



University
of Basel

Polypheny-DB*: Towards a Distributed and Self-Adaptive Polystore

Heiko Schuldt (heiko.schuldt@unibas.ch)

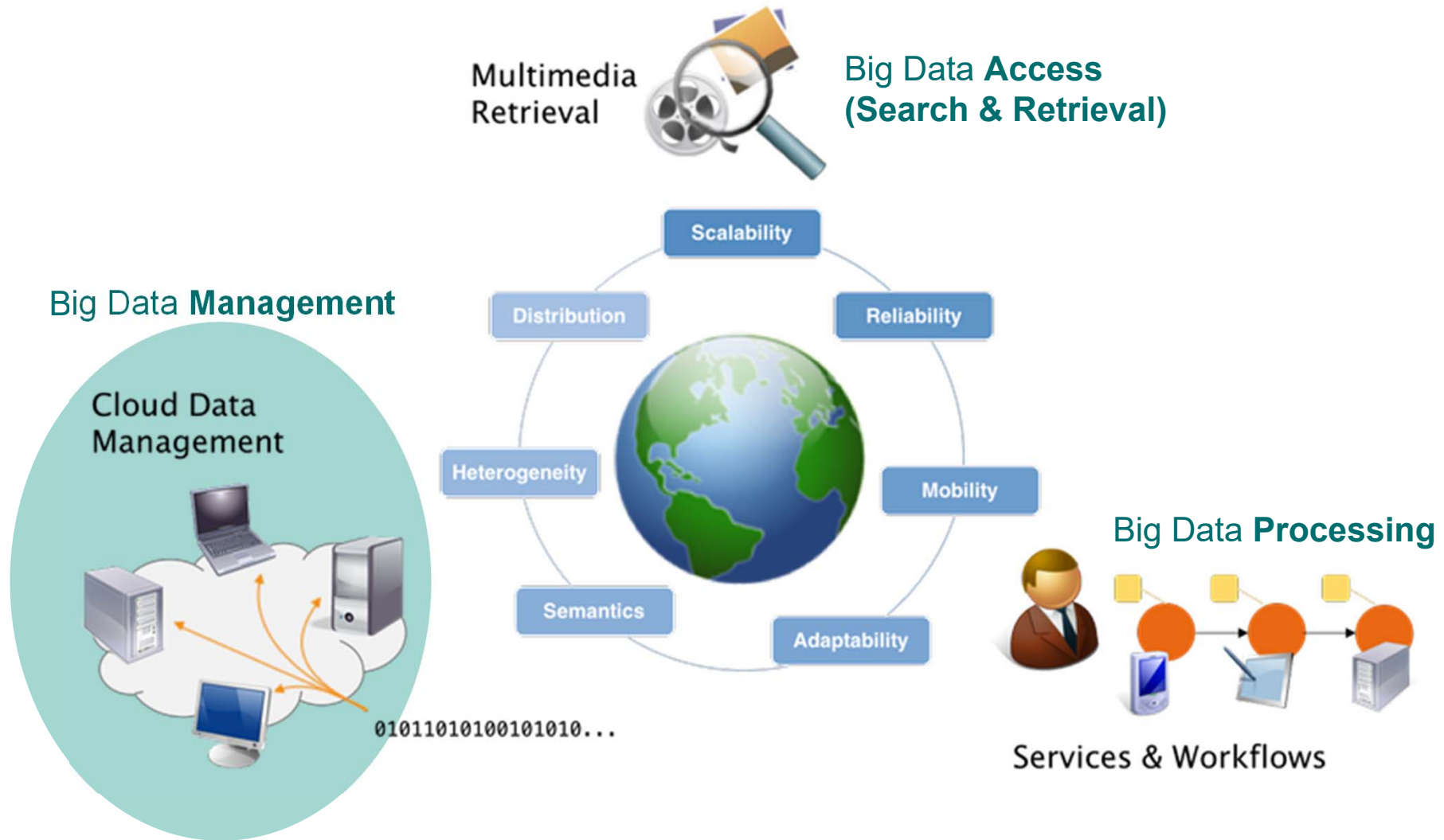
<https://dbis.dmi.unibas.ch/>

** “Polypheny-DB” is derived from **Polyphenism**, a concept from genetics which means that multiple, discrete phenotypes can arise from a single genotype as a result of differing environmental conditions.
(<https://en.wikipedia.org/wiki/Polyphenism>).*



Polypheny-DB

DBIS Research Group



Sample Scenario

- *Gavel*: a fictitious auction house
 - Runs millions of auctions per day
 - Servers deployed around the globe
- Requirements
 - Data availability
 - Performance and latency
 - Different query types
 - Minimal costs
 - GDPR compliance, privacy
 - ...



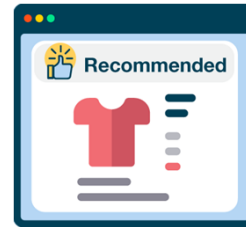
GAVEL

Gavel: Different Query Types and Data Models



Transactional workload (OLTP)

- Relational data model
- SQL Queries



Product recommendations

- Graph data model
- OpenCypher queries



Business analytics

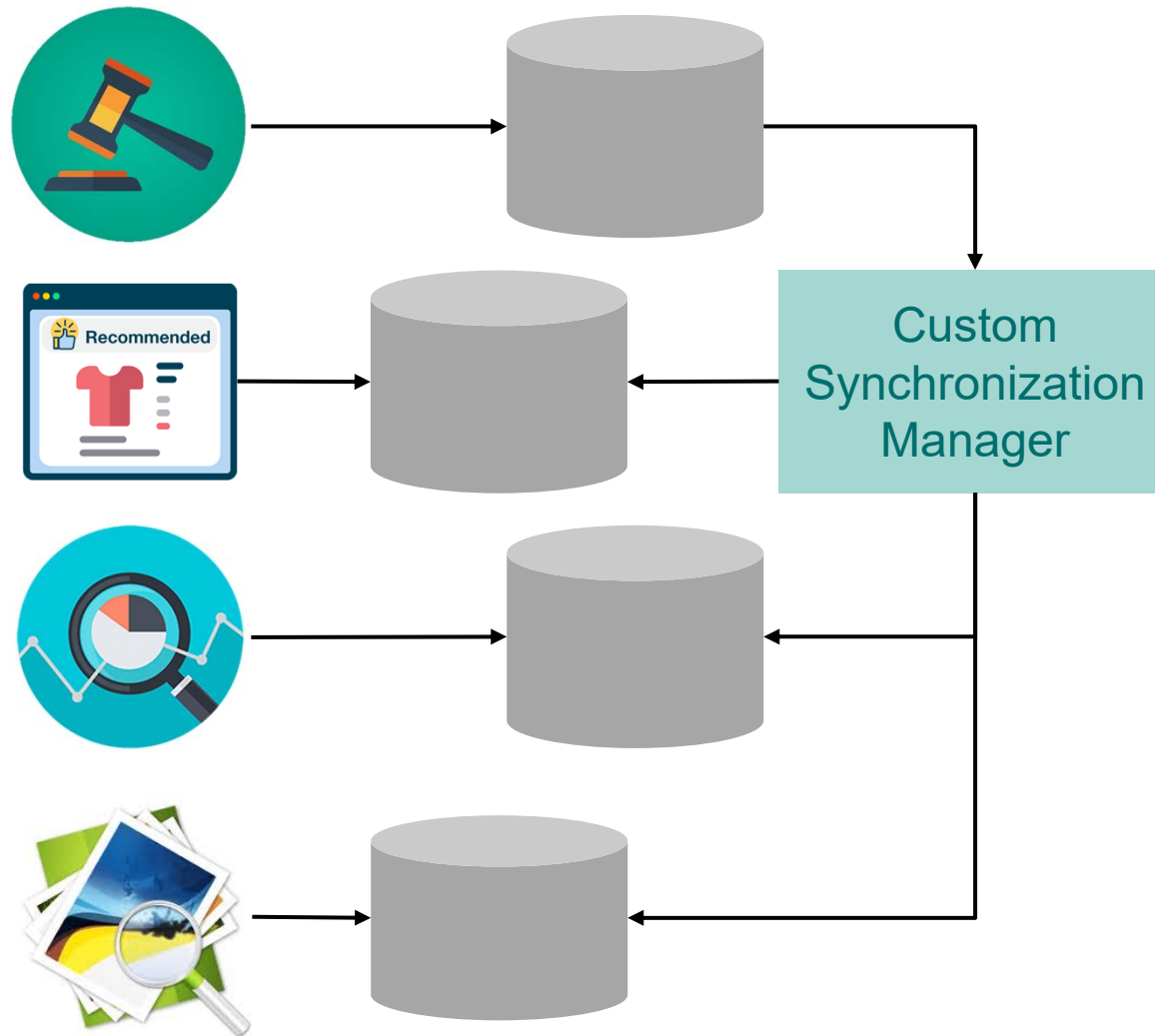
- Cube-based data model
- OLAP Queries



Auction image similarity search

- High-dimensional feature spaces
- Vector space retrieval

How to Address these Requirements?



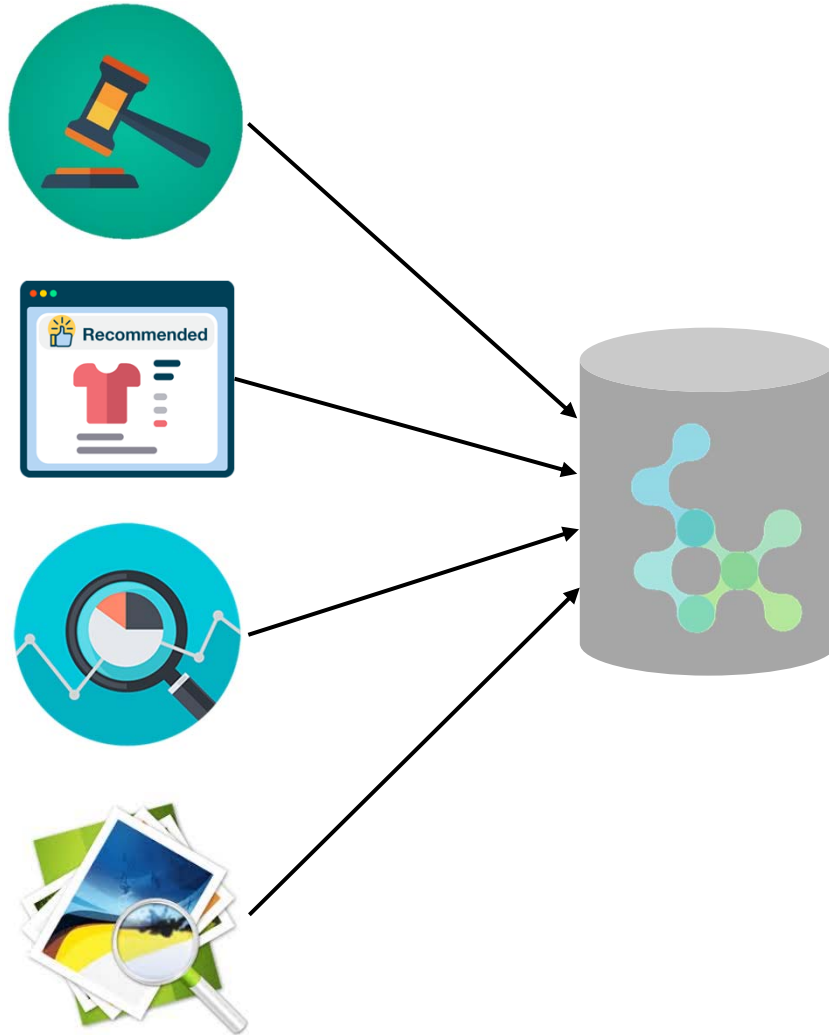
Option 1:

- Specialized Systems

Drawbacks:

- Complex to setup and maintain
- Difficult to optimize (automatically)
- High resource consumption
- Difficult to synchronize
- Not generic
-

How to Address these Requirements?



