DBRepo: A Data Repository System for Research Data in Databases

Martin Weise
Data Science Research Unit
ISE, TU Wien
Vienna, Austria
martin.weise@tuwien.ac.at

Andreas Rauber
Data Science Research Unit
ISE, TU Wien
Vienna, Austria
andreas.rauber@tuwien.ac.at

2024 IEEE International Conference on Big Data (BigData) | 979-8-3503-6248-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/BigData62323.2024.10825401

***Abstract*—In the era of big data, research has become increasingly data-driven, with vast amounts of information being generated and analyzed to produce new insights and discoveries. This data deluge requires a combination of methods and technologies to store, process, share and preserve research data. With many of the world's most valuable data being stored in relational databases where it evolves over time as new knowledge is gained and old knowledge invalidated, current repository systems fail to provide researchers with interfaces to conveniently work with this kind of data within their research environments. For this reason, we have developed *DBRepo*, an institutional data repository for research data in databases (DBRepo) supporting guidelines of the Working Group on Data Citation of the Research Data Alliance. The system has been in use at TU Wien for almost three years now and provides a variety of data science-related interfaces and can be integrated into many workflows and tools. Further, it assists researchers in depositing their datasets by suggesting the table schema (column names, data types, primary key constraints) and it addresses data interoperability issues by suggesting semantic concepts for dataset columns and units of measurements, where applicable. DBRepo is currently in use by six universities globally who use it as data store for hot and cold research data sets. In the paper, we describe their use-cases and provide lessons learned from the various deployments and workflows. Finally, we show how depositing research data into DBRepo increases the data's visibility.**

***Index Terms*—research data repositories, relational databases, research data management**

I. Introduction

The increasing complexity of research and the growing popularity of data-centered research domains such as machine learning created a deluge of research data [1] that is available for reuse to the global research community beyond the scope of its original intent. With data becoming first-class research output such that many journals, conferences and funders consider data publication practices more critically, mandating researchers to deposit the experiment data into data repositories where it is preserved. The data receives a license for reuse and then can be accessed through protocols and tools prevalent in the research community.

Many of the world's most valuable data is stored in relational databases who ensure that the data remains consistent and durable as data evolves over time as new knowledge is gained. The rate of which new knowledge is gained and old knowledge potentially invalidated ranges from few months (e.g. in machine learning) to decades (e.g. in humanities). At the peak of this ongoing process, researchers are publishing their findings in journals or conference proceedings to share it with their domain-specific research community. In the last decade, researchers started to link experiment data to those paper outputs with the use of data repositories (instead of raw data archives on e.g. a web server) that preserve the data. These store machine-readable metadata in addition to hosting the experiment data and offer a wide range of machine-actionable interfaces to find (i.e. indexing in metadata aggregators) and access the (meta-)data to work with it.

A naïve approach to preserve a relational research database in a repository would consist of depositing a (compressed) relational database dump into a data repository and link the persistent identifier of the dump in the paper. Typically, no curation happens after depositing the dump in a data repository, creating preservation issues of handling end-of-life software decades later. It also creates several issues for the researcher that wants to reuse the relational database dump: (i) allocate storage for download and optional extraction, (ii) install the relational database version used, (iii) import into the relational database, (iv) formulate queries to explore the data and create a subset. This approach not only delays the time when research can happen, but also requires compute infrastructure (i.e. a notebook with sufficient resources to run the database), storage, technical knowledge how to install a database and import the dataset and time to perform these steps from the researcher that wants to explore the data to verify if the data is useful for research. Going forward and assuming the researcher found reusable data and wants to publish her research as journal paper with the subset(s) used as supplementary material. How can she efficiently make those available to others? Releasing them as new datasets in data repositories, potentially creating 10's or 100's of similar data dumps? What happens if the original data is modified and released to a new version?

In the context of big data, these problems quickly become severe as the above data publication strategy no longer works and the skills required to manage the storage increase drastically. They further motivate a separation of concerns where the university IT-department maintains a on-demand database infrastructure with abstract interfaces to import data (such

as database dumps, data streams/feeds from sensors, datasets as .csv/dataframe) that researchers from many domains know how to use and a team of data stewards curating the data for future (re-)use.

To help address this concern, we propose a standardized research database repository infrastructure (DBRepo) and data analytics process for proper management, preservation, retrieval, exploration and description for research data in databases. Specifically, we aim to define the data capturing and data management approaches from experience gathered of operating DBRepo for almost three years as well as six friendly-user universities operating DBRepo on their own on-premise infrastructure. The contents covered in this paper extend our first publication [2] by great extent.

The rest of the paper is organized as follows: Sec. II surveys methods to organize relational research data in repositories and similar systems as well. Sec. III proposes our system design from a technical perspective, Sec. IV describes use cases observed during our friendly-user evaluation period and Sec. V outlines the capabilities for researchers. In Sec. VI we describe challenges encountered and how we solved them and finally in Sec. VII, we conclude on the paper and give a perspective on future work.

II. Background

A relational database management system (RDBMS) organizes data in the form of tables that are connected with primary-foreign key relationships. Every interaction with the database (i.e. to read or manipulate data) requires a transaction that ensures that data is visible to other transactions in a consistent manner by enforcing the atomicity, consistency, isolation and durability (ACID) principles. Temporal features [3] in the SQL 2011 standard allow database engineers to assign one or more time periods denoted by its start timestamp t_{s} and end timestamp t_{e} in which the temporal data is valid, e.g. $(t_{1725148800}, \infty)$ denotes a temporal data period that is valid from UTC 2024-09-01 00:00:00 onward. Despite being present more than a decade, few relational databases such as MariaDB, DB2, Microsoft SQL Server/Azure Database, Oracle Database have decided to incorporate[1] them into their RDBMS.

Research data repositories are infrastructures that store research data with the purpose of making it findable, accessible, interoperable and reusable [4] (FAIR) for machine and human agents to access it and provide a basis for future research. They are acting as a central point for researchers to deposit all their raw (e.g. sensor measurements) and processed (e.g. cleaned analysis results) research data. Depending on their scope, some research data repositories collect any data (generalist repository) to preserve as much research data as possible, or only consider data from a specific domain to offer a domain-specific list of features to increase the utility of the research data repository towards reusability. To make the data findable, the researcher needs to provide metadata (data about the data) in a structured form, following a protocol or standard (e.g. DataCite [5], Dublin Core[2] or discipline-specific standards such as the FGDC digital cartographic standard for geologic map symbolization [6].

