Doctoral Project Presentation
DEDS Winter School 2022

April 5, 2022

Abduvoris Abduvakhobov

Center for Data Intensive Systems, Daisy
Department of Computer Science
Aalborg University
Denmark

AALBORG UNIVERSITY
DENMARK

Agenda

Introduction

PhD Topic: ESR:2.3. Model-based storage for time series

Supervisors:

- Associate Professor Christian Thomsen, AAU
- Professor Esteban Zimanyi, ULB

Secondment: Siemens Gamesa Renewable Energy

PhD Start Date: February 1, 2022

Current progress:

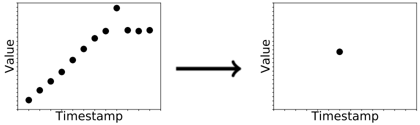
- Finalizing doctoral project plan
- Started collaboration with secondment partner and got access to data

Motivation

Research undertaken by Jensen et al. 2021

Meetings with manufacturers, owners, and energy traders:

- Modern turbines are monitored by up to 7,000 high-quality sensors
- The storage needed makes storing high-frequency sensor data infeasible
- Simple aggregates (e.g. 10-minute averages) are stored instead of the high-frequent series, thereby removing useful fluctuations and outliers:



- Users believe problems can be found earlier with high-frequency data.
- Compression need only be lossless for some types of time series.

Motivation
Data

Meetings with manufacturers, owners, and energy traders:

- The sensors are installed with wired power and connectivity.
- Each sensor produces a data stream sampled to, e.g., a 10 Hz series.
- Collected measures include: Air Pressure, Humidity, Voltage, Power, Rotation Speed, Temperature, Wind Direction, Wind Speed, Internal Controller Measurements.
- The time series are regular, cleaned, but gaps without values can occur.
- Metadata for each time series must also be stored, e.g., as dimensions.

Motivation

Paramount Properties

Paramount properties for a system managing wind turbine data:

Distribution: The system must be able to scale to many nodes.

Stream Processing: Data points are arriving continuously as a regular time series and must be queryable with a short latency.

Compression: High compression is needed for high-frequency data.

Efficient Retrieval: Indexes or ordered storage for fast retrieval.

Approximate Query Processing: Approximate answers can be accepted for some time series and enables use of lossy compression.

Extensibility: Allows users with domain knowledge to implement new storage methods optimized specifically for their data sets.

State of the Art
Time Series Management Systems

Time Series Management System or TSDB:

- Stores time series that consists of time stamp.
- Optionally contain metadata or tags.
- Processes queries on time series.
- Queries contain timestamp or a time range.

Categorization based on architecture:

- Internal Data Stores:
    - Mostly centralized.
    - Tightly coupled data storage to processing component.
    - Few mature implementations.
    - Examples: Plato, LittleTable, VergeDB, Chronos, Apache IoTDB

State of the Art
Time Series Management Systems

Categorization based on architecture:

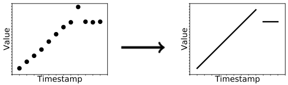
- External Data Stores:
  - Predominantly distributed.
  - New processing engine on top of external data store.
  - Most number of mature implementations.
  - Examples: Apache Druid, Bolt, Gorilla, BTrDB and ModelarDB
- Extension for RDMS:
  - Extends popular RDBMS.
  - predominantly centralized.
  - Small number of mature implementations.
  - Examples: Chronix, EdgeDB, and Heracles

State of the Art

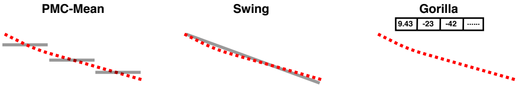
ModelarDB

ModelarDB

- Individual time series can be described with models:
- E.g., $v = a * t + b$ can represent a sub-sequence using only a and b.



- Uses Apache Cassandra for storage and Apache Spark for query processing.
- Approximates the time series values using mathematical functions (models) and stores only model coefficients.
- Currently includes three different model types:

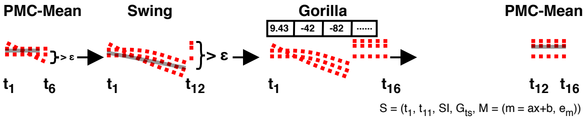


Jensen et al. 2021

ModelarDB

Correlated Time Series

- A data set often contains redundant information across time series:
  - E.g, co-located temperature sensors often produce similar values.
- ModelarDB can group correlated time series together and compress them as one stream of models to reduce the storage required.
- A list of model types fit models to data points, e.g., a constant (PMC-Mean), linear (Swing), and lossless (Gorilla) model type:



