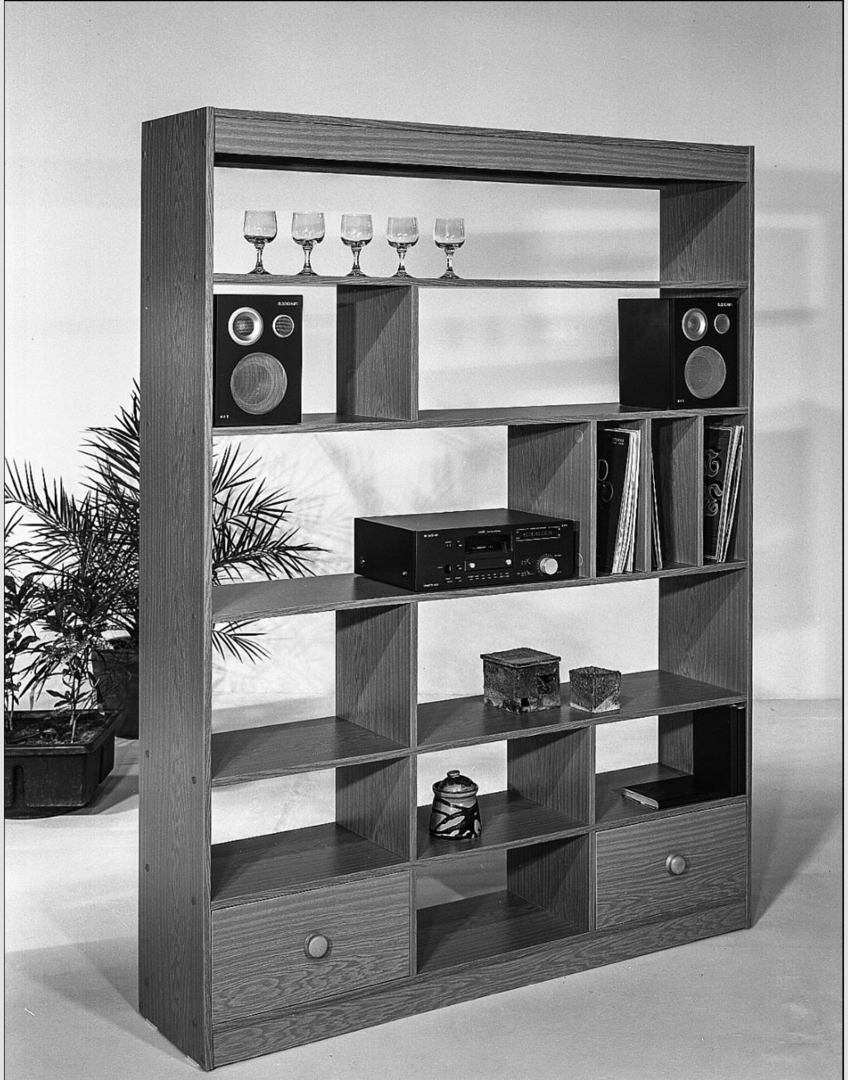
Beyond Relational Databases

Challenges of Archiving NoSQL or Graph

Sven Schlarb

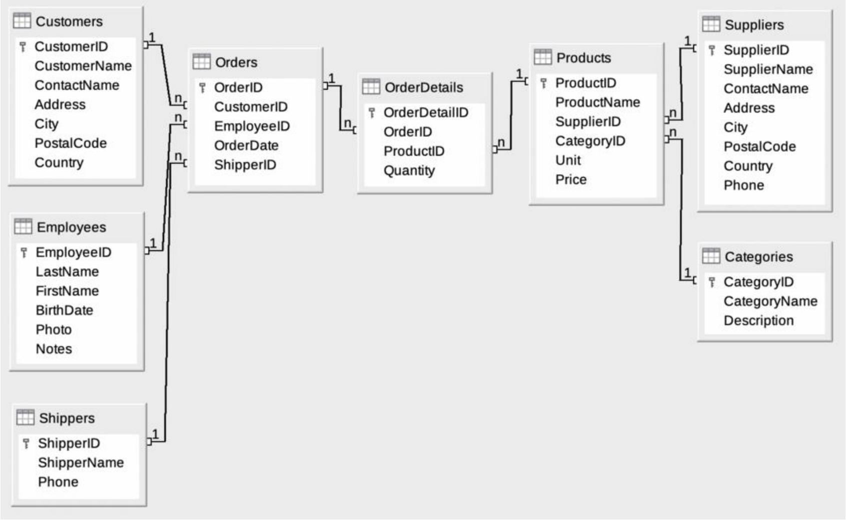
14 MAY 2025

Albert Borschette Building, Rue Froissart36, 1040 Etterbeek, Brussels



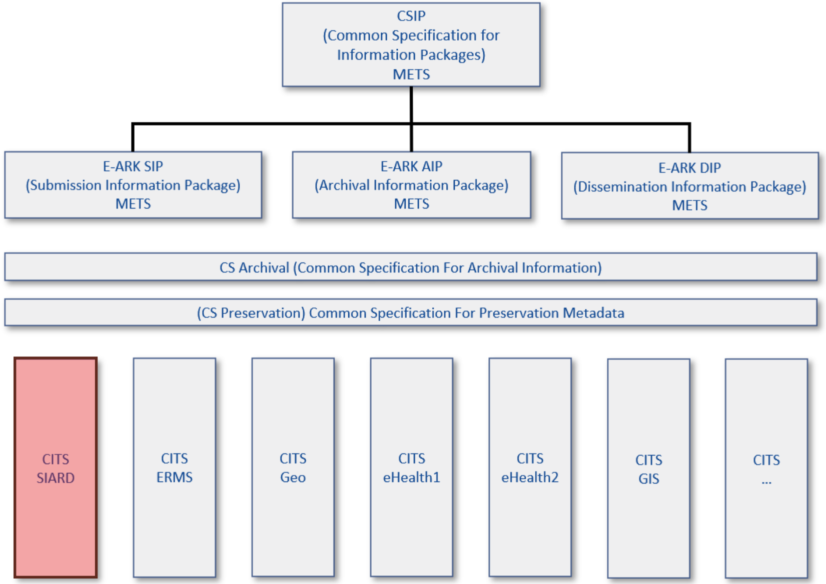
The Order of Relational Databases

- Back in the 80ies database were nearly exclusively relational.
    - RDBMS, SQL, etc. were the predominant standards
- ISO/IEC 9075
    - SQL Language revisions: SQL-86, SQL-89, SQL92, SQL:1999, SQL:2003, SQL:2006, SQL:2008, SQL:2011, SQL:2016, SQL:2019, SQL:2023



Northwind Traders is a database sample that is shipped along with the Microsoft Access application.
The Northwind database is available under a Microsoft Public License.

E-ARK Specifications



CSIP
(Common Specification for Information Packages)
METS

E-ARK SIP
(Submission Information Package)
METS

E-ARK AIP
(Archival Information Package)
METS

E-ARK DIP
(Dissemination Information Package)
METS

CS Archival (Common Specification For Archival Information)