With this metadata, a persistent identifier (PID) can be generated that points to the data, shows the metadata entered in the previous step in a human- and machine-friendly interface, enabling both to traverse through the repository's metadata catalog and explore the dataset, making it findable and interoperable by machines and humans through open interfaces. This increased availability of first-class research data enhances the reuse of research data and automates retrieval of relevant data. This representation on how research data exists on the Internet has motivated discussions on how to improve the representation of research data on the Internet to a machine-readable and -actionable form [7] based on metadata links embedded in the website code.

The above outlines why research data repositories are not just file-/web servers that archive research data and make them public, but offer a wide range of additional features/services that make them a *research* data repository like the Weizmann Interactive Supernova Data Repository [8] that is a domain-specific research data repository. It archives an extensive collection of supernovae spectroscopic data (at time of writing 64.000 of which 81% are public) in a database and offers an interactive web-based graphical interface to explore and reuse scientific data. The development was heavily motivated by the volume of spectroscopic data that "can no longer be easily described in single publications" and data sharing in mind.

The Research Data Alliance (RDA) is a loose research community organization centered around enabling open sharing of data across disciplines, technologies and borders. Despite having no dedicated budget/resources for member activities to contribute towards coming closer to these goals, many (volunteer) working groups have been established to solve a "low hanging fruit" problem and achieve endorsement by this community of data. The Working Group on Data Citation (RDA-WGDC) released such community-validated and RDA-endorsed recommendations on precisely and persistently identifying and citing arbitrary subsets of dynamic data [9].

The Working Group on Dynamic Data Citation developed and published 14 recommendations [9] centered around time stamping, versioning evolving data sources and identifying subsets dynamically via persistent identifiers (PIDs) that store metadata along with the data to ensure a minimal set of information is always available even when the original data is lost. Instead of assigning PIDs to the materialized data (e.g. files in InvenioRDM, DSpace [10], Dataverse [11], Dryad [12]), Rauber et al. propose to assign PIDs to queries which select the subsets instead. Known adoptions include Ocean Networks Canada [13] whose observatories on Canada's west coast collect marine data and store them in databases routinely. The data is available in real time to the general public through a

[1]We excluded PostgreSQL from this list since it lacks native support, temporal tables can only be used through community-developed extensions and not out of the box.

[2]https://www.dublincore.org/specifications/dublin-core/dces/

web portal[3], demonstrating the high utility of using databases compared to e.g. CSV/JSON/PARQUET files. Despite the recommendations being generic and for "virtually any data", we focus only on relational databases.

The Open Archives Initiative developed the Open Archives Initiative Protocol for Metadata Harvesting[4] (OAI-PMH) to create a specification for research data repository interoperability. A research data repository such as DBRepo exposes structured metadata that can be harvested by agents that interactively query the repository via the OAI-PMH protocol.

The Swiss Federal Archives developed a specification for Software Independent Archival of Relational Databases[5] (SIARD) with the aim of long-term preserving relational databases such as DBRepo's data database(s) in an open file format, independent of (proprietary) dump formats of relational databases implementing the XML- or SQL standard where each database is packaged into a container file that is preserved.

III. System Design

We define the technical building blocks of data repository systems for research data in databases such as DBRepo to be: (i) data infrastructure (resources), (ii) research data repository (metadata management), and (iii) database management system (data management and minimal query metadata for database preservation).

A. Data Infrastructure

With the increased demand and race towards more resources that require steady investment, -maintenance and -availability many organizations started outsourcing their data infrastructure, losing intellectual capital on how to operate such infrastructures in the future in the long run. Only few (mostly technical) organizations such as technical universities invest into physical infrastructure and personnel maintaining the infrastructure. Some organizations join shared infrastructure consortia to steadily invest into new infrastructure (e.g. VSC[6] super computer).

With increasing network communication speed and cost of transmitting information essentially being free for anyone, the micro service architecture [14] style is becoming the gold standard of modern service-oriented information systems engineering that supports scaling by design. Packaging the micro services into containers that are managed by a container orchestration tool (e.g. Kubernetes, Apache Mesos) that largely automates the operation of a micro service application. The micro service architecture (c.f. Fig. 1) of DBRepo enables horizontal scaling in e.g. Kubernetes[7]. We use externally maintained software such as MariaDB Galera Cluster[8] for e.g. the database holding the research data (data-db).

[3]https://data.oceannetworks.ca/
[4]https://www.openarchives.org/pmh/
[5]https://siard.dilcis.eu/SIARD_2-2/format/2021-08-31/SIARD_2-2.pdf
[6]https://vsc.ac.at/
[7]https://kubernetes.io/
[8]https://mariadb.com/kb/en/galera-cluster/

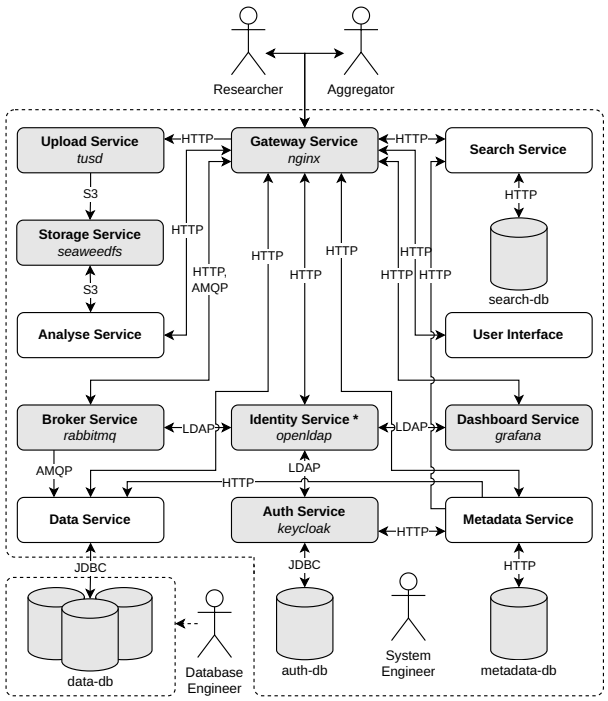

Fig. 1: DBRepo's micro service architecture. White boxes are micro services developed and maintained by TU Wien, gray boxes are external micro services.