Option 2:

- Polystore / Multistore

Advantages:

- Only one system to setup and to maintain
- Resources can be used more efficiently → optimization
- Resource optimization
- Generic approach
- ...

The Polypheny-DB Vision



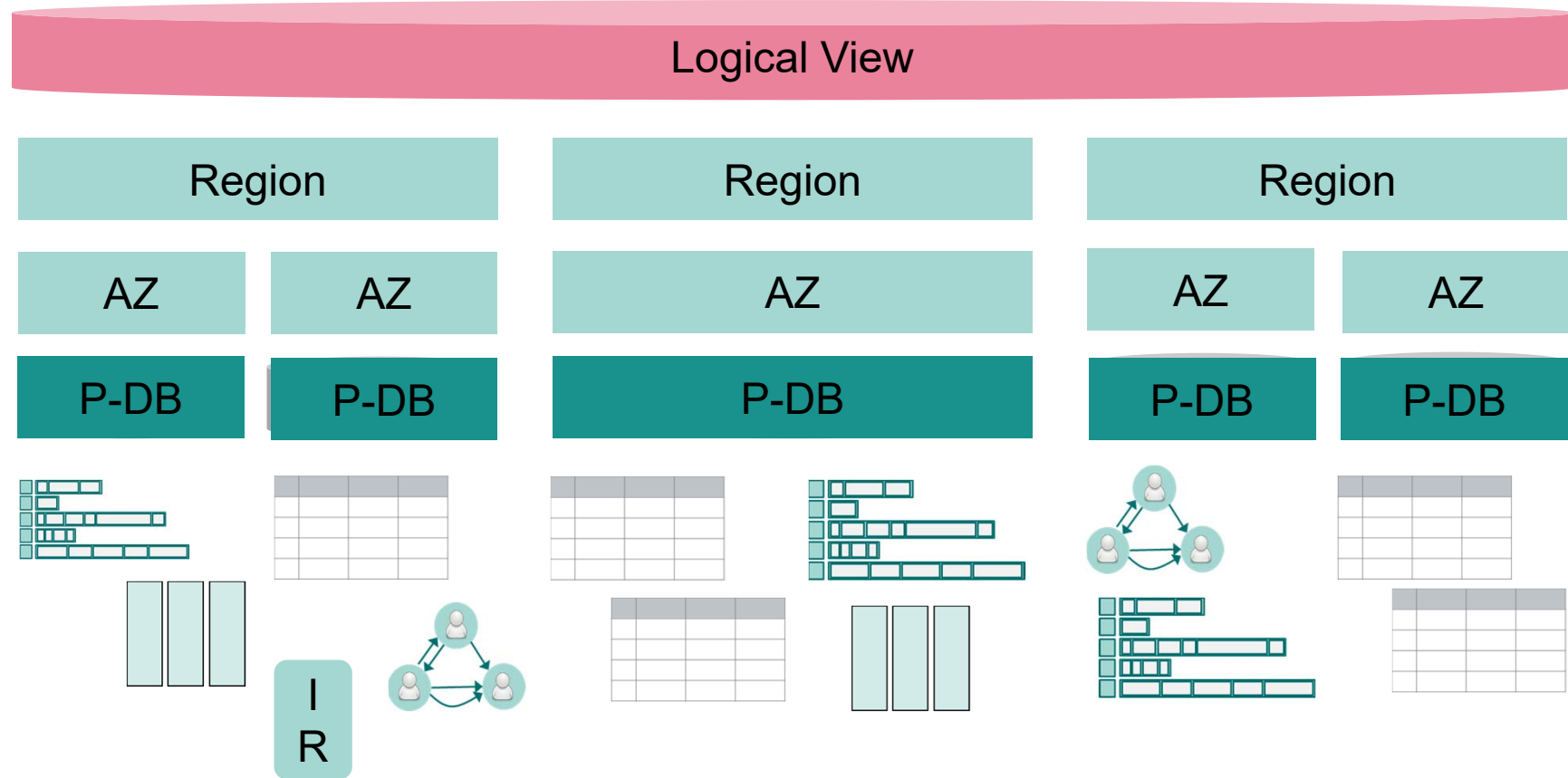
Image Source: stutteringhelp.org/blog/problem-one-size-fits-all

A polystore* that ...

- ... provides good performance for mixed (OLTP and OLAP) workloads
- ... offers different data stores (data models, storage media, ...)
- ... also supports multimedia retrieval
- ... automatically optimizes queries (by selecting the best sites / data stores)
- ... allows to globally distribute data to be used cost-effectively in the Cloud
- ... adapts itself to changing workloads

* A *polystore* system enables query processing across heterogeneous data stores and supports multiple query interfaces. [Tan *et al* BigData '17]

Polypheny-DB: Logical Overview

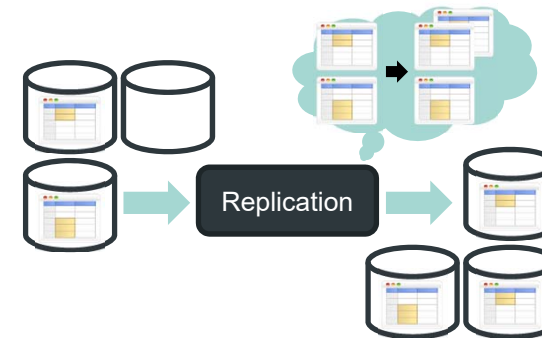
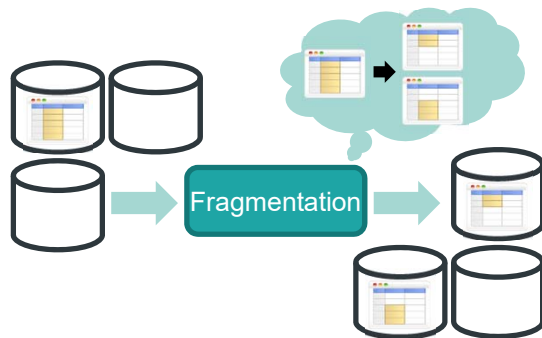


Overview

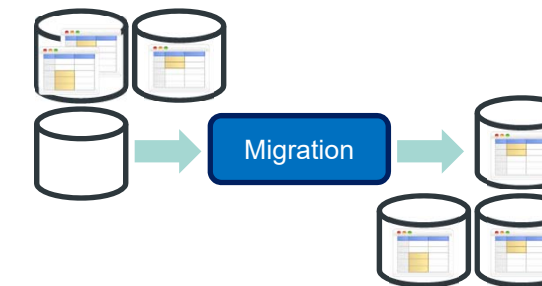
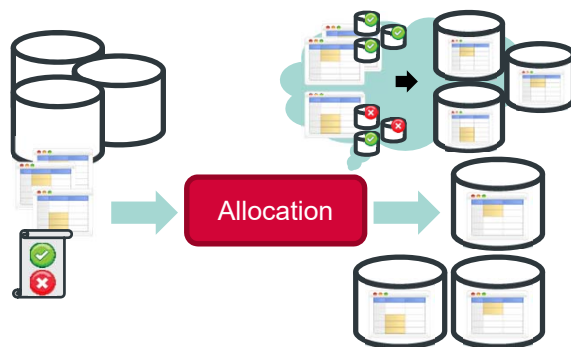
-
- 1 Motivation
 - 2 Polypheny-DB in the Large
 - 3 Polypheny-DB in the Small
 - 4 Use Case: Multimedia Data
 - 5 Conclusions
-

Polypheny in the Large: Distribution

- Data **Fragmentation** – Splitting Data
- Data **Replication** – Duplicating Data



- Data **Allocation** – Assigning Data
- Data **Migration** – Reorganizing Data



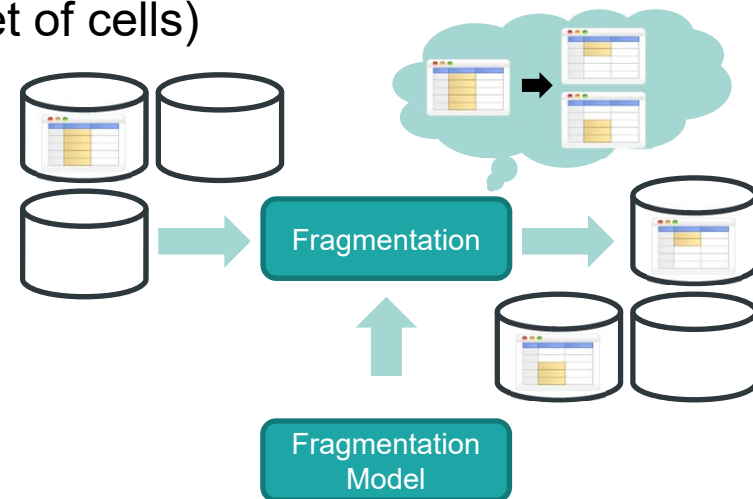
Fragmentation (Partitioning / Sharding)

- Splitting data into fragments and their distribution
 - Load balancing (separating “unrelated” transactions)
 - Storage capacity (horizontal scaling)
- Approaches
 - Horizontal (set of records)
 - Vertical (set of attributes)

} Hybrid (set of cells)

Fragmentation Model

- Input: Database schema, meta data, transactional workload, ...
- Process: Transaction → Record relationship (modelled as graph)
- Output: Partitioned graph determining the new fragments