(CS Preservation) Common Specification For Preservation Metadata

CITS
SIARD

CITS
ERMS

CITS
Geo

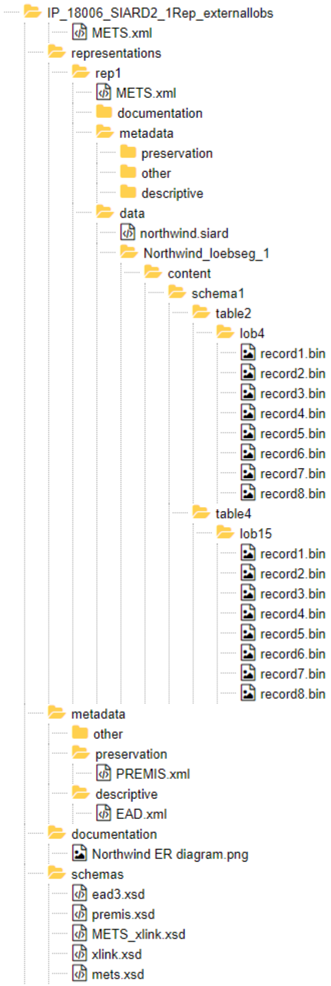
CITS
eHealth1

CITS
eHealth2

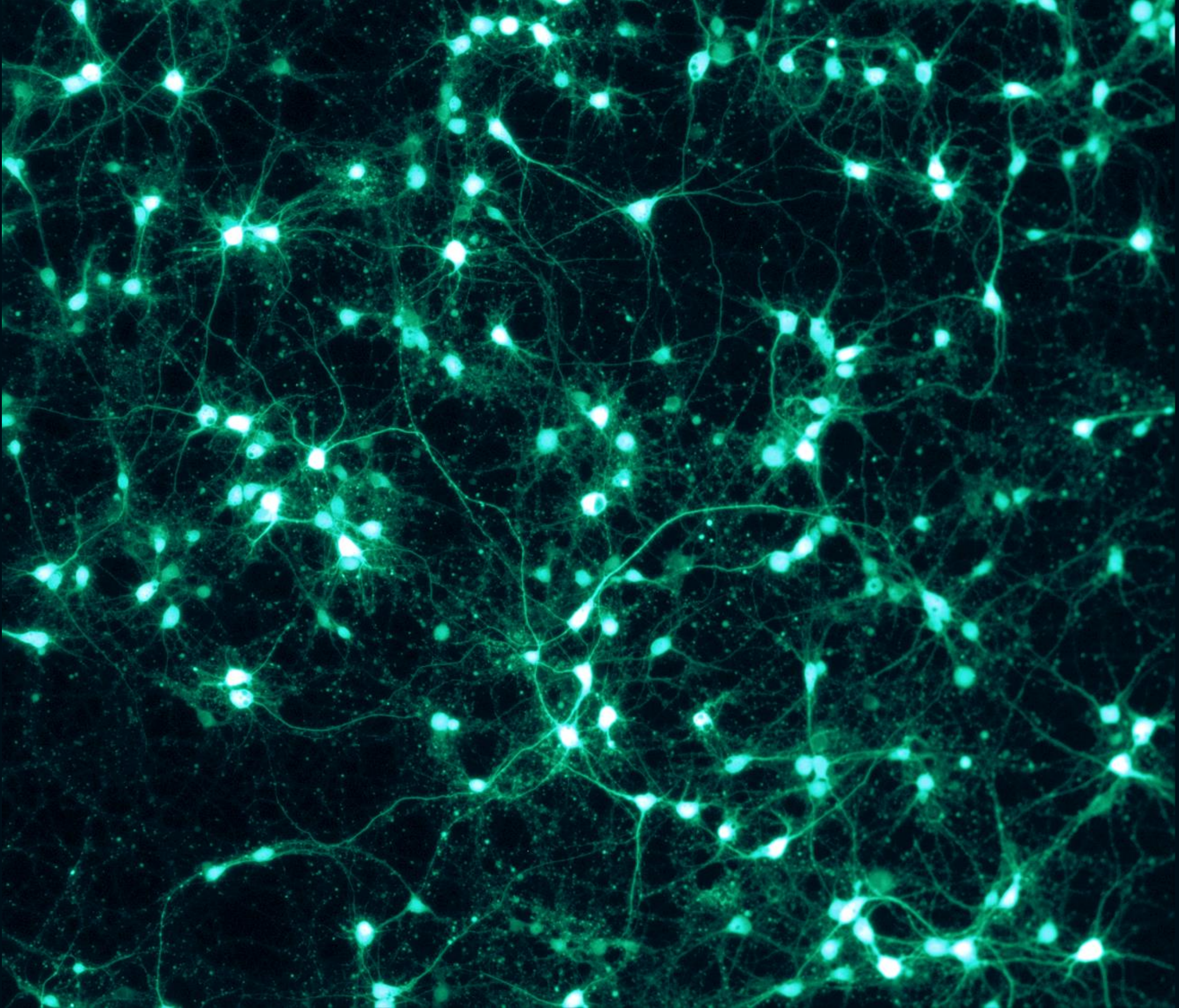
CITS
GIS

CITS
...

Folder Structure of Northwind Sample Database



```
IP_18006_SIARD2_1Rep_externallobs
    METS.xml
    representations
        rep1
            METS.xml
            documentation
            metadata
                preservation
                other
                descriptive
            data
                northwind.siard
                Northwind_loebseg_1
                    content
                        schema1
                            table2
                                lob4
                                    record1.bin
                                    record2.bin
                                    record3.bin
                                    record4.bin
                                    record5.bin
                                    record6.bin
                                    record7.bin
                                    record8.bin
                            table4
                                lob15
                                    record1.bin
                                    record2.bin
                                    record3.bin
                                    record4.bin
