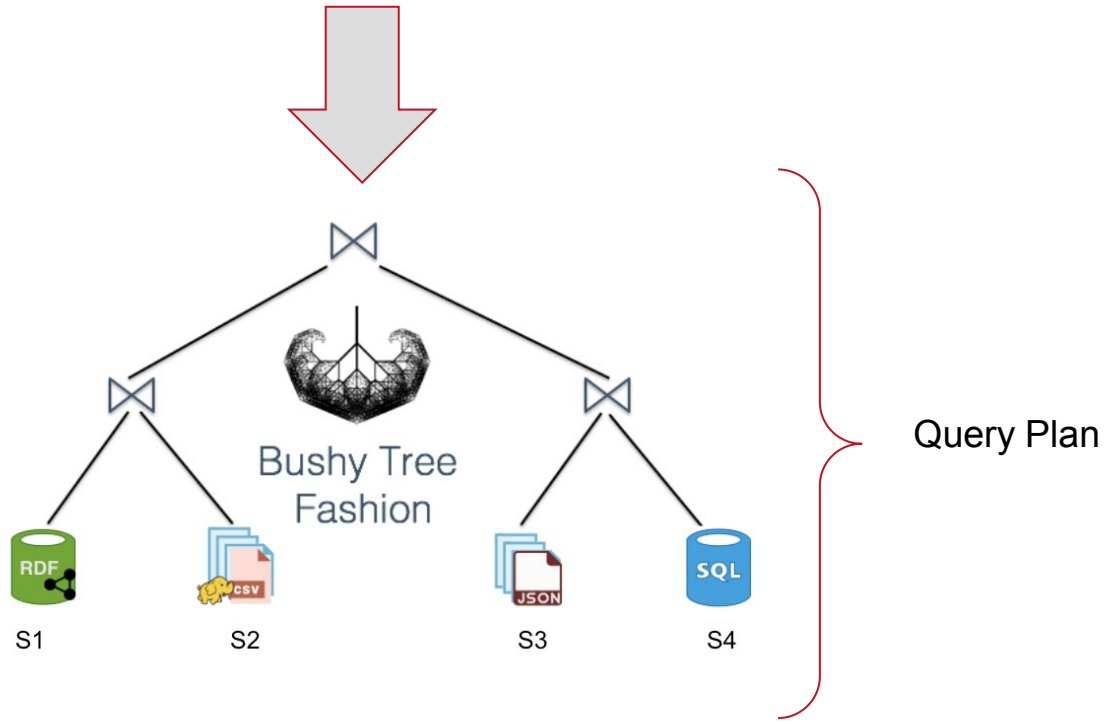


Federated Engine Architecture

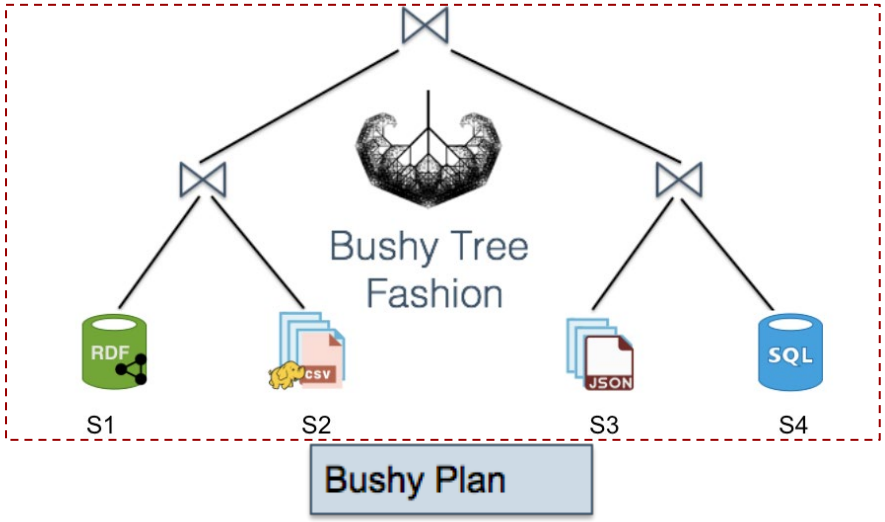


Query Planning Over Heterogeneous Data Sources

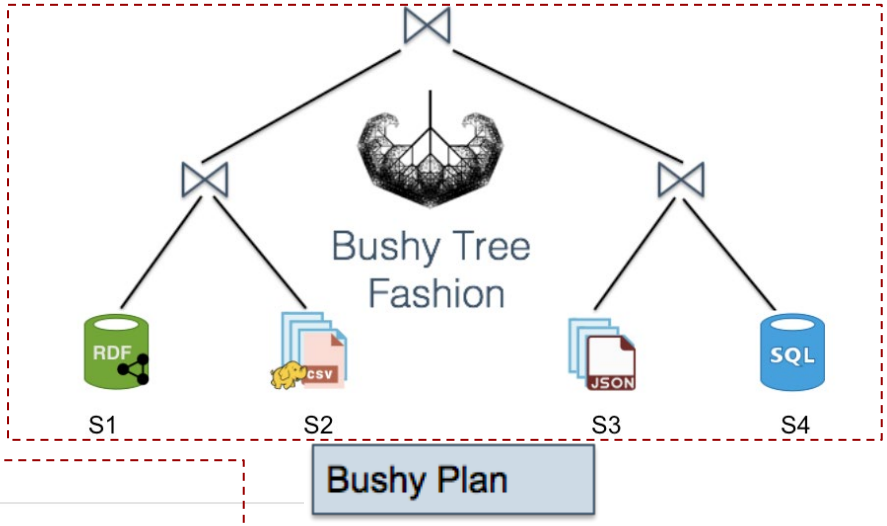
```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
  t1  ?drug      dailymed:activeIngredient        dailymed:Simvastatin .
  t2  ?drug      dailymed:genericDrug             ?dbdrug .
  t3  ?drug      dailymed:possibleDiseaseTarget   ?disease .
  t4  ?drug      owl:sameAs                       ?sadrug .
  t5  ?disease   rdfs:label                       ?disName
  t6  ?sadrug    sider:sideEffect                 ?seffect .
  t7  ?seffect   sider:sideEffectName             ?sename .
  t8  ?dbdrug    drugbank:chemicalFormula         ?drugformula
}
```

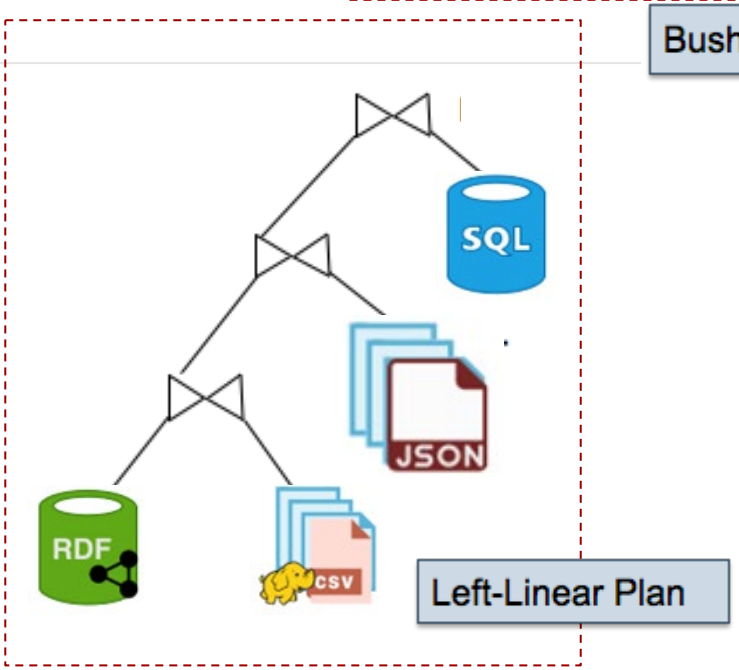


Join Orderings

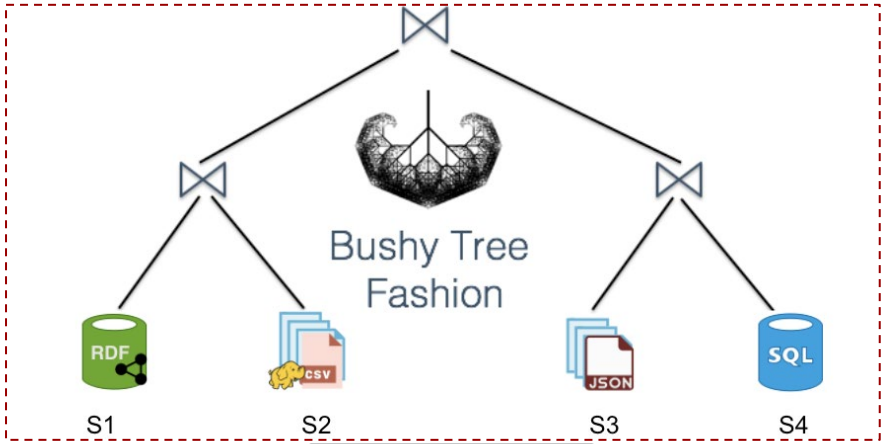


Join Orderings

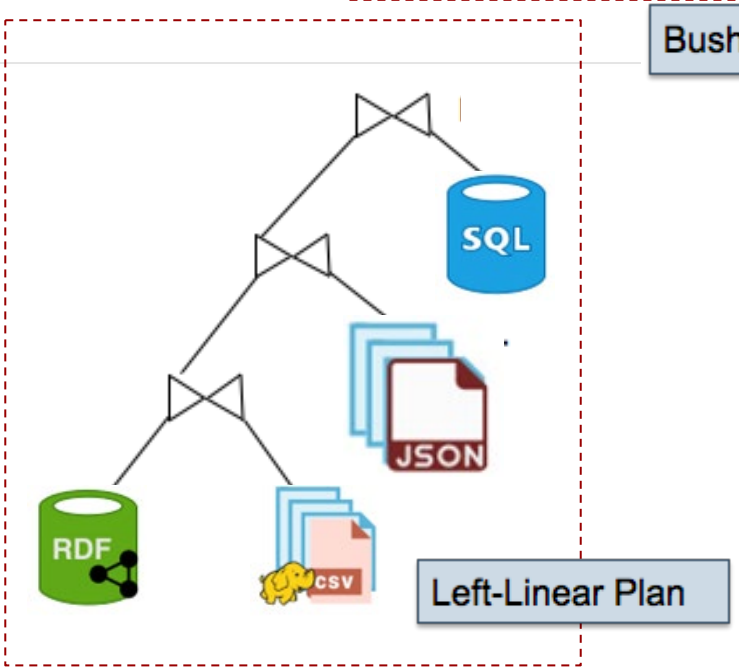




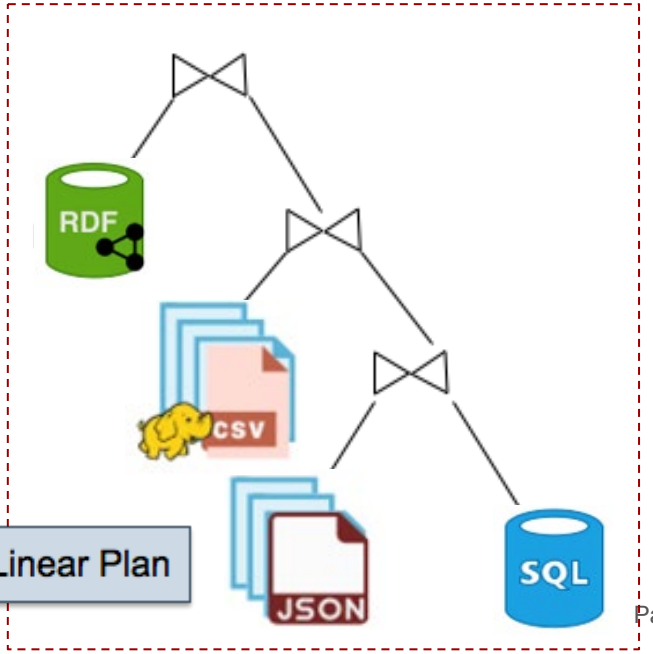
Join Orderings



Bushy Plan



Left-Linear Plan



Right-Linear Plan

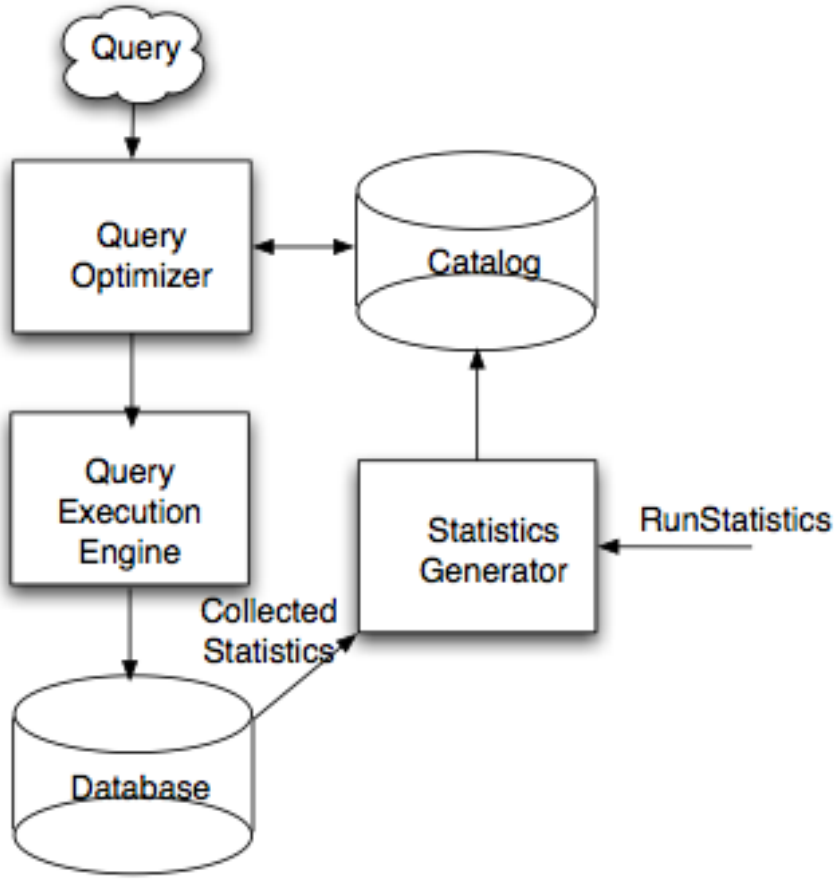
Query Processing Steps

Query Processing is divided into three major steps:

Statistics generation.

Query optimization.

Query Execution.



The Optimize-Then-Execute Paradigm

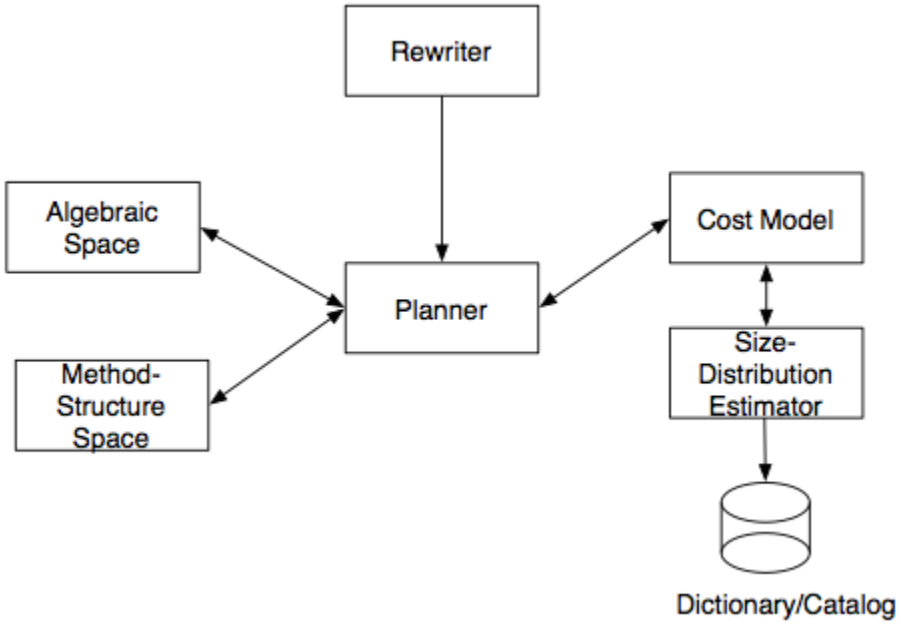
Traditional Query Processing techniques:

- **Parse** a declarative query.
- Generate an **intermediate** representation of the query **(Query Blocks).**
- Produce an efficient **logical and physical plan**; minimize disk I/O access.
- Execute the **query plan** without **making** runtime decisions.

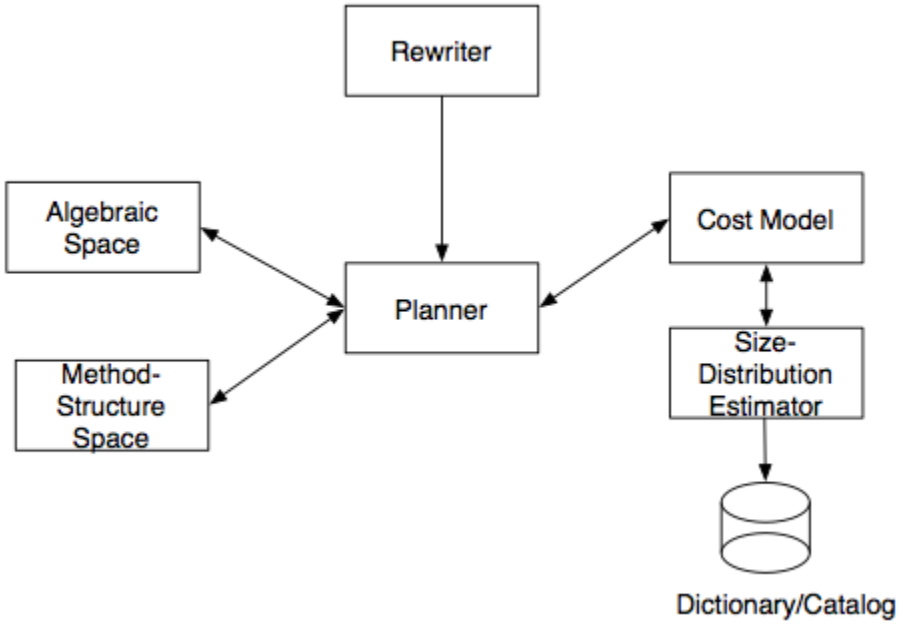
A logical plan is a tree
- Non-leaf nodes correspond to operations of in an algebra (e.g., the relational algebra)
- Leaf nodes correspond to relations or subqueries to be executed over a data source

A physical plan is a logical plan
- Non-leaf nodes are annotated with the algorithms used to execute the algebra operators
- Leaf nodes are annotated with the technique used to access the relations of the subquery

Traditional Query Optimizer



Traditional Query Optimizer

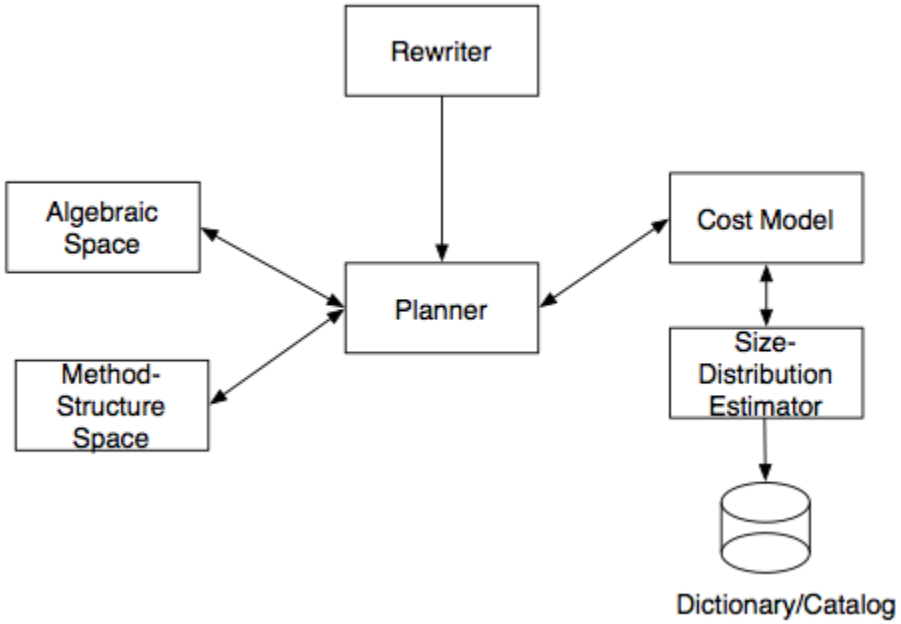


Query Rewriter

Applies algebraic transformations to the query to produce a more efficient equivalent query; some transformations are:

- Flatten out queries, using views, Nested queries are decomposed into query blocks.