Analyse Service assists the researcher in depositing datasets by suggesting column names and types, enumerations, primary keys, semantic concepts and units of measurements for machine-understandable context.

Auth Service authenticates users of DBRepo based on their group membership in the Identity Service and returns a set of roles associated with the group that allow accessing certain resources or perform actions.

Broker Service operates a single exchange and distributed (quorum) topic message queue and pushes AMQP-tuples to the Data Service who adds it to a table from e.g. sensors.

Dashboard Service visualizes databases (table data preview, statistical properties of numerical columns) to explore datasets for reuse. Detailed dashboards can be added manually, e.g. to display air quality[9] in Austria.

Data Service manages the research data in research databases using object relational mapper-specific logic currently supporting MariaDB-based databases.

Gateway Service bundles the communication to the micro services as a single point of entry in the Docker Deployment. For cloud deployments this micro service will be replaced by an ingress controller or load balancer.

Identity Service stores the identity credentials. For produc-

[9]https://dbrepo1.ec.tuwien.ac.at/dashboard/d/FLB9eAv4z/airquality

tion deployments, this service is replaced with the organization's (federated) identity management system, granting the user access to certain resources based on their affiliation.

Metadata Service manages the metadata generated by the system (e.g. database size, table rowcount, user access) or input by the researcher (e.g. persistent identifier, table schema, semantic concepts and/or units of measurement per table column).

Upload Service deposits datasets into the Storage Service.

Search Service mirrors metadata related to databases (e.g. table information, schema, semantic concepts and units of measurement per table column) for advanced search queries through the User Interface.

Storage Service provides a file system for importing datasets into the research databases and exporting datasets as files temporarily (they are being deleted automatically after a timeout).

User Interface facing the vast majority of researchers and machine agents (e.g. web crawlers) to read information and to change the system state.

B. Research Data Repository

Suppose a researcher wants to collect observation data from sensor measurements in a relational database that is used for some machine learning-related task at a later stage. Then, she needs to create a virtual machine in a cloud infrastructure, install and configure the database and only then can start the actual work. The burden of setting up databases and maintaining them (until the end of a project where they are deposited into a research data repository) still remains with the researcher, despite this not being their primary concern. Similarly, the researcher who wants to reuse the data from the research data repository also has to setup a data as outlined earlier in this paper.

We therefore propose a separation of concerns for professional research data management for institutional research data repositories into the roles of Researcher, Aggregator, System Engineer, Database Engineer and Data Steward. We want to emphasize that the roles of System Engineer and Database Engineer are not mutually exclusive.

Researcher deposits and works with research data.

Aggregator indexes and queries research data.

System Engineer monitors the deployment, performs backups and applies updates & security patches.

Database Engineer deposits large datasets that cannot be imported through the User Interface (i.e. larger than the upload limit), assists in integration of research data into tabular schema, generates subsets that are not supported by DBRepo, i.e. contain problematic keywords (c.f. Sec. III-C).

Data Steward curates the deposited data according to the organization's policy, assists in selecting ontologies for semantic concepts and units of measurements and ensures the preservation of research databases.

TABLE I: Data Infrastructure Matrix

RDA-WGDC Recommendation	Research Database	Data Service	Metadata Service	User Interface
R1 Data Versioning	✓			
R2 Timestamping	✓			
R3 Query Store Facilities	✓			
R4 Query Uniqueness		✓		
R5 Stable Sorting		✓		
R6 Result Set Verification		✓		
R7 Query Timestamping	✓			
R8 Query PID	✓			
R9 Store Query	✓			
R10 Automated Citation Texts			✓	✓
R11 Landing Page			✓	✓
R12 Machine Actionability		✓	✓	✓
R13 Technology Migration	✓			
R14 Migration Verification	✓			

The curation of the deposited data entails a surface-level understanding of the data by the Data Steward and relies on cooperation from the Researcher to e.g. provide a description of the data in natural language and a detailed elucidation of schema context to be able to assign a semantic concept and/or unit of measurement to a column. In DBRepo, the Data Steward additionally elicits database views from the researcher, i.e. what meaningful subsets can be presented to researchers that want to reuse the data later.

Contrary to discipline- or domain-specific repositories who are aiming at completeness and providing a one stop shop for researchers from heterogeneous affiliations depositing, analyzing and reusing research data of narrow scope, an institutional repository [15] is only available to researchers of a certain institution, promoting a decentralized scholarly infrastructure approach that can organize in federations to share these infrastructure resources among trusted institutions.

C. Database Management System

A relational database is an efficient data storage for data. In response to the rising expectations of data provisioning in research databases, database engineers are challenged with: (i) Data versioning (ii) Data preservation (iii) User-transparent reproducibility. The RDA-WGDC recommendations [9] discuss these challenges for relational research databases in a proof-of-concept implementation. Following the recommendations further, a research database must implement the query store in the database itself but can shelve modifying operations into the application. Since relational databases are excellent computation resources for transactional data (i.e. when data is created, read, updated or deleted) and for database preservation purposes, recommendations on versioning and query store preparation need to be implemented directly in the database.

In DBRepo, we implemented data versioning and time stamping (R1-2, c.f. Table I) via enabling system versioned tables in the research database (MariaDB). System versioned tables maintain the (t_s, t_e) validity period transparent to the

user such that select-queries with a timestamp select a subset with data being valid at this timestamp. Further, DBRepo implements R3 directly in the research database by creating a special table `qs_queries` in the research databases themselves to hold metadata about the query. DBRepo ensures query uniqueness, stable sorting and result set verification (R4-6) in the application logic of the Data Service by rewriting the query and computing a `sha256`-checksum of both the normalized query and the result set and stores this information in the query store. On subset generation, the query that produced the subset is stored in the query store with the current timestamp (R7). In case the (normalized) query- and the result hash are unique, the query is assigned a new ID (R8) and query metadata is added to the query store as new row (R9) through the use of stored procedures. Since the information about PIDs is managed by the Metadata Serive, this service is also responsible for assisting the researcher in metadata citation and the landing page (R10-12). For completeness, we note that R13-14 are only applicable in ex post migration scenarios, thus affecting the research database.