                                    record5.bin
                                    record6.bin
                                    record7.bin
                                    record8.bin
    metadata
        other
        preservation
            PREMIS.xml
        descriptive
            EAD.xml
    documentation
        Northwind ER diagram.png
    schemas
        ead3.xsd
        premis.xsd
        METS_xlink.xsd
        xlink.xsd
        mets.xsd
```



But sometimes, things just don't fit into the schema

© Manuel Schottdorf
Creative Commons
Attribution-Share Alike 4.0

The Downside of the Order

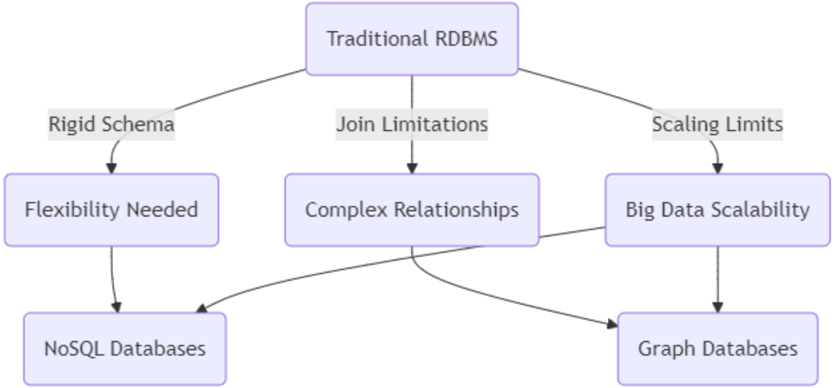
- **Schemas** are **strict**: Every piece of data must fit a predefined structure (tables, columns, data types, constraints).
- **Schema changes are painful**: Altering tables can require downtime, complex migrations, and the risk of breaking applications.
- **Scaling is rigid**: Vertical scaling (adding more power to a single server) has physical limits and is expensive.

Column Name	Data Type	Constraints
id	INTEGER	PRIMARY KEY, NOT NULL
title	VARCHAR(255)	NOT NULL
author	VARCHAR(255)	NOT NULL
published_year	INTEGER	CHECK (published_year > 0)
isbn	CHAR(13)	UNIQUE, NOT NULL

From: https://agilemanifesto.org/principles.html
Welcome changing requirements, even late in development.