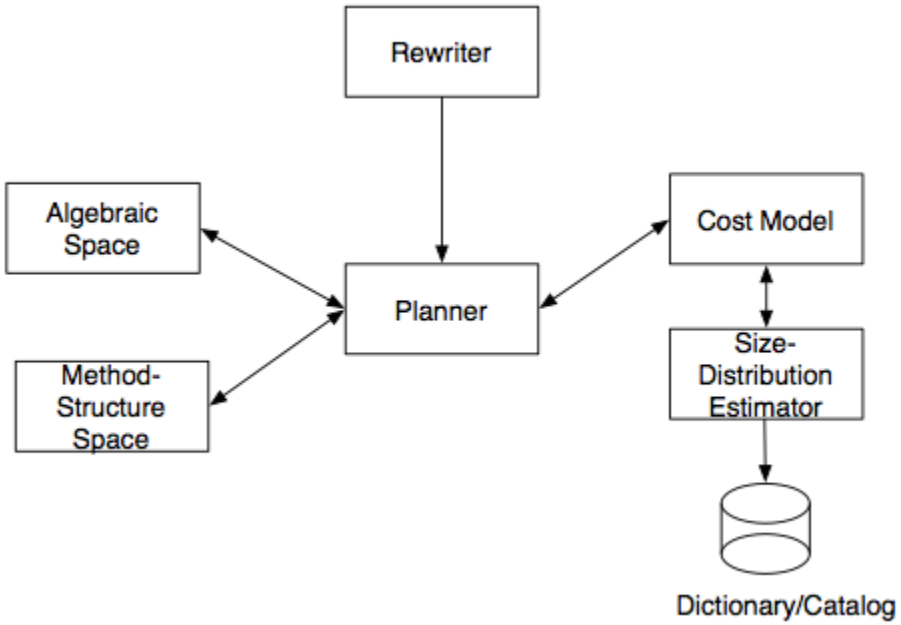
Traditional Query Optimizer



Planner

- Traverses the space of possible execution plans following a particular search strategy, e.g., dynamic programming, randomized algorithms.
- Query blocks are optimized one at a time.
- All available access methods are considered for each relation.
- All the ways to join the relations one-at-a-time; permutations of relations and different join methods are considered.

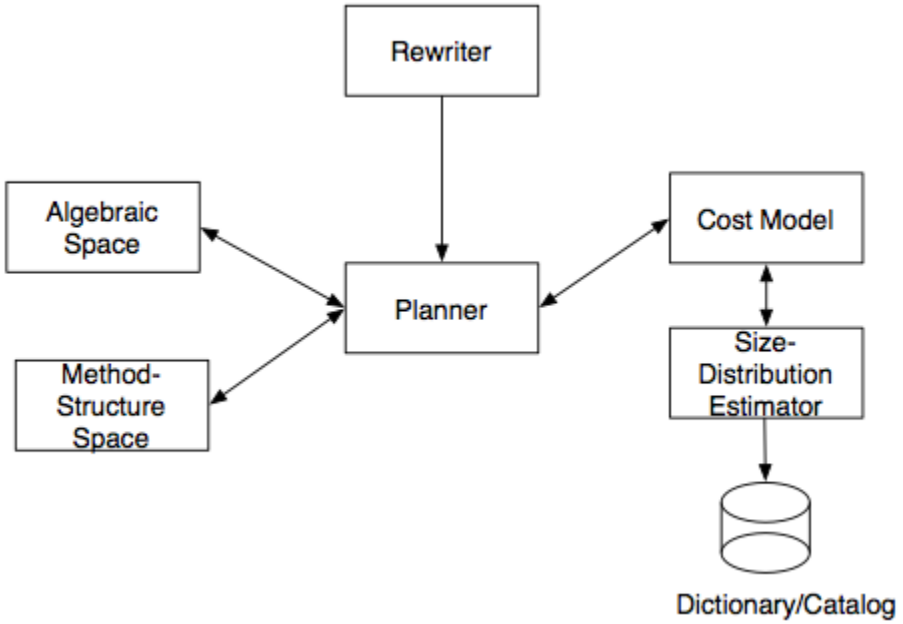
Traditional Query Optimizer



Algebraic Space

Set of algebraic rules that restrict the space of plans transformations, and guide the planner into the space of efficient logical plans.

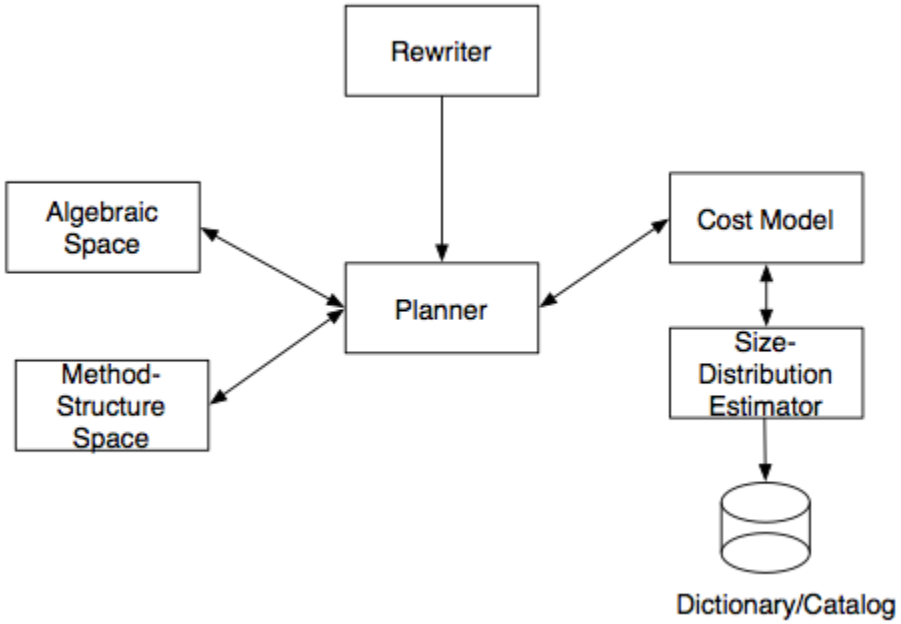
Traditional Query Optimizer



Method-Structure Space

Set of rules that restrict the space of physical implementations of each logical plan, and guide the planner into the space of efficient physical plans.

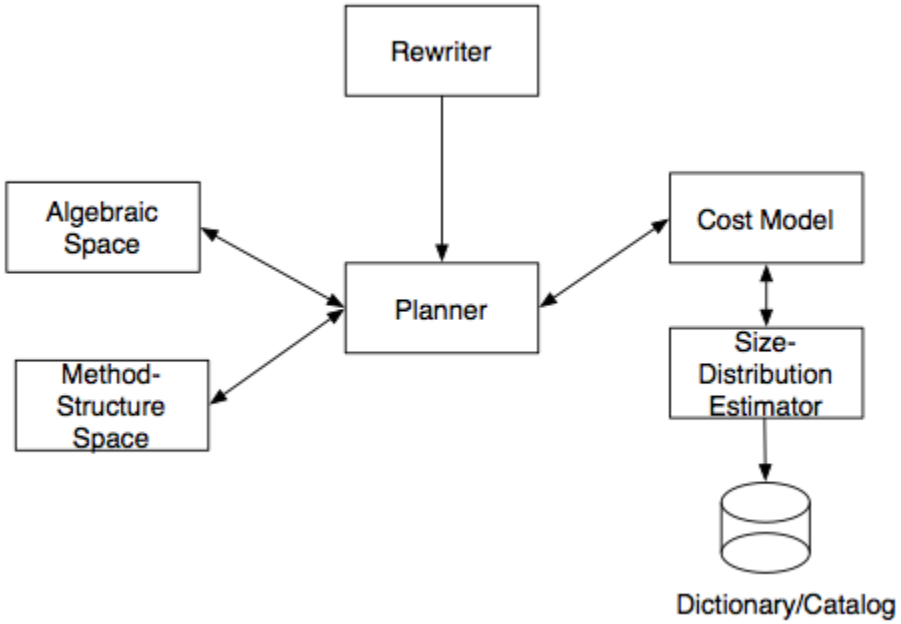
Traditional Query Optimizer



Method-Structure Space

Set of rules that restrict the space of physical implementations of each logical plan, and guide the planner into the space of efficient physical plans.

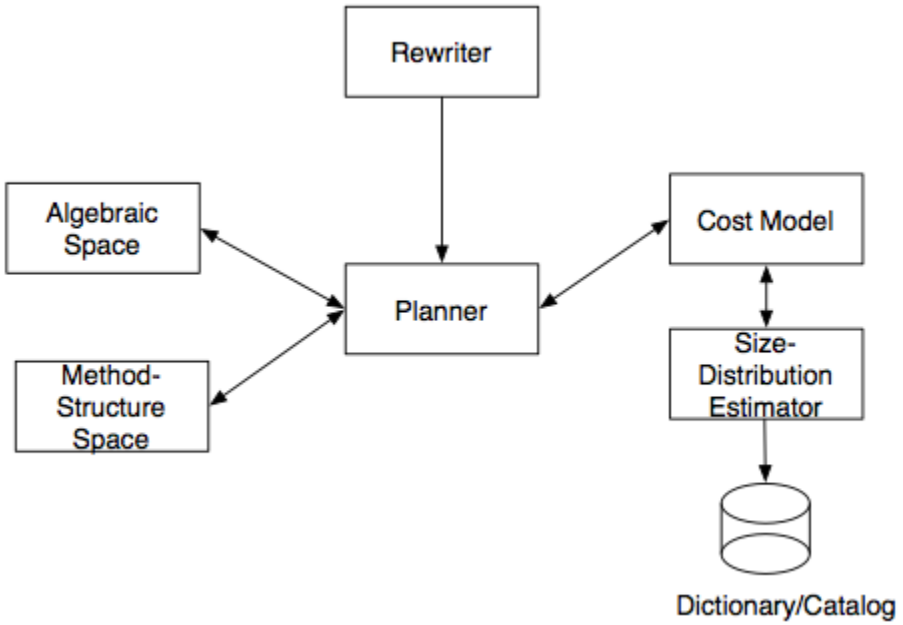
Traditional Query Optimizer



Cost Model

Set of arithmetic rules or statistical techniques, to estimate the execution cost and cardinality of logical and physical plans.

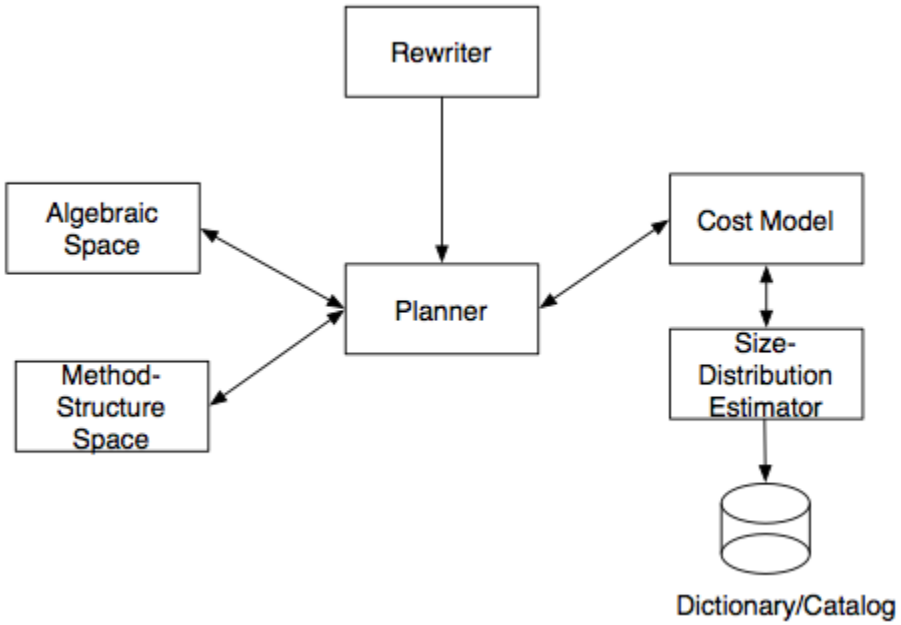
Traditional Query Optimizer



Size-Distributor Estimator

Set of arithmetic rules or statistical techniques to estimate the cardinalities of dataset to be accessed as well as its distributions, and the selectivity of a query select condition.

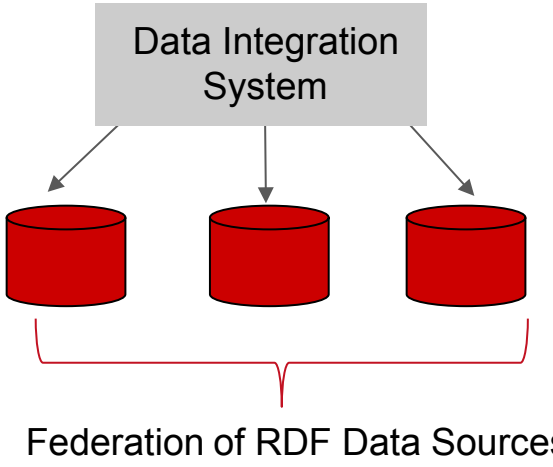
Traditional Query Optimizer



Dictionary

Set of statistics that characterize the dataset, e.g., number of different values of an attribute, or different subjects, properties, or values; distributions, etc. Stored statistics depend on the type of size-distribution estimator.

Federated SPARQL Query Engines



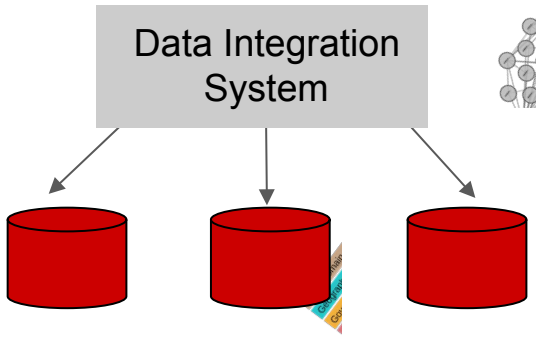
#LD

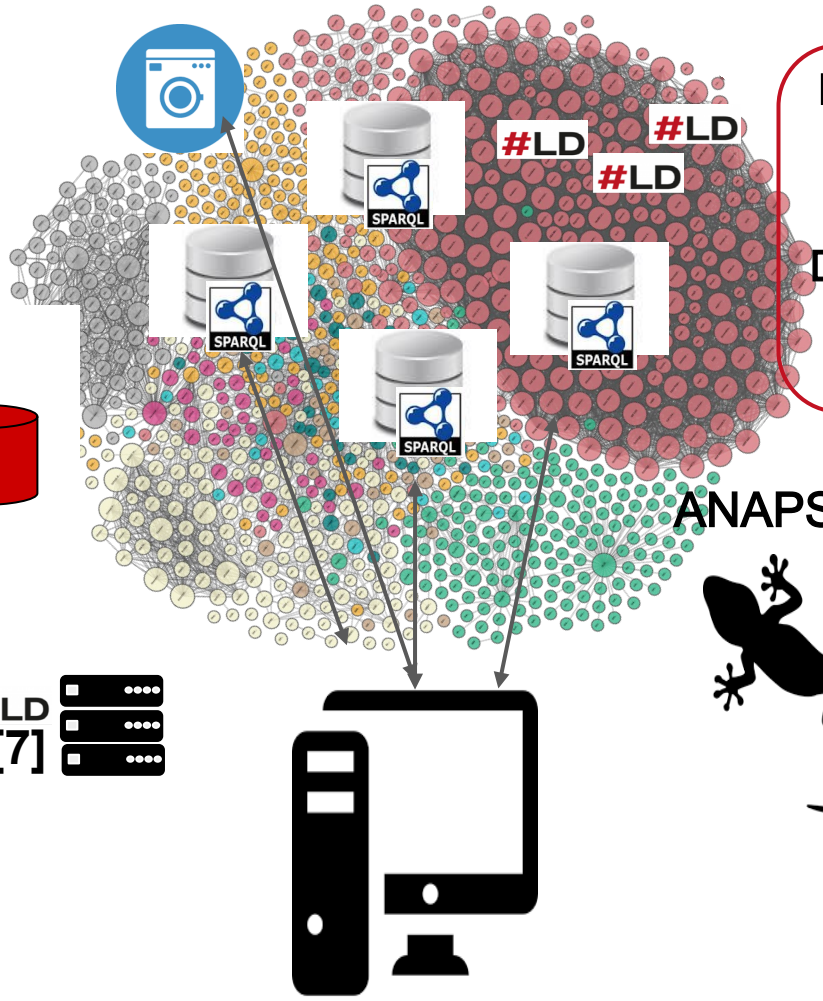
Web-access interfaces that allow for querying RDF data:

- SPARQL Endpoints: respect **SPARQL** protocol, i.e., **any** SPARQL query
- Triple Pattern Fragments: limited query capabilities, i.e., **only one** triple pattern

Challenges: Query processing is impacted by different parameters, e.g., **query capabilities, data** fragmentation, dataset **size** and **connectivity**, and query **selectivity**

Federated SPARQL Query Engines





Extensions

LILAC[5] FEDRA[6]

Fed-DESATUR[3]

DAW[9] MULDER[10]

HIBISCUS[15]

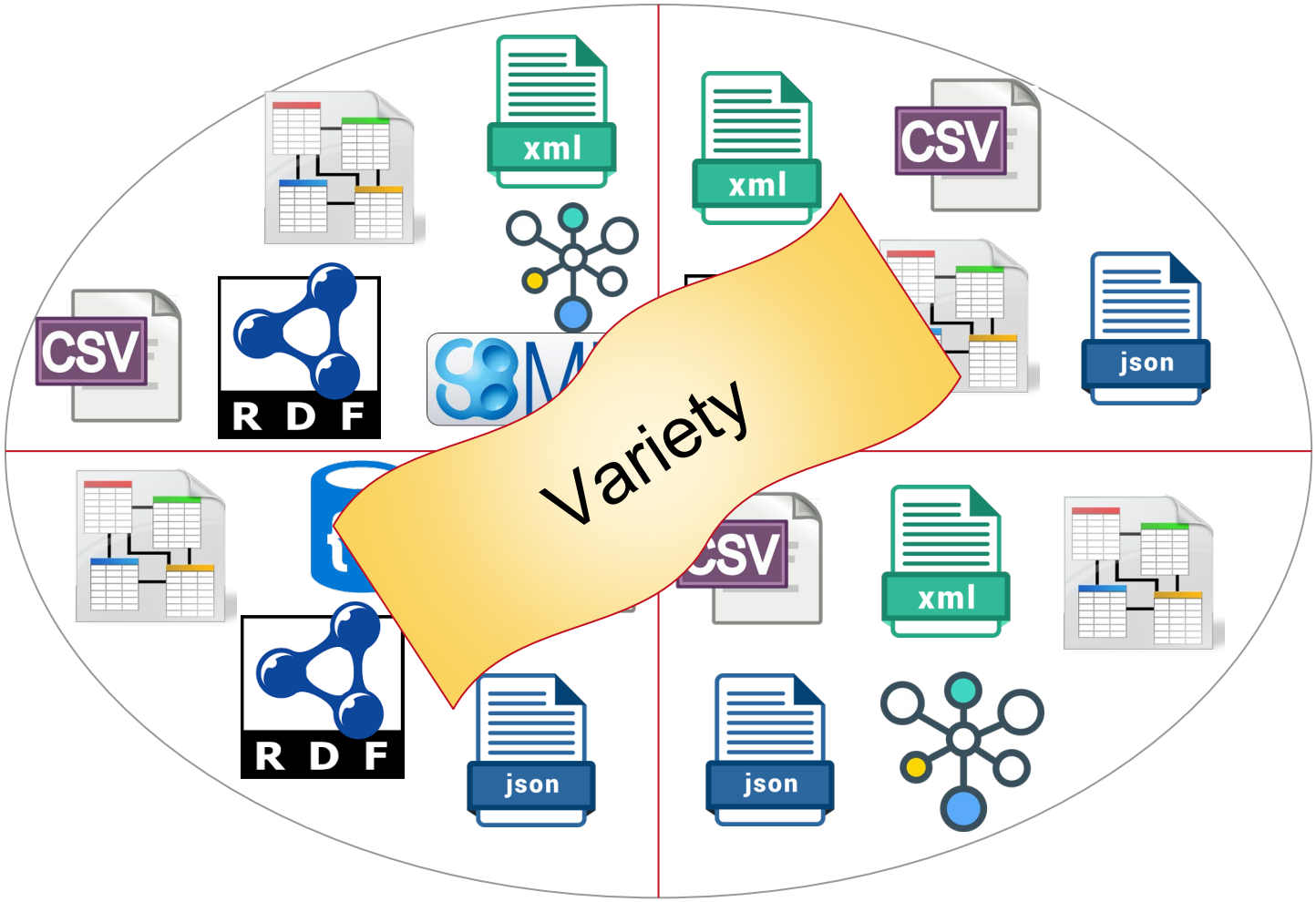
ANAPSID[1]

FedX [4]
Linked Data in a Federation

SPLENDID [3]

SemaGrow [12]

Impacting Data Complexity Dimensions



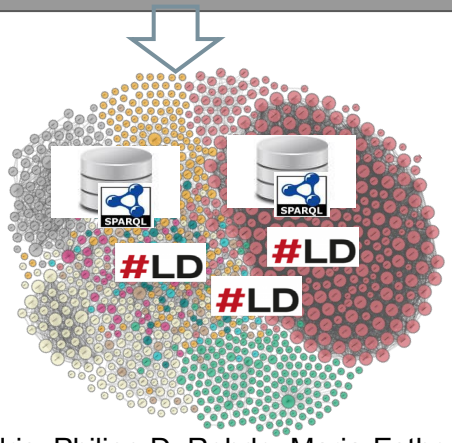
Hybrid Federated Query Engines

SPARQL Query **Q**

Source Selection & Query Decomposition

Query Optimizer

Execution Strategies



Kemele M. Endris, Philipp D. Rohde, Maria-Esther Vidal, Sören Auer: Ontario: Federated Query Processing Against a Semantic Data Lake. DEXA (1) 2019

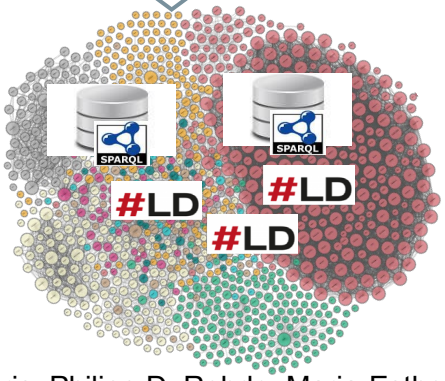
Hybrid Federated Query Engines

SPARQL Query **Q**

Source Selection & Query Decomposition

Query Optimizer

Execution Strategies

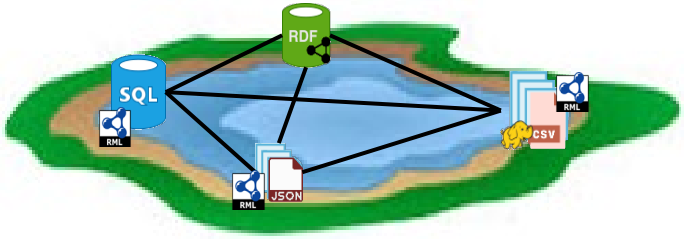


SPARQL Query **Q**

Source Selection & Query Decomposition over Heterogeneous Sources

Query Optimizer

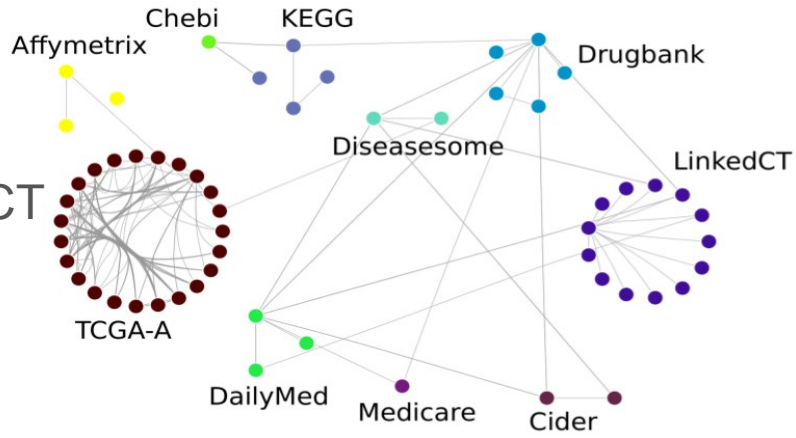
Hybrid Execution Strategies **over Heterogeneous Sources**



Kemele M. Endris, Philipp D. Rohde, Maria-Esther Vidal, Sören Auer: Ontario: Federated Query Processing Against a Semantic Data Lake. DEXA (1) 2019

Experimental Setup

- Benchmark:
  - Life Science Linked Open Data (LSLOD)
  - 10 RDF Data Source
  - 10 Simple Queries
    - UNION, OPTIONAL, DISTINCT
    - 3 - 8 triple patterns
    - 2 - 4 star-shaped sub-queries



#triples	#subjects	#predicates	#objects	RDF file size
96.10 M	8.32 M	742	27.47 M	16.0 GB

A. Hasnain, Q. Mehmood, S. Sana e Zainab, M. Saleem, C. Warren, D. Zehra, S. Decker, and D. Rebholz-Schuhmann. Biofed: federated query processing over life sciences linked open data. Journal of Biomedical Semantics, 8(1):13, Mar 2017.