Additionally to the RDA-WGDC recommendations listed above, a research database must ensure reproducibility at all times, even in presence of e.g. floating point operations leading to different results without modifying the underlying data [16] [17]. This can be achieved by denying problematic keywords that are known to (potentially) leading to non-deterministic subset results. DBRepo therefore disallows problematic keywords[10] and the $\star$ for selection. Further, to provide the necessary machine-understandable context, a research database must assign each table column a semantic concept and unit of measurement (through linking it with the ontology entity).

To the best of our knowledge, no (commercial) database vendor implements the RDA-WGDC recommendations and our additional requirement on a sufficient level such that the database can be considered a research database.

IV. Use Cases

This selection of use cases shows how researchers use DBRepo. A full list of demonstrative use cases is available on the documentation website[11].

A. Teaching Industry 4.0

Industry 4.0 initiatives are expanding beyond the traditional manufacturing industry, influencing all three key employment sectors: primary, secondary, and tertiary. These advancements are impacting various fields, from agriculture to service industries. This trend highlights the importance of equipping the current and future workforce with digital knowledge.

In a position paper [18], we addressed the challenge of integrating practical application of this knowledge into the tertiary education system for TU Wien Pilot Factory [19] who focuses both on discrete, multi-variant series production and on production in very small quantities. The TU Wien Pilot Factory encompasses every stage of product development, from design through to assembly. The primary target group is small and medium-sized enterprises. Their specific needs and areas of interest have been identified through close collaboration with the participating institutes. The topics explored in both research and training, within the context of demonstration scenarios, focus on current industrial challenges and aim to provide potential solutions.

One of these demonstration scenarios that increasingly carry over to teaching at TU Wien is the usage of real-world data which is routinely collected at TU Wien Pilot Factory by a network of sensors and available as MQTT data stream (throughput 40 t/s, 4.3kB per second) for continuous internal monitoring. As part of a research unit collaboration, we send the power consumption by CNC mills to the Broker Service of DBRepo which routes them to the Data Service for storage in the research database, growing the dataset ≈ 95MB per day. In there, students will be able to use the data for exercises (e.g. introduction to machine learning, data oriented programming paradigms) within our research unit. For this, students receive access to a virtual research environment based on Jupyterhub [12] where the connection code to DBRepo is already present (in Python), letting the students work transparently with the live real-world data from a research data repository as if the data would be present on their machines. Until now, we collected around two years of live sensor measurement data with a rate of ≈ 40 tuples per second (one tuple every 0.025 seconds). Another aim that TU Pilot Factory explores is predictive maintenance [20] of components using sensor measurements.

Similarly, UTeM Teaching Factory[13] setup a small production line who focuses on providing a computer integrated manufacturing line to a local multi-national haberdashery manufacturer, combining teaching and industry needs. Further, we identified three requirements:

- data collection of sensors and actors (e.g. pressure valve) connected to a programmable logic controller
- separation of concerns, i.e. trained IT-personnel who maintains the deployment of DBRepo
- environment to compute on the collected data.

We recently, in September 2024, visited UTeM again to follow up on our joint position paper from two years ago and could integrate DBRepo deeper into the teaching process by collecting process data of the Reconfigurable Manufacturing System (c.f. Fig. 3) via the programmable logic controller (PLC) to DBRepo by integrating a MQTT-to-AMQP bridge adapter[14] we also use to collect sensor data from our Pilot Factory (c.f. Fig. 2). Unfortunately, their computer integrated manufacturing line for haberdashery packaging is no longer in operation, putting a halt to sensor data collection.

[10] `AVG`, `BIT_AND`, `BIT_OR`, `BIT_XOR`, `COUNT`, `COUNTDISTINCT`, `GROUP_CONCAT`, `JSON_ARRAYAGG`, `JSON_OBJECTAGG`, `MAX`, `MIN`, `STD`, `STDDEV`, `STDDEV_POP`, `STDDEV_SAMP`, `SUM`, `VARIANCE`, `VAR_POP`, `VAR_SAMP`

[11] https://www.ifs.tuwien.ac.at/infrastructures/dbrepo/1.5/examples/air/

[12] https://science.datalab.tuwien.ac.at/

[13] https://tf.utem.edu.my/

[14] https://gitlab.tuwien.ac.at/crdm/dbrepo-pilots/-/tree/master/worker-pilot-factory

Fig. 2: Siemens EMCO MaxxTurn TU Wien Pilot Factory

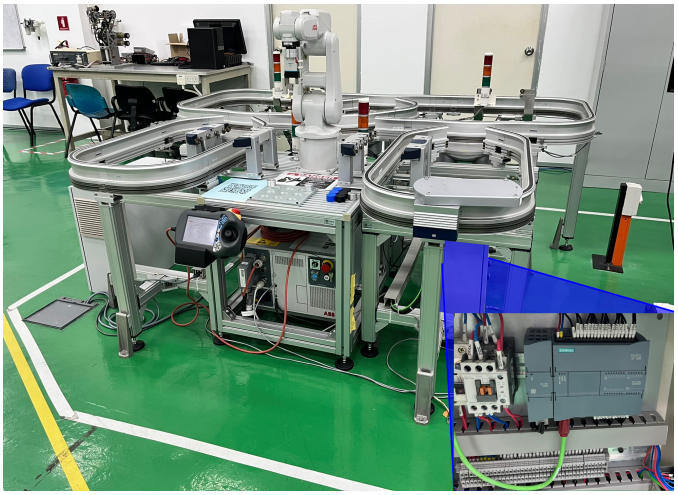

Fig. 3: UTeM Teaching Factory Reconfigurable Manufactoring System, highlighting the programmable logic controller that sends data to DBRepo.