Limitations of Classical Relational Databases

- Rigid Schema
  - Lack of flexibility of adapting a model to a changing environment
- Join Limitations
  - Unable to cope with complex Relationships
- Scaling Limits
  - Unable to cope with Big Data needs



An Evolving Landscape of Data Storage

Relational

- PostgreSQL
- SQLite
- MariaDB
- …

Distributed

- CitusData
- CockroachDB
- YugabyteDB
- …

Cached

- Redis
- Memcached
- Dragonfly
- …

Document Store

- MongoDB
- CouchDB
- RavenDB
- …

Graph

- Neo4j
- JanusGraph
- NebulaGraph
- …

Distributed Key-Value

- Riak
- FoundationDB
- Etcd
- …

Wide-Column Key-Value

- Cassandra
- HBase
- Scylla
- …

Embedded Key-Value

- MyRocks
- LevelDB
- RocksDB
- …

Search Engine

- Solr
- ElasticSearch
- Sphinx
- …

Streaming

- Materialize
- EventStoreDB
- Ksqldb
- …

Time-Series

- InfluxDB
- IoTDB
- KairosDB
- …

Columnar OLAP

- Kudu
- Greenplum
- MonetDB
- …

Real-Time OLAP

- Pinot
- Druid
- Apache Kylin
- …

…

- …
- …
- …

NoSQL and Graph-Databases

Graph Database

```
[Painting: "Portrait of a Woman"]
   ├── created_by ──▶ [Artist: "Vermeer"]
   ├── exhibited_at ──▶ [Exhibition: "Dutch Masters"]
   └── depicts_theme ──▶ [Theme: "Baroque Portraiture"]
```

NoSQL (JSON)

```
{
  "_id": "L001",
  "title": "Letter from Einstein to Bohr",
  "date": "1926-11-05",
  "participants": ["Einstein", "Bohr"],
  "location": "Berlin",
  "language": "German",
  "content_summary": “Uncertainty in quantum measurements.",
  "references": ["L000"],
  "annotations": [
    {
      "author": "Historical Editor",
      "note": "Possibly drafted after 5th Solvay Conference.",
      "date_added": "1978-06-10"
    }
  ]
}
```

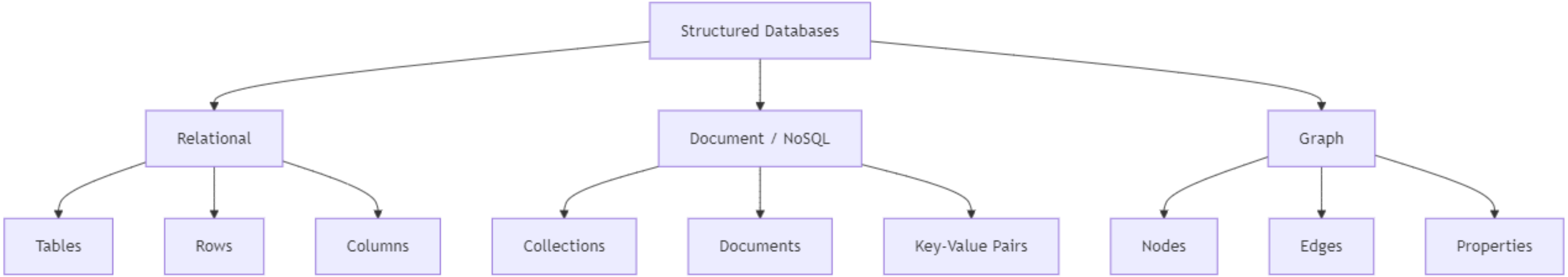
HBase

```
Row Key: L001

Column Family: meta
  - title → "Letter from Einstein to Bohr"
  - date → "1926-11-05"
  - location → "Berlin"
  - language → "German"
  - content_summary → “Uncertainty in quantum measurements."
  - participants:0 → "Einstein"
  - participants:1 → "Bohr"
  - references:0 → "L000"

Column Family: annotations
  - 0:author → "Historical Editor"
  - 0:note → "Possibly drafted after 5th Solvay Conference."
  - 0:date_added → "1978-06-10"
```