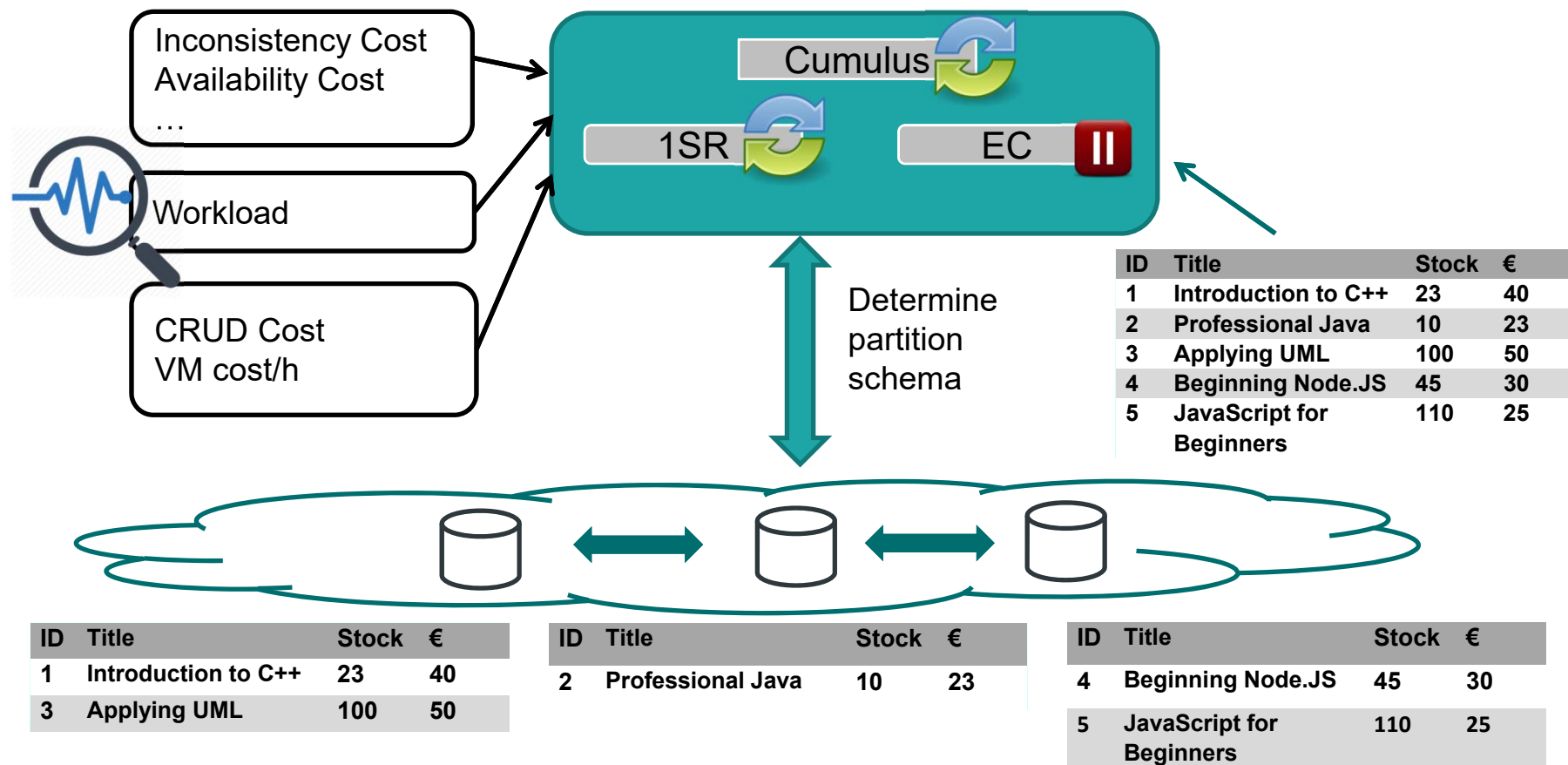


D. Nashat, and A. Amer. 2018. **A Comprehensive Taxonomy of Fragmentation and Allocation Techniques in Distributed Database Design**. *ACM Computing Surveys*. <https://doi.org/10.1145/3150223>

I. Fetai, D. Murezzan, and H. Schuldt. 2015. **Workload-driven adaptive data partitioning and distribution — The Cumulus approach**. *IEEE International Conference on Big Data*. <https://doi.org/10.1109/BigData.2015.7363940>

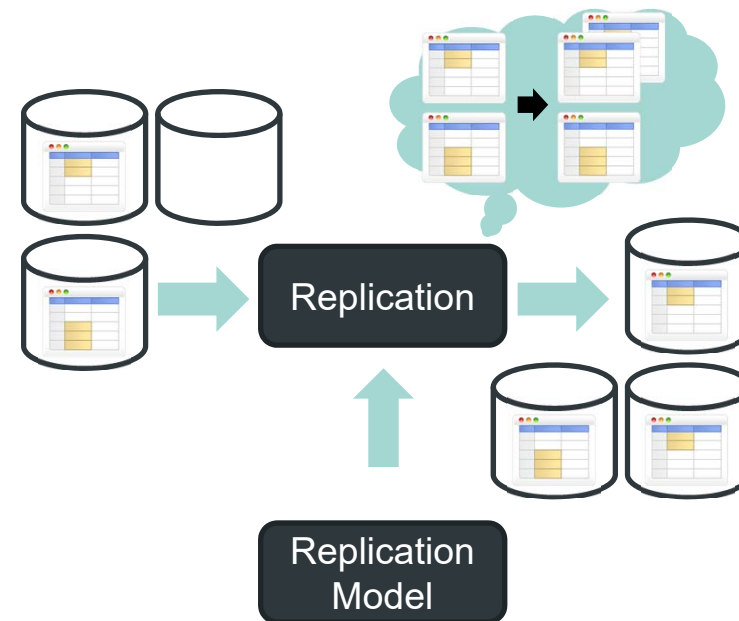
Example: Cumulus

- Workload-driven and adaptive partition protocol providing 1SR executions



Replication

- Duplicating data items and their synchronization
 - Availability
 - Load balancing (in case of read-heavy workloads)
- Challenges: PACELC
 - (Multi-)Master–Slave architecture
 - 2PC, Quorums, Weighted Voting, ...



Replication Model

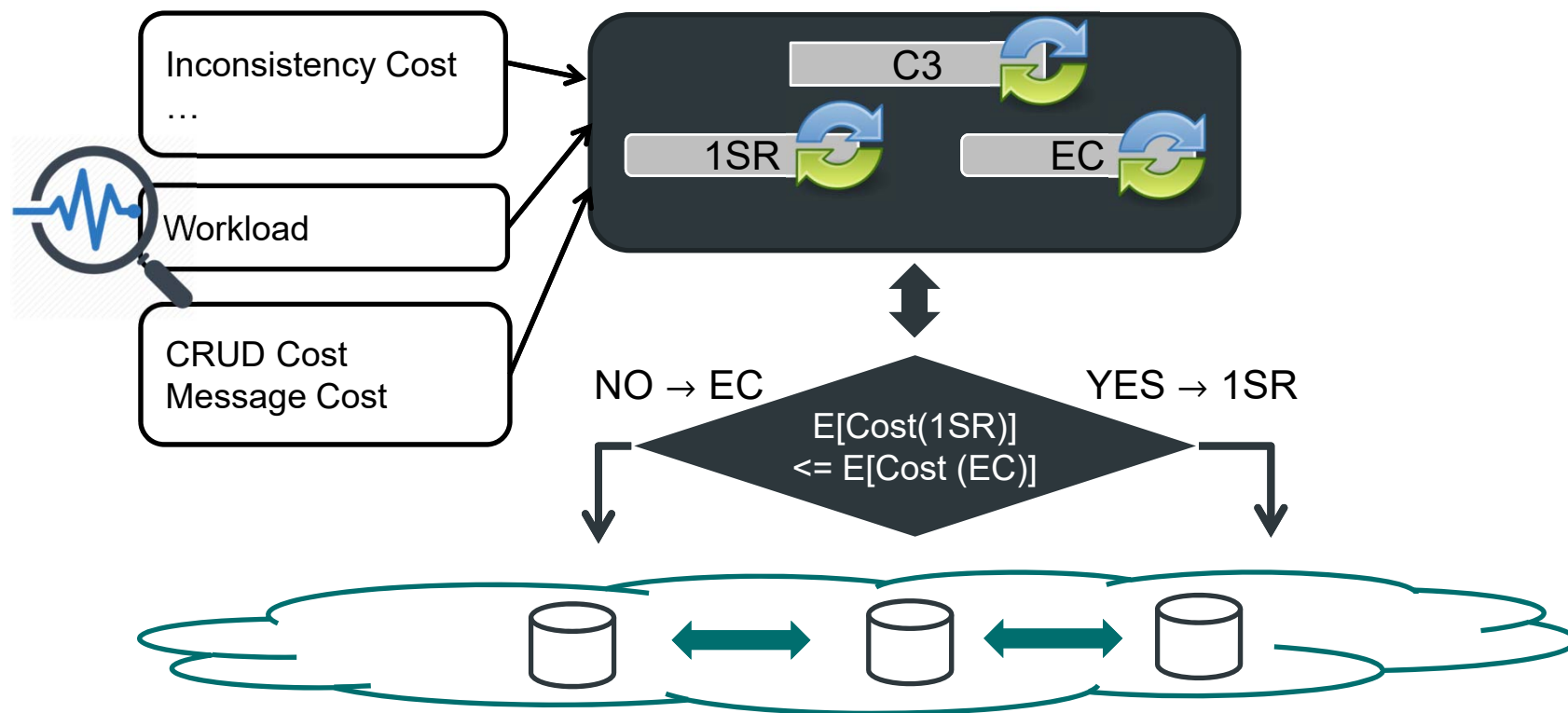
- Input: Data fragments, transactional workload, policies, ...
- Process: Determine replication factor and method
- Output: Replicated fragments and their quorums

I. Fetai, A. Stiemer, and H. Schuldt. 2017. **QuAD: A Quorum Protocol for Adaptive Data Management in the Cloud**. *IEEE International Conference on Big Data*. <https://doi.org/10.1109/BigData.2017.8257952>

A. Stiemer, I. Fetai, and H. Schuldt. 2016. **Analyzing the performance of data replication and data partitioning in the cloud: The BEOWULF approach**. *IEEE International Conference on Big Data*. <https://doi.org/10.1109/BigData.2016.7840932>

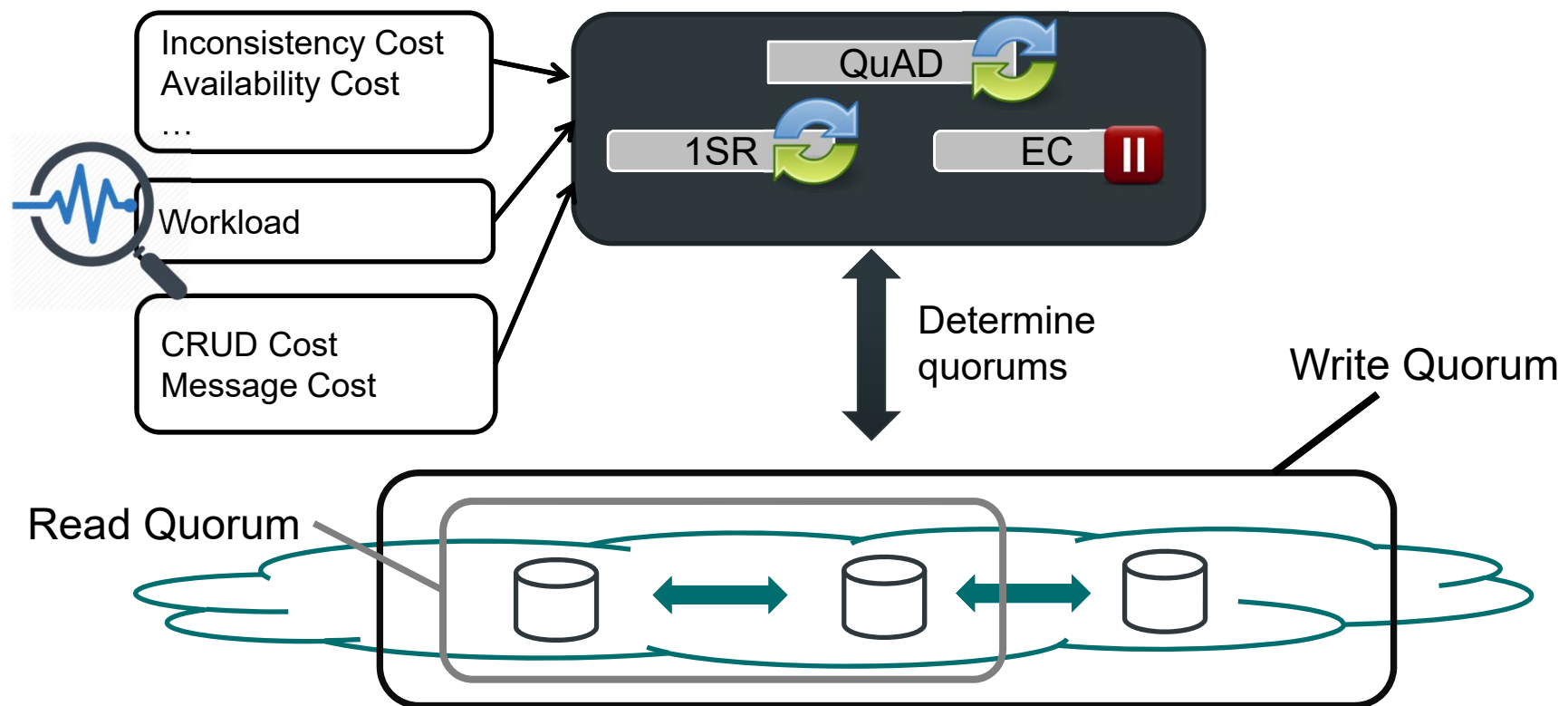
Example: C3

- Adjust consistency at runtime in order to reduce overall cost. Trade-off between
 - Consistency costs (lock messages, 2PC messages, ...)
 - Inconsistency costs (application-specific)