(Jensen et al. 2021)

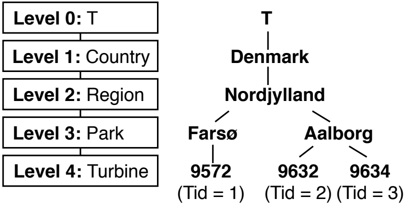
ModelarDB

Multi-model Group Compression

- Time series are grouped based on user hints given using primitives.
- The primitives can be combined and allow users to state that series are correlated based on their source or their dimensional hierarchy.
- Users can use their domain knowledge, analyze historical data, or use ModelarDB's automatic grouping method built on the primitives.

Grouping 9632 and 9634:

- From specific sources: 9632 and 9634
- Sharing a specific member: Location 3 Aalborg
- Share members until a level: Location 3
- The dimension's distance: 0.25
- Automatically (distance): auto



(Jensen et al. 2021)

ModelarDB

Open Issues

Open Issues

- No integrated functionality for evaluating the efficiency and use of model types;
- Model types are quite generic;
- Sampling interval and error bound could be changed dynamically;
- System only supports automatic and manual grouping using heuristics (domain knowledge and metadata);
- Time series must be ordered and have a regular sampling interval;

Project Objectives
Research Questions

RQ1: How can we efficiently evaluate the compression performance of model types and the quality of compression to varying error bounds of ModelarDB on different datasets?

RQ2: Depending on the the outcomes of RQ1, what other model types can be implemented to improve the compression and query performance of ModelarDB on real-life RES datasets?

RQ3: How can time series automatically be grouped using different correlation statistics and provided heuristics during the ingestion process?

RQ4: How can model-based ingestion of time series with a dynamic sampling interval and error bound be supported in ModelarDB?

Work and Publication Plan

Work and Time Plan

Time	Plan
Semester 1, Spring' 22 (Home)	Literature study Problem formulation Preparation of the Doctoral Project Plan Begin work on Paper 1 Begin development of ModelarDB performance evaluation tool Data collection Start general and project-related courses Trying out ModelarDB on real-life datasets and analyzing the results
Milestones	**Submission of 2-month Doctoral Project Plan** **Establish collaboration with secondment partner and get access to data**
Semester 2, Fall' 22 (Home)	Continue project-related courses Develop and test performance evaluation tool for ModelarDB
Milestones	**Submission of Paper 1** **Submission of 11-month Doctoral Project Plan** **Secondment completed**

Work and Publication Plan

Work and Time Plan

Time	Plan
Semester 3, Spring' 23 (Host)	Develop new model types for ModelarDB Begin work on Paper 2
Milestones	**Solution design of new model types**
Semester 4, Fall' 23 (Host)	Refine and test new model types with real-life datasets Begin work on Paper 3
Milestones	**Submission of Paper 2**
Semester 5, Spring' 24 (Home)	Develop new method for correlation-based grouping in ModelarDB Begin work on Paper 4
Milestones	**Submission of Paper 3** **Completed all general and project-related courses**
Semester 6, Fall' 24 (Home)	Develop and test new method for compressing dynamic sampling intervals and model error bounds Writing the Thesis
Milestones	**Submission of Paper 4** **Submission of PhD Thesis**

Work and Publication Plan

Tentative Publication List

Tentative Title of Paper 1: A tool for analysis of the efficiency of model-based compression in ModelarDB
Type: Conference paper.
Description:

- Analysis of the efficiency of current model types deployed by ModelarDB.
- Integrated tool that explains the system performance and its usage of model types.
- Performance indicators and visualization.

Datasets: Siemens Gamesa Renewable Energy and ENGIE data.
Authors: A. Abduvakhobov, S.K. Jensen, C. Thomsen, T. B. Pedersen, E. Zimányi, T. Pasma.
Length: 12 pages.
Time of submission: September, 2022.

Outlet: IEEE BigData.

Work and Publication Plan

Tentative Publication List

Tentative Title of Paper 2: New model types to achieve better compression rate and lower error bound for ModelarDB.
Type: Conference paper.
Description:

- Develops new model types for better ingestion and storage use.
- Mainly tailored to match real-life use cases.
- Novel time series compression method.

Datasets: Siemens Gamesa Renewable Energy and ENGIE data.
Authors: A. Abduvakhobov, S.K. Jensen, C. Thomsen, T. B. Pedersen, E. Zimányi, T. Pasma.
Length: 12 pages.
Time of submission: June, 2023.