Entities of Structured Databases



Structured Databases

Relational

Document / NoSQL

Graph

Tables

Rows

Columns

Collections

Documents

Key-Value Pairs

Nodes

Edges

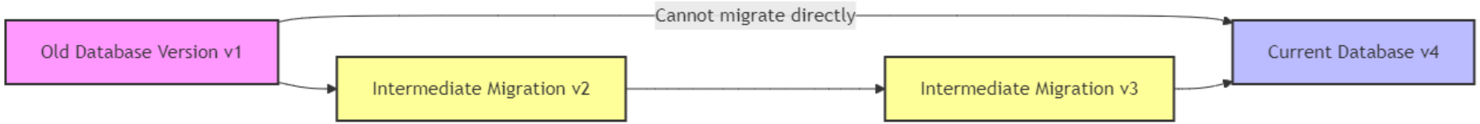
Properties

Example: Hbase Preservation

- HBase is a distributed, scalable, NoSQL database built on top of Hadoop and HDFS.
- HBase comes with a built-in MapReduce job called Export which can dump all the data from a table into HDFS in sequence file
- Archive SequenceFiles
- Can be re-imported into HBase if format remains compatible ✓
- SequenceFile is binary and tied to Hadoop ⚠
  - Requires HBase or Hadoop – in the correct version – in the future.
- No schema metadata, just rows and columns. ⚠
- **Raw HBase exports may not be future-proof** ⚠

Technical Challenges: Migration

- **Intermediate migrations** (hops) might be necessary to "transform" the data step-by-step



- **Versions** of the database management system available?
- **Licenses** available?

Digital Preservation: Not only Formats – Interpretability/Understandability!

- Keeping bits and formats intact is necessary but not sufficient
- A future user must still be able to understand the digital object as intended.

The OAIS Model:
"Preservation must ensure that the information remains *independently understandable* by the *Designated Community* without requiring the assistance of the original producers." **(OAIS, ISO 14721, 2012, Section 1.7.2)**

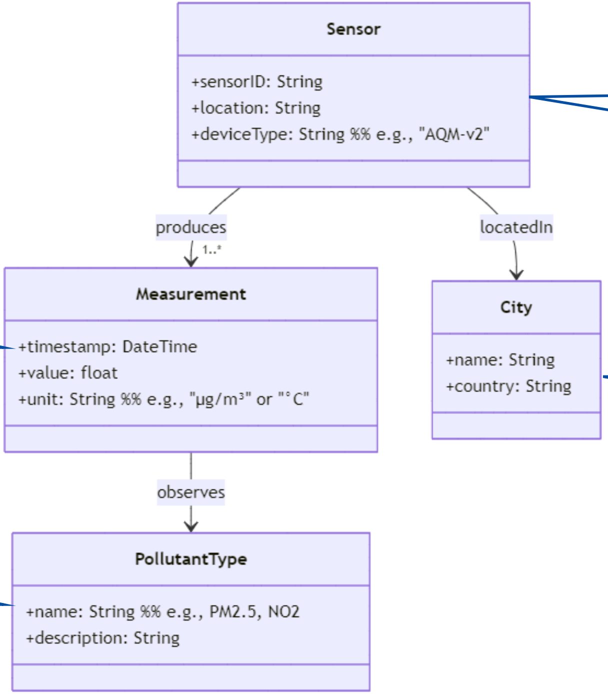
Context is key for preservation!

- Imagine an air quality monitoring system storing data in **HBase**.
  - Each sensor logs time-stamped readings for particulate matter (PM2.5), temperature, humidity, and location metadata.

```
Row Key: sensor-vienna_2025-05-12T10:00Z
Column Families:
  data:pm25 -> "32"
  data:temp -> "19.5"
  meta:city -> "Vienna"
  meta:deviceType -> "AQM-v2"
```

- What happens to this data in 5, 10, or 50 years?
  - Can we still interpret or reuse it? Without structure, context, and preservation, we risk losing insights.

Structure is key for perservation!