Example: QuAD

- Considers load and network proximity between sites
 - Avoid 'weak' sites from read and commit path
 - Strength determined by the site score (load, round-trip-time)

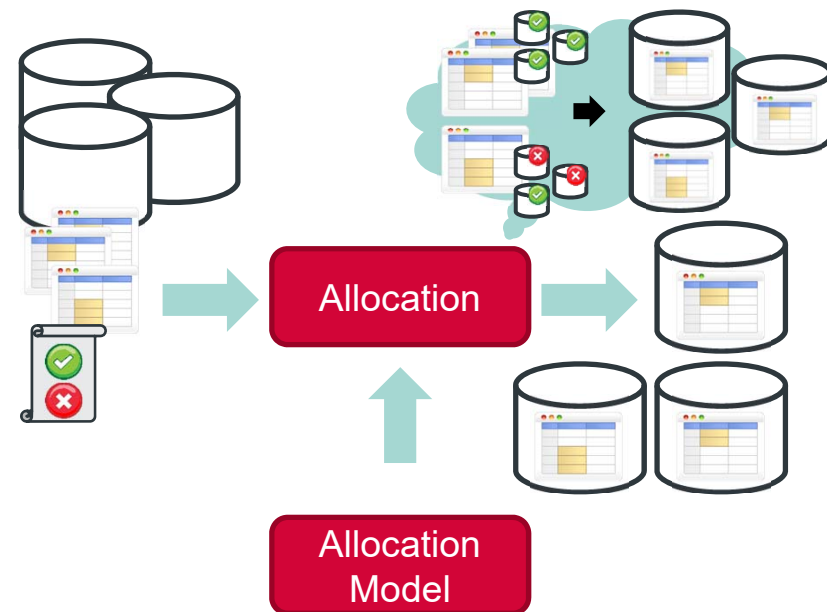


Allocation

- Assigning data to nodes in a distributed environment
 - Optimizing performance
 - Applying / respecting policies
- Approaches
 - Static
 - Dynamic

Allocation Model

- Input: Fragments to distribute, network properties, policies, ...
- Process: Assignment matrix (Knapsack Problem)
- Output: Assignments of fragments to nodes

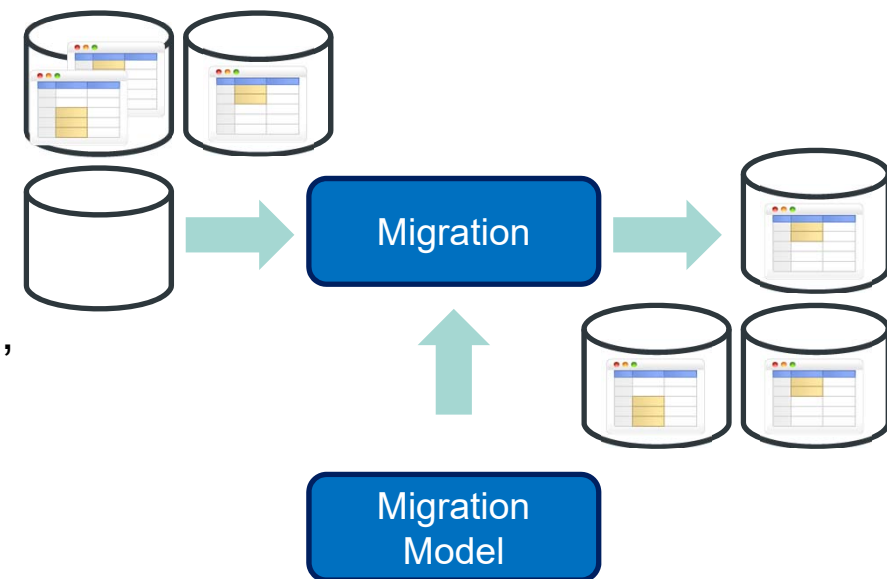


D. Nashat, and A. Amer. 2018. **A Comprehensive Taxonomy of Fragmentation and Allocation Techniques in Distributed Database Design.** *ACM Computing Surveys*. <https://doi.org/10.1145/3150223>

I. Fetai, A. Stierner, and H. Schuldt. 2017. **QuAD: A Quorum Protocol for Adaptive Data Management in the Cloud.** *IEEE International Conference on Big Data*. <https://doi.org/10.1109/BigData.2017.8257952>

Migration

- Reorganization of the distributed data
 - Dynamic environment requires adaptation
- Approaches
 - Offline: Stop – Migrate – Start
 - Online: Background Migration



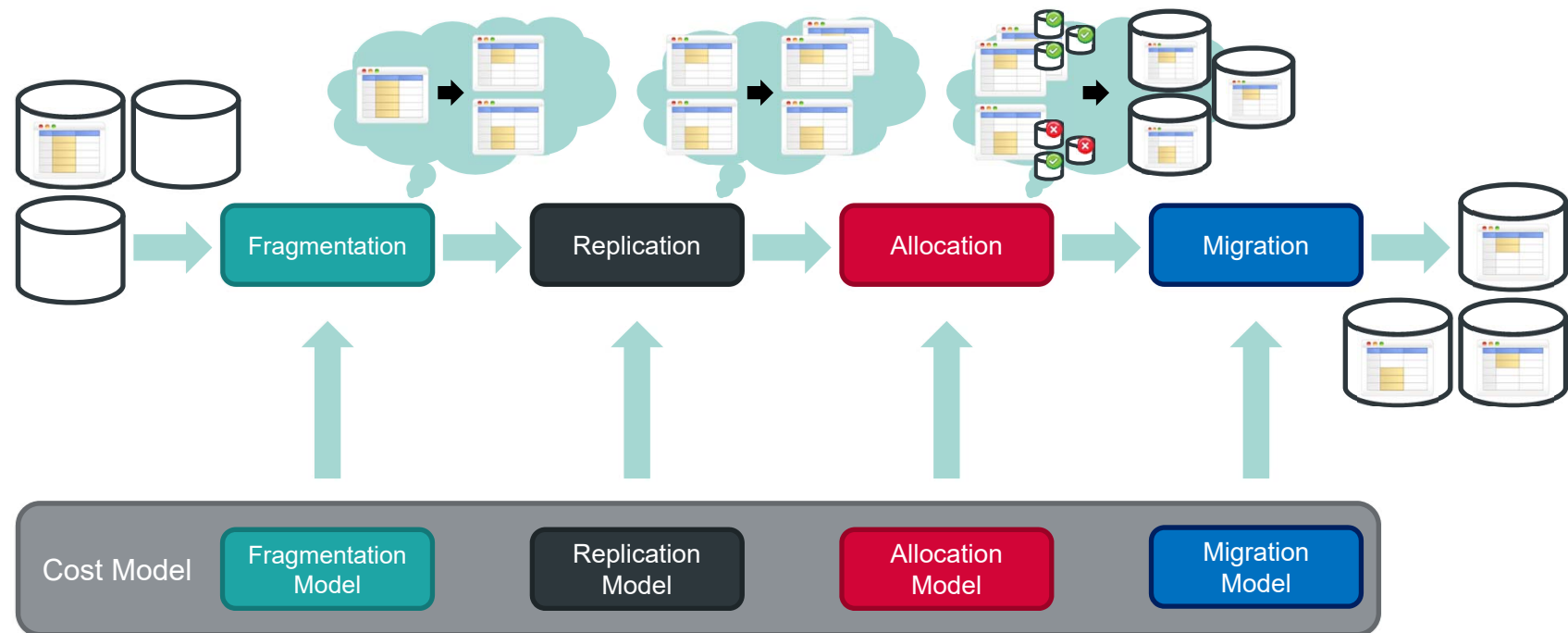
Migration Model

- Input: Meta data, current workload, network properties, ...
- Process: Evaluation of migration time vs. latency impact
- Output: Migration method

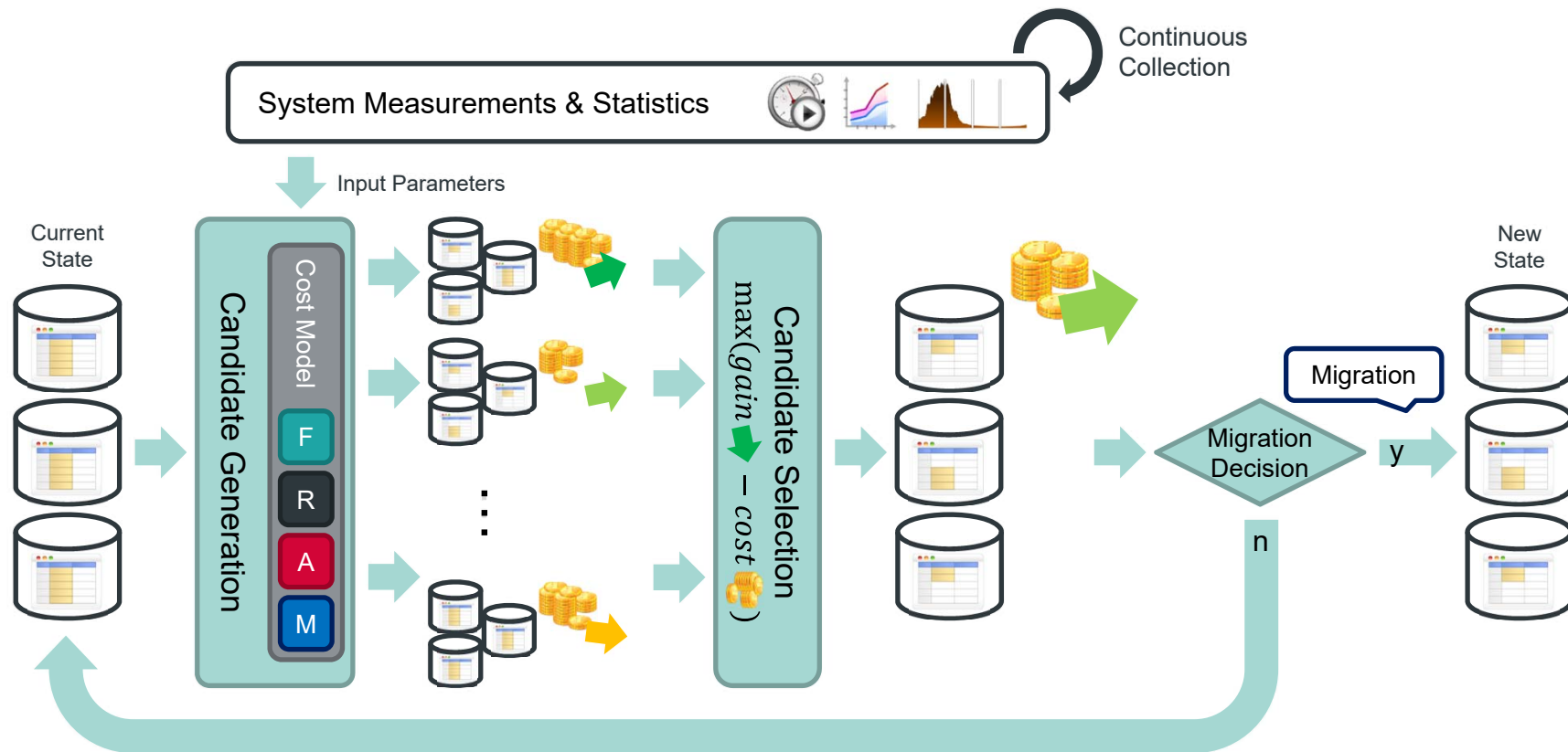
G. H. Sockut, and B. R. Iyer. 2009. **Online Reorganization of Databases**. *ACM Computing Surveys*. <https://doi.org/10.1145/1541880.1541881>