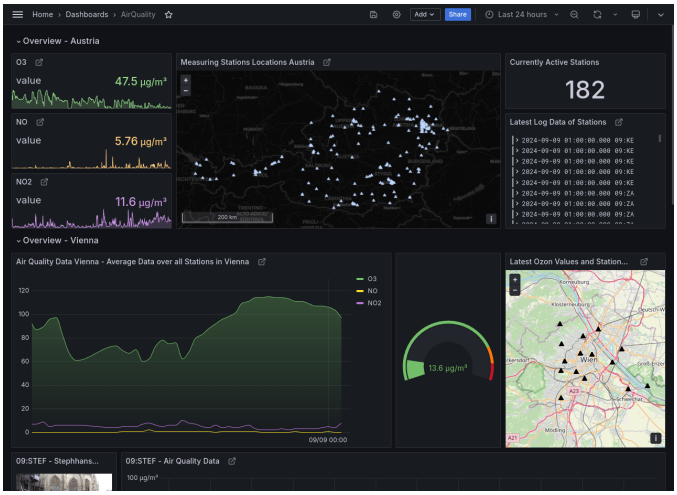

Fig. 4: Visualizing the routinely collected sensor data measurements in an interactive dashboard to monitor the sensor's update and explore the dataset[15].

B. Collecting Decades of Air Quality Data

The Environment Agency Austria is the national government agency responsible for monitoring the climate environment. It publishes an annual report [21] on harmful air quality emissions caused by humans (e.g. particle matter, substances related to Ozone, heavy metals), based on 178 sensor stations distributed all over Austria. While the data is publicly available in the form of an interactive map, the historic data dating back to 1990 is not open. As part of a academia-government cooperation, we received the complete historic dataset of sensor measurement data being collected in intervals of ≈ 30 minutes. To collect new measurements, we additionally scrape their public interactive map now for almost two years with a Python Flask application that appends new measurements to the historic dataset we received. Our dataset is updated every 15 minutes (half of the collection interval) and contains ≈ 7 million measurement tuples of ≈ 1 gigabyte storage.

Dashboards are one of the most common use cases for data visualization and interactive data exploration [22]. To demonstrate the utility of DBRepo's interfaces and showcase the live nature of the research database that is updated continuously, we visualize the dataset through a Grafana dashboard (c.f. Fig. 4). The dashboard[15] aims at communicating the data to researchers that are interested in researching on the dataset, interacting with the 34 years of detailed air quality data before the researcher needs to setup an experiment environment. Similar to the teaching industry 4.0 (c.f. Sec. IV-A), we want to use this data in summer term 2025 for teaching with real-world data.

C. Evolving Dataset

Water quality monitoring is an ongoing effort to assess the water quality that starts with selecting sampling sites,

[15]https://dbrepo1.ec.tuwien.ac.at/dashboard/d/FLB9eAv4z/airquality

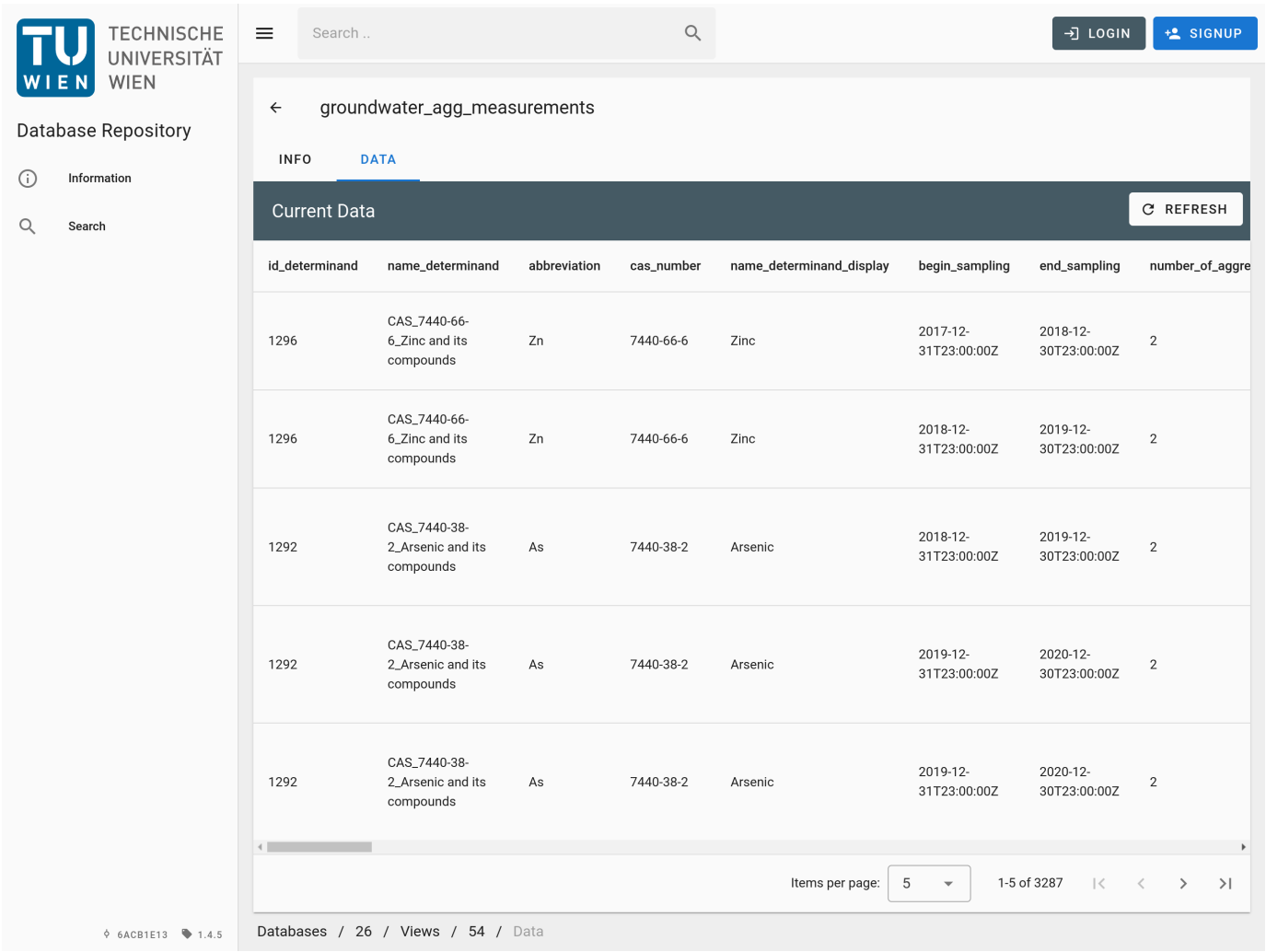

Fig. 5: User Interface of DBRepo, showing a database view of the Danube Hazard dataset that aggregates groundwater measurements. After login, the database can be further explored interactively through applying filters and generating subsets.

the water quality parameters, sampling frequency and leads eventually to implementation of actions [23] before starting another iteration. In order to contribute towards the European Water Framework Directive, mandating member states of the European Union (like Austria) to establish emission inventories for micro pollutants that enable evidence-based development of mitigation measures.

