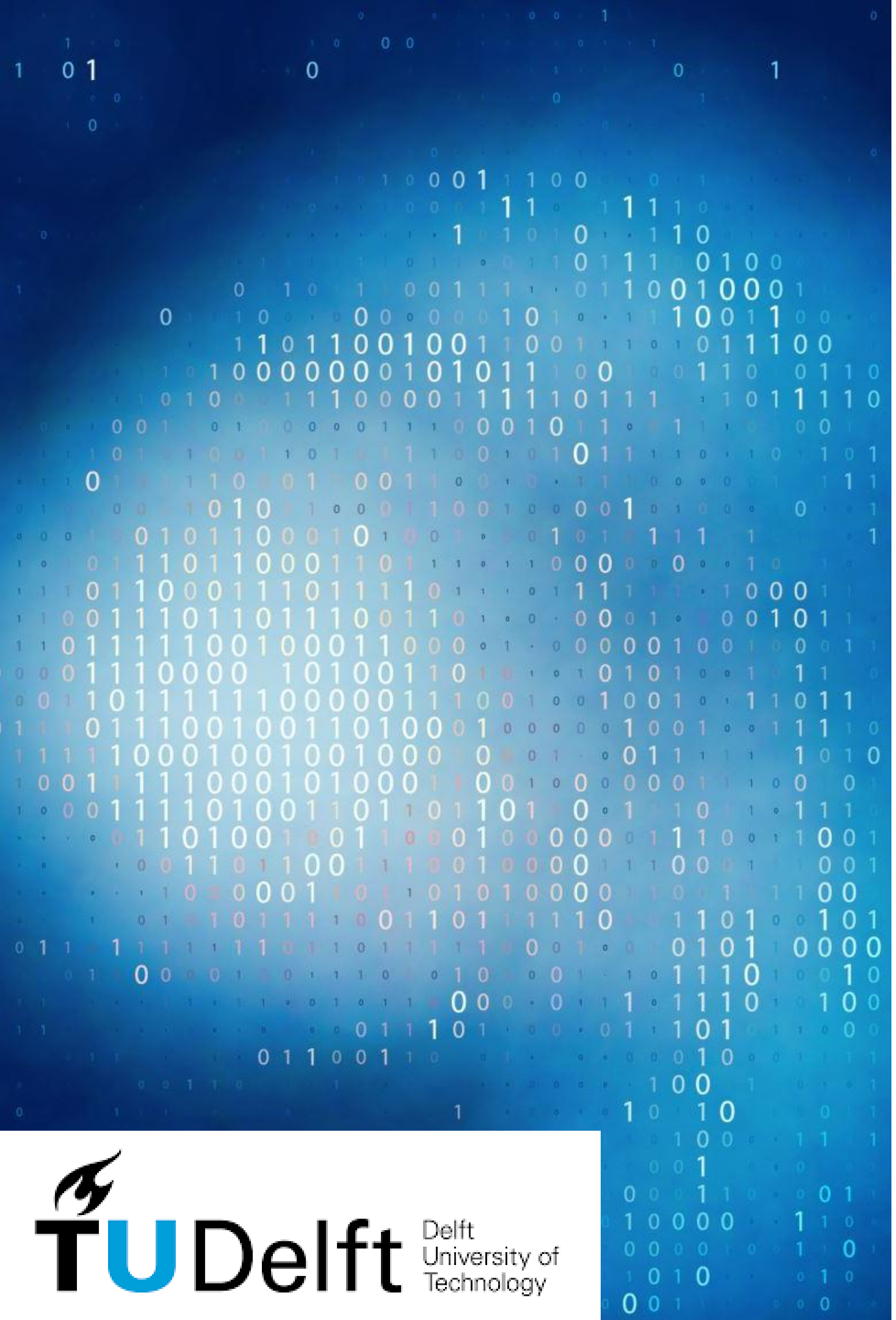


Data Integration in Data Lakes

Rihan Hai

Web Information Systems

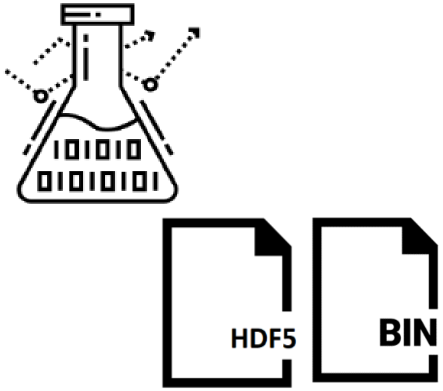
27.06.2022

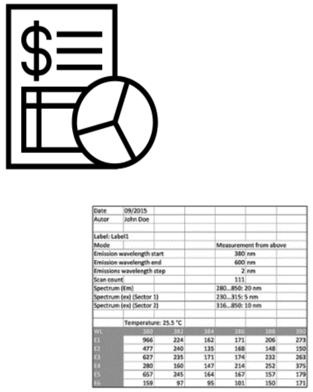
Data Lake survey

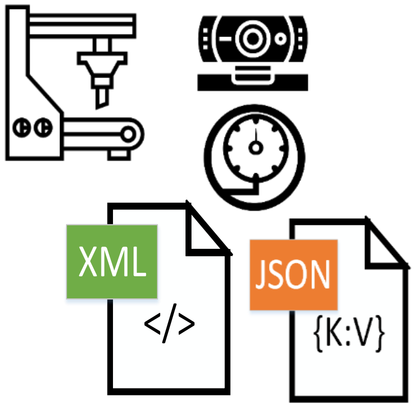
TU Delft Delft University of Technology

Problems of Big Data Management

Scientific Experiments	Business Transactions	Industrial Production	Mobile Applications