Experimental Setup

Experimental Configuration

- 23 Docker containers
  - 10 RDF sources (Virtuoso 6.01.3127)
  - 10 RDB sources (MySQL 5.7)
  - Three engines (FedX, MULDER, Ontario)
- Metrics:
  - **Execution time:** Time elapsed between query submission and retrieval of last answer

Types of Subqueries

CI: Star-shaped subqueries with no instantiations or filter clauses

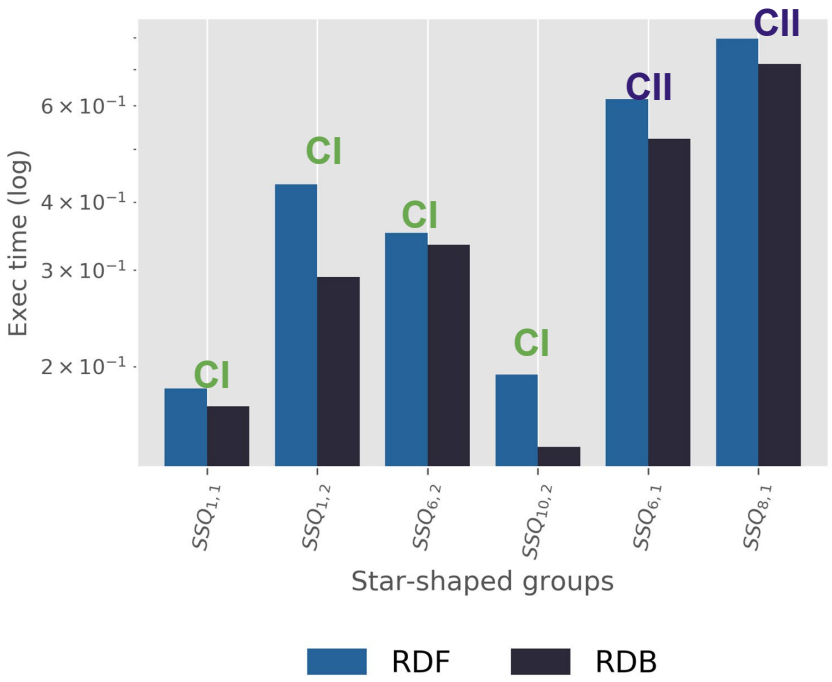
CII: Star-shaped subqueries with no instantiations or filter clauses, and defined over an RDF class implemented by joining several relational tables in a data lake

CIII: Star-shaped subqueries with instanstiations in object variables

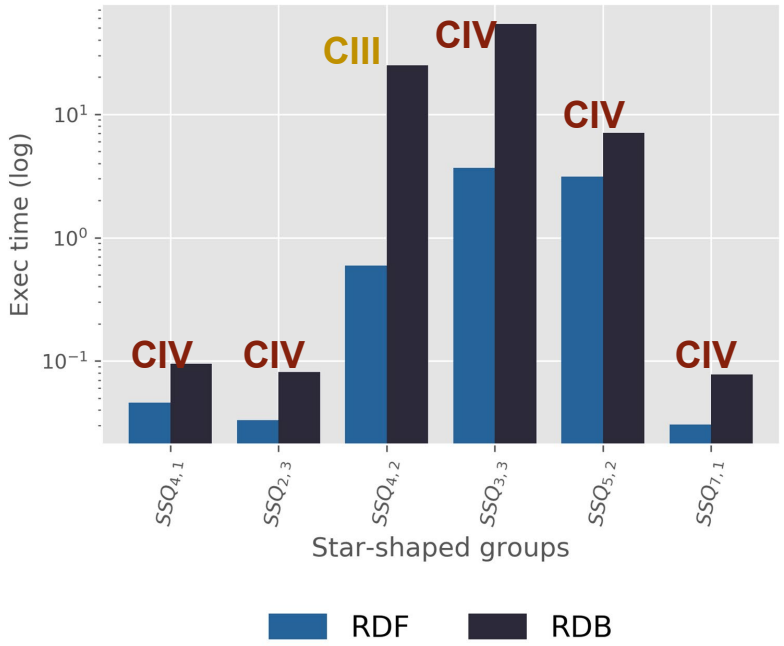
CIV: Star-shaped subqueries with instantiations or filter clauses, and defined over an RDF class implemented by joining several relational tables in a data lake

Exp I: Impact of Star-shaped Groups

Goal: Evaluate the impact of different subqueries--**star-shaped groups (SSQs)**-- on the performance of a query engine.



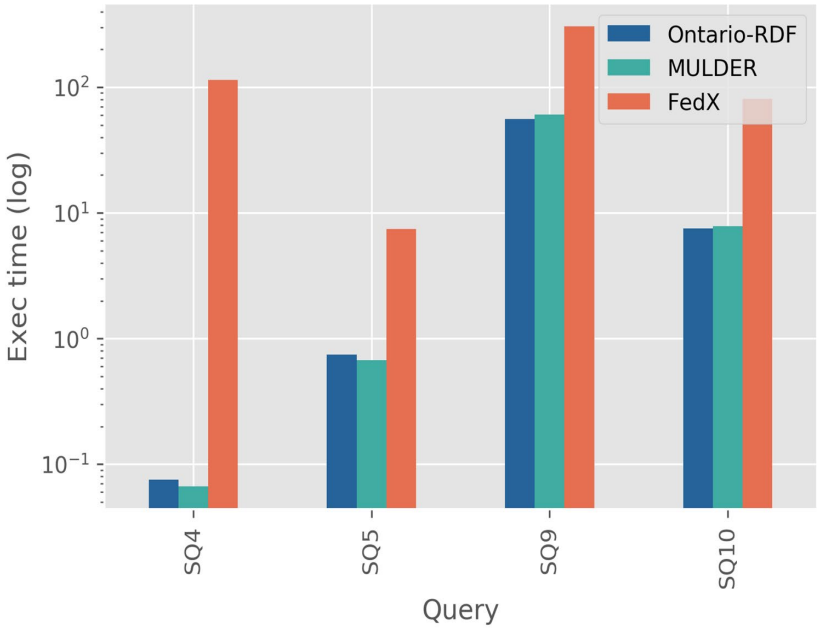
RDB scans a relation or a set of relations, while an **RDF** engine scans over all data. Thus, RDB engines **outperform** RDF engines

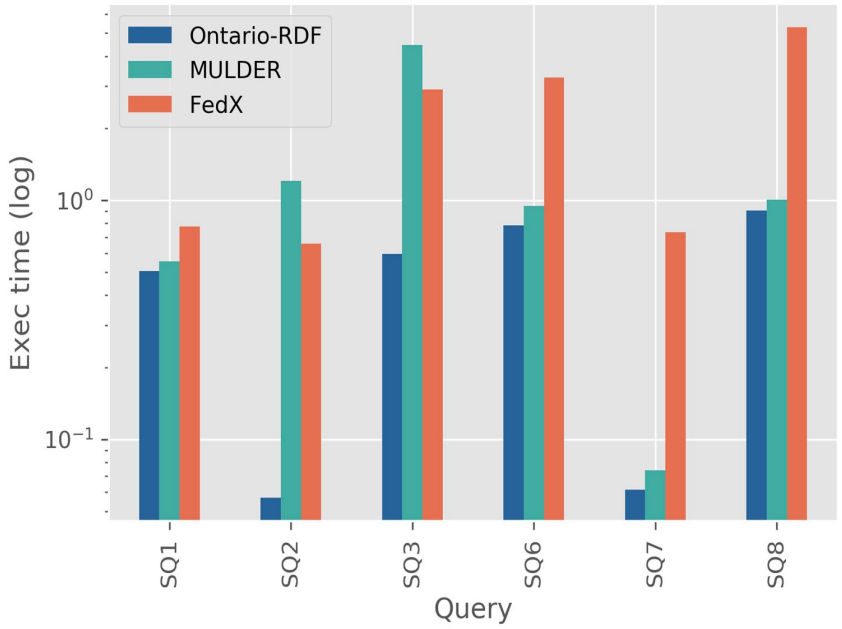


RDB only has indexes on primary keys, while an **RDF** engine has indexes over combinations of subject, predicate, and object. Thus, RDF engines **outperform** RDB engines

Exp II: Impact of Considering Heterogeneity

Goal: Performance of Ontario engine over RDF data sources and the overhead introduced while considering heterogeneity

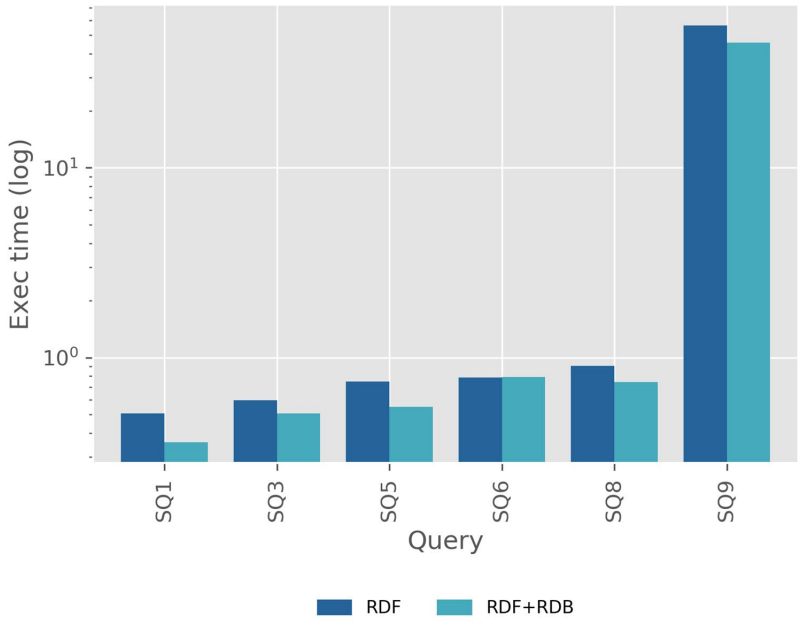


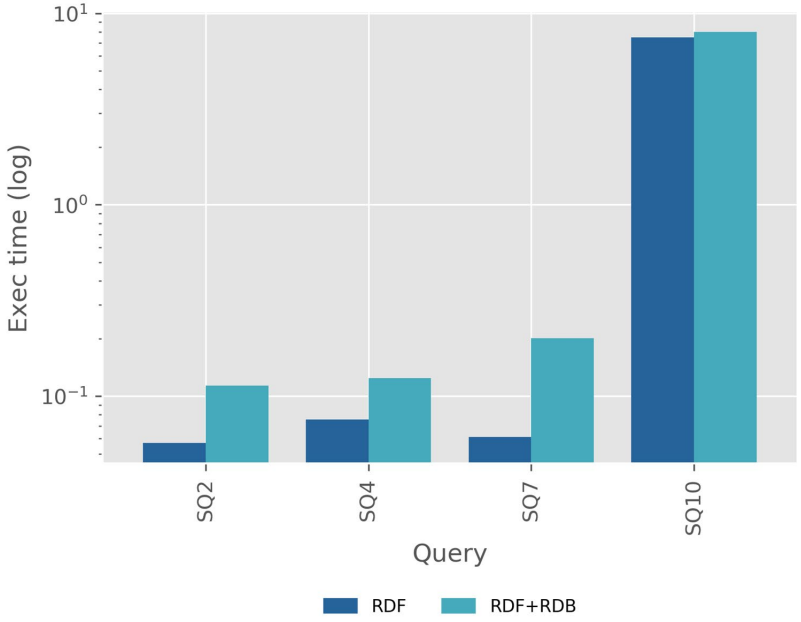


Ontario pays the price of **considering heterogeneous data sources**. Ontario outperforms both **FedX and MULDER** by generating efficient plans and using optimization rules tailored for RDF sources on the rest of the queries

Exp III: Impact of Heterogeneity

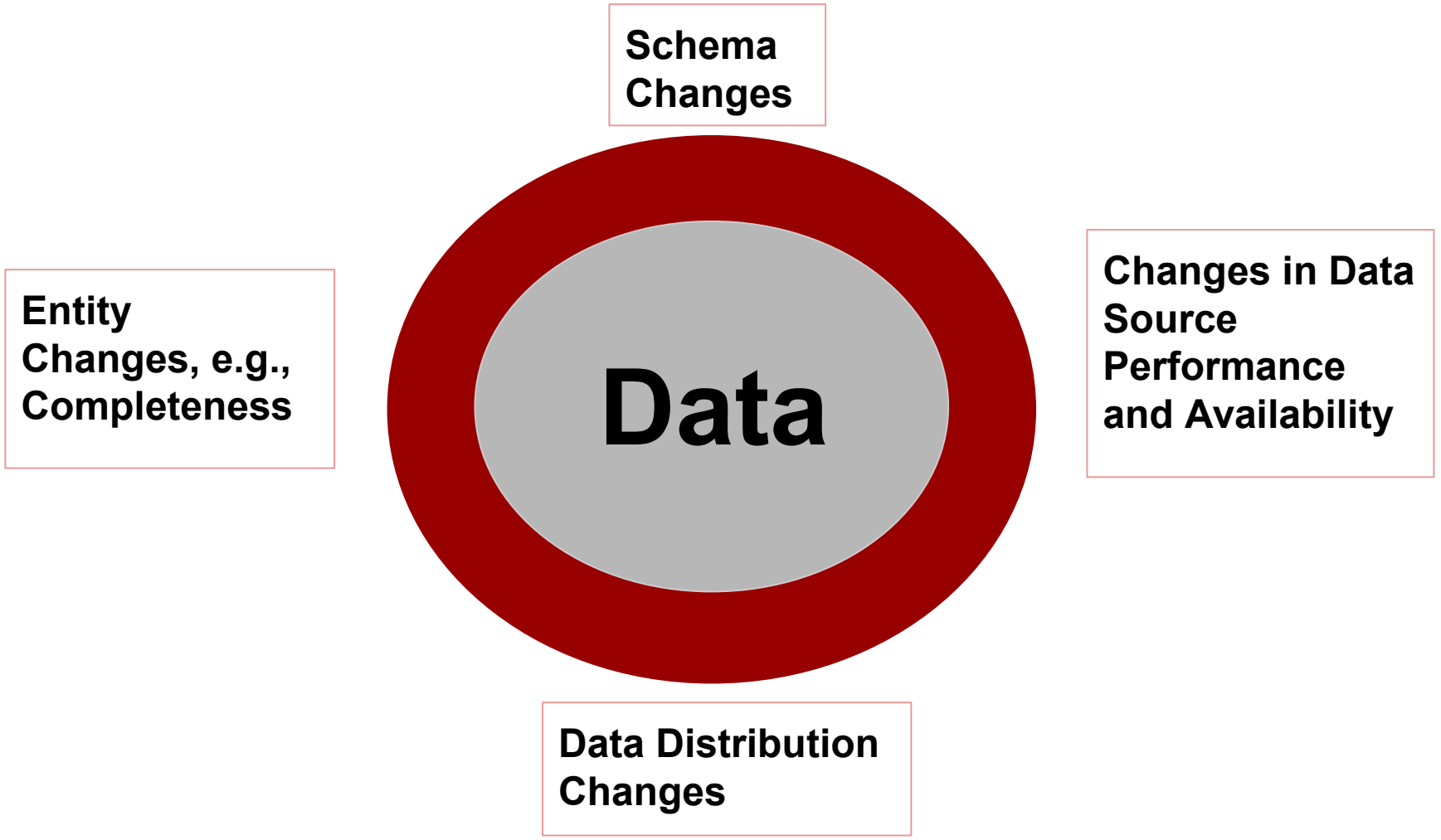
Goal: Performance of Ontario over heterogeneous sources, i.e., RDF and RDB



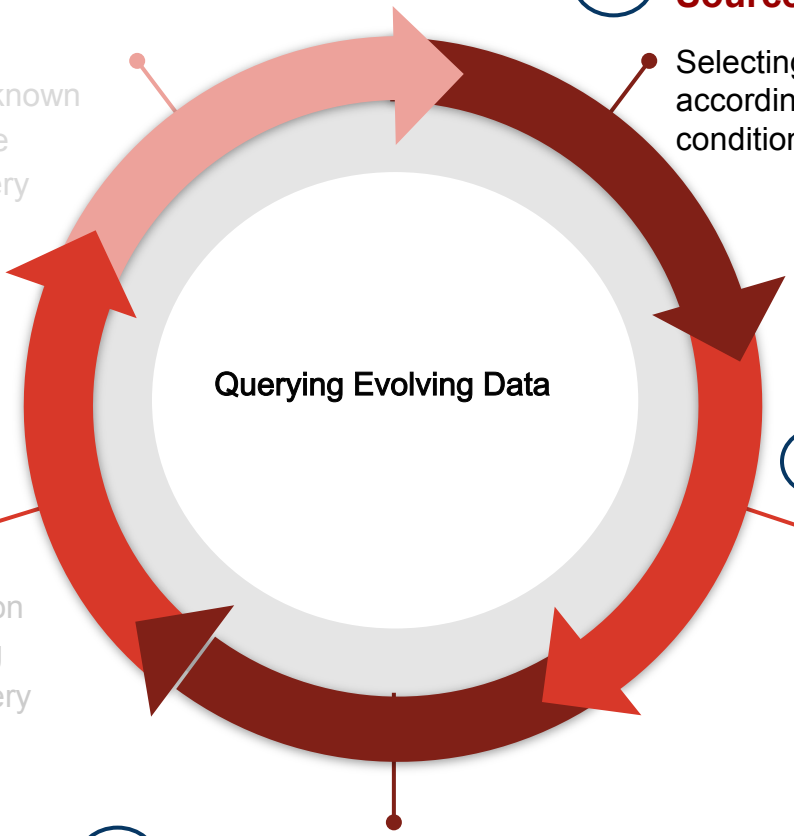


Characteristics of the queries impact on the performance of the federated query engine. **Ontario** is able to identify according to the data source implementations which is the most effective plan.

Data Evolution….