To this end, Kittlaus et al. [24] collected and harmonized (meta-)data on micro pollutant measurements across multiple compartments (river water, ground water, waste water, soil, etc.) and countries along Danube's course in a relational database[16] hosted at the Research Unit of Water Quality and Resource Management. Once the project ended beginning of 2023, they released the database in the following months in our institutional research data repository as database dump that can be reused publicly. This database contains 74 tables and 16 views and has a volume of $\approx$ 1.9GB. Following another year (2024), they released an updated version that has slightly more volume of $\approx$ 2GB, corrected errors that led to wrong results in the inventory and an extended the database schema. Once released, the latest release followed the next day, correcting coordinate data for soil sampling sites, motivating an excellent use case for using research databases that version evolving data automatically (c.f. Sec. III-C).

As part of a database interoperability migration, we mapped their first database dump that is available as plain text PostgreSQL dialect dump to MariaDB dialect SQL statements. Only minor adjustment needed to be made, e.g. naming conventions of internal procedure names, data type naming conventions, comparison operator renaming and a minor query rewrite of a database view. After import into our institutional DBRepo[17] demonstration instance, all data from the following

[16]Danube Hazard m^{3}c's project duration (2020-2022) funding the inventory collection was identical to FAIR Data Austria (2020-2022) funding the early development of DBRepo. Therefore using DBRepo to collect the data from the beginning was not an option.

[17]https://dbrepo1.ec.tuwien.ac.at/database/26/view/54/data

versions was imported retrospectively such that each created/updated/deleted row reflects the correct timestamp as the changes would have been made in the research database of DBRepo. This use case demonstrates the necessity of research databases implementing the RDA-WGDC recommendations and aims at increasing the reusability of the Danube Hazard dataset of Kittlaus et al. by making it available as research database that can be explored in the User Interface (c.f. Fig. 5) and reused with our multitude of open, documented and well-understood interfaces without having to setup a PostgreSQL database anywhere.

V. Working with Research Data in DBRepo

A. Data Science

Once a data science researcher has provisioned a relational database in DBRepo, she can start importing data from other repositories or use DBRepo as data store for e.g. sensor measurement data. To lower the entry barrier for this undertaking, DBRepo offers multiple open, documented and well-understood interfaces for data scientists to deposit and access data (c.f. Table II). Started as internal Python client, we recently released our internal Python Library that covers the data management operations of DBRepo and bundles them into a single package that can be used with e.g. Jupyter Notebooks. With this step we aim to increase the utility for data scientists in using DBRepo.

For example, a data scientist wants to reuse training data from her institutional DBRepo repository as `torch` tensor for a machine learning task. The easiest way to achieve this is to use the `dbrepo` Python Library, setting the repository URL (c.f. Listing 1, line 4) and get the data by typing the database id (46) and table id (1516) in line 5. Then she can use the tensor stored in the column "targets" and create a `torch` tensor from the returned `pandas` dataframe (df=True).

```
1 from dbrepo.RestClient import RestClient
2 import torch
3
4 client = RestClient("https://dbrepo.example.com")
5 df = client.get_table_data(48, 1516, df=True)
6 torch_tensor = torch.tensor(df['targets'].values)
7 ...
```

Listing 1: Reusing data from DBRepo for Machine Learning

Similarly, if the data scientist wants to create subsets from the dataset to e.g. select all stations that exceed the threshold of 120μg/m^3 in 2023 (c.f. Listing 2). Note that the research database automatically filters invalid tuples at timestamp t_x such that tuples of year 2023 that were deleted in 2024 are not in the result set. If these tuples should not be filtered automatically, she needs to set the optional `ts` parameter of the `client.create_subset` method (c.f. Listing 2, line 10) to a `datetime` timestamp to read historical data as of this timestamp.

TABLE II: Managing Data in DBRepo

Operations	User Interface	HTTP API	AMQP API	Python Library
M1 Import .csv dataset	✓	✓		✓
M2a Create data tuple	✓	✓	✓	✓
M2b Read/Update/Delete data tuple	✓	✓		✓
M3 Create/Read/Delete database view	✓	✓		✓
M3 Create/Read/Delete database table	✓	✓		✓
M4 Create database subset	✓	✓		✓

```
1  from dbrepo.RestClient import RestClient
2
3  client = RestClient("https://dbrepo.example.com",
4                      "username", "password")
5  q = ("SELECT component, value FROM sensor s"
6       "JOIN location l ON s.loc_id = l.id"
7       "WHERE s.component = 'Ozon' AND"
8       "s.value > 120 AND "
9       "YEAR(ROW_START) = 2023")
10 df = client.create_subset(4, q, df=True)
```

Listing 2: Create database subset in DBRepo

If the researcher wants to create a data tuple in DBRepo, she needs to set the repository URL and her credentials (c.f. Listing 3, lines 3-4), specify the data tuple (lines 5-7) and import it into the desired table where she has write access to (line 8).

```
1 from dbrepo.RestClient import RestClient
2
3 client = RestClient("https://dbrepo.example.com",
4                     "username", "password")
5 df = pd.DataFrame(data={"uuid": ...,
6                         "component": "Ozone",
7                         "value": 37.4})
8 client.import_table_data(4, 20, df)
```

Listing 3: Creating data tuple in DBRepo

B. Position in the Repository Landscape

The scientific method requires reproducibility, that is, given the original data, a researcher must be able to come to the same conclusions as the original research. For machine learning, this can become a challenging task quickly as it requires the researcher to document the environment, activities and provide the trained models as well as the training- and test datasets. In many cases, the validity of research overall can be challenged as reproducibility is not possible given reasonable time and resources [25] [20].