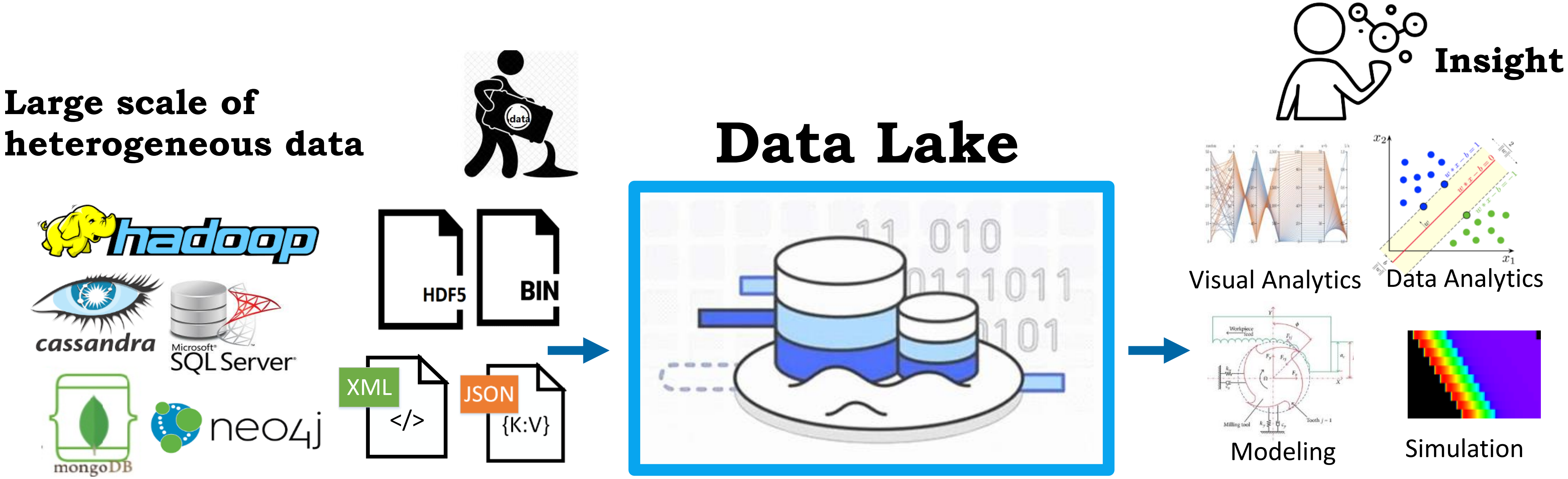




Data Lake

What Is a Data Lake?

*A **data lake** is a flexible, scalable **data storage and management system**, where **raw** data from **heterogeneous** sources can be ingested and stored in their **original** format, and later queried in an **on-the-fly** manner.*



Our contribution in data lake solution landscape

- **Metadata Extraction**
  - Structures: DATAMARAN [Gao et al., 2018]
  - Content and context: Skluma [Skluzacek et al., 2018]
  - **Schema and metadata properties: GEMMS [Quix et al., 2016b]**
- **Metadata Modeling**
  - **Generic metadata model [Quix et al., 2016b， Hai et al., 2019]**
  - Data Vault [Nogueira et al., 2018; Giebler et al., 2019]
  - Network-based metadata model [Diamantini et al., 2018a,b]
  - Enterprise knowledge graph [Fernandez et al., 2018]

- **Data Integratio**
  - **Mapping**
  - **Query re al., 2018**
  - Schema
- **Dataset Organi**
- **Discover Relat**

Table 1: Classification of data lake solutions based on functions

Tier	Functions	Systems
Ingestion	Metadata extraction	GEMMS [112]
		DATAMARAN [49]
		Skluma [129]
	Metadata modeling	GEMMS [61], [112]
		HANDLE [40]
		Data vault [54], [101]
		Diamantiniet al. [31], [32], [33]
		Aurum [45]
		Sawadogoet et al. [122]
Maintenance	Dataset organization	GOODS [64], [65]
		DS-Prox [4], [5], [6]
		KAYAK [85], [86]
		Nargesian et al. [98]
		Ronin [105]
		Juneau [143]
	Related dataset discovery	Aurum [45]
		Brackenbury et.al. [14]
		JOSIE [145]
		D^3L [13]
		Juneau [70], [142], [143]
		PEXESO [37]
		RNLIM [116]
		DLN [11]
	Data integration	Constance [58], [59], [60], [62]
	Metadata enrichment	CoreDB [8], [9]
		D^4 [104]
		DomainNet [79]
		Constance [61]
		GOODS [64], [65]
	Data cleaning	CLAMS [44]
		Constance [61]
		Song et al. [130]
	Schema evolution	Klettkeet et al. [75]
	Data provenance	IBM tool [134]
		Suriarachchi et al. [132]
		GOODS [64], [65]
		CoreDB [8], [9]
		Juneau [70], [142], [143]
Exploration	Query-driven data discovery	JOSIE [145]
		D^3L [13]
		Juneau [70], [142], [143]
		Aurum [45]
	Heterogeneous data querying	Constance [58], [62]
		CoreDB [8], [9]
		Ontario [41], [74]
		Squerall [89]

15]

- **Query-driven Discovery**
  - Explore related datasets with keyword queries/primitive-based query language [Nargesian et al., 2018; Brackenbury et al., 2018; Fernandez et al., 2018; Zhu et al., 2019]
- **Query Heterogeneous Data with a Unified Interface**
  - **Support multiple query languages** [Beheshti et al., 2017;2018; **Hai et al., 2016,2018b**]

nrichment

and named entities [Beheshti et al., 2017; 2018]