Required Solutions to Support Evolution



5

Knowledge Incompleteness

Considering that unknown facts may need to be predicted during query execution

1

Source Evolution

Selecting the sources according to their current conditions and availability

4

Knowledge Evolution

Considering the evolution of the knowledge during the execution of the query

2

Environment Evolution

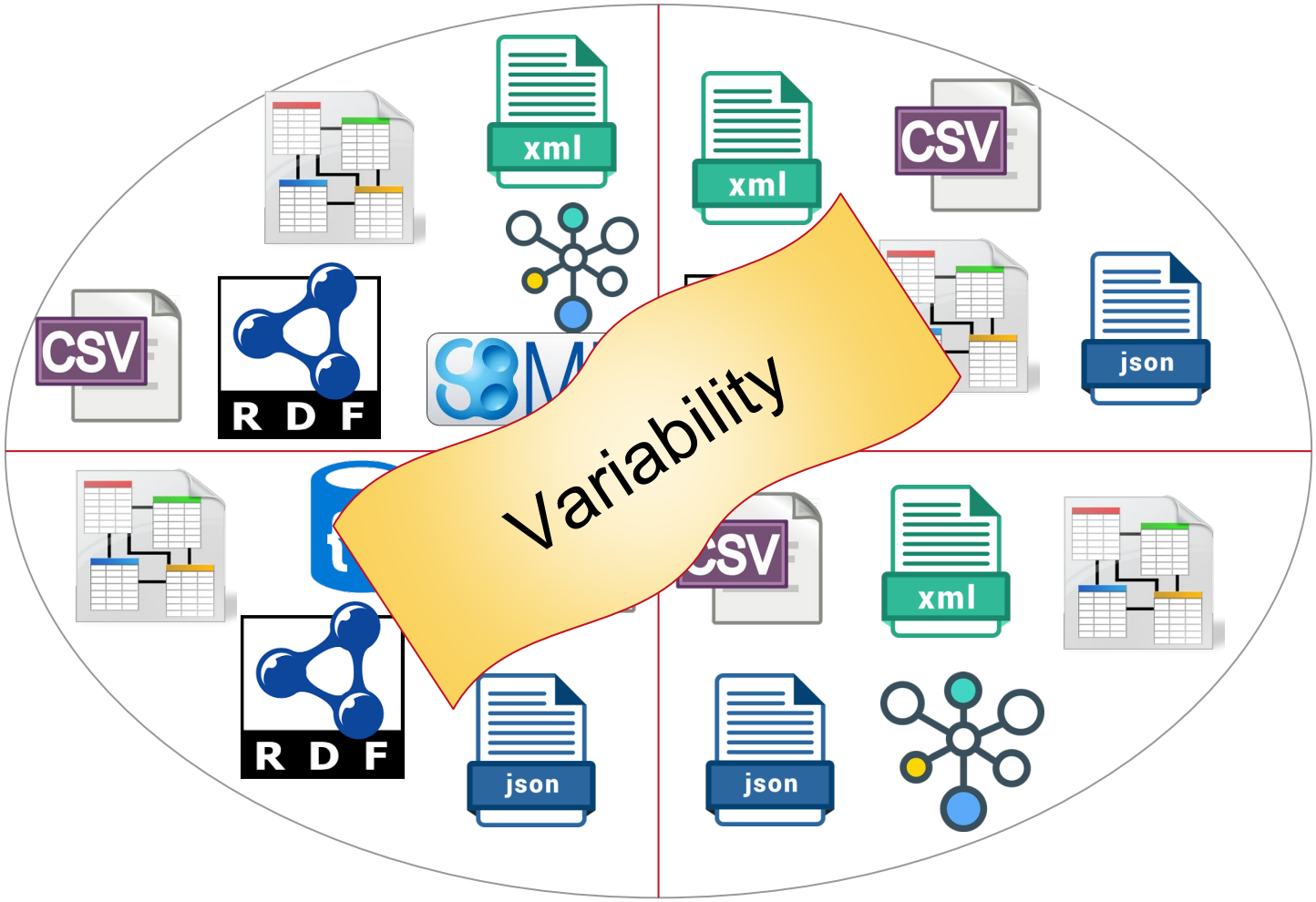
Executing queries according to current conditions of the environment

3

Data Evolution

Considering the status of the data, e.g., completeness, during the execution of the query

Impacting Data Complexity Dimensions



Ideal Federated Query Engines

- Systems able to **change** their **behavior** by learning **behavior** of **data providers**.
- **Receive** information from the **environment**.
- Use **up-to-date** information to **change** their **behavior**.
- Keep **iterating** over time to **adapt** their **behavior** based on the **environment** conditions.

Challenges: Federated Query Processing

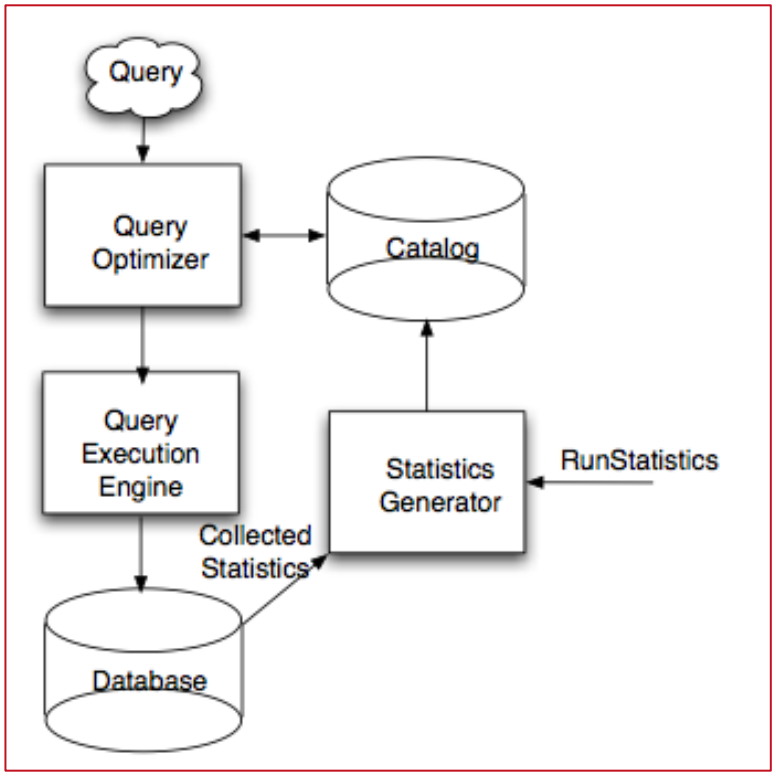
- SPARQL endpoints have been implemented for a large number of data sets
- Missing dataset statistics

Source selection & Query planning

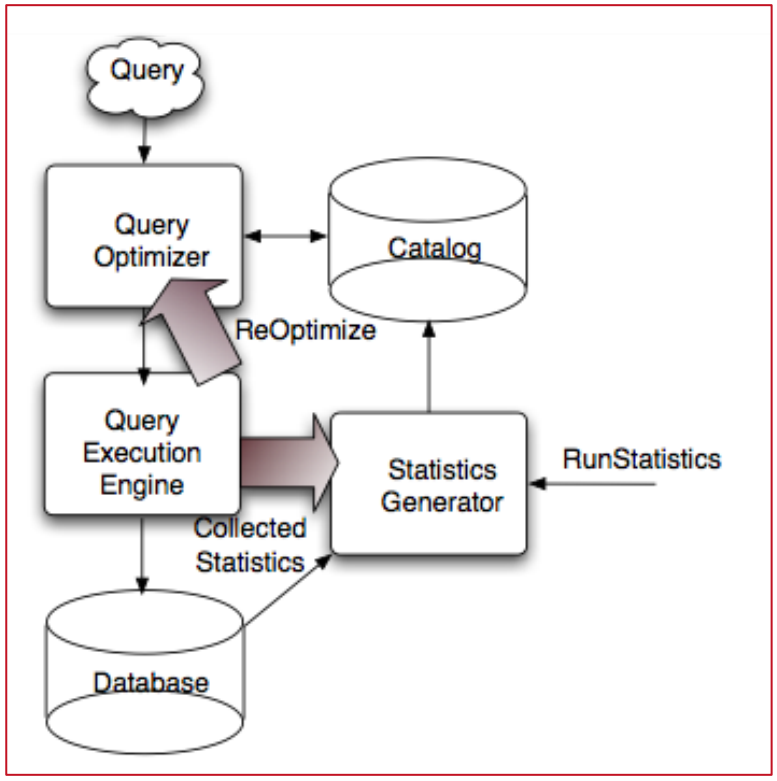
- Unpredictable network delays
- Unpredictable endpoint workload

Query execution **(adaptivity is required)**

Adaptive Query Processing



Optimized-Then-Execute Paradigm



Adaptive Query Processing

Adaptive SPARQL Query Engines

Adapt to Source and Environment Evolution:

- **Misestimated** or **missing** statistics.
- **Unexpected** correlations.
- **Unpredictable** costs.
- **Dynamically** changing **data**, **workload,** and source **availability**.
- **Changes** at **rates** at which tuples **arrive** from sources
  - **Initial** Delays.
  - **Slow** Delivery.
  - **Bursty** Arrivals.

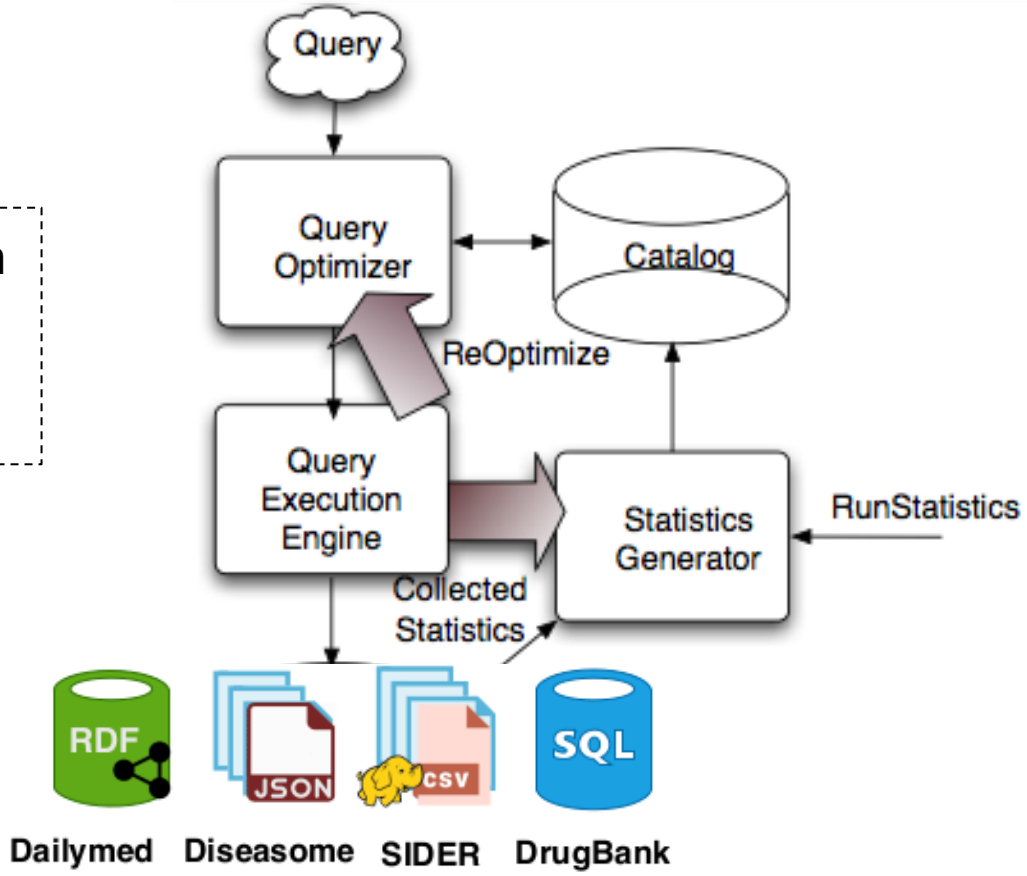
Adaptivity in Federated Query Processing

Adaptive Query Federated Engines are able to:

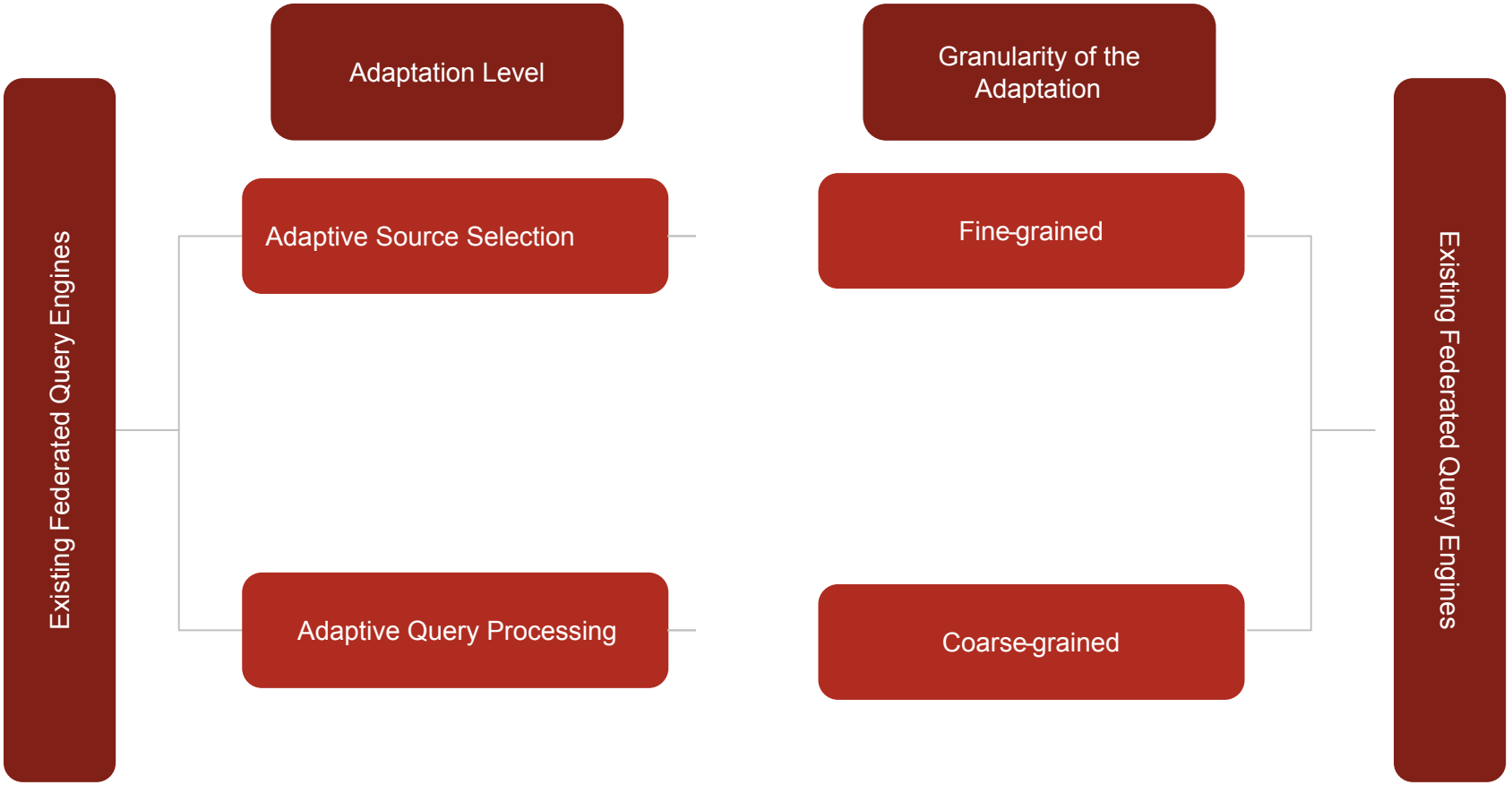
- Change their behavior by learning the behavior of data providers
- Receive and exploit information from the environment
- Use up-to-date information to change their behavior
- Keep iterating over time to adapt their behavior based on the environment conditions

Adaptive Federated Query Engines

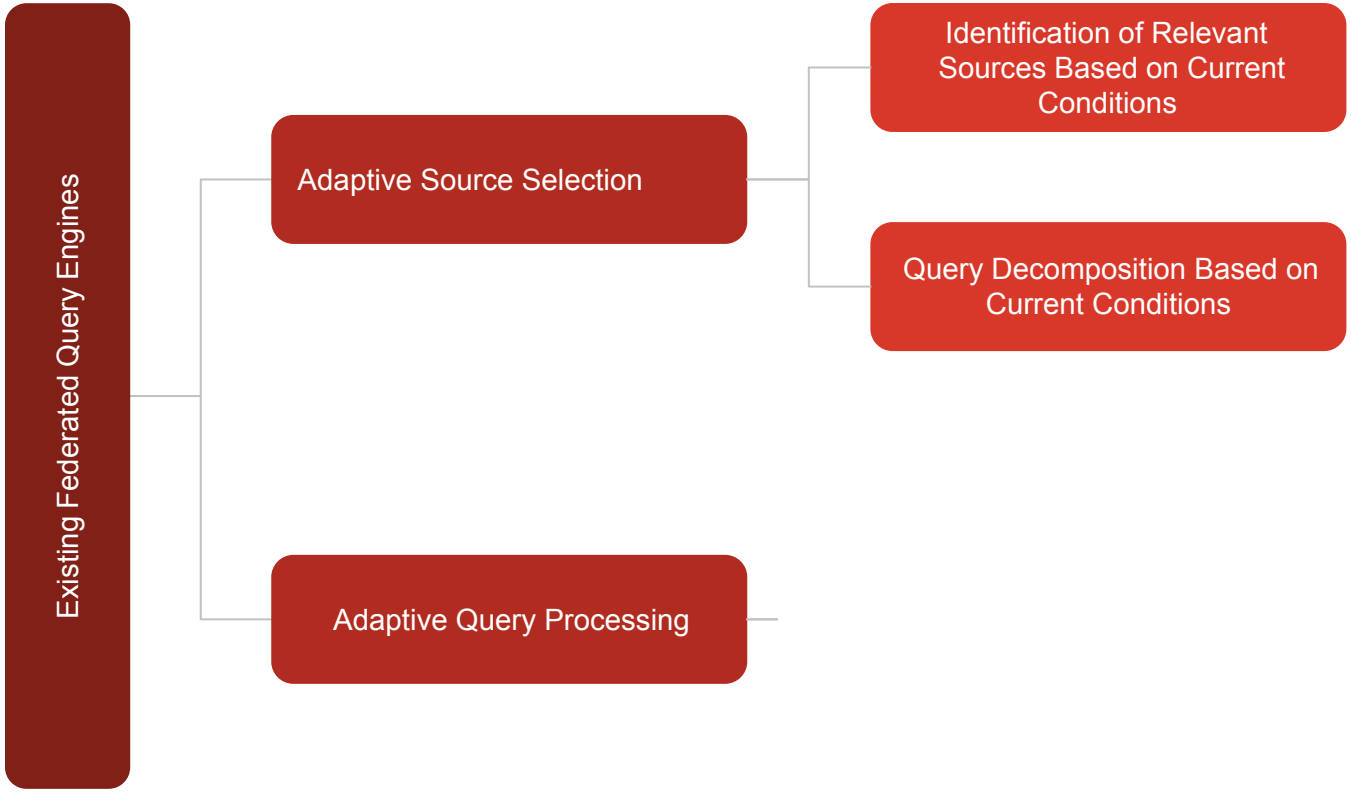
Re-optimize the original **plan on-the-fly** according to the source conditions



Existing Federated SPARQL Query Engines



Existing Federated SPARQL Query Engines



Adaptivity at Source Selection Level

Source Selection: searching strategies to select the sources for answering a query according to the real-time source conditions:

- **Schema** changes
- **Source** availability
- **Data distribution** changes

Adaptivity During Source Selection

LILAC FEDRA

Fed-DESATUR

DAW MULDER

HIBISCUS

SemaGrow

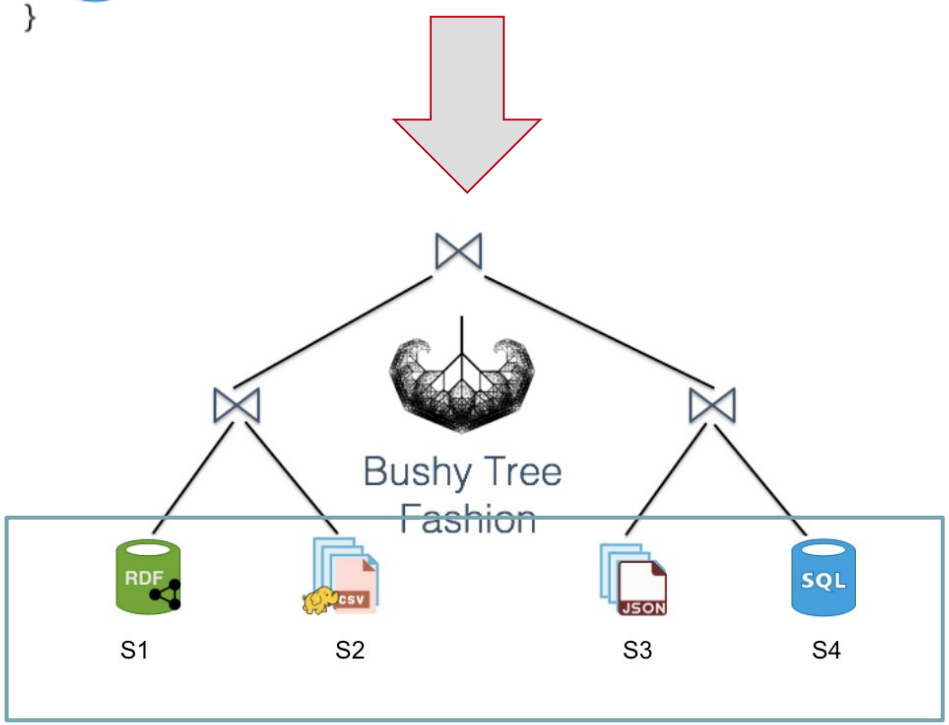
ANAPSID SPLENDID

FedX
Linked Data in a Federation

Source Selection techniques that allow for identifying the sources that can be used to answer a query based on the current conditions of the sources

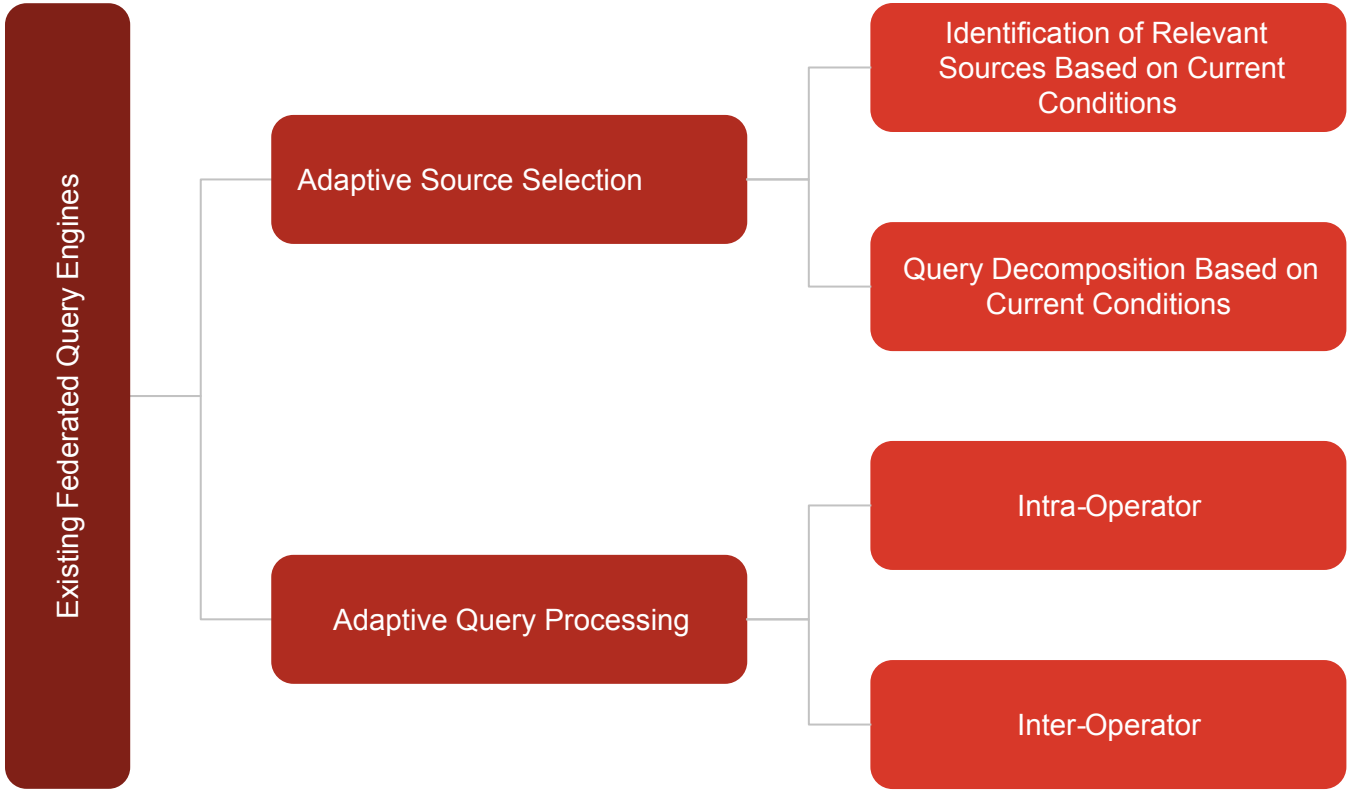
Query Planning Over Heterogeneous Data Sources

```
SELECT DISTINCT ?drug ?disName ?drugformula ?sename
WHERE {
  t1  ?drug      dailymed:activeIngredient        dailymed:Simvastatin .
  t2  ?drug      dailymed:genericDrug             ?dbdrug .
  t3  ?drug      dailymed:possibleDiseaseTarget   ?disease .
  t4  ?drug      owl:sameAs                       ?sadrug .
  t5  ?disease   rdfs:label                       ?disName
  t6  ?sadrug    sider:sideEffect                 ?seffect .
  t7  ?seffect   sider:sideEffectName             ?sename .
  t8  ?dbdrug    drugbank:chemicalFormula         ?drugformula
}
```



Source Selection

Existing Federated SPARQL Query Engines



Only adaptivity to changes in the environment is addressed!!