Outlet: EDBT.

Work and Publication Plan

Tentative Publication List

Tentative Title of Paper 3: Automatic grouping of time series by deploying correlation statistics in ModelarDB
Type: Journal paper.
Description:

- More optimized and faster grouping (possibly in a streaming fashion).
- Leverages correlation and other statistical attributes of time series.
- Supported by user heuristics and metadata.

Datasets: Siemens Gamesa Renewable Energy and ENGIE data.
Authors: A. Abduvakhobov, S.K. Jensen, C. Thomsen, T. B. Pedersen, E. Zimányi, T. Pasma.
Length: 12 pages.
Time of submission: March, 2024.

Outlet: PVLDB.

Work and Publication Plan

Tentative Publication List

Tentative Title of Paper 4: Adding dynamic sampling intervals and error bounds for time series ingestion of ModelarDB.
Type: Conference paper.
Description:

- Dynamic error bound and sampling interval.
- User controls the error bound and sampling interval.
- More fine-grained data for exceptional cases.
- Also discusses automatic adjustment of error bound.

Datasets: Siemens Gamesa Renewable Energy and ENGIE data.
Authors: A. Abduvakhobov, S.K. Jensen, C. Thomsen, T. B. Pedersen, E. Zimányi, T. Pasma.
Length: 12 pages.
Time of submission: October, 2024.

Outlet: ICDE.

Plan for PhD Courses

Course Name	At	Type	ECTS	Time	Status
General Courses	AAU	General	13.75	’22-’25	Planned
Project-related Courses	AAU	General	17	’22-’25	Planned
Winter School (ARC)	ARC	General	3	Spring’22	Mandatory
Summer School (ULB)	ULB	Project	3	Summer’22	Mandatory
Winter School (AAU)	AAU	General	3	Winter’22	Mandatory
Summer School (UPC)	UPC	Project	3	Summer’23	Mandatory
Conference Attendance	TBD	Project	3	TBD	Planned
Danish Lessons	TBD	General	TBD	TBD	Mandatory
Total: 30.75					

Secondment

Partner Organization: Siemens Gamesa Renewable Energy
Secondment Supervisor: Tjip Pasma
Secondment start date: March 21, 2022
Data: power, voltage, reactive power, controller measurements, and other metadata
Size: ≈110TB

Out[13]:

	TimeStamp	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	...	[illegible]
0	2021-10-01 16:00:00.123	1	200.0	173.825897	165.654297	165.905594	165.905380	173.295059	-9.18	200.0	...	
1	2021-10-01 16:00:00.273	1	200.0	173.824905	165.663910	165.903992	165.905212	173.291351	-9.18	200.0	...	
2	2021-10-01 16:00:00.423	1	200.0	173.825897	165.673157	165.902466	165.905060	173.287689	-9.18	200.0	...	
3	2021-10-01 16:00:00.570	1	200.0	173.821899	165.680008	165.900955	165.904907	173.284042	-9.18	200.0	...	
4	2021-10-01 16:00:00.723	1	200.0	173.820908	165.723297	165.900543	165.904861	173.280426	-9.18	200.0	...	
5	2021-10-01 16:00:00.870	1	200.0	173.820908	165.806335	165.901611	165.904968	173.276825	-9.18	200.0	...	
6	2021-10-01 16:00:01.023	1	200.0	173.824905	165.848724	165.902130	165.905014	173.273270	-9.18	200.0	...	
7	2021-10-01 16:00:01.173	1	200.0	173.825897	165.911987	165.903549	165.905151	173.269730	-9.18	200.0	...	
8	2021-10-01 16:00:01.323	1	200.0	173.821899	165.975845	165.905457	165.905350	173.266220	-9.18	200.0	...	
9	2021-10-01 16:00:01.470	1	200.0	173.825897	166.084717	165.909119	165.905716	173.262741	-9.18	200.0	...	

10 rows × 50 columns

Bibliography

[1] Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen, "Time Series Management Systems: A Survey", *TKDE, 29(11)*, 2017.

[2] Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen, "ModelarDB: Modular Model-Based Time Series Management with Spark and Cassandra", *PVLDB, 11(11)*, 2018.

[3] Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen, "Demonstration of ModelarDB: Model-Based Management of Dimensional Time Series", in *SIGMOD*, 2019.

[4] Søren Kejser Jensen "Model-Based Time Series Management at Scale", *PhD Thesis*, 2019.