ırcing for descriptive metadata [Halevy et al., 2016a;b]

functional dependencies [Hai et al., 2019b]

lity

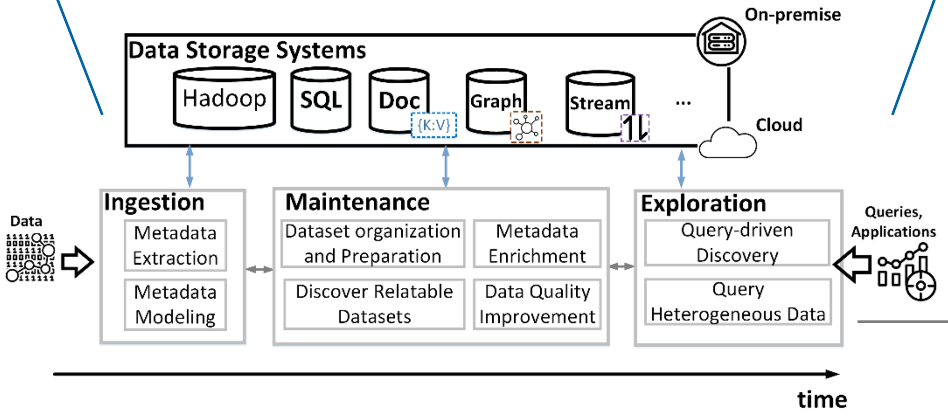
ata objects violating functional dependencies [Hai et al.,

RDF triples violating conditional denial constraints [Farid et al., 2016]

Our contribution in data lake solution landscape

- **Hadoop and File-based Storage Systems**
  - Hadoop Distributed File System [Stein et al., 2014; Boci et al., 2015]
  - Azure data lake store [Ramakrishnan et al., 2017]
- **Single Data Store**
  - RDBMS [Zhu et al., 2019]
  - NoSQL store [Walker et al., 2015]
- **Polystore Systems**
  - Google Dataset Search (GOODS) [Halevy et al., 2016b]
  - CoreDB [Beheshti et al., 2017]
  - **Constance [Hai et al., 2016]**

- **Metadata Extraction**
  - Structures: DATAMARAN [Gao et al., 2018]
  - Content and context: Skluma [Skluzacek et al., 2018]
  - **Schema and metadata properties: GEMMS [Quix et al., 2016b]**
- **Metadata Modeling**
  - **Generic metadata model [Quix et al., 2016b，Hai et al., 2019]**
  - Data Vault [Nogueira et al., 2018; Giebler et al., 2019]
  - Network-based metadata model [Diamantini et al., 2018a,b]
  - Enterprise knowledge graph [Fernandez et al., 2018]



- **Query-driven Discovery**
  - Explore related datasets with keyword queries/primitive-based query language [Nargesian et al., 2018; Brackenbury et al., 2018; Fernandez et al., 2018; Zhu et al., 2019]
- **Query Heterogeneous Data with a Unified Interface**
  - **Support multiple query languages** [Beheshti et al., 2017;2018; **Hai et al., 2016,2018b**]

- **Data Integration**
  - **Mapping generation: [Hai et al., 2018b, 2019a]**
  - **Query rewriting in Polystore based data lakes [Hai et al., 2018a]**
  - Schema matching [Alserafi et al., 2020]
- **Dataset Organization and Preparation**
- **Discover Relatable Datasets**

- **Metadata Enrichment**
  - Keywords and named entities [Beheshti et al., 2017; 2018]
  - Crowd-sourcing for descriptive metadata [Halevy et al., 2016a;b]
  - **Relaxed functional dependencies [Hai et al., 2019b]**
- **Data Quality**
  - **Detect data objects violating functional dependencies [Hai et al., 2019b]**
  - Examine RDF triples violating conditional denial constraints [Farid et al., 2016]

When is data **integration?**

Goal:

- Combine multiple heterogeneous data sources
- Provide a uniform query interface for heterogeneous data

UI

user

Query *q*

```
SELECT flightNumber
FROM  Flight
WHERE city1= 'Basel'  AND city2='Amsterdam'
```

Flight				
city1	**city2**	**flightNo**	**depTime**	**arrTime**

How to rewrite q to q1 and q2?

Subquery q1

DS1

Schedule		
flightNo	**depTime**	**arrTime**
B737	0600	0845
LH311	0830	1945

Subquery q2

DS2

Route		
source	**destination**	**FNo**
Basel	Amsterdam	B737
Munich	Tokyo	LH311

Motivation | Data Integration and Metadata Management

- **Schema Matching**
Discover the correspondences among source schemas
- **Schema Merging**
Resolve several related source schemas, and build the integrated schema
- **Schema Mapping**
Capture semantic relationships between the source schemas and the integrated/target schema
- **Entity linkage**
Discover records across different data sources, which represent the same entity
- **Data cleaning**
Detect and correct corrupt or inaccurate records
- **Query Reformulation**
Translate user queries posed on the global/target schema to a set of subqueries based on source schemas

Schedule		
flightNo	depTime	arrTime
B737	0600	0845
LH311	0830	1945

Route		
source	destination	FNo
Basel	Amsterdam	B737
Munich	Tokyo	LH311

Data Integration key operations

- **Schema Matching**

Discover the correspondences among source schemas

- **Schema Merging**

Resolve several related source schemas, and build the integrated schema

- **Schema Mapping**

Capture semantic relationships between the source schemas and the integrated/target schema

- **Entity linkage**

Discover records across different data sources, which represent the same entity

- **Data cleaning**

Detect and correct corrupt or inaccurate records

- **Query Reformulation**

Translate user queries posed on the global/target schema to a set of subqueries based on source schemas