I. Fetai, D. Murezzan, and H. Schuldt. 2015. **Workload-driven adaptive data partitioning and distribution — The Cumulus approach**. *IEEE International Conference on Big Data*. <https://doi.org/10.1109/BigData.2015.7363940>

F-R-A-M in Polypheny

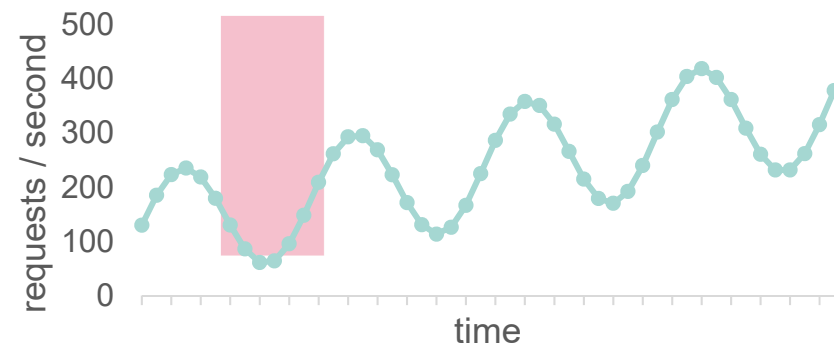
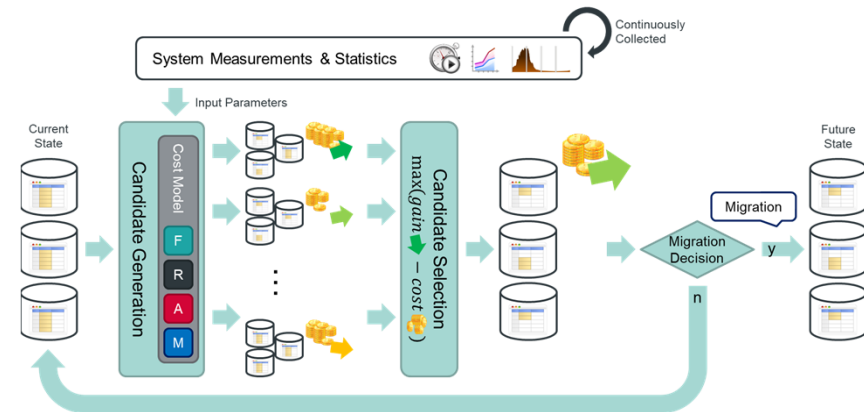


Adaptive Data Management Pipeline



Adaptive Data Management Pipeline: Challenges

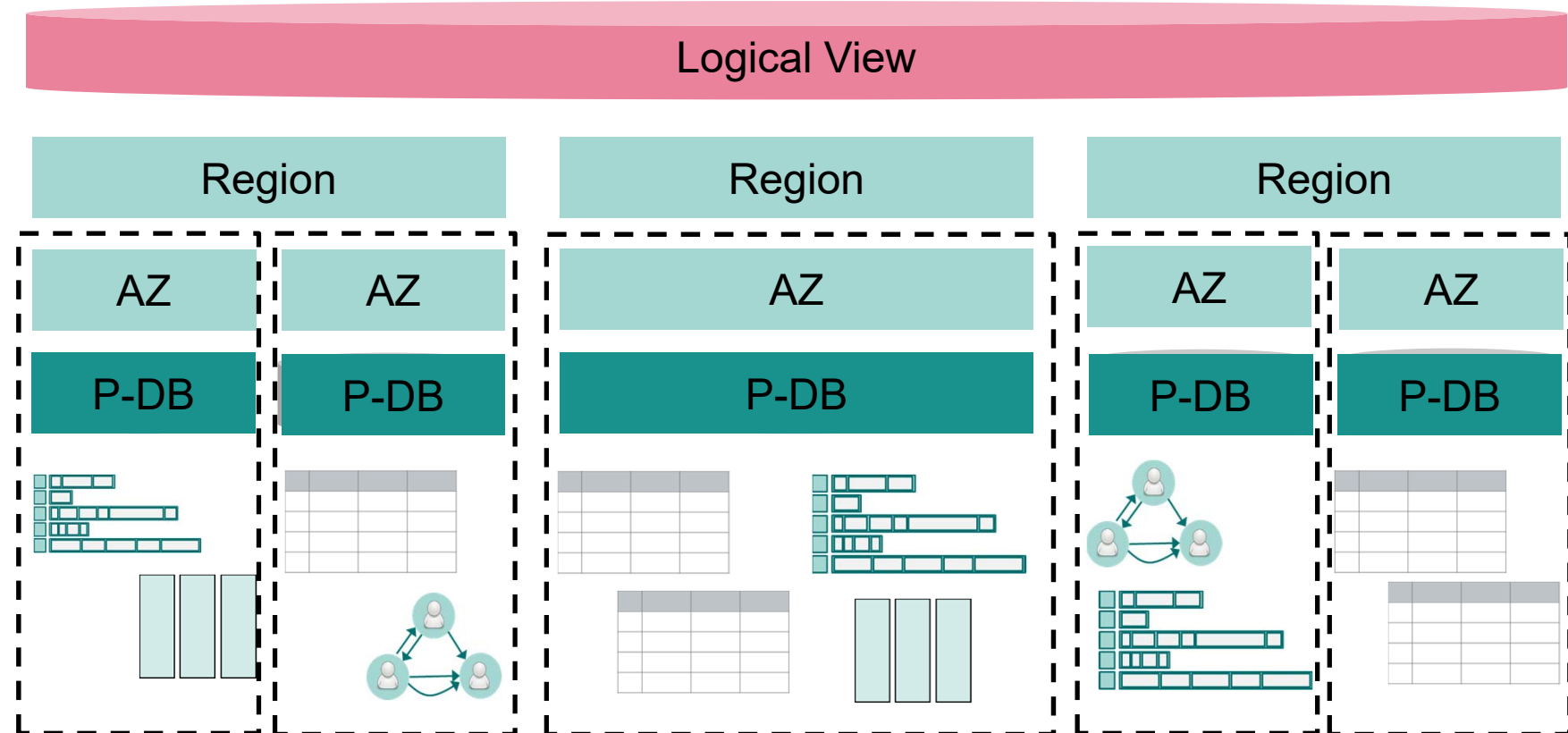
- Modularization
 - Exchangeability of methods and models
 - Influences and dependencies
- Combination
 - Finer-grained data management
 - Performance & resource utilization
- Adaptation
 - Which time frame?
 - How avoid “cyclic” migrations?
 - Selection of thresholds
 - How to detect long term gain?



Overview

-
- 1 Motivation
 - 2 Polypheny-DB in the Large
 - 3 Polypheny-DB in the Small
 - 4 Use Case: Multimedia Data
 - 5 Conclusions
-

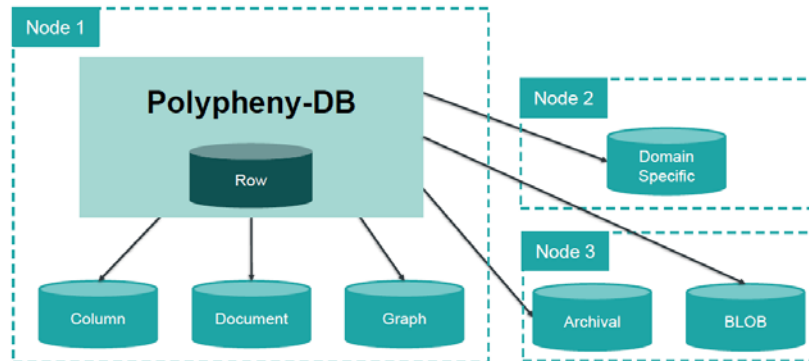
Polypheny-DB: Logical Overview



Polystore Layer – Challenges at Data Storage Level

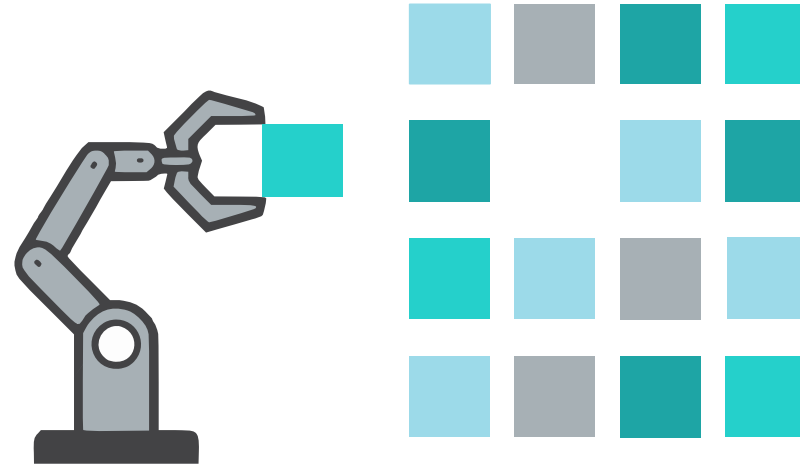


- Some data stores are better suited for a specific scenario than others
- Data stores can be loaded and unloaded at runtime
- Data stores can be located on the same physical node or on other nodes
- Special store located within Polypheny-DB