[5] Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen, "Scalable Model-Based Management of Correlated Dimensional Time Series in ModelarDB_{+}", in *ICDE*, 2021.

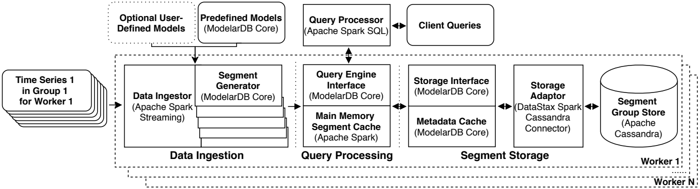
Thank you!

Abduvoris Abduvakhobov
abduvorisa@cs.aau.dk

AALBORG UNIVERSITY
DENMARK

ModelarDB Architecture

- ModelarDB is a portable Java library (ModelarDB Core) interfaced with a query engine (Apache Spark) and storage (Apache Cassandra).



- The architecture of a worker consists of three sets of components: Data Ingestion, Query Processing, and Segment Storage.

Reusing slides by Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen

ModelarDB Segment Structure

Segment

Gid (PK)	StartTime (PK)	Gaps (PK)	EndTime	Mid	Parameters
1	1460442200000	[]	1460442620000	1	0x3f50cfc0
3	1460642900000	[2]	1460645060000	2	0x3f1e ...
...	...	...	...	...	...

Model

Mid (PK)	Classpath
1	PMC-Mean
2	Swing
3	Gorilla

Time Series

Tid (PK)	Gid	Scaling	SI	Country	Region	Park	Entity	Level 1	Level 2	Level 3	
1	1	1.0	60000	Denmark	Nordjylland	Farsø	9572				
2	3	1.0	30000	Denmark	Nordjylland	Aalborg	9632				...
3	3	4.75	30000	Denmark	Nordjylland	Aalborg	9634				
...	...	...	...	...	...	...	...				
				Location Dimension				2nd Dimension			

- Time Series and Model store metadata for time series and model types.
- Segment stores sub-sequences of time series as segments with a model.

Reusing slides by Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen

ModelarDB

Model-Based Time Series Management at Scale

October 30th, 2020

Søren Kejser Jensen, Torben Bach Pedersen, Christian Thomsen
skj,tbp,chr@cs.aau.dk

Center for Data Intensive Systems, Daisy
Department of Computer Science
Aalborg University
Denmark

AALBORG UNIVERSITY
DENMARK

daisy
Center for Data-Intensive Systems

Swing
Implementation

- Models can always reconstruct their data points within the error bound:

```
@Override
protected float get(long timestamp, int index) {
    return (float) (this.a * timestamp + this.b);
}
```

- Many aggregates (e.g, SUM) can be computed directly from the models:

```
@Override
public double sum() {
    double first = this.a * this.getStartTime() + this.b;
    double last = this.a * this.getEndTime() + this.b;
    double average = (first + last) / 2;
    return average * this.length();
}
```

Agenda

Configuration

Using ModelarDB


```
# Engine, Storage, and Query Interface
modelardb.engine spark://10.0.0.1:7077     # local,spark
modelardb.storage cassandra://10.0.0.1     # RDBMS,Cassandra,more
modelardb.interface http                   # none,socket,file,http

# Engine, Storage, and Query Interface
modelardb.source /home/user/DataSet/*.gz   # filepath,ip:port
modelardb.dimensions /home/user/dimensions.txt
modelardb.correlation auto                 # primitives

# Model Types
modelardb.model dk.aau.modelardb.core.models.PMC_MeanModel
modelardb.model dk.aau.modelardb.core.models.SwingFilterModel
modelardb.model dk.aau.modelardb.core.models.FacebookGorillaModel

# Error Bound, Length Limit Bound, and Latency Bound
modelardb.error 1       # per data point error bound in percentage
modelardb.limit 50      # length limit used by lossless model types
modelardb.latency 0     # maximum latency in terms of data points
```

Deployment
Using ModelarDB

- ModelarDB is simple to run standalone on a single node:

```
java -jar target/scala-2.11/ModelarDB-assembly-1.0.jar
```

- ModelarDB is simple to run on a Spark cluster by using `spark-submit`:

```
spark-submit
--conf spark.sql.parquet.filterPushdown=true
--conf spark.sql.orc.filterPushdown=true
--conf spark.cassandra.connection.host=10.0.0.2
--conf spark.hadoop.dfs.replication=1
--driver-memory 4G
--executor-memory 3G
--master spark://10.0.0.1:7077
--jars UserDefinedModels
--class dk.aau.modelardb.Main
ModelarDB-assembly-1.0.jar
```

- Additional query engines can easily be interfaced with ModelarDB Core.