Adaptivity During Query Execution

No Adaptivity

Fine-Grained Adaptivity

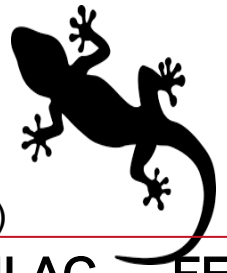
FedX
Linked Data in a Federation

SemaGrow
SPLENDID

DAW HIBISCUS

ANAPSID

Network of Linked Data Eddies (nLDE)



LILAC FEDRA

MULDER

Fed-DESATUR

Implement physical operators and query processing techniques to adjust query schedulers to the conditions of the sources and the network

Adaptive Query Engine

Intra-Operator

Operators able to detect when sources become **blocked** or data traffic is **bursty**

- **Opportunistically** produce results as **quickly** as data arrives from the sources
- Results are produced **incrementally**

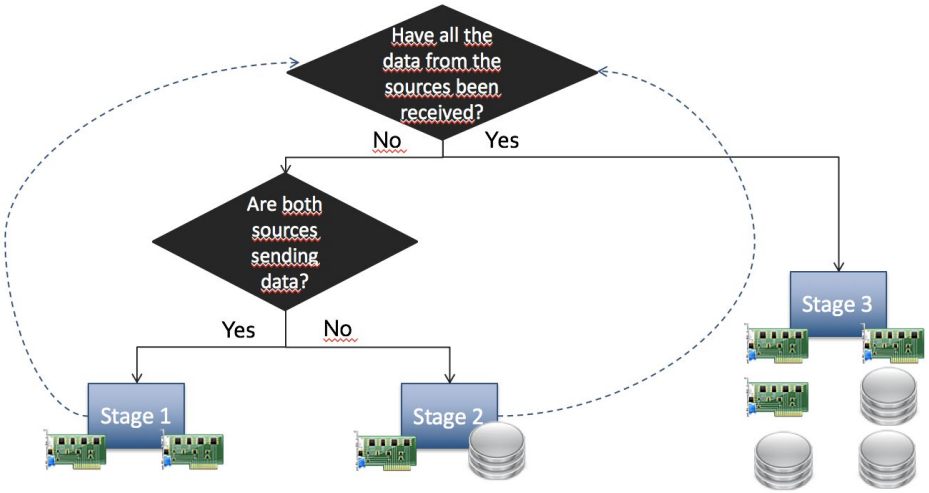
Adaptive Query Engine

Intra-Operator

Operators able to detect when sources become **blocked** or data traffic is **bursty**

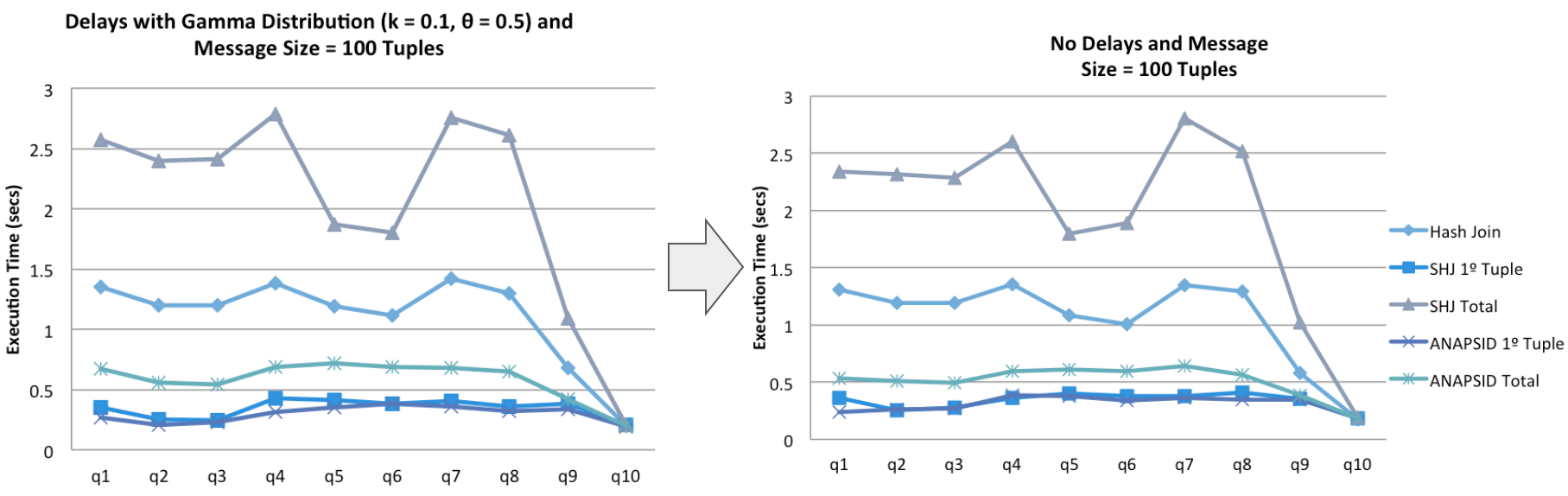
- **Opportunistically** produce results as **quickly** as data arrives from the sources
- Results are produced **incrementally**

GJoin [Acosta and Vidal et al. 2011]



M. Acosta, M.E. Vidal, T. Lampo, J. Castillo, E. Ruckhaus: ANAPSID: An Adaptive Query Processing Engine for SPARQL Endpoints. ISWC, 2011.

Intra-Operator



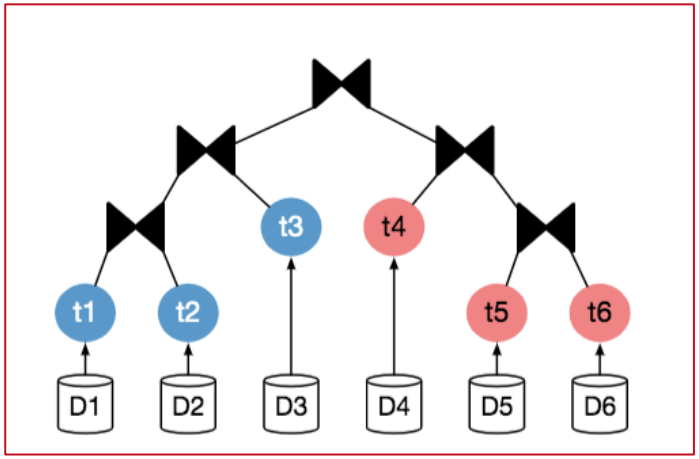
ANAPSID intra-operator strategies are able to **produce all the results faster** than state-of-the-art **join operators** in presence of **delays**

Adaptive Query Engine

Intra-Operator

Operators able to detect when sources become **blocked** or data traffic is **bursty**

- **Opportunistically** produce results as **quickly** as data arrives from the sources
- Results are produced **incrementally**



SMJoin [Galkin et al. 2017]

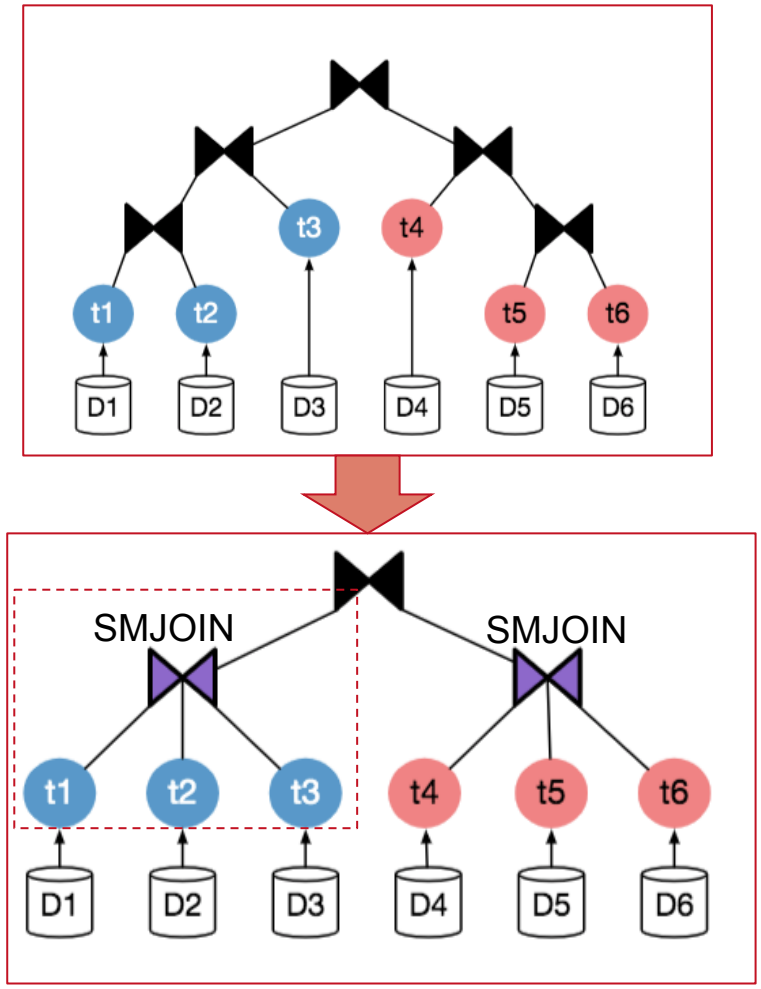
M. Galkin, K. M. Endris, M. Acosta, D. Collarana, M.-E.r Vidal, S. Auer: SMJoin: A Multi-way Join Operator for SPARQL Queries. SEMANTICS 2017:

Adaptive Query Engine

Intra-Operator

Operators able to detect when sources become **blocked** or data traffic is **bursty**

- **Opportunistically** produce results as **quickly** as data arrives from the sources
- Results are produced **incrementally**



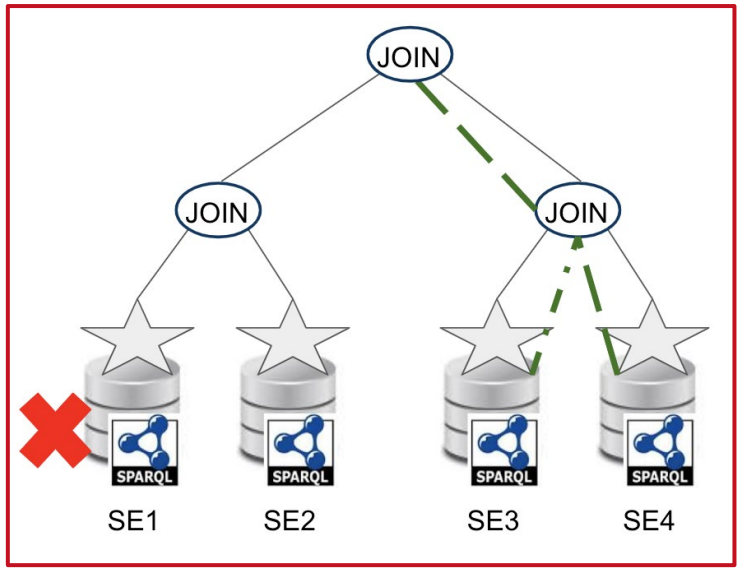
SMJoin [Galkin et al. 2017]

M. Galkin, K. M. Endris, M. Acosta, D. Collarana, M.-E.r Vidal, S. Auer: SMJoin: A Multi-way Join Operator for SPARQL Queries. SEMANTICS 2017:

Adaptive Query Engine

Intra-Operator

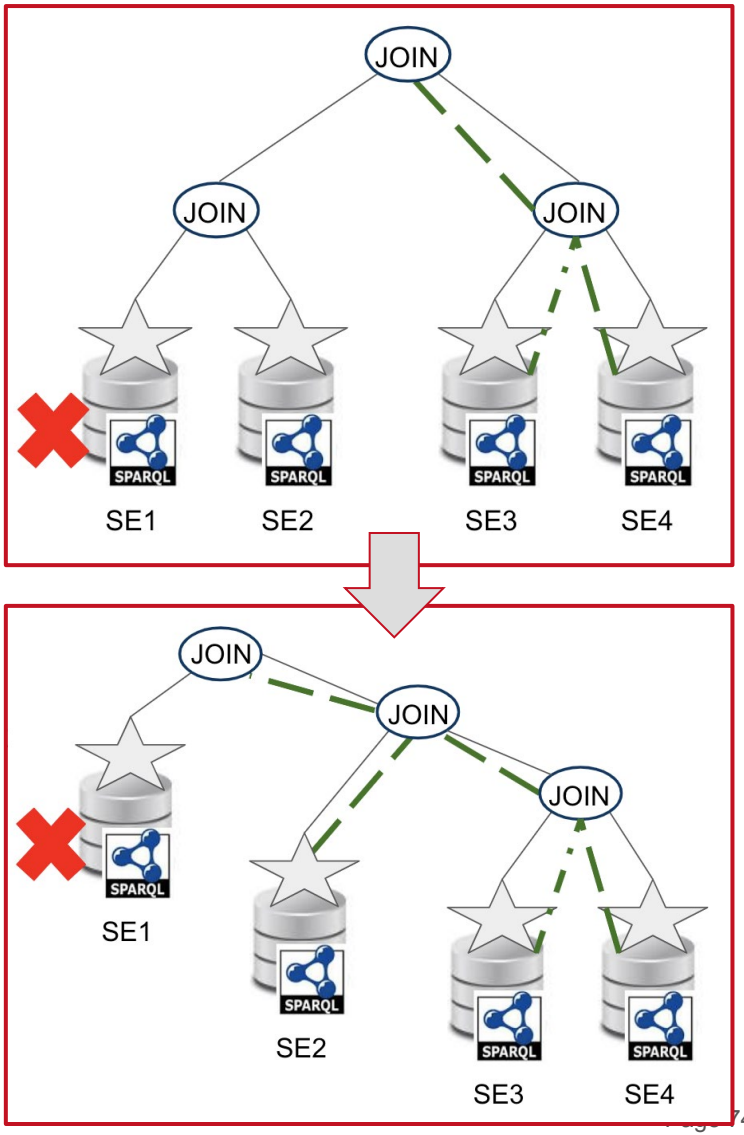
- Produce an **answer** as **soon** as it is computed
- Can **keep** producing **intermediate** results **even** when data a source becomes **blocked**



Adaptive Query Engine

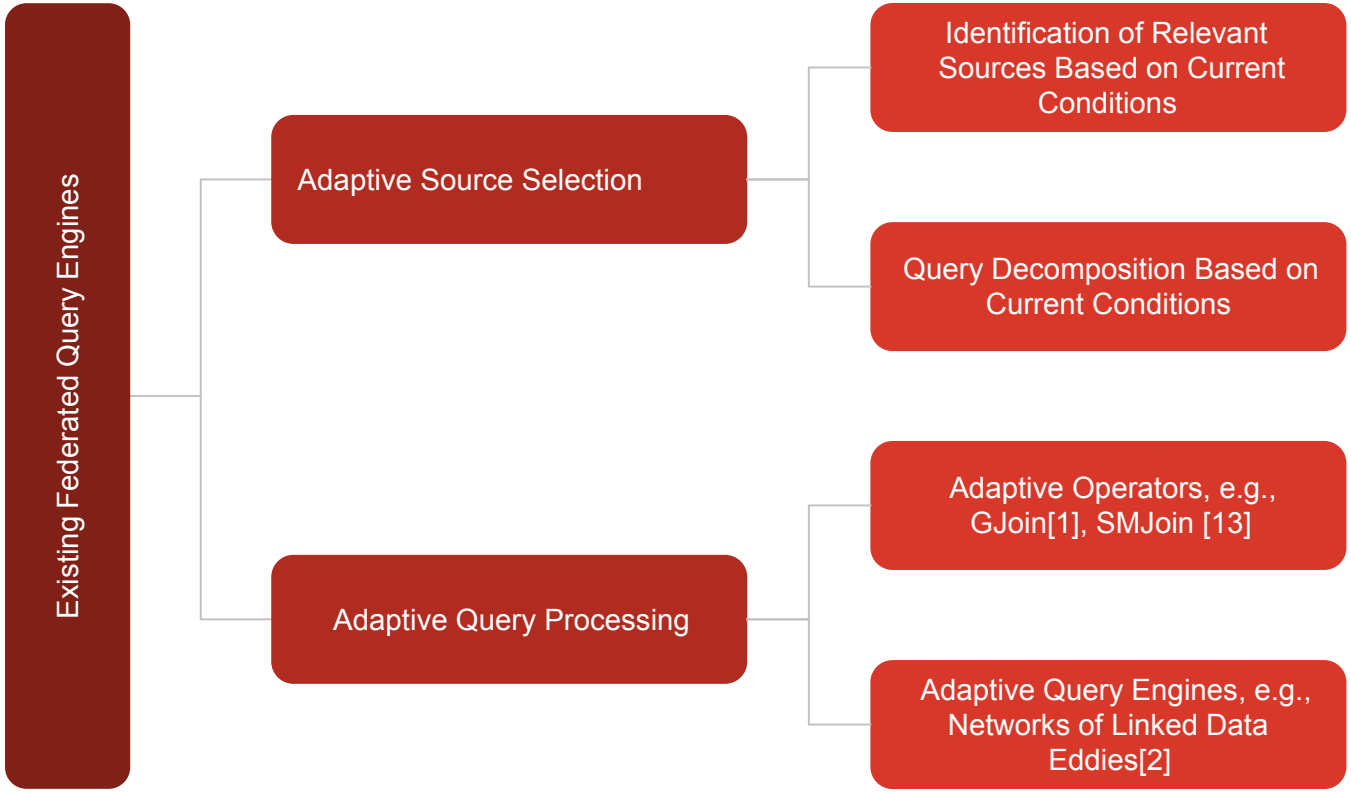
Intra-Operator

- Produce an **answer** as **soon** as it is computed
- Can **keep** producing **intermediate** results **even** when data a source becomes **blocked**



M. Acosta, M.E. Vidal: Networks of Linked Data Eddies: An Adaptive Web Query Processing Engine for RDF Data. ISWC 2015

Existing Federated SPARQL Query Engines



Only adaptivity to changes in the environment is addressed!!

Evaluation

Dataset: DBpedia 2015 (HDT on top of TPF server), 837M triples

Benchmark 1: **14** high-selective queries (<1000 int. res.)

Benchmark 2: **Four** low-selective queries (>1000 int. res.)