Schedule

flightNo	depTime	arrTime
B737	0600	0845
LH311	0830	1945

Route

source	destination	FNo
Basel	Amsterdam	B737
Munich	Tokyo	LH311

Flight

city1	city2	flightNo	depTime	arrTime

Data Integration key operations

- **Schema Matching**
  Discover the correspondences among source schemas
- **Schema Merging**
  Resolve several related source schemas, and build the integrated schema
- **Schema Mapping**
  Capture semantic relationships between the source schemas and the integrated/target schema
- **Entity linkage**
  Discover records across different data sources, which represent the same entity
- **Data cleaning**
  Detect and correct corrupt or inaccurate records
- **Query Reformulation**
  Translate user queries posed on the global/target schema to a set of subqueries based on source schemas

Schedule		
flightNo	depTime	arrTime
B737	0600	0845
LH311	0830	1945

Route		
source	destination	FNo
Basel	Amsterdam	B737
Munich	Tokyo	LH311

Flight				
city1	city2	flightNo	depTime	arrTime

- Transformation: executable mapping constraint
  - Constructs target instances from source instances
  - E.g., SQL query, XSLT, C# program
- Mapping formalism: Tuple-generating dependency (TGD)

$$\forall c_1, c_2, d \ (Route(c_1, c_2, d) \rightarrow \exists a_1, a_2 \ (Flight(c_1, c_2, d, a_1, a_2)) \ )$$

Data Integration key operations

- **Schema Matching**
  Discover the correspondences among source schemas
- **Schema Merging**
  Resolve several related source schemas, and build the integrated schema
- **Schema Mapping**
  Capture semantic relationships between the source schemas and the integrated/target schema
- **Entity linkage**
  Discover records across different data sources, which represent the same entity
- **Data cleaning**
  Detect and correct corrupt or inaccurate records
- **Query Reformulation**
  Translate user queries posed on the global/target schema to a set of subqueries based on source schemas

Schedule		
flightNo	depTime	arrTime
B737	0600	0845
LH311	0830	1945

⇔

Route		
source	destination	FNo
Basel	Amsterdam	B737
Munich	Tokyo	LH311

Data Integration key operations

- **Schema Matching**

Discover the correspondences among source schemas

- **Schema Merging**

Resolve several related source schemas, and build the integrated schema

- **Schema Mapping**

Capture semantic relationships between the source schemas and the integrated/target schema

- **Entity linkage**

Discover records across different data sources, which represent the same entity

- **Data cleaning**

Detect and correct corrupt or inaccurate records

Schedule		
flightNo	**depTime**	**arrTime**
B737	0600	0845
LH311	0830	1945

Route		
source	**destination**	**FNo**
Basel	Amsterdam	B737?
Munich	Tokyo	LH311

- **Query Reformulation**

Translate user queries posed on the global/target schema to a set of subqueries based on source schemas

Data Integration key operations

- **Schema Matching**

Discover the correspondences among source schemas

- **Schema Merging**

Resolve several related source schemas, and build the integrated schema

- **Schema Mapping**

Capture semantic relationships between the source schemas and the integrated/target schema

- **Entity linkage**

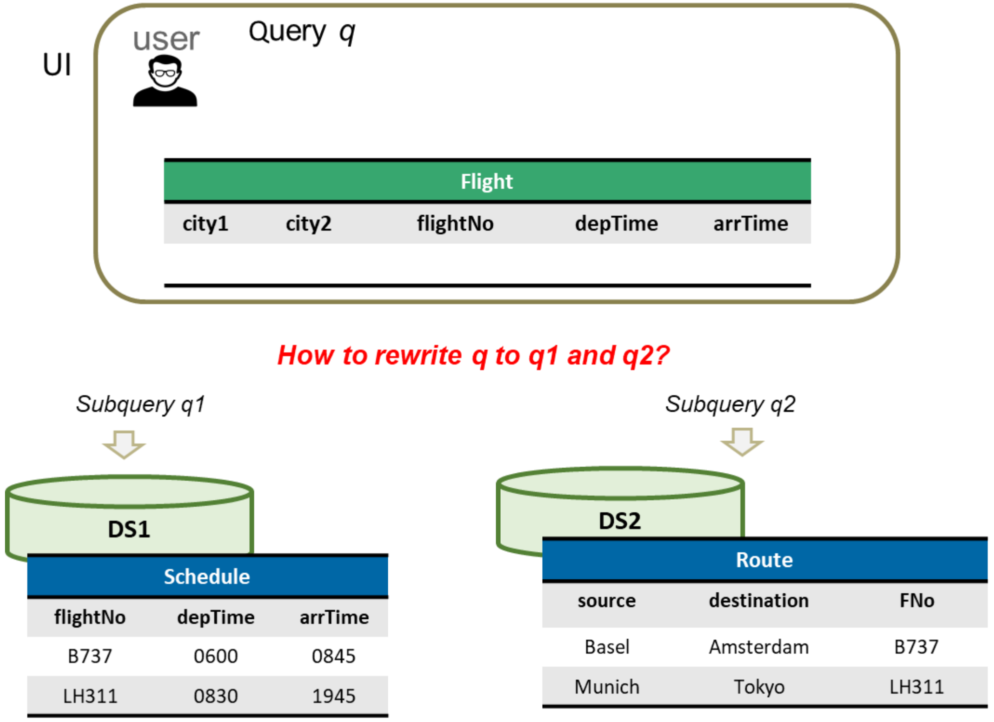
Discover records across different data sources, which represent the same entity