We address this challenge briefly and propose a reproducibility approach to generate machine-learning documentation in a semi-automated manner that utilizes research data repositories for storing the model as well as training- and test datasets. The method was developed at our Data Science Research Unit and increasingly diffuses into lectures as standard

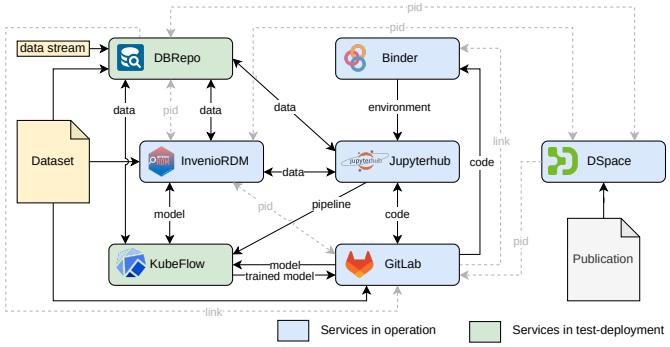

Fig. 6: TU Wien's research data infrastructure consists of operational services (blue) and services in test deployment (green) that are available to selected pilot users.

methodology for conducting machine learning experiments. We describe the method for the virtual research environment present at TU Wien that covers machine learning activities from the data selection to the final publication (c.f. Fig. 6).

The method starts with a researcher depositing the (structured) training- and test datasets into our institutional DBRepo[18] database repository, the trained model into our institutional InvenioRDM file repository and the code/algorithms used for the data cleaning/preprocessing as well as the training itself into our instutional Gitlab code repository according to TU Wien's research data policy. Then, a researcher can reproduce the machine learning environment by taking the code from Gitlab and the data from the InvenioRDM/DBRepo repositories into our institutional Jupyterhub compute environment that performs the machine learning, deposits the trained model automatically into the InvenioRDM repository, the training-/test datasets into DBRepo and the code into Gitlab. At any step, the researcher receives a PID for each of the research data which can be linked with each other.

The researcher benefits from this method too. In case of mistakes during the process, the trail of PIDs allows to identify the erroneous research data and notify affected researchers that used the erroneous research data (based on their publications in the publication repository DSpace) to correct their findings and re-run the experiments, increasing the accountability in machine learning experiments, potentially using big data.

VI. Challenges

A. Stakeholder Engagement

Data is essential for research and mission-critical to Universities. From a strategic point of view, it is imperative for organizations that the research data remains within the on-premise infrastructure similar to a carpenter having a hammer within reach is mission-critical. Important strategic reasons to have a research data repository within on-premise infrastructure are: (i) protection of intellectual property, (ii) collection of available data on management level, (iii) liberation of public cloud vendors.

Scientific freedom allows for a bottom-up change of practice approach where researchers use an information system as part of their day to day activities without being considered a core service of the University yet. From our experience of interdisciplinary collaborations with universities spanning around the globe, the vast majority of researchers are accepting for technological changes in the data management activities on data infrastructure as long as the offered infrastructure is available without extra effort, i.e. without the researcher having to setup/configure it.

B. Requirements Elicitation

A lack of understanding of the actual requirements and a sense of mistrust can result in users avoiding the use of certain infrastructures and data. Thus, it is crucial to both identify and address the real needs and to build confidence in the infrastructures and data quality. This naturally leads to the question of what defines trust in infrastructures and data quality, and what makes them trustworthy. In the end, trust in infrastructures and their components is crucial for researchers to have confidence in their own research findings and to take responsibility when sharing their results with the public.

We noticed that almost all stakeholders from the 10's of Universities we are in contact with are facing similar issues regarding making research data available in appropriate manner, i.e. research data in databases. Most research data management-departments look for a technical solution which they can in bottom-up fashion present to researchers which convinces the technical stakeholders (IT-department and their system-/database engineers) to share access to virtualization resources for setting up DBRepo on their own infrastructure. These small deployment on friendly-user basis then are the seed necessary to scale and commit to the data infrastructure change.

VII. Conclusions & Future Work

We have presented DBRepo, an institutional data repository system for research data in databases supporting guidelines of the Working Group on Data Citation of the Research Data Alliance. We described the technical building blocks of the system in detail, defining the micro service architecture, separation of concerns of the research data repository and give a definition of research databases. We highlighted three use cases related to big data (velocity, volume) and data versioning. We demonstrated how data scientists work with research data in DBRepo through three examples on reusing data, creating a subset, creating data and positioned DBRepo as structured research data repository in the repository landscape. Additionally, we proposed a research data management method to improve reproducibility of machine learning experiments using DBRepo. Last, we dicussed challenges surrounding stakeholder engagement and requirements elicitation.

[18]https://dbrepo1.ec.tuwien.ac.at/ at time of writing consists of 23 databases, 437 data sources (tables, queries, views) and manages 15 GB of pilot user's data

To scale DBRepo in the future, a detailed understanding of researcher's activities and needs for depositing research data is necessary in order to benefit the researchers, increasing the technological acceptance additionally. From regular interviews with our friendly users, we concluded that it is important for researchers to assign semantic concepts and units of measurements from low-level ontologies, especially institutional ontologies[19] developed by the Resarch Unit of Automation Systems surrounding the topic of smart homes that precisely assign machine-understandable context to a dataset. Development has already started with a prototypic implementation suggesting the correct label and URI using BGE-M3 Embeddings [26]. Initial results suggest that our semi-automatic method [27] is 9.8 times faster than mapping the semantic context by hand.

Finally, we envision a future where machines retrieve data for the researcher's current interest e.g. for automated systematic reviews where data can be extracted by machines instead of human investigators [28].

Supplementary Material

DBRepo's source code[20] is open source and can be deployed fast using Docker Compose for testing purposes and as Helm Chart[21] for operational use in Kubernetes (private-) cloud environments. We maintain a public demo environment[22] to explore DBRepo without installing it locally, any data is deleted sporadically. The Python library can be installed from PyPI[23].

Acknowledgment

We thank the Center for Research Data Management at TU Wien and TU.it for their excellent cooperation regarding the development of DBRepo. Research reported in this publication was jointly supported by the ASEAN-European Academic University Network (ASEA-UNINET), the Austrian Federal Ministry of Education, Science and Research (BMBWF) and the OeAD - Austria's Agency for Education and Internationalization.

References

[1] T. Hey and A. Trefethen, "The Data Deluge: An e-Science Perspective," p. 809–824, 2003.