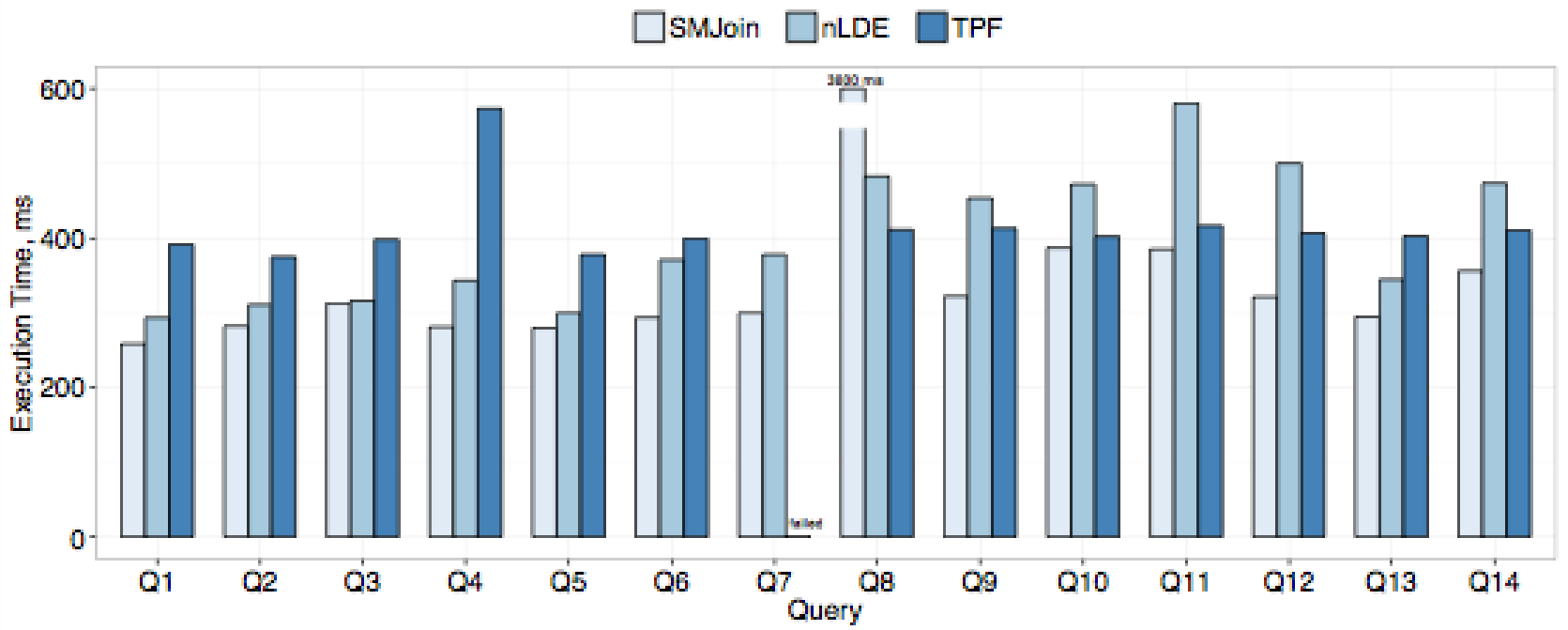
Metrics:

- Execution Time, ms
- Completeness over time, %

Compared tools:

- **TPF**: triple pattern fragment client [7]
- **nLDE**: network of Linked Data Eddies [2]
- **SMJoin**: multi-way join operator for SPARQL [13]

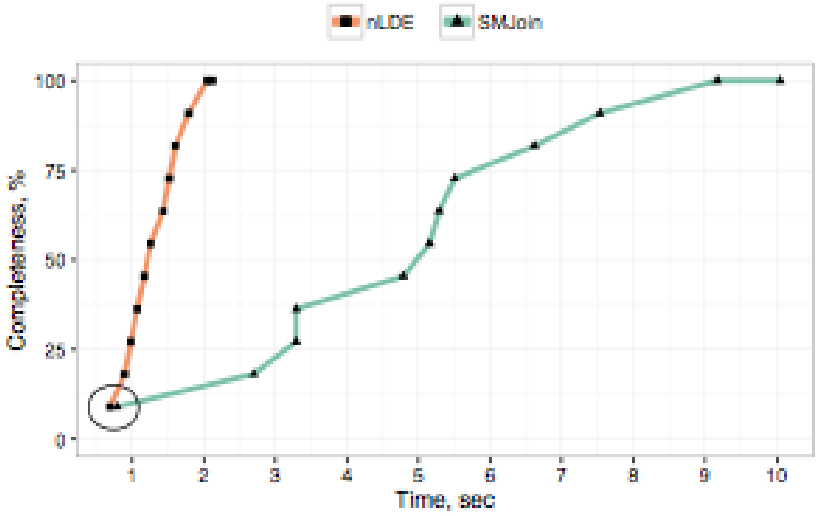
Benchmark 1: High Selective Queries



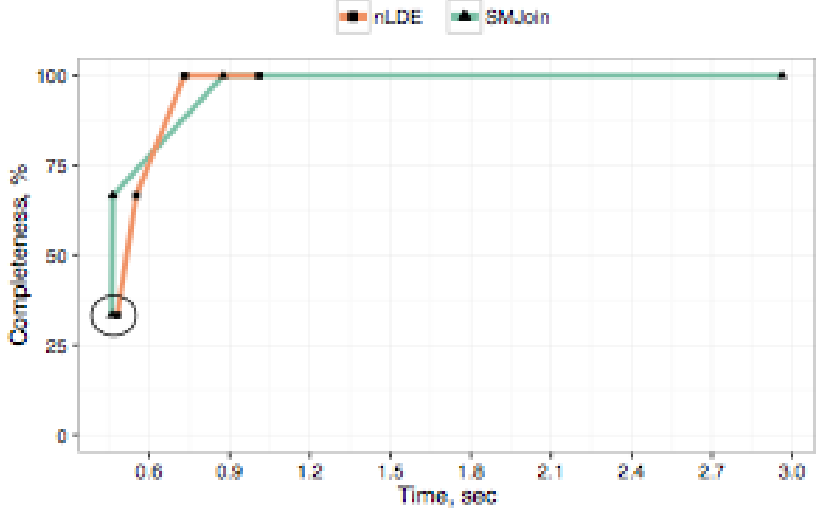
An adaptive approach like SMJoin outperforms other approaches in high-selective queries that produce small number of intermediate results

Benchmark 2: Low Selective Queries

Q1



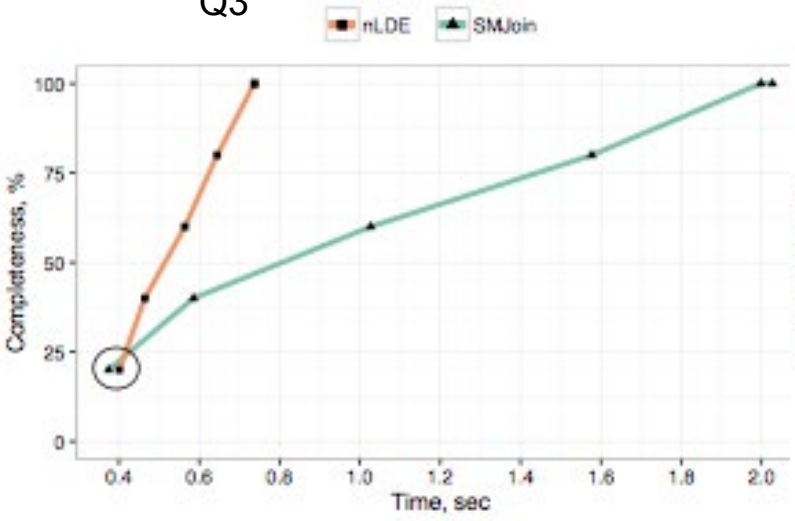
Q2



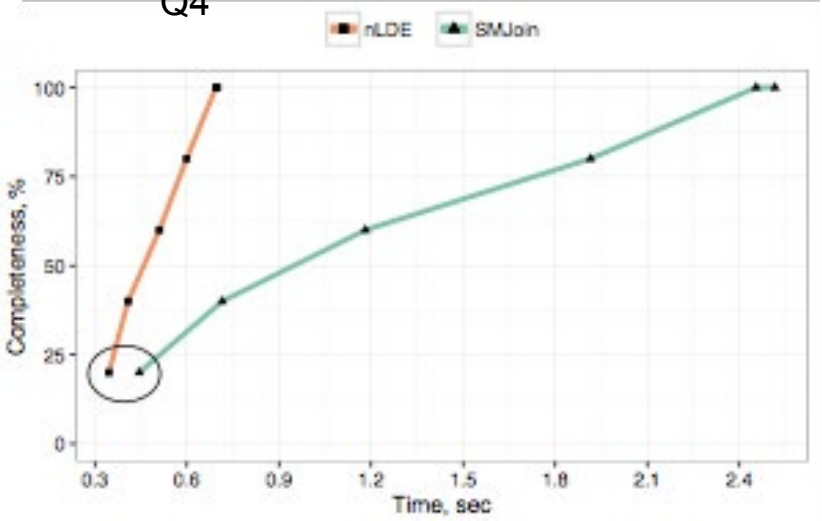
- SMJoin yields the first answer at about the same time as nLDE
- SMJoin has to process more intermediate results
- Q2: results are yielded but all intermediate tuples have to be processed

Benchmark 2: Low Selective Queries

Q3



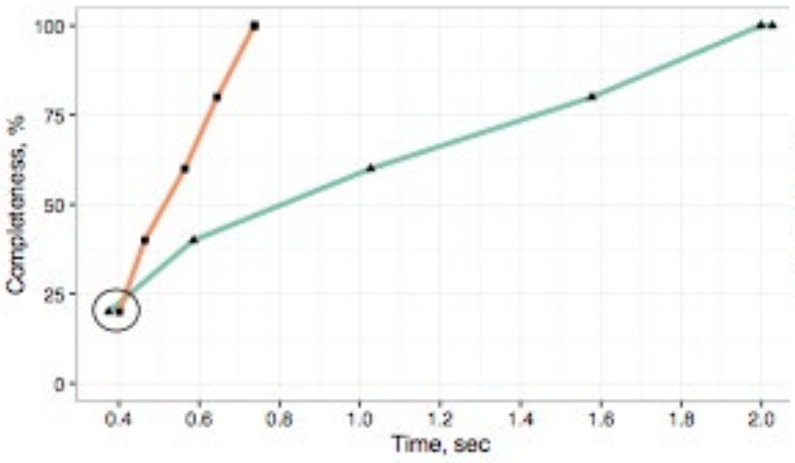
Q4



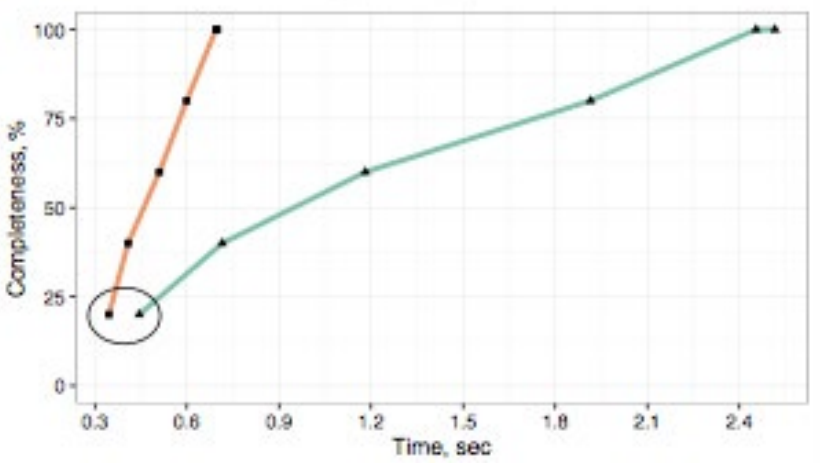
- SMJoin yields the first answer at about the same time as nLDE
- SMJoin has to process more intermediate results

Benchmark 2: Low Selective Queries

Q3



Q4



- SMJoin yields the first answer at about the same time as nLDE
- SMJoin has to process more intermediate results

Evaluation

Dataset: DBpedia 2015 (HDT on top of TPF server), 837M triples

Benchmark 3: **25** queries against DBpedia; basic graph patterns with up to 15 triple patterns.

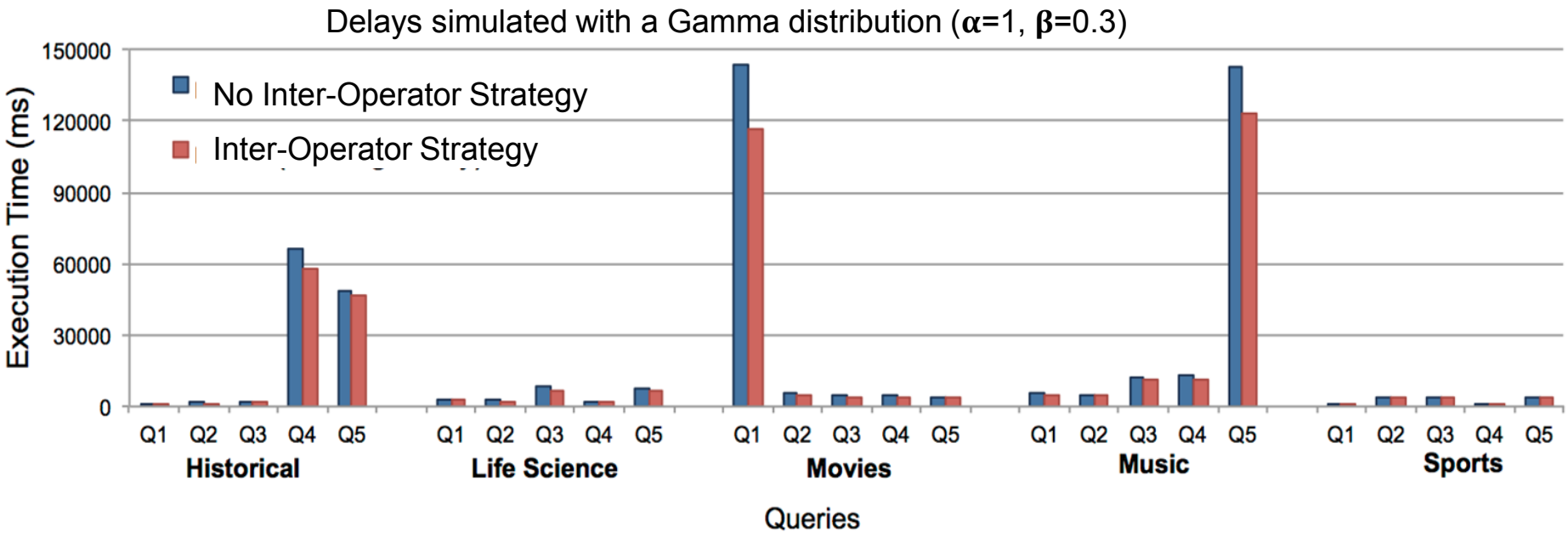
Metrics:

- Execution Time, ms

Compared tools:

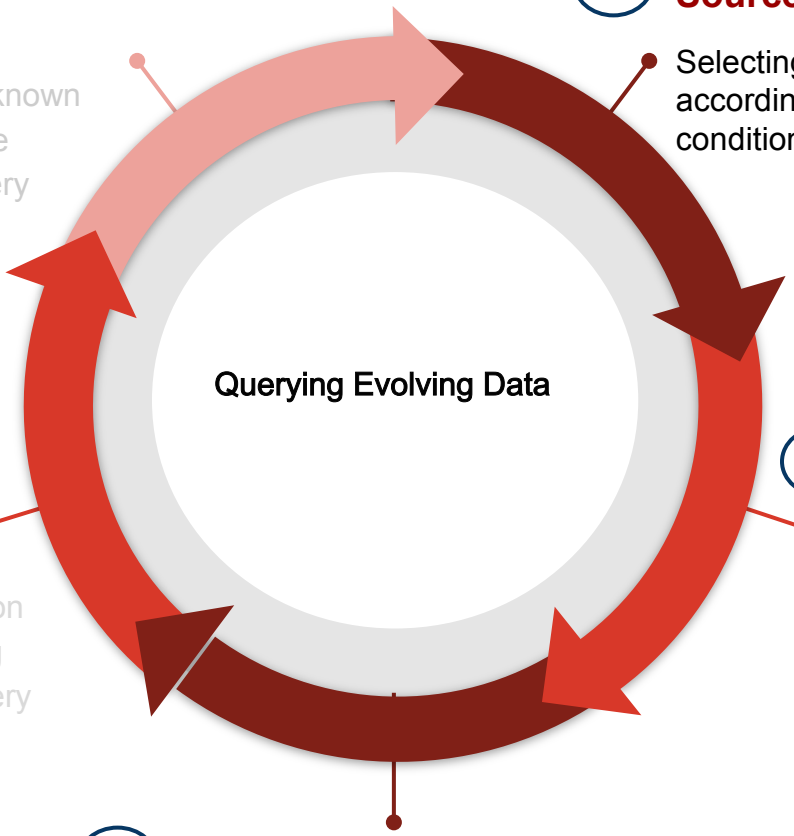
- **nLDE**: network of Linked Data Eddies
- **No Inter-Operator** Strategy

Evaluation



Adaptive query processing strategies are able to speed up query execution in presence of delays

Required Solutions to Support Evolution



1 **Source Evolution**

Selecting the sources according to their current conditions and availability

2 **Environment Evolution**

Executing queries according to current conditions of the environment

3 **Data Evolution**

Considering the status of the data, e.g., completeness, during the execution of the query

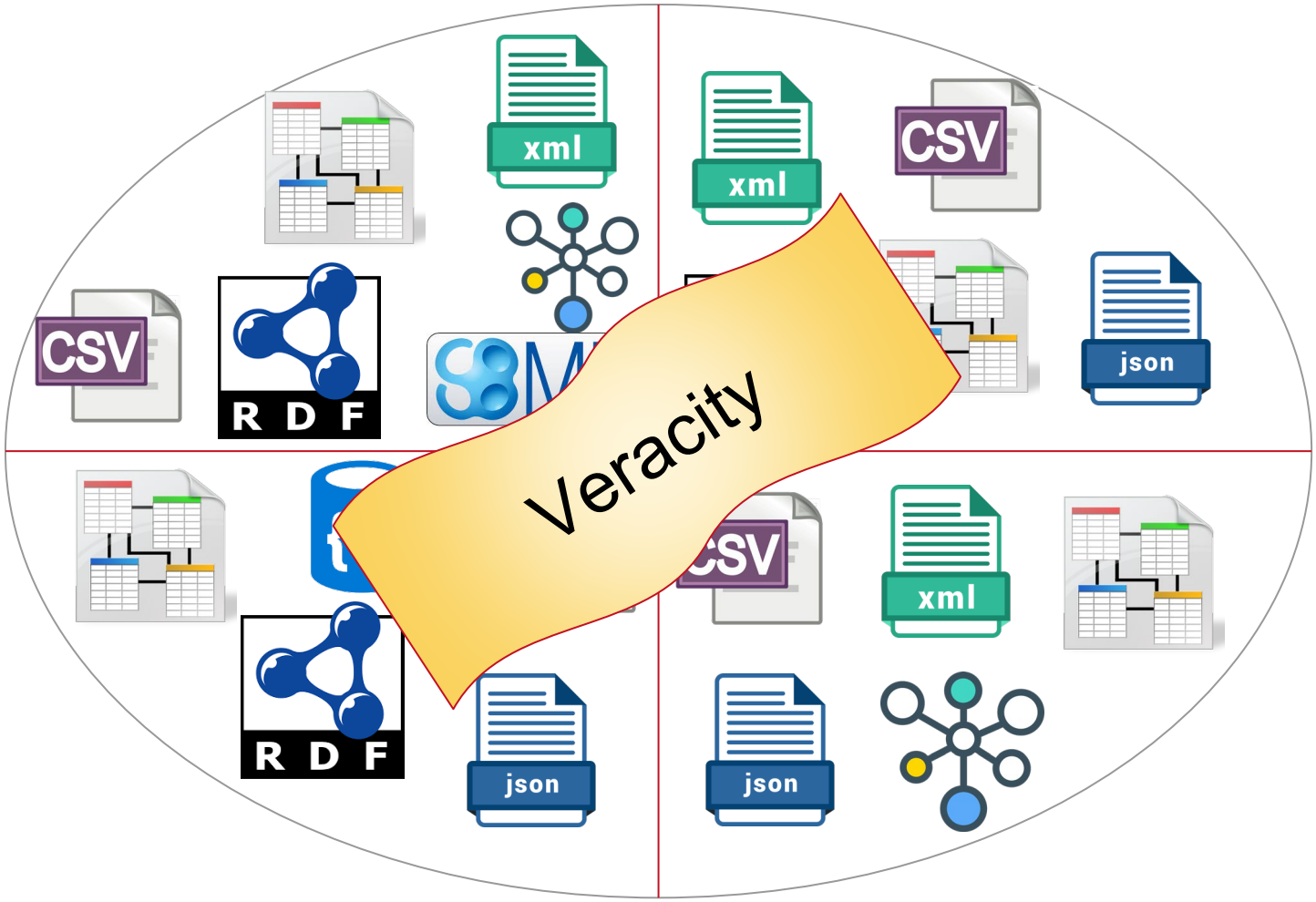
4 **Knowledge Evolution**

Considering the evolution of the knowledge during the execution of the query

5 **Knowledge Incompleteness**

Considering that unknown facts may need to be predicted during query execution

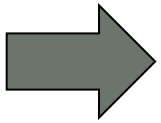
Impacting Data Complexity Dimensions



Data Changes….

Lung Cancer Biomarkers?

```
PREFIX iasis:<http://iasis/vocab/>
SELECT ?id  ?stage ?limit
 WHERE {
  ?bm  a  iasis:LungCancerBiomarker .
  ?id iasis:associated ?bm .
  ?bm iasis:associated ?obs .
  ?bm iasis:limit ?limit .
  ?bm iasis:stage ?stage
    }
```

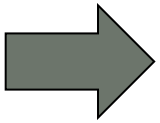


iasis:CYFRA21-1	iasis:II	iasis:50
iasis:CYFRA21-1	iasis:III	iasis:70
iasis:NSE	iasis:III	iasis:70

Data Changes….