Data Point View

Using ModelarDB


```
# Client
curl -d 'SELECT MIN(val) FROM DataPoint WHERE sid = 17861' ...
{
  "time": "PT0.55S",
  "query": "SELECT MIN(val) FROM DataPoint WHERE sid = 17861",
  "result":  [
  {"min(val)":48.0}
  ]
}

# Server
Data Point required columns{ val sid }
Data Point filters{ EqualTo(sid,17861) }
Segment required columns{ sid st et res mid param gaps }
Segment provided filters{ EqualTo(sid,17861) }
Segment rewritten filters{ EqualTo(gid,15408) }
Cache miss
Constructed predicates(gid=15408, takeWhile(st <= null, Gid IN )
Caching RDD
```

Segment View
Using ModelarDB


```
# Client
curl -d 'SELECT MIN_S(#) FROM Segment WHERE sid = 17861' ...
{
  "time": "PT0.295S",
  "query": "SELECT MIN_S(#) FROM Segment WHERE sid = 17861",
  "result":  [
  {"maxs(sid, st, et, res, mid, param, gaps)":48.0}
  ]
}

# Server
Segment required columns { sid st et res mid param gaps }
Segment provided filters { EqualTo(sid,17861) }
Segment rewritten filters { EqualTo(gid,15408) }
Cache hit
```

- The Segment View has UDAFs for simple and aggregates over time.
- Multi-dimensional queries are supported with the use of `GROUP BY`.

Agenda

Environment

Performance

- The evaluation uses real-life data sets from the energy domain.
- Most experiments are performed on a modest local cluster.
- Scalability experiments are performed using Microsoft Azure.
- ModelarDB configurations: with no grouping (MDB_{+}-G), with `auto` (MDB_{+}+GA), and with the best primitives per data set (MDB_{+}+GB).
- For comparison the evaluation includes the state-of-the-art big data systems and file formats most commonly used in industry.[1]
  - InfluxDB, Apache Cassandra, Apache Parquet, and Apache ORC.
- No model-based time series management system is publicly available.
- Small, large, and multi-dimensional aggregates are evaluated using:
  - A Java Library (J), a Spark DataFrame (F), the Segment View (S), or the Data Point View (DP).

[1] From meeting with companies, our survey, and https://db-engines.com/en/

Data Sets
Performance

EP 45,353 time series collected from energy producers with a sampling interval of 60s and occupying 339 GiB as uncompressed CSV.

- It contains two dimensions with each containing two levels:
  - **Production:** *Entity* $\rightarrow$ *Type* $\rightarrow \top$
  - **Measure:** *Concrete* $\rightarrow$ *Category* $\rightarrow \top$
- ActiveProductionMWh, AirPressure, DerateDetectionStage, EnergyInAccumulationTankMWh, EstimatedProductionMWh, EstimatedReductionMWh, EstimatedRunningCapacityMw, EstimatedSetPointPercentage, HorizontalIrradiance, Humidity, LinearScaledProductionMWh, NacelleDirection, PossibleProdMwh, Temperature, TiltedIrradiance, Tso, TsoOnshore, WindDirection, WindSpeedEastWest, WindSpeed, WindSpeedNorthSouth
- MDB_{+}+GA time series with the same category are grouped per entity.
- MDB_{+}+GB energy production measurements are grouped per entity.

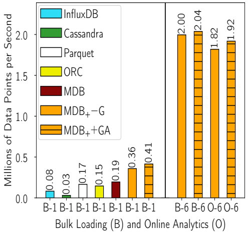
Data Sets
Performance

EF 197 time series collected from wind turbines with a sampling interval of 200ms and occupying 372 GiB as uncompressed CSV.

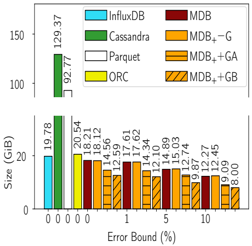
- It contains two dimensions with three and two levels respectively:
  - **Location**: *Entity* → *Park* → *Country* → ⊤
  - **Measure**: *Concrete* → *Category* → ⊤
- AlmActCount, BecBulbSt, ConSt, DateTimeUTCDiff, ExTmp, GnTmpBrg1, GnTmpBrg2, GriPF, GriPhV1, GriPhV2, GriPhV3, Hz, PtAngValBl1, RotSpd, Spd, VAr, W, YwSt
- MDB_{+}+GA groups all time series on concrete measure and park.
- MDB_{+}+GB in general, group time series on the concrete measure and country, but time series are also grouped on the category and park.