[2] M. Weise, M. Staudinger, C. Michlits, E. Gergely, K. Stytsenko, R. Ganguly, and A. Rauber, "Dbrepo: a Semantic Digital Repository for Relational Databases," *International Journal of Digital Curation*, vol. 17, no. 1, p. 11, 2022.

[3] K. Kulkarni and J.-E. Michels, "Temporal Features in SQL:2011," *ACM SIGMOD Record*, vol. 41, no. 3, p. 34–43, 2012.

[4] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten *et al.*, "The FAIR Guiding Principles for Scientific Data Management and Stewardship," *Scientific Data*, vol. 3, no. 1, 2016.

[5] DataCite Metadata Working Group, "DataCite Metadata Schema Documentation for the Publication and Citation of Research Data and Other Research Outputs v4.5," 2024.

[19]https://www.auto.tuwien.ac.at/downloads/thinkhome/ontology/

[20]https://gitlab.phaidra.org/fair-data-austria-db-repository/fda-services/

[21]https://artifacthub.io/packages/helm/dbrepo/dbrepo/

[22]https://test.dbrepo.tuwien.ac.at/

[23]https://pypi.org/project/dbrepo/

[6] G. D. Subcommittee, "Digital Cartographic Standard for Geologic Map Symbolization," Federal Geographic Data Committee, Tech. Rep., 2006.

[7] H. Van de Sompel, "FAIR Digital Objects and FAIR Signposting," 2023.

[8] O. Yaron and A. Gal-Yam, "WISeREP: An Interactive Supernova Data Repository," *Publications of the Astronomical Society of the Pacific*, vol. 124, no. 917, p. 668–681, 2012.

[9] A. Rauber, B. Gößwein, C. M. Zwölf, C. Schubert, F. Wörister, and J. Duncan, "Precisely and Persistently Identifying and Citing Arbitrary Subsets of Dynamic Data," *Harvard Data Science Review*, vol. 3, no. 4, 2021.

[10] M. Smith, M. Barton, M. Branschofsky, G. McClellan, J. H. Walker, M. Bass, D. Stuve, and R. Tansley, "DSpace: An Open Source Dynamic Digital Repository," *D-Lib Magazine*, vol. 9, no. 1, 2003.

[11] G. King, "An Introduction to the Dataverse Network as an Infrastructure for Data Sharing," *Sociological Methods & Research*, vol. 36, no. 2, p. 173–199, 2007.

[12] H. C. White, S. Carrier, A. Thompson, J. Greenberg, and R. Scherle, "The Dryad Data Repository: a Singapore Framework metadata Architecture in a DSpace Environment," in *Proceedings of the 2008 International Conference on Dublin Core and Metadata Applications*. Dublin Core Metadata Initiative, 2008, p. 157–162.

[13] M. Heesemann, T. Insua, M. Scherwath, K. Juniper, and K. Moran, "Ocean networks canada: From geohazards research laboratories to smart ocean systems," *Oceanography*, vol. 27, no. 2, p. 151–153, 2014.

[14] N. Alshuqayran, N. Ali, and R. Evans, "A Systematic Mapping Study in Microservice Architecture," in *2016 IEEE 9th International Conference on Service-Oriented Computing and Applications*, 2016.

[15] M. Witt and M. Cragin, "Introduction to Institutional Data Repositories Workshop," *Libraries Research Publications*, p. 83, 2008.

[16] I. Mueller, A. Arteaga, T. Hoefler, and G. Alonso, "Reproducible Floating-Point Aggregation in RDBMSs," in *2018 IEEE 34th International Conference on Data Engineering*, 2018.

[17] P. Panchekha, A. Sanchez-Stern, J. R. Wilcox, and Z. Tatlock, "Automatically Improving Accuracy for Floating Point Expressions," in *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*. ACM, 2015.

[18] F. J. Ekaputra, M. Weise, K. Flicker, M. R. Bin Salleh, M. N. Abd Rahman, A. A. Abd Rahman, T. Miksa, and A. Rauber, "Towards A Data Repository for Educational Factories," in *2022 International Conference on Data and Software Engineering*, 2022.

[19] M. Hennig, G. Reisinger, T. Trautner, P. Hold, D. Gerhard, and A. Mazak, "TU Wien Pilot Factory Industry 4.0," *Procedia Manufacturing*, vol. 31, pp. 200–205, 2019.

[20] C. Schneidhofer, K. Dubek, and N. Dörr, "Robust Sensors Enabling Condition-based Maintenance of Lubricated Components in Locomotives and Wagons," *Transportation Research Procedia*, vol. 72, p. 3236–3243, 2023.

[21] M. Anderl, M. Gangl, L. Makoschitz, S. Mayer, K. Pazdernik, S. Poupa *et al.*, *Emissionstrends 1990-2022*. Umweltbundesamt, 2024.

[22] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher, "What Do We Talk About When We Talk About Dashboards?" *IEEE Transactions on Visualization and Computer Graphics*, vol. 25, no. 1, p. 682–692, 2019.

[23] S. Behmel, M. Damour, R. Ludwig, and M. Rodriguez, "Water Quality Monitoring Strategies - A Review and Future Perspectives," *Science of The Total Environment*, vol. 571, p. 1312–1329, 2016.

[24] S. Kittlaus, M. K. Kardos, K. M. Dudás, N. Weber, A. Clement, S. Petkova, D. Sukovic *et al.*, "A Harmonized Danube Basin-wide Multi-compartment Concentration Database to Support Inventories of Micropollutant Emissions to Surface Waters," *Environmental Sciences Europe*, vol. 36, no. 1, 2024.

[25] M. Baker, "1,500 Scientists Lift the Lid on Reproducibility," *Nature*, vol. 533, no. 7604, p. 452–454, 2016.

[26] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation," 2024.

[27] B. Altug, M. Weise, and A. Rauber, "Generating Semantic Context for Data Interoperability in Relational Databases using bge M3-Embeddings."

[28] I. J. Saldanha, B. T. Smith, E. Ntzani, J. Jap, E. M. Balk, and J. Lau, "The systematic review data repository (srdr): descriptive characteristics of publicly available data and opportunities for research," *Systematic Reviews*, vol. 8, no. 1, 2019.