- **Data cleaning**

Detect and correct corrupt or inaccurate records

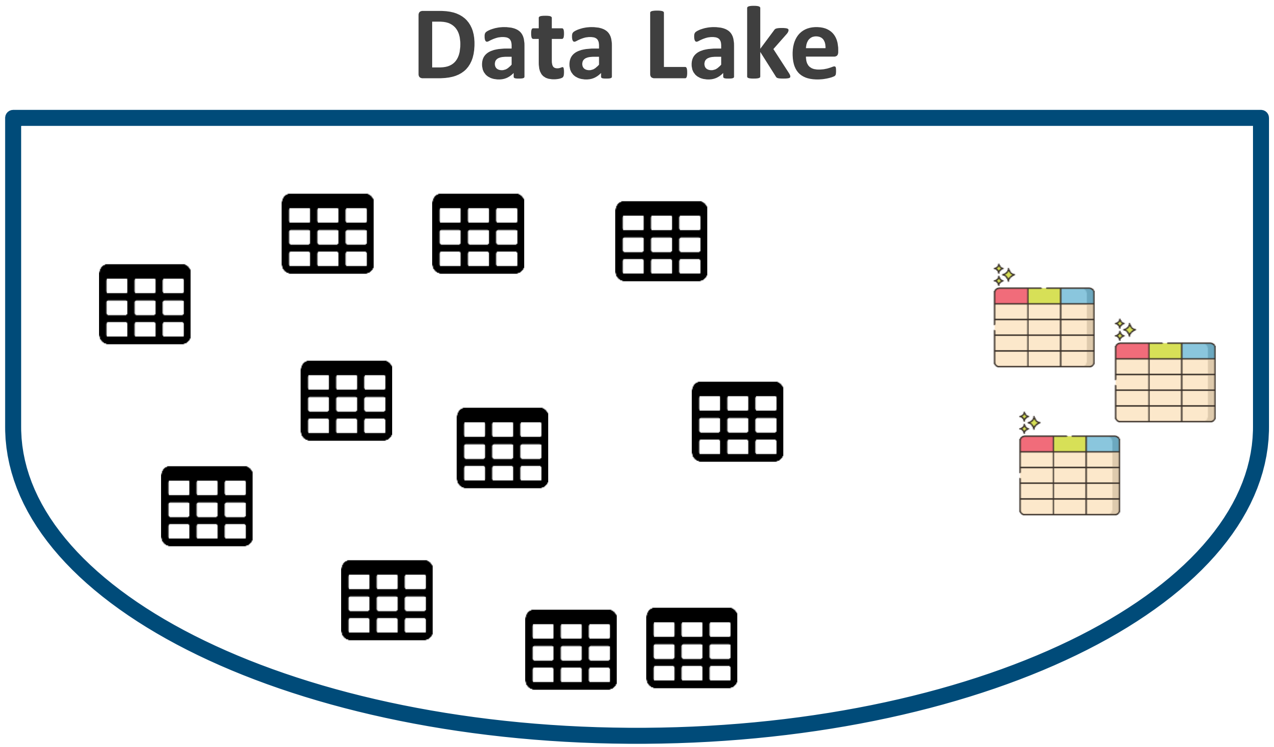
- **Query Reformulation**

Translate user queries posed on the global/target schema to a set of subqueries based on source schemas



Challenges: Data integration in data lakes

- Open data: massive datasets, not all needed
- Various data: heterogeneous data models (e.g., relational, semi-structured)
- Insufficient metadata: structureless datasets, missing semantics



Challenges: Data integration in data lakes

- Open data: massive datasets, not all needed
- Various data: heterogeneous data models (e.g., relational, semi-structured)
- Insufficient metadata: structureless datasets, missing semantics

Table 3: Comparison of related dataset discovery approaches in data lakes

Systems	**Relatedness criteria**	**Similarity metrics**	**Applied technique**
Aurum [45]	Instance value overlap Attribute name PK-FK candidate	Jaccard similarity (MinHash) Cosine similarity (TF-IDF)	Hypergraph
Brackenbury et.al. [14]	Instance value overlap Attribute name Semantics Descriptive metadata	Jaccard similarity (MinHash)	-
JOSIE [145]	Instance value overlap	Intersection size of sets	Inverted Index
D^3L [13]	Instance value overlap Attribute name Semantics Data value representation pattern (Numerical) data distribution	Jaccard similarity (MinHash) Cosine similarity (Random projections)	5–dim Euclidean space
Juneau [70], [142], [143]	Instance value overlap Domain overlap Attribute name Key constraint New attributes rate New instance rate Variable dependency Descriptive metadata Null Values	Jaccard similarity	Workflow graph Variable dependency graph
PEXESO [37]	(Textual) instance values	Any similarity function in a metric space	High-dimensional vectors Hierarchical grids Inverted Index
RNLIM [116]	Table name Attribute name Attribute data type Attribute value domain	–	BERT [30]
DLN [11]	Attribute name Instance values	Jaccard similarity Cosine similarity	Classification models

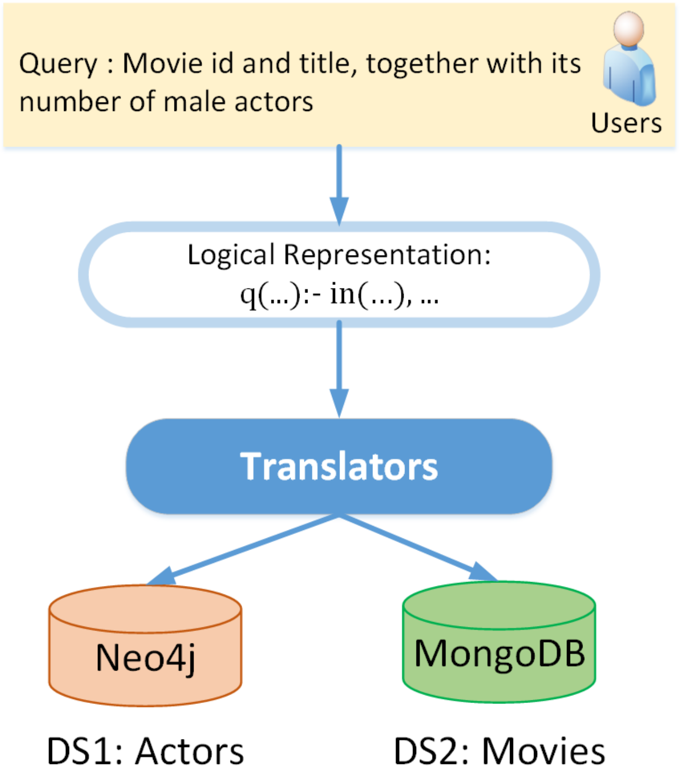
Challenges: Data integration in data lakes