Ingestion and Storage
Performance

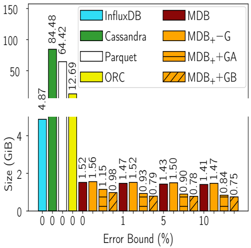
- Results with groups of correlated time series are shown with stripes.



Ingestion Rate, EP



Storage Used, EP

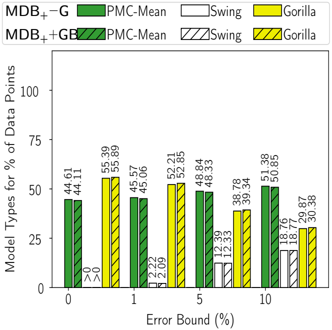


Storage Used, EH

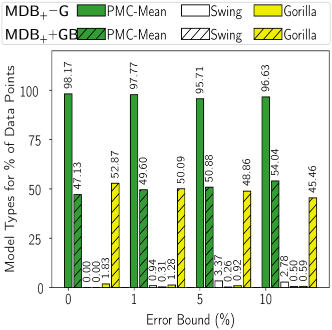
- Ingestion rate is 2.12–13.7 times faster than the industry formats.
- The storage required is reduced by 1.09–16.17 times for EP and by 3.12–112.64 times for EH compared to the industrial formats.
- Grouping series increases ingestion rate and can lower storage usage.

Model Types

Performance



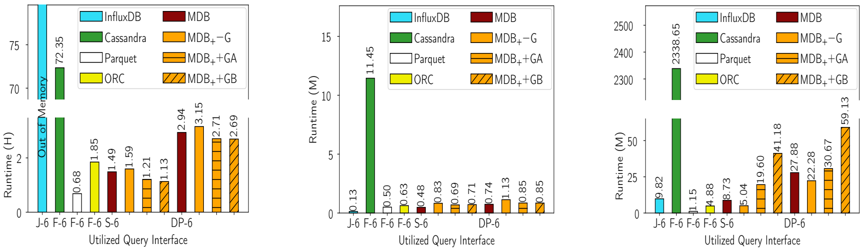
Model Types, EP



Model Types, EH

- ModelarDB automatically adapts to each data set and error bound.
- Grouping increases ModelarDB's use of the lossless model type.

Aggregate Queries
Performance

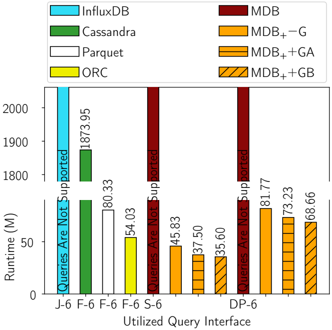


Large Scale, EP   Small Scale, EP   Small Scale, EF

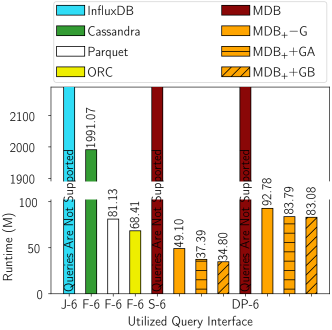
- Grouping time series decreases query time for large scale aggregate queries, but can increase query time for small scale aggregate queries.
- Experiments run on Azure show that ModelarDB scales linearly.

Multi-Dimensional Queries EP

Performance



Month and Category, EP

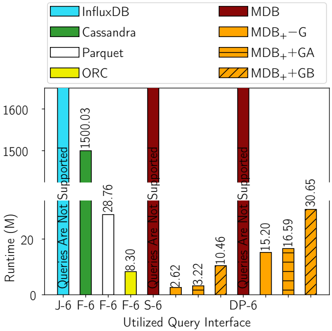


Month and Concrete, EP

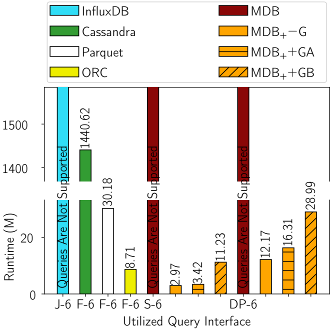
- Grouping reduces query time as each query only reads whole groups.

Multi-Dimensional Queries EF

Performance



Month and Park, EF



Month and Entity, EF

- Grouping increases query time as the cluster is not fully utilized.