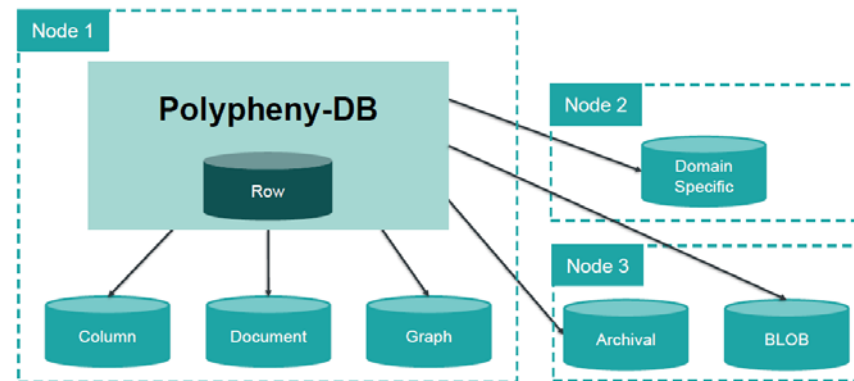


Polystore Layer – Adaptive Data Placement

- Different approaches:
 - Each data item is stored on all stores
 - Each data item is stored on one store
- Dynamic placement of data items across the underlying data stores based on the workload



Source: tintri.com/blog/2017/09/optimize-vm-placement-machine-learning-reduced-costs



Query Interfaces



- Polypheny-DB supports (will support) a wide range of different query languages based on different data models and through different query interfaces
 - Different SQL dialects
 - A CRUD based “query language”
 - A graph-based query language (OpenCypher)
 - Vector space retrieval

Temporal / Multi-version Data Management

- Updates can be done in-place or by storing several versions of the same data item
- More options for data placement
- Updates to outdated stores periodically or in times of low workload
- Option to provide an “acceptable” freshness level for each database transaction



Self-Adaptivity

- Continuously change configuration according to the current and predicted workload
- Fulfil requirements with minimal costs by:
 - Adjusting global data management strategy
 - Adding and removing indexes
 - Adding and removing data stores

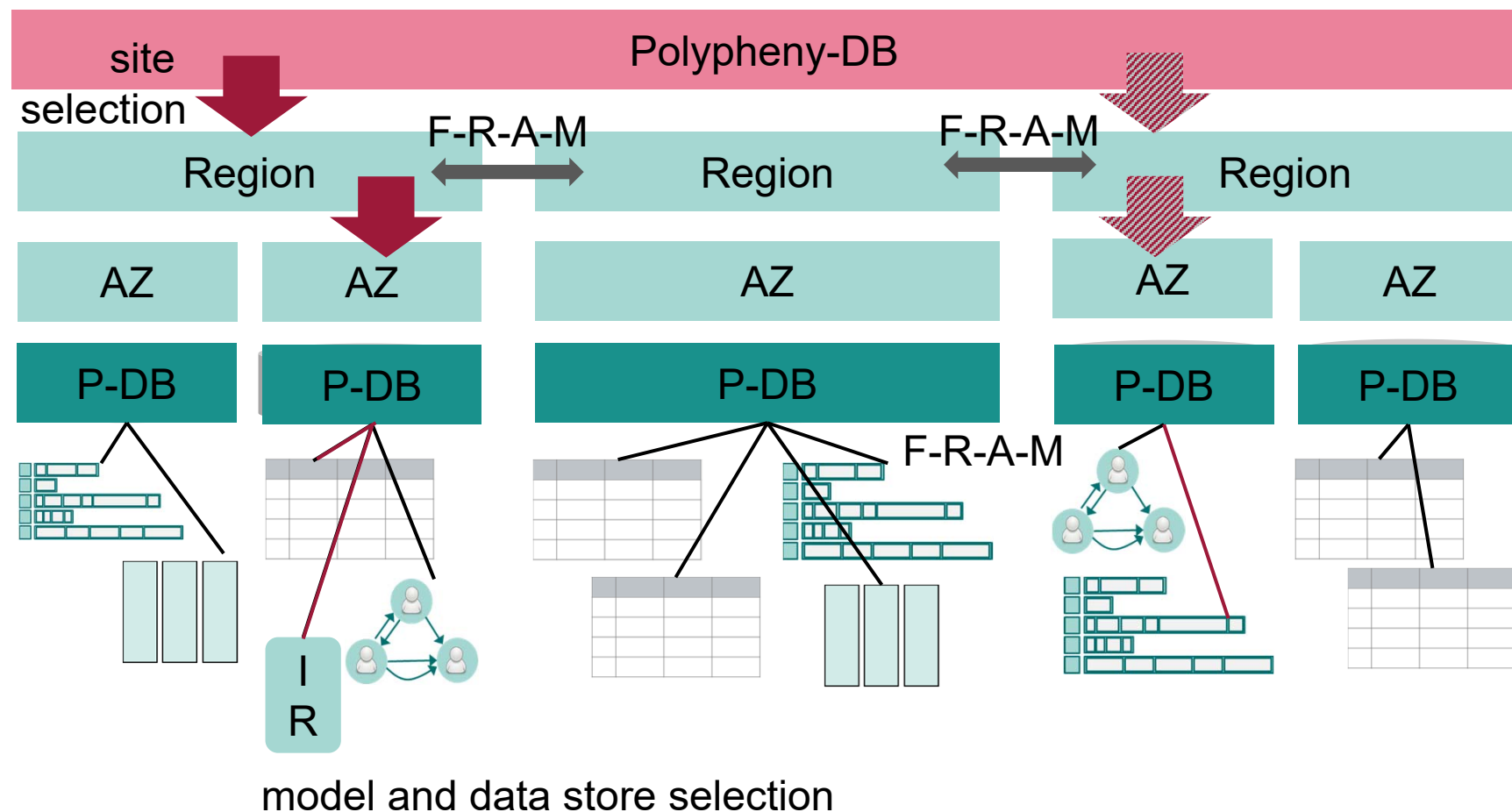


Source: <http://www.beyondthefear.com/four-things-to-know-about-adaptation/>

Polypheny-DB: Logical Overview

 at run-time

 at query-time

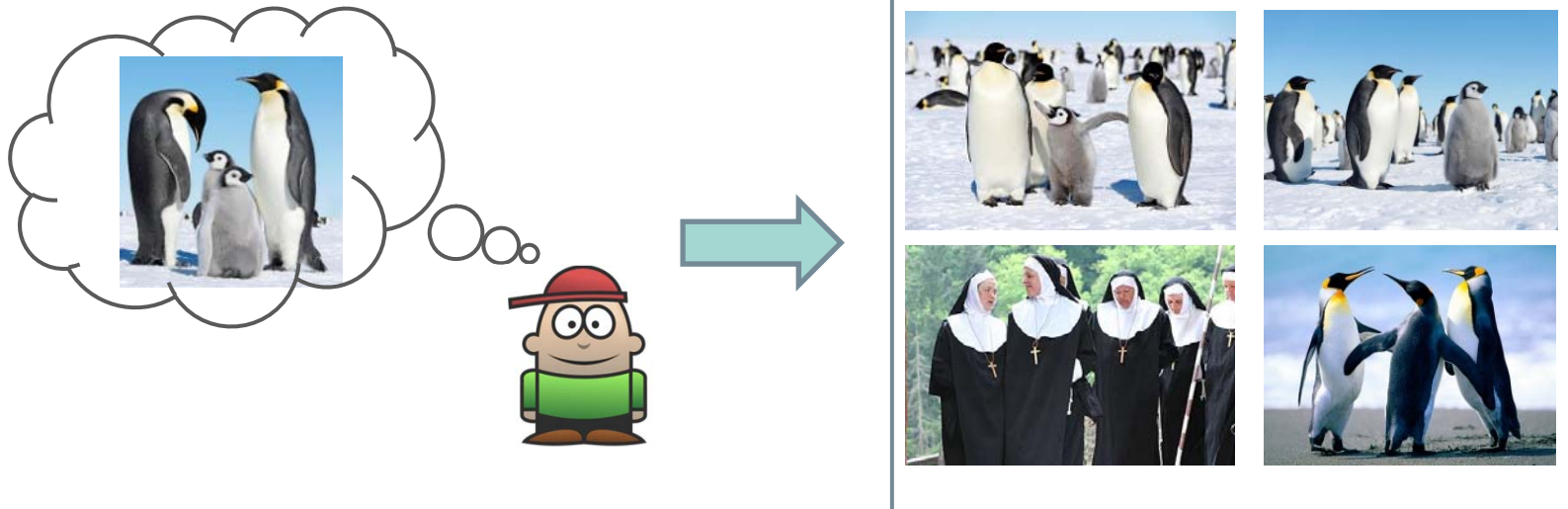


Overview

-
- 1 Motivation
 - 2 Polypheny-DB in the Large
 - 3 Polypheny-DB in the Small
 - 4 Use Case: Multimedia Data
 - 5 Conclusions
-

New Approaches to Searching in Multimedia Data

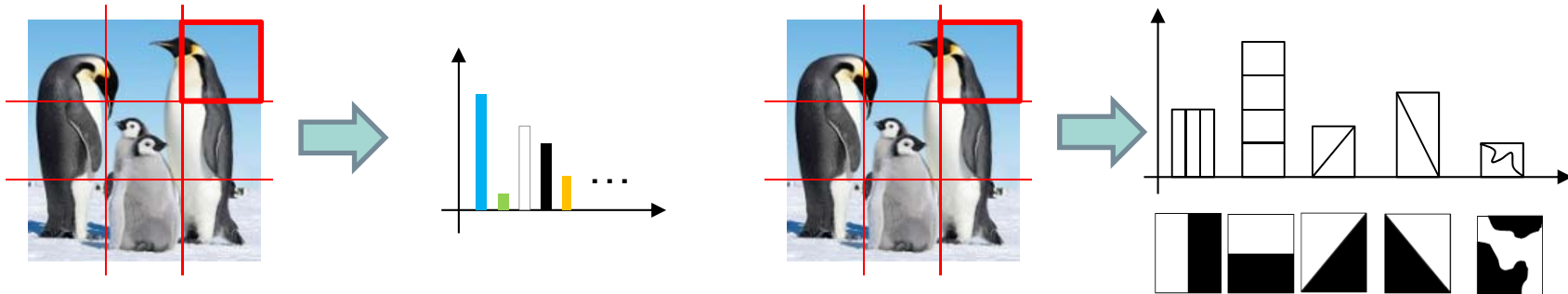
- Keyword Search: Search based on (manually added) textual descriptions
- Query-by-Example: Similarity to query object



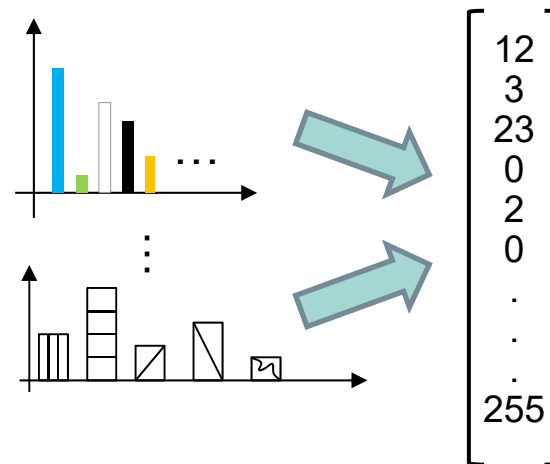
Sources: • <http://moviespictures.org>
• Penguins by kyuubidemon98 Michel CC BY-SA 3.0 via <http://kyuubidemon98.deviantart.com/art/penguins-156283137>
• Emperor Penguins by Christopher Michel CC BY-SA 2.0 via flickr -- <https://www.flickr.com/photos/cmichel67/11240231654>,
<https://www.flickr.com/photos/cmichel67/11240225716>, <https://www.flickr.com/photos/cmichel67/11240219084>