- Open data: massive datasets, not all needed
- Various data: heterogeneous data models (e.g., relational, semi-structured)
- Insufficient metadata: structureless datasets, missing semantics

Query Rewriting in Data Lakes [Hai et al., ADBIS 2018]

Query rewriting for heterogeneous data

- User query might need data from multiple sources
  - Different source schemas and formats
  - Diverse query languages



Query Processing over Diverse Polystore [Hai et al., ADBIS 2018]

- Reformulate and process subqueries
  - 1 Rewrite an input query on integrated schema to logical rules in predicates
  - 2 3 Generate subqueries based on source schemas and stores
  - 4 Execute subqueries on each data store
  - 5 6 Retrieve the query results, and create the final integrated results
- Optimize query execution
  - 3 Push down selection predicates to the data sources
  - 5 Reduce the amount of data that has to be returned and loaded into Spark

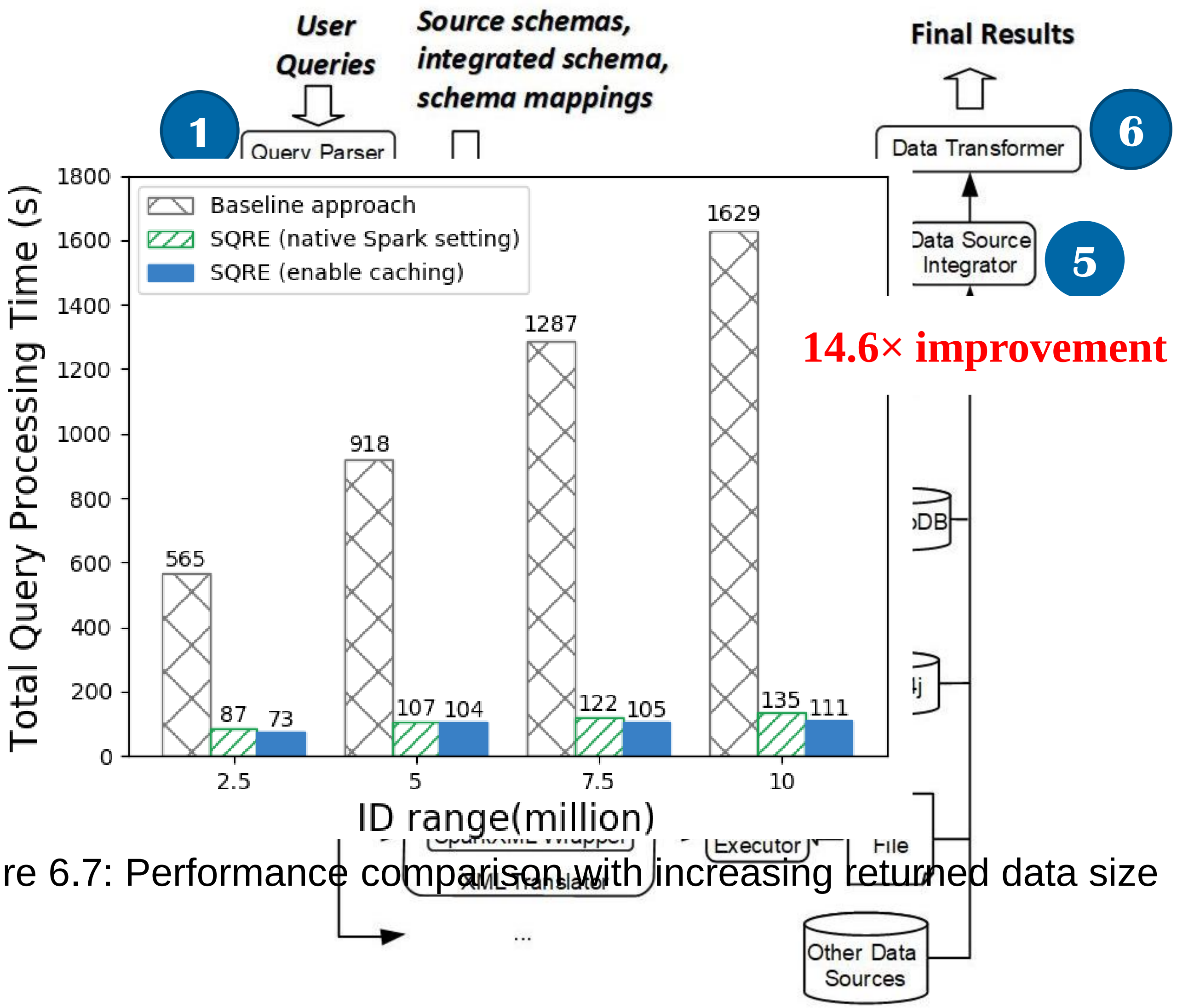


Figure 6.7: Performance comparison with increasing returned data size

Challenges: Data integration in data lakes

- Open data: massive datasets, not all needed
- Various data: heterogeneous data models (e.g., relational, semi-structured)
- Insufficient metadata: structureless datasets, missing semantics