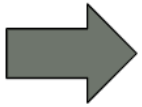
Lung Cancer Biomarkers?

```
PREFIX iasis:<http://iasis/vocab/>
SELECT ?id  ?stage ?limit
 WHERE {
  ?bm  a  iasis:LungCancerBiomarker .
  ?id iasis:associated ?bm .
  ?bm iasis:associated ?obs .
  ?bm iasis:limit ?limit .
  ?bm iasis:stage ?stage
    }
```



iasis:CYFRA-21-1	iasis:II	iasis:50
iasis:CYFRA-21-1	iasis:III	iasis:70
iasis:NSE	iasis:III	iasis:70

```
PREFIX iasis:<http://iasis/vocab/>
SELECT distinct ?id
 WHERE {
  ?bm  a  iasis:LungCancerBiomarker .
  ?id iasis:associated ?bm .
  }
```



- iasis:CYFRA-21-1
- iasis:CEA
- iasis:NSE
- iasis:CA-125

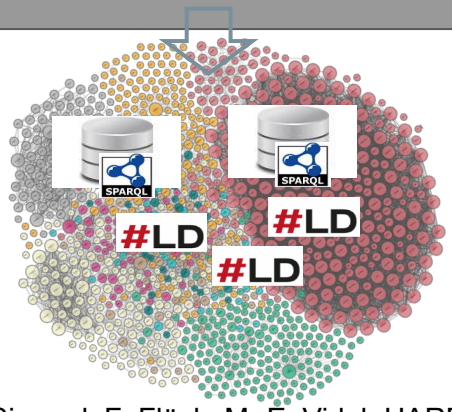
Hybrid Federated Query Engines

SPARQL Query **Q**

Source Selection & Query Decomposition

Query Optimizer

Execution Strategies



SPARQL Query **Q**

Source Selection & Query Decomposition

Query Optimizer

Hybrid Execution Strategies
Crowd Microtask Manager

Crowd

M. Acosta, E. Simperl, F. Flöck, M.-E. Vidal: HARE: A Hybrid SPARQL Enhancing answer completeness of SPARQL queries via crowdsourcing. J. Web Sem. 45: 41-62 (2017)

Hybrid Query Processing

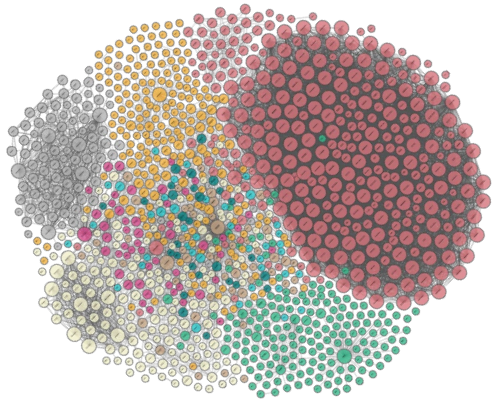
Lung Cancer Biomarkers?

```
PREFIX iasis:<http://iasis/vocab/>
SELECT ?id  ?stage ?limit
 WHERE {
  ?bm  a  iasis:LungCancerBiomarker .
  ?id iasis:associated ?bm .
  ?bm iasis:associated ?obs .
  ?bm iasis:limit ?limit .
  ?bm iasis:stage ?stage

  }
```

```
PREFIX iasis:<http://iasis/vocab/>
SELECT ?id
 WHERE {
  ?bm  a  iasis:LungCancerBiomarker .
   ?bm iasis:associated ?obs .
   ?id iasis:associated ?bm .
    ?bm iasis:stage ?stage
}
```

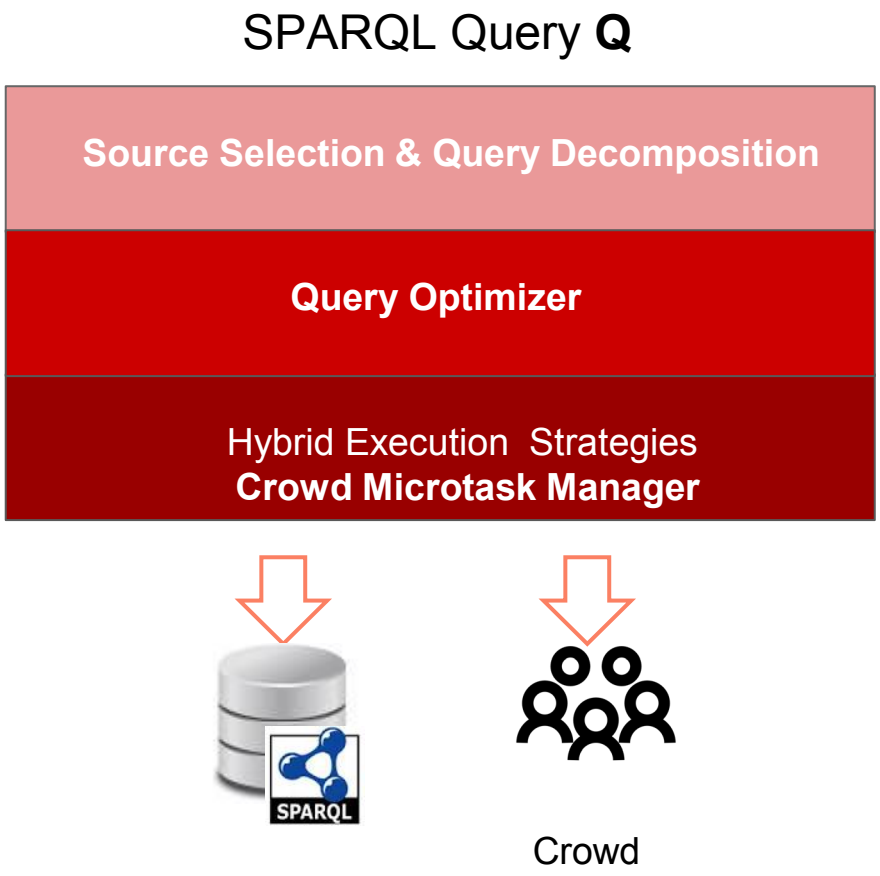
```
PREFIX iasis:<http://iasis/vocab/>
SELECT ?limit
 WHERE {
?bm iasis:limit ?limit .
  ?bm iasis:stage ?stage
  ?id iasis:associated ?bm .
```



Crowd

HARE: A Hybrid Query Engine

- **Completeness model** to estimate dataset completeness
- **Crowd knowledge bases** to capture crowd answers about missing data
- **Query engine** that combines **knowledge** in knowledge bases and **estimates** from the **completeness** model to **decompose** and plan sub-query execution
- **Microtask manager** that exploits **metadata** to **crowdsource** subqueries as **microtasks** and update the **knowledge bases** according to the **crowd answers**



M. Acosta, E. Simperl, F. Flöck, M.-E. Vidal: HARE: A Hybrid SPARQL Enhancing answer completeness of SPARQL queries via crowdsourcing. J. Web Sem. 45: 41-62 (2017)

HARE Microtasks

Metadata is utilized by the microtask manager to **automatically** generate **well-described** crowd tasks
Microtasks are submitted to **crowdsourcing platforms**, e.g., CrowdFlower or Mechanical Turk
Answers collected from the **crowd** are represented as **structured data**

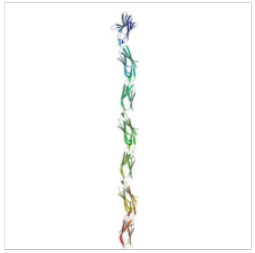
What is the **value** of the **Marker CEA** for **Lung Cancer Stage III**?

Search in Google: Carcinoembryonic antigen

Short Description: Carcinoembryonic antigen (CEA) describes a set of highly related glycoproteins involved in cell adhesion. CEA is normally produced in gastrointestinal tissue during fetal development, but the production stops before birth. Therefore, CEA is usually present only at very low levels in the blood of healthy adults. However, the serum levels are raised in some types of cancer, which means that it can be used as a tumor marker in clinical tests. Serum levels can also be elevated in heavy smokers.

Wikipedia Page: https://en.wikipedia.org/wiki/Carcinoembryonic_antigen

Picture:



Does the Marker CEA have a value for Lung Cancer Stage III?

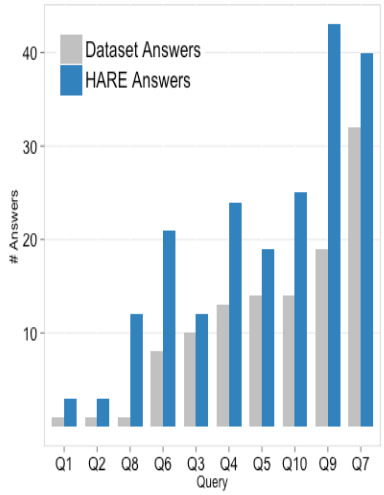
- Yes
- No
- I do not know

Experimental Study - Set Up

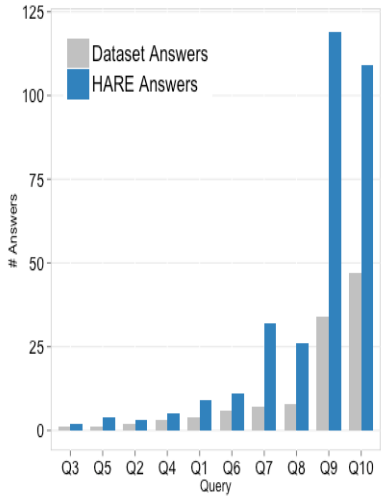
- **Benchmark:** 50 queries against DBpedia (v. 2014).
  - Ten queries in five different knowledge domains:
    History, Life Sciences, Movies, Music, and Sports.
- **Implementation details:**
  - HARE is implemented in Python 2.7.6,
  - The crowd is reached via CrowdFlower.
- **Crowdsourcing configuration:**
  - Four different RDF triples per task, 0.07 US$ per task.
  - At least three judgments were collected per task.
- Total RDF triple patterns crowdsourced: 502
- Total answers collected from the crowd: 1,609

Experimental Evaluation

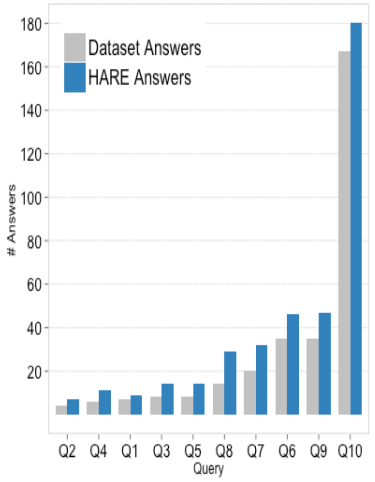
Sports



Music



Life Sciences

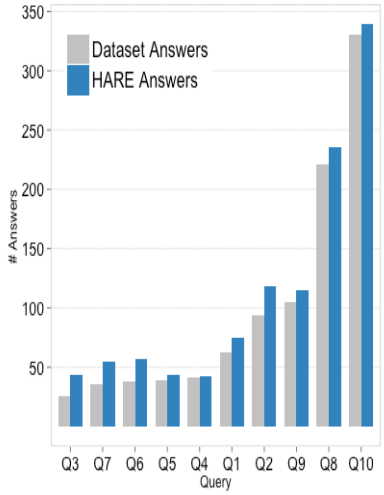


Crowdsourced answers and answers **collected** from DBpedia

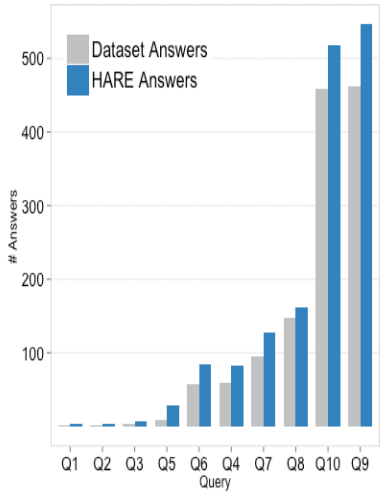
HARE **identifies** subqueries with **incomplete answers**

Hybrid query processing **enhances** query answer **completeness**

Movies

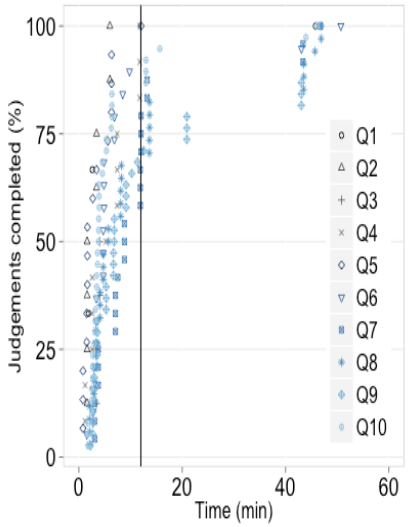


History

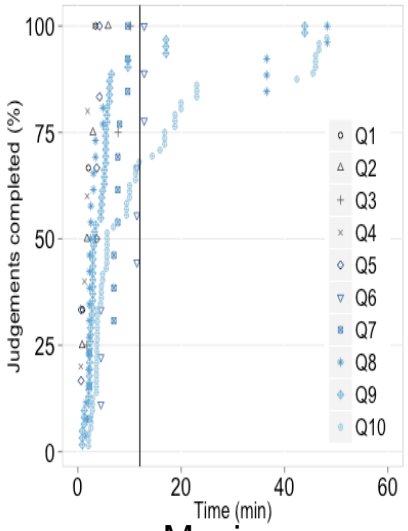


Experimental Evaluation

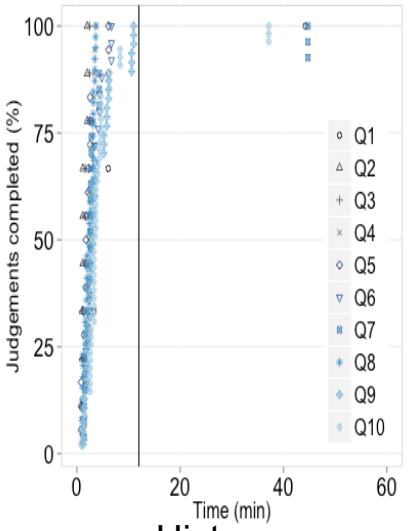
Sports



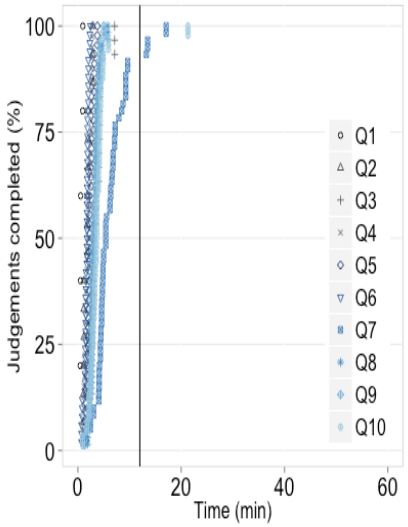
Music



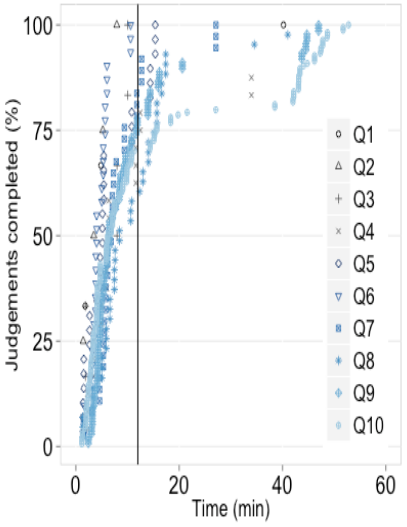
Life Sciences



Movies



History



HARE **is able** to produce more than 75% of the answers at the 12th minute

Experimental Evaluation

Precision

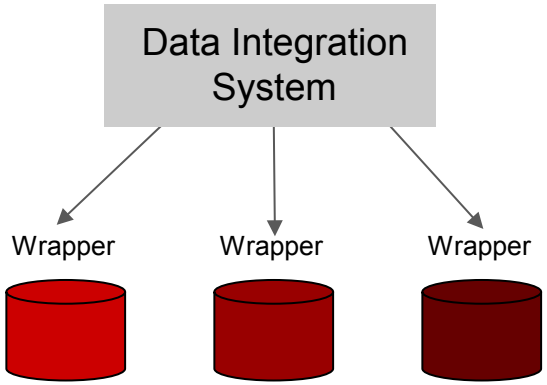
	Sports	Music	Life Sciences	Movies	History
Q1	1.00	1.00	0.67	0.88	1.00
Q2	1.00	1.00	1.00	0.96	1.00
Q3	1.00	1.00	0.89	0.79	0.67
Q4	0.55	0.67	1.00	1.00	0.96
Q5	0.86	0.67	1.00	1.00	0.95
Q6	0.69	0.83	1.00	1.00	0.96
Q7	1.00	0.63	0.71	1.00	0.57
Q8	1.00	0.67	0.88	0.94	0.72
Q9	0.46	0.73	1.00	1.00	0.64
Q10	0.92	0.49	1.00	1.00	0.95
Avg	**0.85**	**0.77**	**0.91**	**0.96**	**0.84**

Recall

	Sports	Music	Life Sciences	Movies	History
Q1	1.00	1.00	1.00	0.47	1.00
Q2	1.00	0.29	1.00	1.00	1.00
Q3	1.00	1.00	1.00	1.00	1.00
Q4	0.83	1.00	1.00	1.00	1.00
Q5	1.00	0.86	1.00	1.00	1.00
Q6	1.00	1.00	1.00	1.00	0.96
Q7	1.00	1.00	1.00	1.00	0.84
Q8	1.00	1.00	1.00	1.00	0.78
Q9	1.00	1.00	1.00	1.00	0.92
Q10	1.00	1.00	1.00	1.00	0.98
Avg	**0.98**	**0.91**	**1.00**	**0.95**	**0.95**

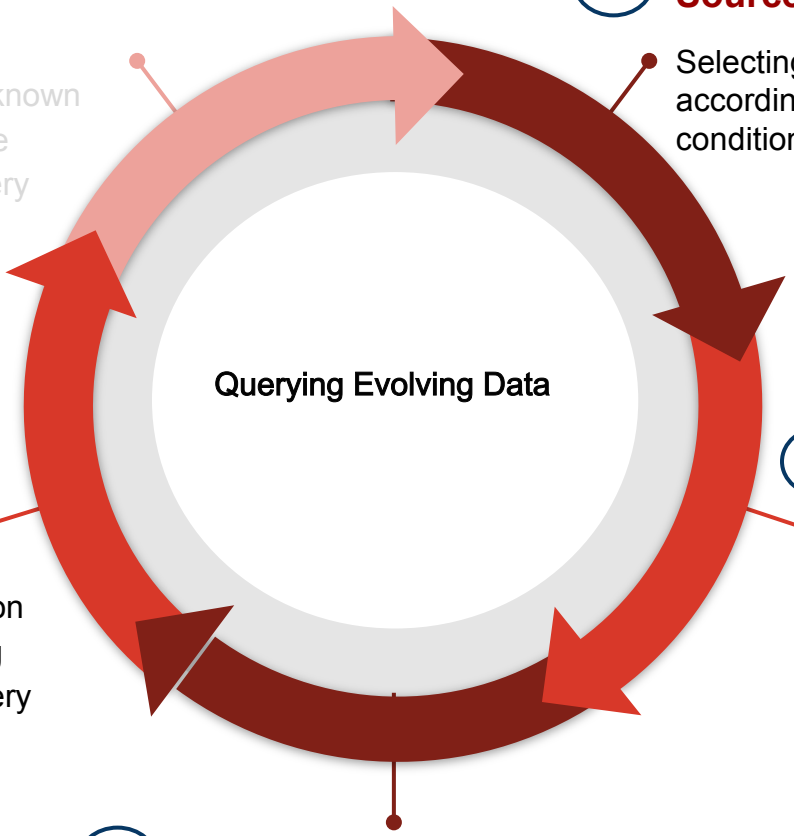
The crowd exhibits heterogeneous performance within domains.
This supports the importance of HARE triple-based approach.