Image (Video) Features

- Similarity search is based on inherent object features
 - e.g., color (global and/or for parts) or edges (global and/or for parts)



- Feature transformation:
Each object is represented
by a feature vector



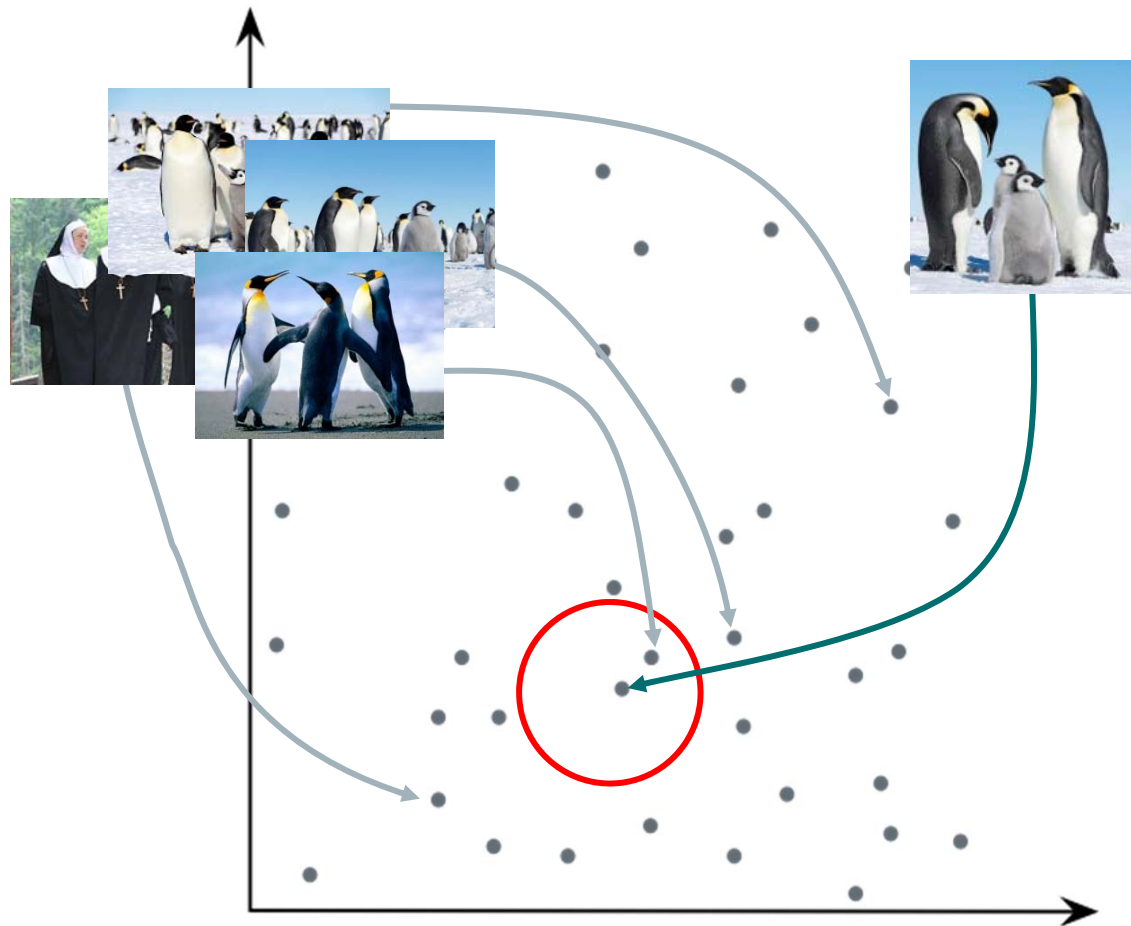
Similarity Search (Vector Space Retrieval)

- Each object is a point in the feature space (also the query object)



$$d = \begin{pmatrix} d_1 \\ d_2 \\ \vdots \\ d_{dim} \end{pmatrix}$$


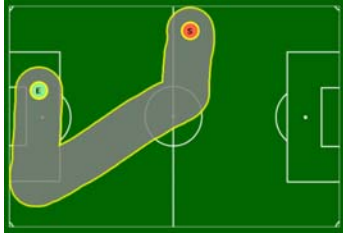
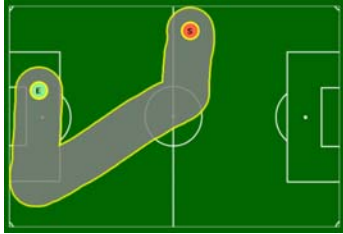
- Similarity search = k -Nearest Neighbor Search

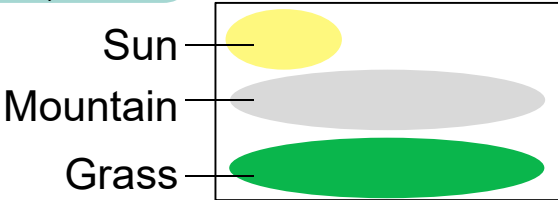
$$\delta_{L_2}(d, q) = \sqrt{\sum_{j=1}^d (d_j - q_j)^2}$$



... Search in Big Multimedia Data

- Query-by-Sketch: Searching using a low-level representation



Edinburgh-Castle by Christian Bickel - <https://commons.wikimedia.org/wiki/File:Edinburg004.jpg>
- Motion Query: Specify motion (in videos)
- Query-by-Concept: Searching using real-world concepts




king penguin (4.96037)
penguin (4.88750)
spenisciform seabird (4.81464)
seabird (4.00825)
aquatic bird (2.31044)
- Query-by-Semantic Sketch: Assign concepts to regions in an object


Sun
Mountain
Grass
- Query-byTime/Space: Similarity to location or time of capture of query object
- ... and a combination of all these modes

Object Recognition (Semantic Features) ...

- Novel approaches use machine learning (deep learning) to analyze images



“man in black shirt
is playing guitar.”



“black and white dog
jumps over bar.”



“man in blue wetsuit
is surfing on wave.”

Source: <http://cs.stanford.edu/people/karpathy/deepimagesent/>

... Object Recognition (Semantic Features)



"a group of boats floating on top of a lake."




"a young boy is holding a baseball bat."



"a bunch of bananas are hanging from a ceiling."

Source: <http://cs.stanford.edu/people/karpathy/deepimagesent/>

Multimedia Data Management: Databases or MIR

Database Systems		Retrieval Systems																					
Data	Structured (clear syntax and semantics)	Semi-structured / unstructured																					
Query	Unambiguous, attribute-based, Boolean matching, short running	Imprecise and incomplete, similarity matching, long-running																					
Inter-action	Single query produces single answer	Interaction sequence, relevance feedback																					
Result	Exact match w.r.t. query	Relevant w.r.t. query																					
	<table><tr><td></td><td><i>id</i></td><td><i>painter</i></td><td><i>title</i></td><td>...</td></tr><tr><td><i>t</i>₁</td><td>1</td><td>van Gogh</td><td>Wheatfield with Crows</td><td>...</td></tr><tr><td><i>t</i>₂</td><td>2</td><td>Monet</td><td>Water Lillies</td><td>...</td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr></table>		<i>id</i>	<i>painter</i>	<i>title</i>	...	<i>t</i> ₁	1	van Gogh	Wheatfield with Crows	...	<i>t</i> ₂	2	Monet	Water Lillies		
	<i>id</i>	<i>painter</i>	<i>title</i>	...																			
<i>t</i> ₁	1	van Gogh	Wheatfield with Crows	...																			
<i>t</i> ₂	2	Monet	Water Lillies	...																			
...																			

[https://en.wikipedia.org/wiki/Wheat_Fields_\(Van_Gogh_series\)#/media/File:Vincent_van_Gogh_-_Wheatfield_with_crows_-_Google_Art_Project.jpg](https://en.wikipedia.org/wiki/Wheat_Fields_(Van_Gogh_series)#/media/File:Vincent_van_Gogh_-_Wheatfield_with_crows_-_Google_Art_Project.jpg)
[https://en.wikipedia.org/wiki/Water_Lilies_\(Monet_series\)#/media/File:Claude_Monet_-_Water_Lilies_-_Google_Art_Project.jpg](https://en.wikipedia.org/wiki/Water_Lilies_(Monet_series)#/media/File:Claude_Monet_-_Water_Lilies_-_Google_Art_Project.jpg)

Multimedia Data Management: Databases **and** MIR

	Database Systems	Retrieval Systems
Data	Structured (clear syntax and semantics)	Semi-structured / unstructured
Query	Unambiguous, attribute-based, Boolean matching, short running	Imprecise and incomplete, similarity matching, long-running
Inter-action	Single query produces single answer	Interaction sequence, relevance feedback
Result	Exact match w.r.t. query	Relevant w.r.t. query

Painter = “van Gogh”

and

Object is similar to:



Multimedia Data Management: Databases and MIR

Multimedia Data Management

Relational Data Model

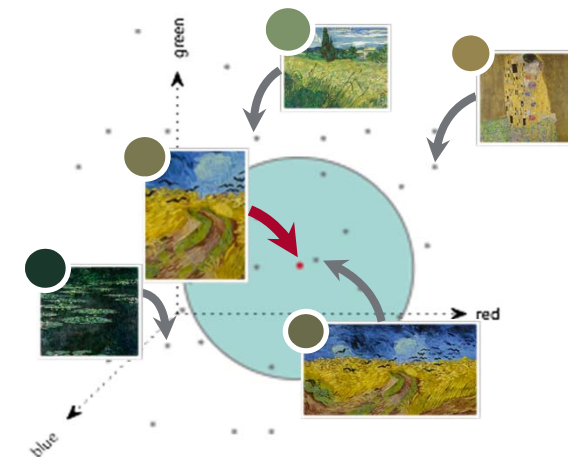
$\text{sch}(R) := \{a_1, a_2, \dots, a_m\}$

$\text{val}(R) := \{t_1, t_2, \dots, t_n\}$

	a_1	a_2	...	a_m
t_1				
t_1				
...				
t_n				

$\times, \pi, \sigma, \cup, \cap, \bowtie$

Vector Space Model



Similarity operator τ

Sample Multimedia Query

Painter = "van Gogh"

and

Object is similar to:



$\sigma_{\text{painter} = \text{"van Gogh"}} (\text{paintings})$

$\tau_{L2, \text{feature_color}, [0.08, 0.42, \dots, 0.83], \kappa=2} (\text{paintings})$

Relational Data Model

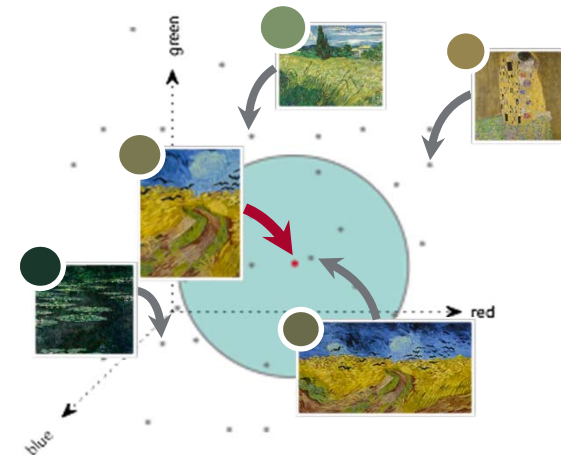
$\text{sch}(R) := \{a_1, a_2, \dots, a_m\}$