Table 2: Comparison of DAG-based dataset organization approaches in Sec. 6.1.3

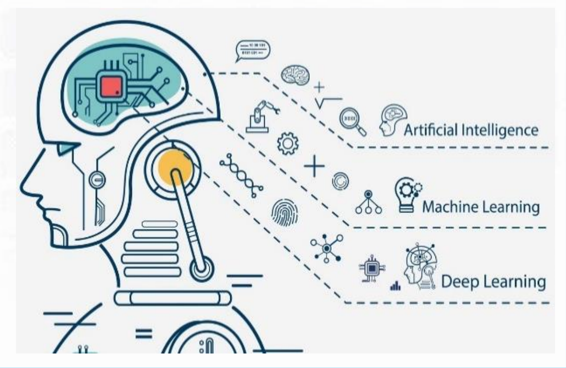
System	*KAYAK [85], [86] (pipeline)*	*KAYAK [85], [86] (task dependency)*	*Nargesian et al. [98]*	*Juneau [143] (variable dependency)*
Function	Represent the primitives of a data preparation pipeline	Enforce correct execution sequence of tasks while parallelization	Semantic navigation	Measure table relatedness w.r.t. notebook workflow
Node	Primitives	Atomic tasks for data preparation operations	Sets of attributes	Notebook variables
Edge	Sequential execution order of two primitives	Sequential execution order of two tasks	Containment relationships	Notebook functions (as edge labels)
Edge direction	From the previous primitive to the subsequent primitive	From the previous task to the subsequent task	From the superset to the subset	From the input variable of the function to the output variable

Future work: data lake for AI

Data Lake (2015-2020)

RWTH AACHEN UNIVERSITY

Intelligent, scalable Data Lake (2021-)



TU Delft Delft University of Technology

Christoph Quix

Matthias Jarke

Rihan Hai

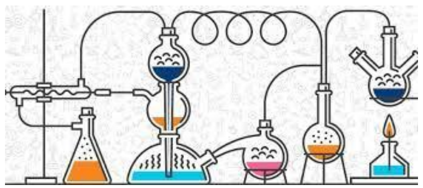
Asterios Katsifodimos

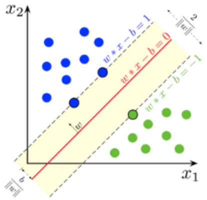
Amalur: a data science platform

Use cases:

Healthcare

Chemistry

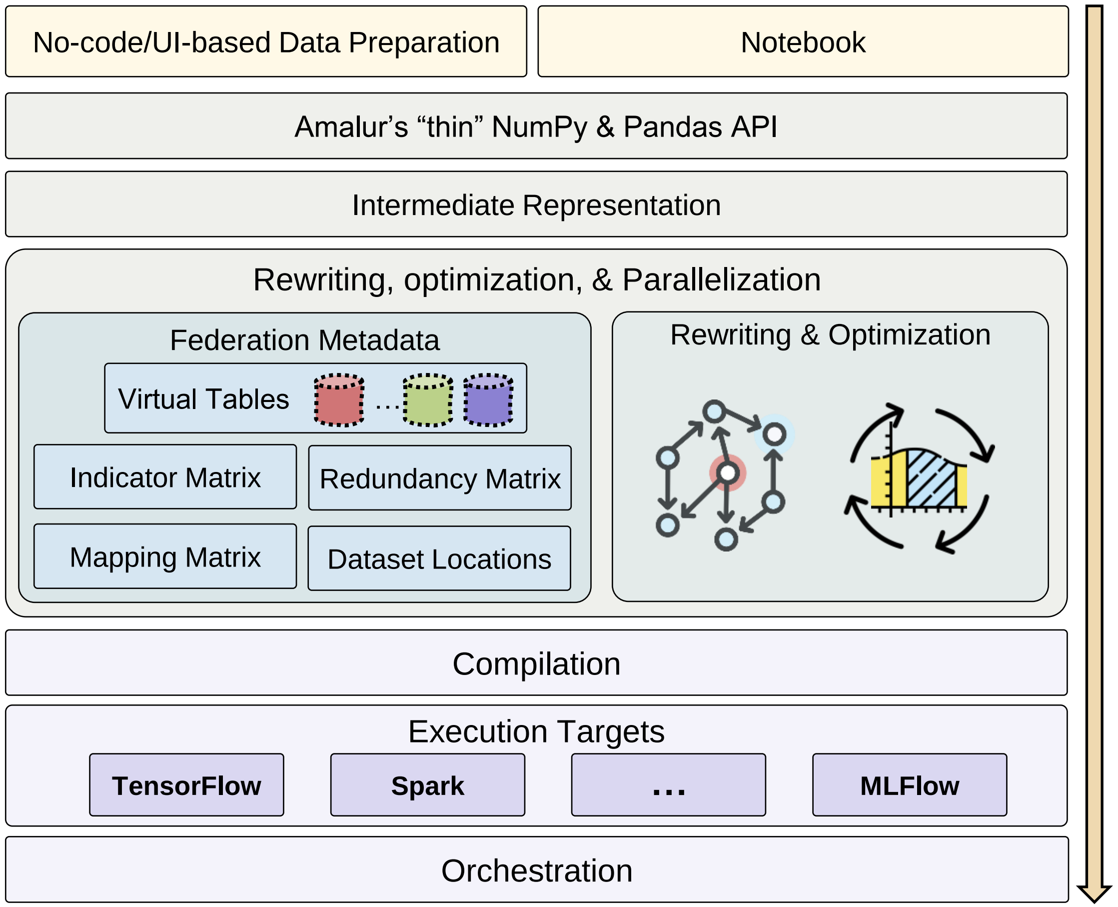




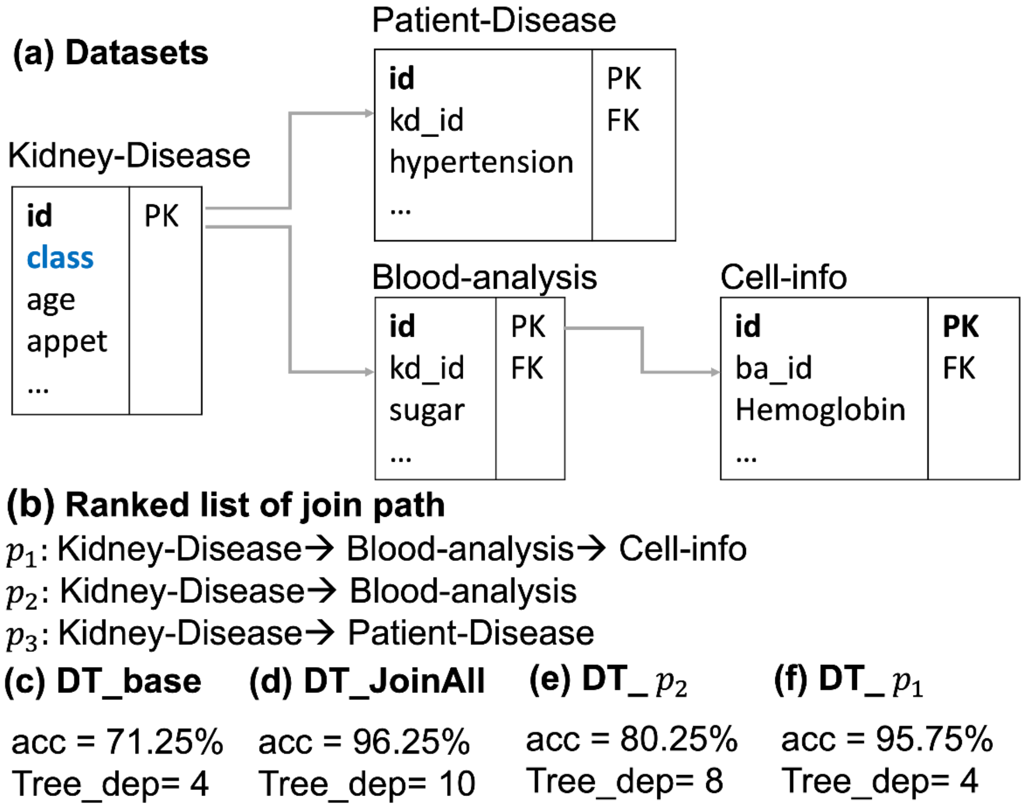
Data Analytics

Insight

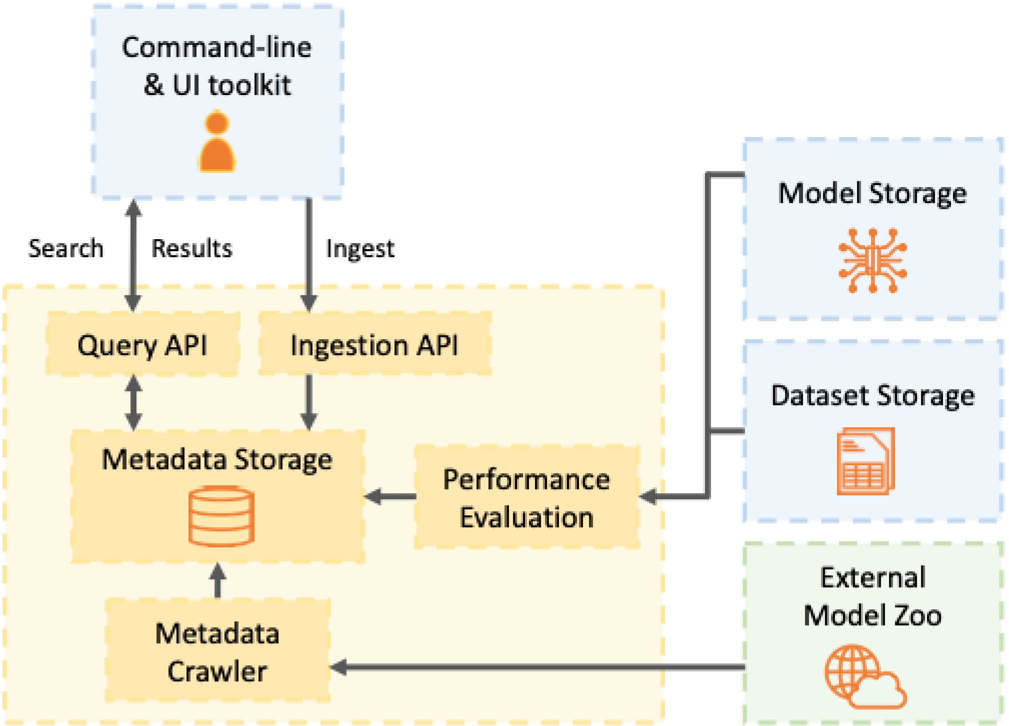
Amalur



Early results so far



Andra Ionescu, et al. "Join Path Based Data Augmentation for Decision Trees." ICDE Workshops. 2022.



Ziyu Li, et al. "Metadata Representations for Queryable ML Model Zoos." ICML Workshops. 2022.

Conclusion

- Data integration is a difficult problem, even more difficult in data lakes
  - Data structured with various schemas
  - Heterogeneous data models
  - Diverse query languages and systems

- New challenges, new technologies
  - Related dataset discovery
  - Dataset organization
  - Metadata extraction/dataset profiling
  - Entity resolution using ML/DL
  - Data cleaning for ML
  - Factorized Databases
  - Federated learning
  - …