Lessons Learned



- **Hybrid data integration** systems allow for the adaptation of the system to the conditions of the data sources
- **Hybrid data integration** systems enable the integration of **heterogeneous** data sources
- **Wisdom of the crowd** can contribute the evolution of the knowledge

Required Solutions to Support Evolution



5

Knowledge Incompleteness

Considering that unknown facts may need to be predicted during query execution

1

Source Evolution

Selecting the sources according to their current conditions and availability

4

Knowledge Evolution

Considering the evolution of the knowledge during the execution of the query

2

Environment Evolution

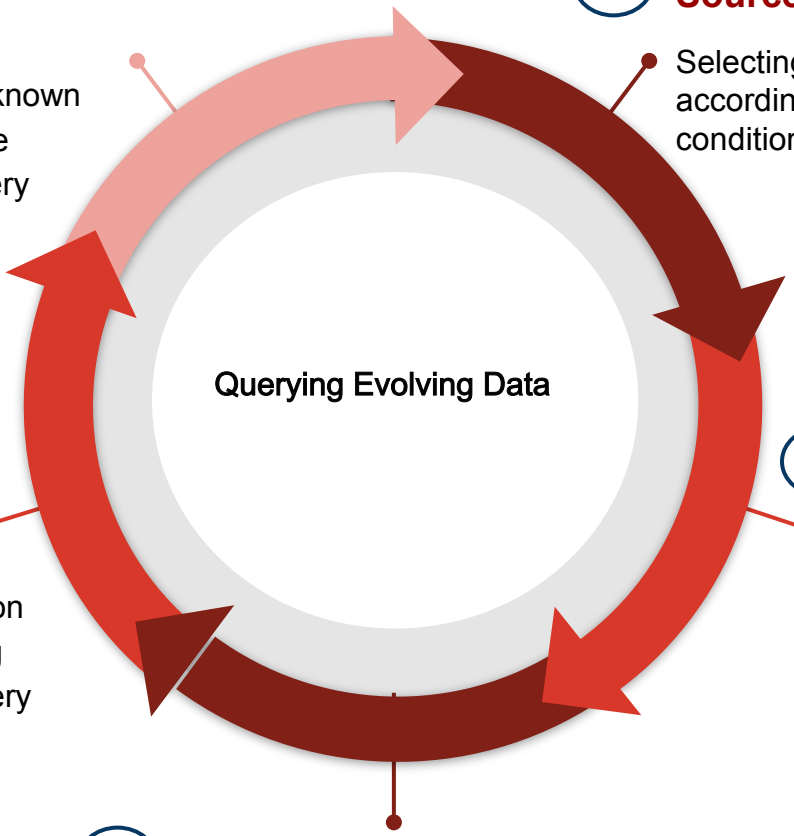
Executing queries according to current conditions of the environment

3

Data Evolution

Considering the status of the data, e.g., completeness, during the execution of the query

Required Solutions to Support Evolution



1 **Source Evolution**

Selecting the sources according to their current conditions and availability

2 **Environment Evolution**

Executing queries according to current conditions of the environment

3 **Data Evolution**

Considering the status of the data, e.g., completeness, during the execution of the query

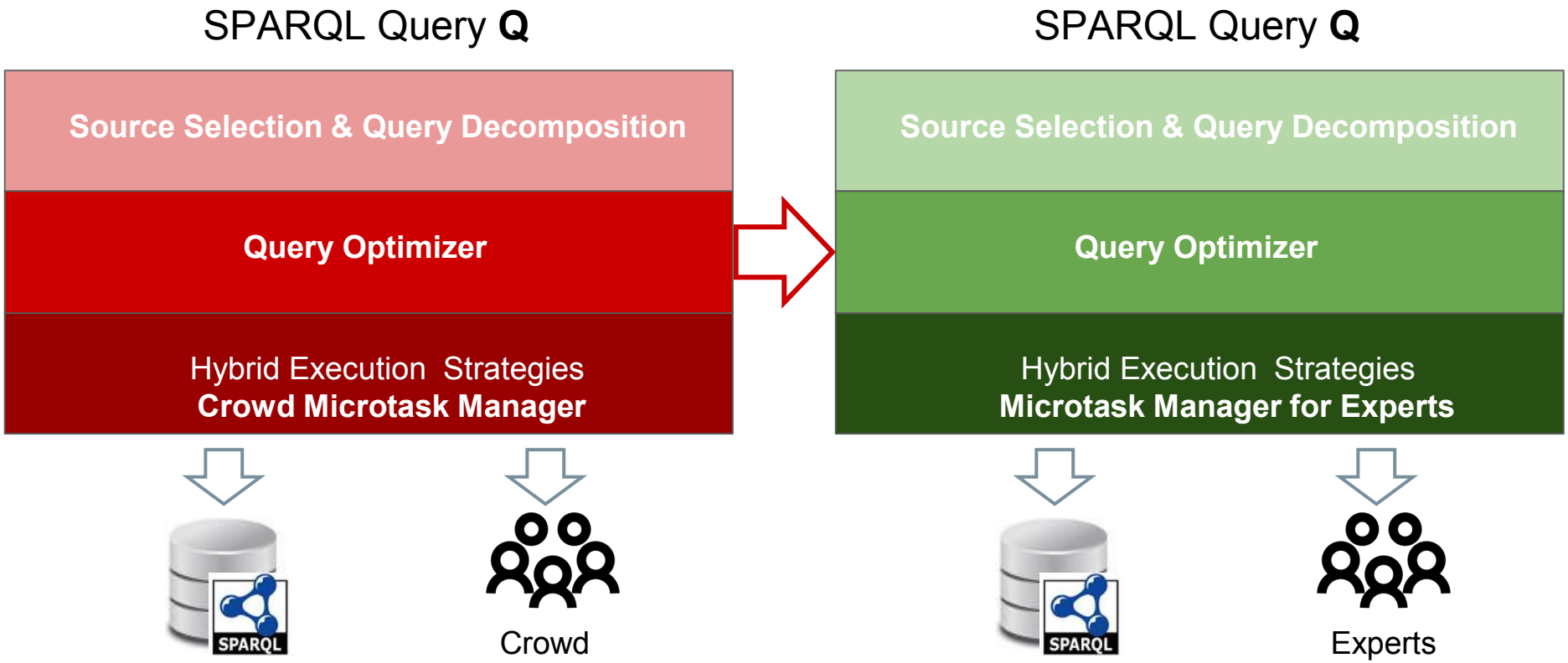
4 **Knowledge Evolution**

Considering the evolution of the knowledge during the execution of the query

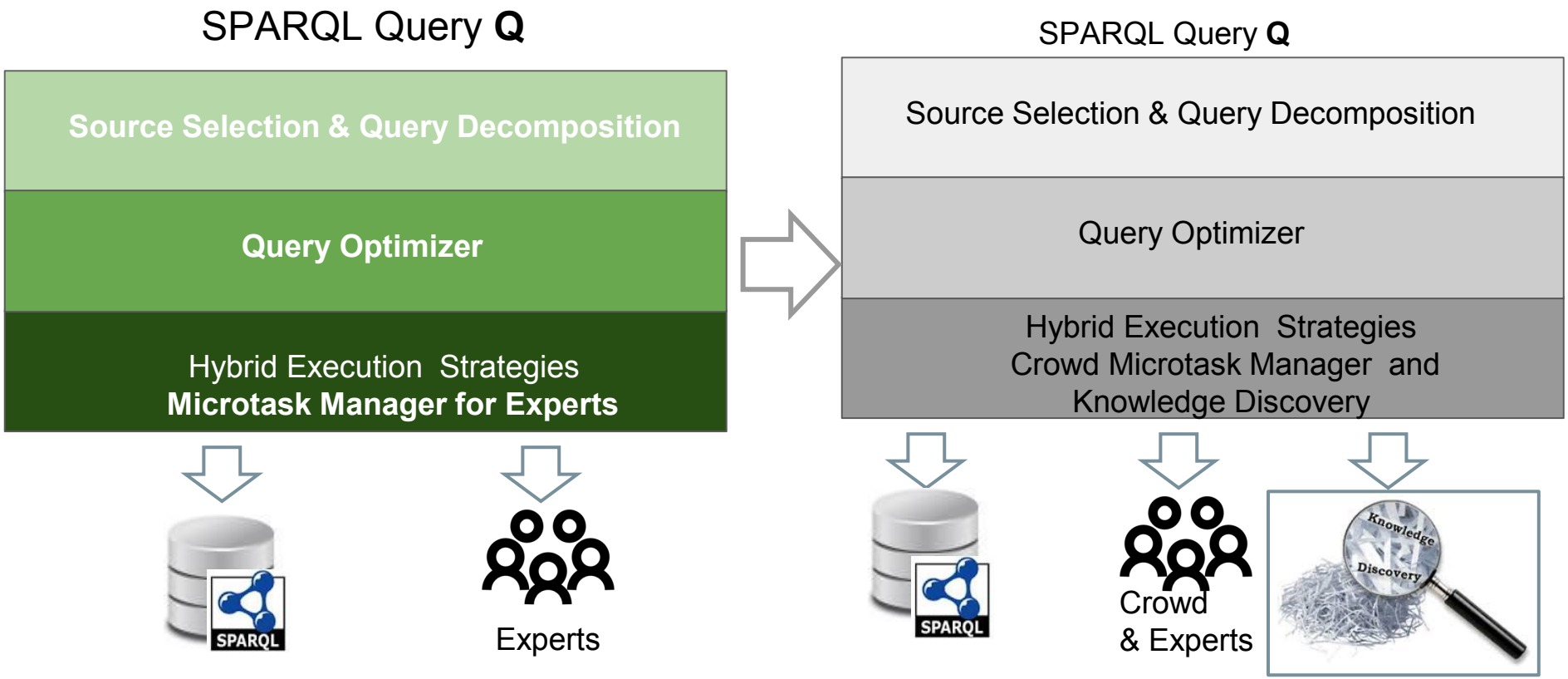
5 **Knowledge Incompleteness**

Considering that unknown facts may need to be predicted during query execution

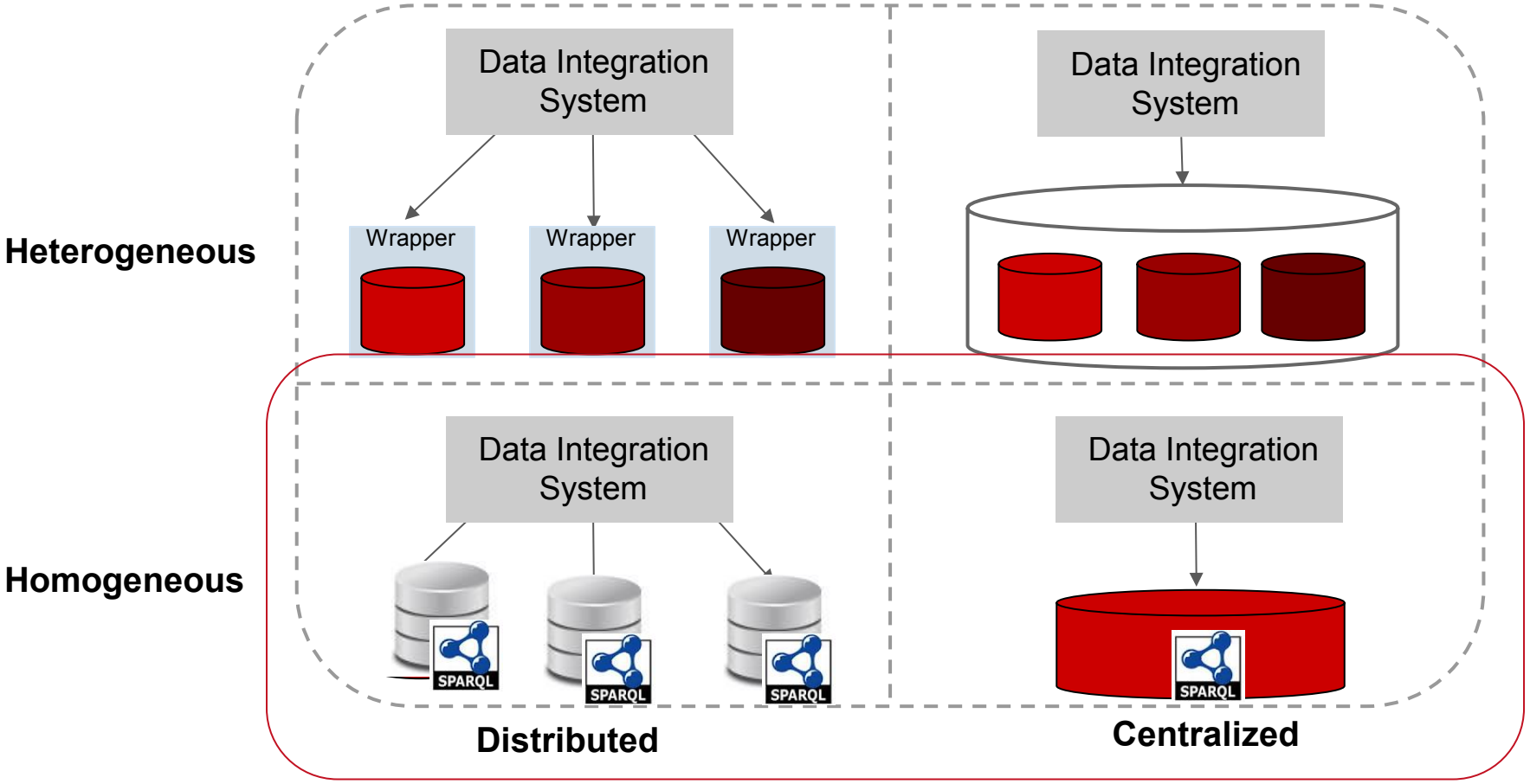
Future Hybrid Federated Query Engines



Future Hybrid Federated Engines

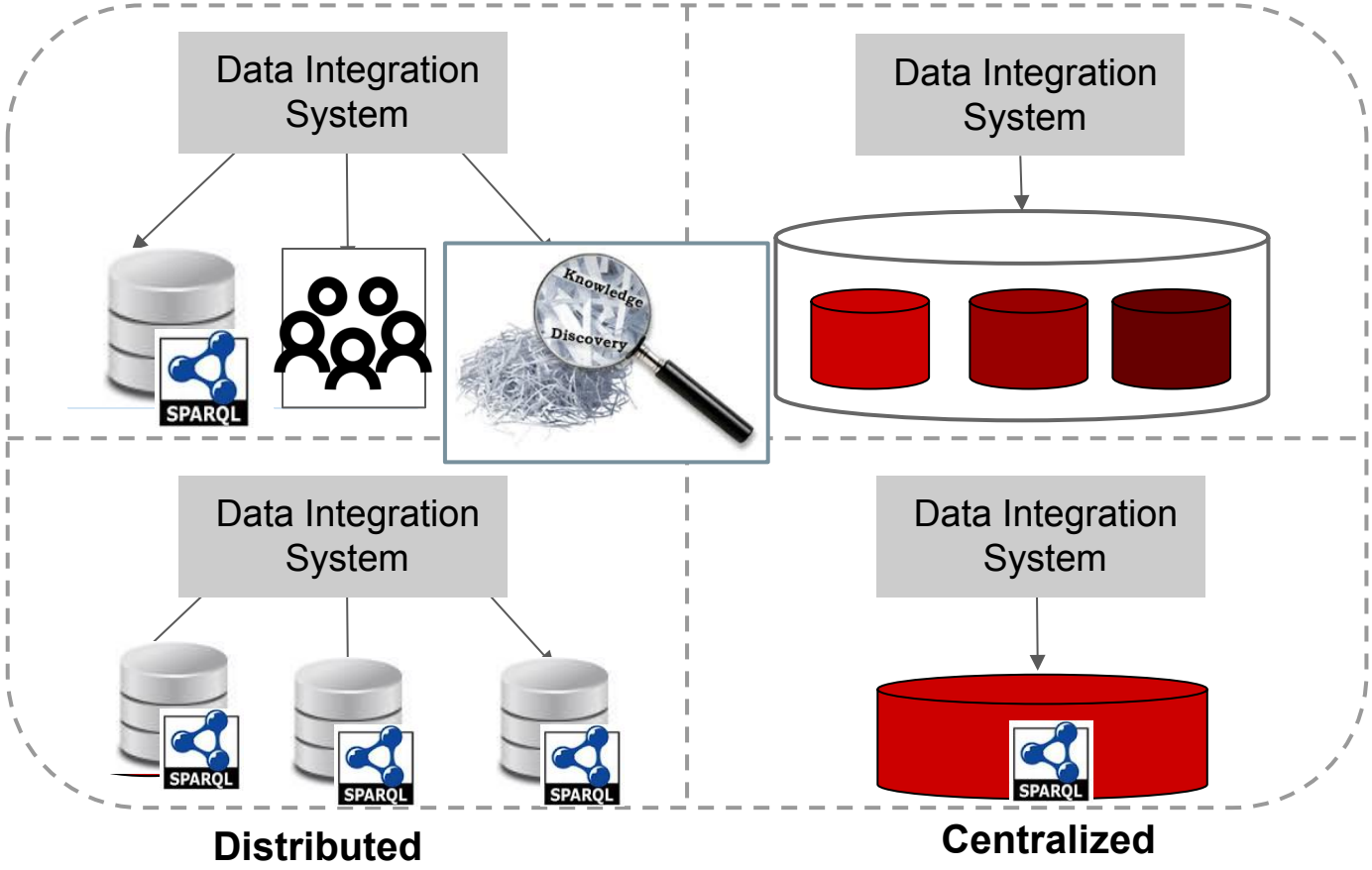


Data Integration Systems



Existing Approaches have focused on adaptive techniques to support SPARQL Query Processing over RDF Data Sources

Data Integration Systems



Future Approaches require to be focused on techniques to support data and knowledge evolution of RDF Data Sources

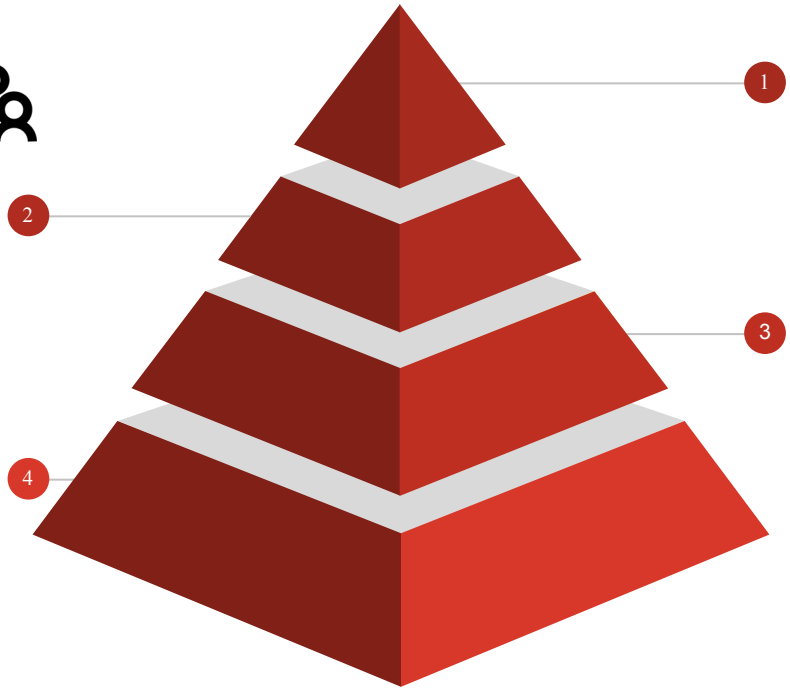
Future Hybrid Query Engines

Knowledge Prediction

Knowledge discovery techniques able to "***predict unknown facts"*** to complete RDF data sources..

Data Curation

Crowd based techniques able to exploit "***public domain"*** knowledge to complete RDF data sources.



Knowledge Curation

Crowd based techniques able to exploit "***specialized knowledge"*** to complete RDF data sources.

RDF Data Sources

Adaptive query processing techniques able to *adjust query execution* schedulers to current conditions of the data sources.


Leibniz
Universität
Hannover
Forschungszentrum · Research Center
L3S
LEIBNIZ INFORMATION CENTRE
FOR SCIENCE AND TECHNOLOGY
UNIVERSITY LIBRARY
TIB

Our Team at the Scientific Data Management Group

Prof.(Uni. Simon Bolivar)
Dr. Maria-Esther Vidal

Katja Bartel

PostDocs

Dr. Ingo Keck

Dr. Kemele Endris

Dr. Farah Karim

Senior Researcher

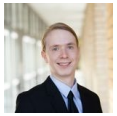
Akhilesh Vyas

Research Assistants

Samaneh Jozashoori

Ariam Rivas

Maria Isabel Castellanos



Philipp Rohde

Ahmad Sakor

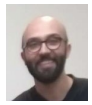
Master Student Assistants

Enrique Iglesias

Monica Figuera

Gabriela Ydler

Supreetha Hanasoge



Mohammad Torabinejad

William Scott

LEIBNIZ INFORMATION CENTRE
FOR SCIENCE AND TECHNOLOGY
UNIVERSITY LIBRARY

TIB

Thank you!
Questions

Contact
Maria-Esther Vidal
Maria.Vidal@tib.eu

Leibniz Association

References

[1] Maribel Acosta, Maria-Esther Vidal, Tomas Lampo, Julio Castillo, Edna Ruckhaus: ANAPSID: An Adaptive Query Processing Engine for SPARQL Endpoints. International Semantic Web Conference (2011)
[2] Maribel Acosta, Maria-Esther Vidal: Networks of Linked Data Eddies: An Adaptive Web Query Processing Engine for RDF Data. International Semantic Web Conference (2015)
[3] Olaf Görlitz, Steffen Staab: SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions. COLD (2011)
[4] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, Michael Schmidt: FedX: Optimization Techniques for Federated Query Processing on Linked Data. International Semantic Web Conference (2011)
[5] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, Maria-Esther Vidal: Decomposing federated queries in presence of replicated fragments. J. Web Sem. (2017)
[6] Gabriela Montoya, Hala Skaf-Molli, Pascal Molli, Maria-Esther Vidal: Federated SPARQL Queries Processing with Replicated Fragments. International Semantic Web Conference (2015)
[7] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, Pieter Colpaert: Triple Pattern Fragments: A low-cost knowledge graph interface for the Web. J. Web Sem.(2016)
[8] Maria-Esther Vidal, Simón Castillo, Maribel Acosta, Gabriela Montoya, Guillermo Palma: On the Selection of SPARQL Endpoints to Efficiently Execute Federated SPARQL Queries. Trans. Large-Scale Data- and Knowledge-Centered Systems 25: 109-149 (2016)

References

[9] Muhammad Saleem, Axel-Cyrille Ngonga Ngomo, Josiane Xavier Parreira, Helena F. Deus, Manfred Hauswirth: DAW: Duplicate-AWare Federated Query Processing over the Web of Data. International Semantic Web Conference (2013)
[10] Kemele M. Endris, Mikhail Galkin, Ioanna Lytra, Mohamed Nadjib Mami, Maria-Esther Vidal, Sören Auer: MULDER: Querying the Linked Data Web by Bridging RDF Molecule Templates. International Conference on Database and Expert Systems Applications (2017)
[11] Muhammad Saleem, Axel-Cyrille Ngonga Ngomo: HiBISCuS: Hypergraph-Based Source Selection for SPARQL Endpoint Federation. Extended Semantic Web Conference (2014)
[12] SemaGrow: Optimizing federated SPARQL queries Angelos Charalambidis, Antonis Troumpoukis and Stasinos Konstantopoulos In Proceedings of the 11th International Conference on Semantic Systems (SEMANTiCS 2015)
[13] Mikhail Galkin, Kemele M. Endris, Maribel Acosta, Diego Collarana, Maria-Esther Vidal, Sören Auer: SMJoin: A Multi-way Join Operator for SPARQL Queries. SEMANTICS 2017: 104-111
[14] Kemele M. Endris, Philipp D. Rohde, Maria-Esther Vidal, Sören Auer: Ontario: Federated Query Processing Against a Semantic Data Lake. DEXA 2019: 379-395
[15] Maribel Acosta, Maria-Esther Vidal, York Sure-Vetter: Diefficiency Metrics: Measuring the Continuous Efficiency of Query Processing Approaches. International Semantic Web Conference, 2017