$\text{val}(R) := \{t_1, t_2, \dots, t_n\}$

	a_1	a_2	...	a_m
t_1				
t_1				
...				
t_n				

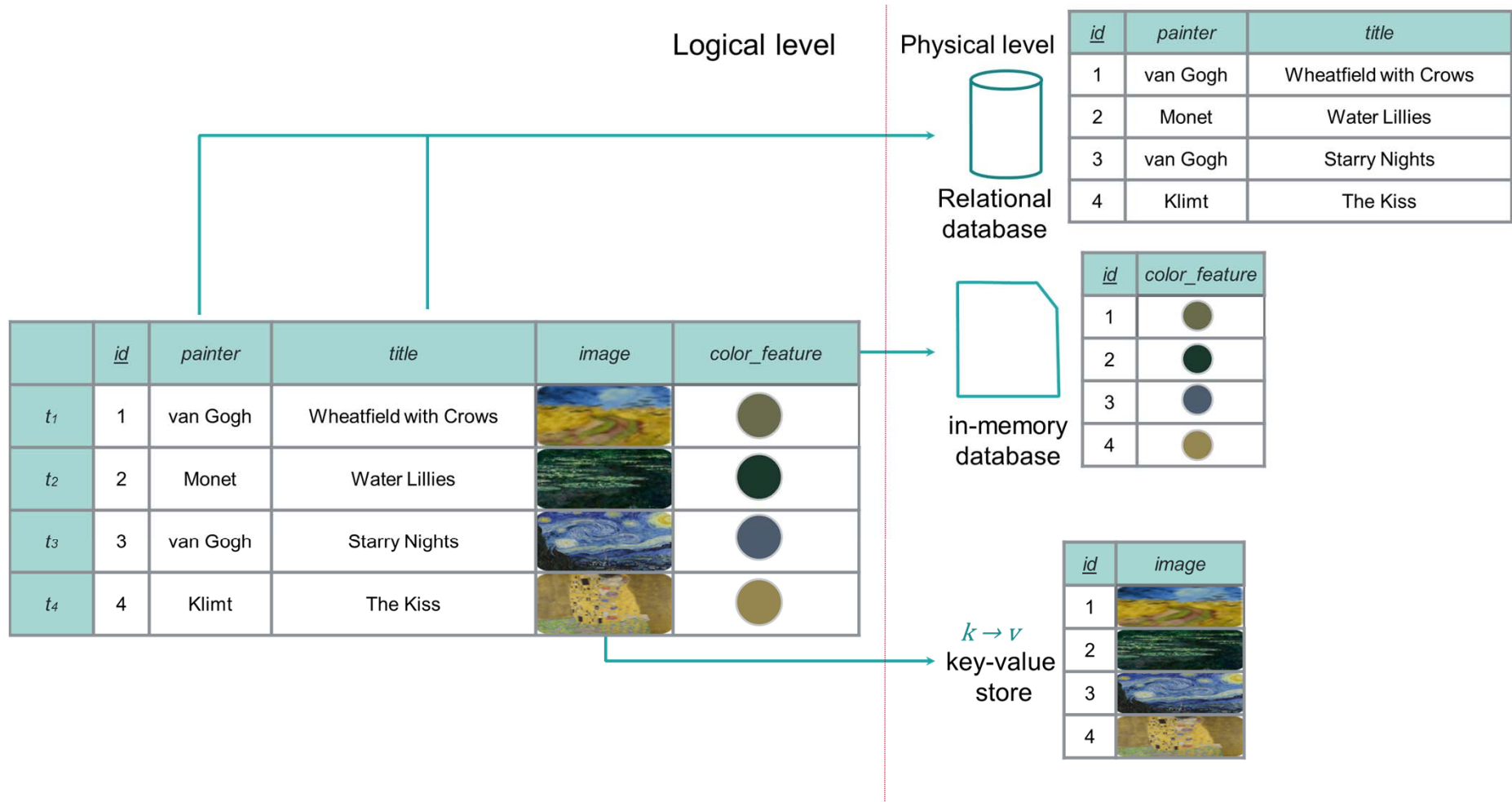
$\times, \pi, \sigma, \cup, \cap, \bowtie$

Vector Space Model

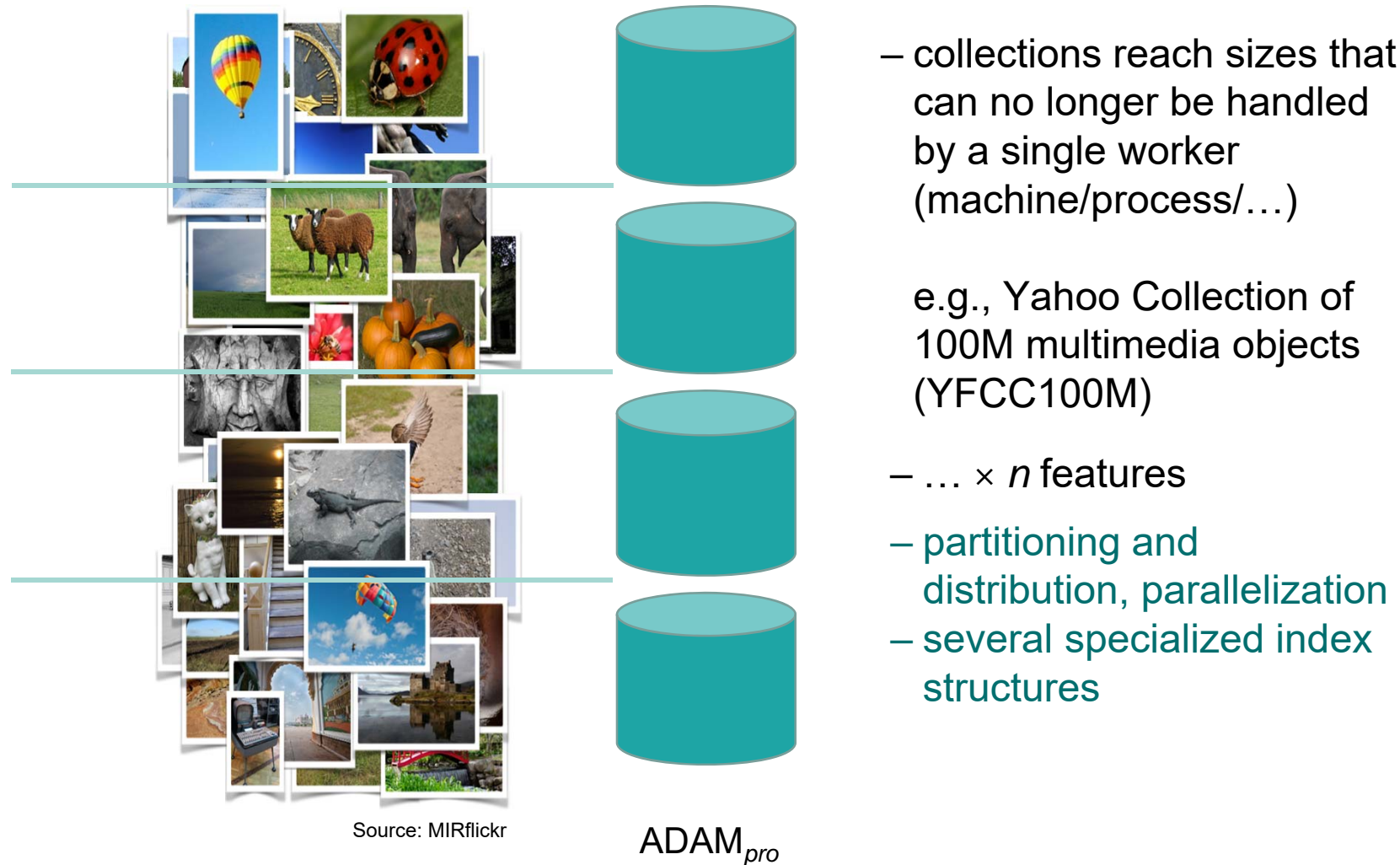


Similarity operator τ

Polystore for Multimedia Data: ADAM_{pro}

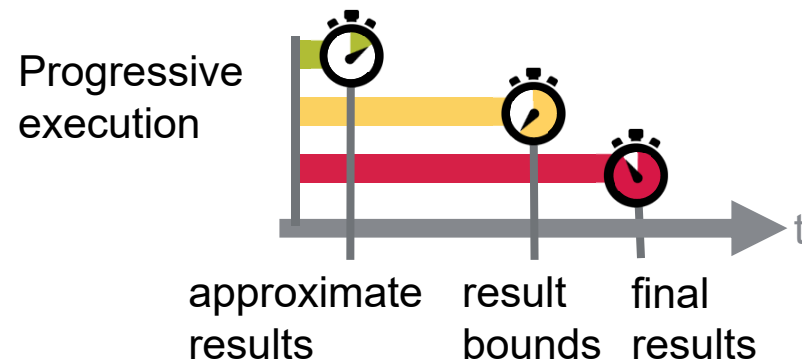


Challenges at the Database Level: Collection Sizes



Index Structured for Multimedia Queries

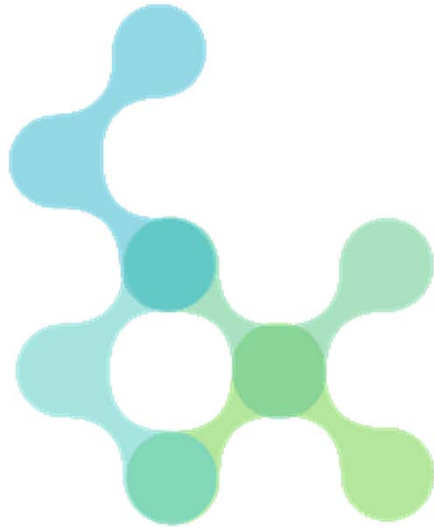
- Similarity search implies long-running queries
- Dedicated index structures exist
 - Exact index structures (e.g., VA file)
 - Provide exact results
 - Long query execution time
 - Approximate index structures (e.g., SH, LSH, PQ, CP, etc.)
 - Quick results
 - False positives
- ADAM_{pro} supports both types of index structures



Overview

-
- 1 Motivation
 - 2 Polypheny-DB in the Large
 - 3 Polypheny-DB in the Small
 - 4 Use Case: Multimedia Data
 - 5 Conclusions

Polypheny-DB



- Global data partitioning and replication based on user requirements and workload
- Polystore leveraging the strength of different data models, data stores (and storage media)
- Optimization in the large and in the small
- Dynamic adaption on both levels according to the current and predicted workload
- Challenges
 - Degree of separation between “in the large” and “in the small”
 - Dependencies between the different levels
 - ...

Acknowledgements

Former and current members of the DBIS group

- Dr. Ilir Fetai
- Dr. Ivan Giangreco
- Alexander Stiemer
- Marco Vogt

Funding Agencies

- SNSF: Project Polypheny-DB
- Horizon 2020/SNSF (ERA-Net): Project IMOTION (vitivr, ADAM_{pro})



University
of Basel

Thank you for your attention!

contact: heiko.schuldt@unibas.ch or dbis-cs@unibas.ch