Sensor

+sensorID: String
+location: String
+deviceType: String %% e.g., "AQM-v2"

produces 1..*

locatedIn

Measurement

+timestamp: DateTime
+value: float
+unit: String %% e.g., "µg/m^3" or "°C"

City

+name: String
+country: String

observes

PollutantType

+name: String %% e.g., PM2.5, NO2
+description: String

represents a physical device that records environmental data

a reading recorded by a sensor at a specific time

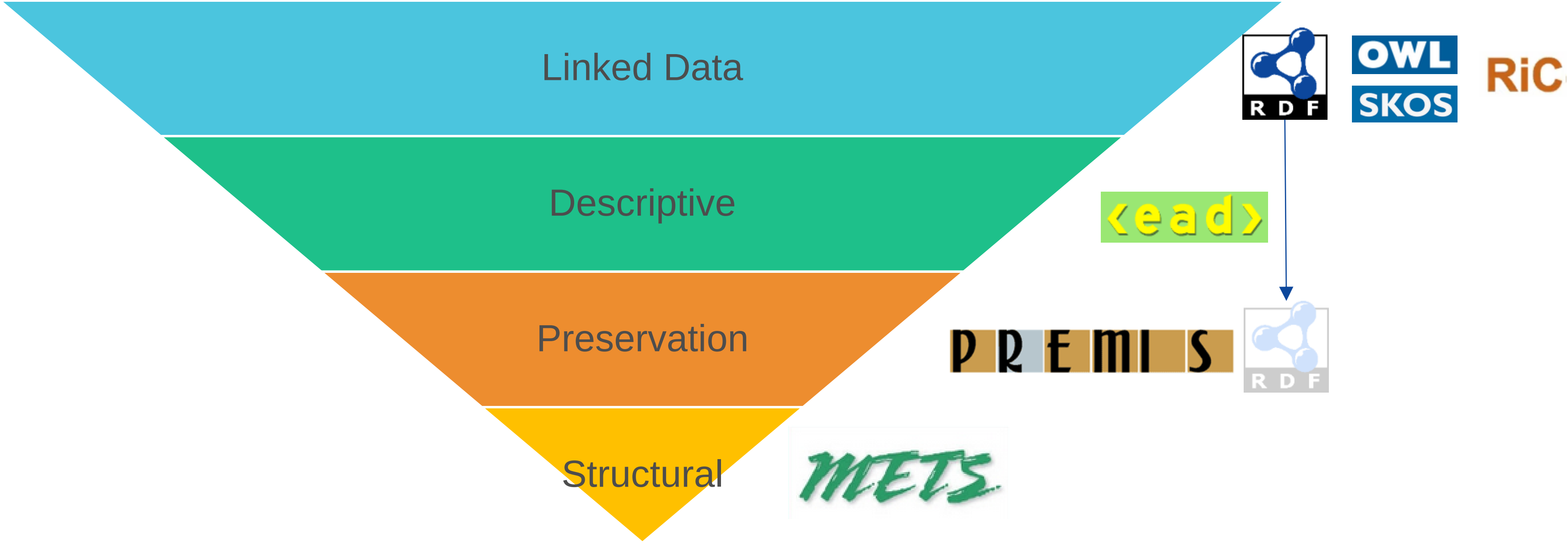
geographical context for sensor placement

what the measurement is about

What AI (an LLM) can do using the Structure

- „Understand“ relations
  - data:pm25 and data:temp are instances of Measurement, tied to a Sensor entity and must be preserved with units and context.
- Data transformation
  - Suggest mapping flat HBase entries into JSON-LD or RDF for long-term semantic preservation.
- Human-readable summaries
  - Auto-document data structures, identify gaps (e.g., missing unit), and recommend enrichment.
- Export Plan
  - From HBase to a graph store using the ontology as a schema map
- Queries
  - SPARQL or Cypher queries or based on natural language, like:
    “Show me all sensors in Vienna reporting PM2.5 above 25 in May 2025.

eArchiving Vision for Interoperability

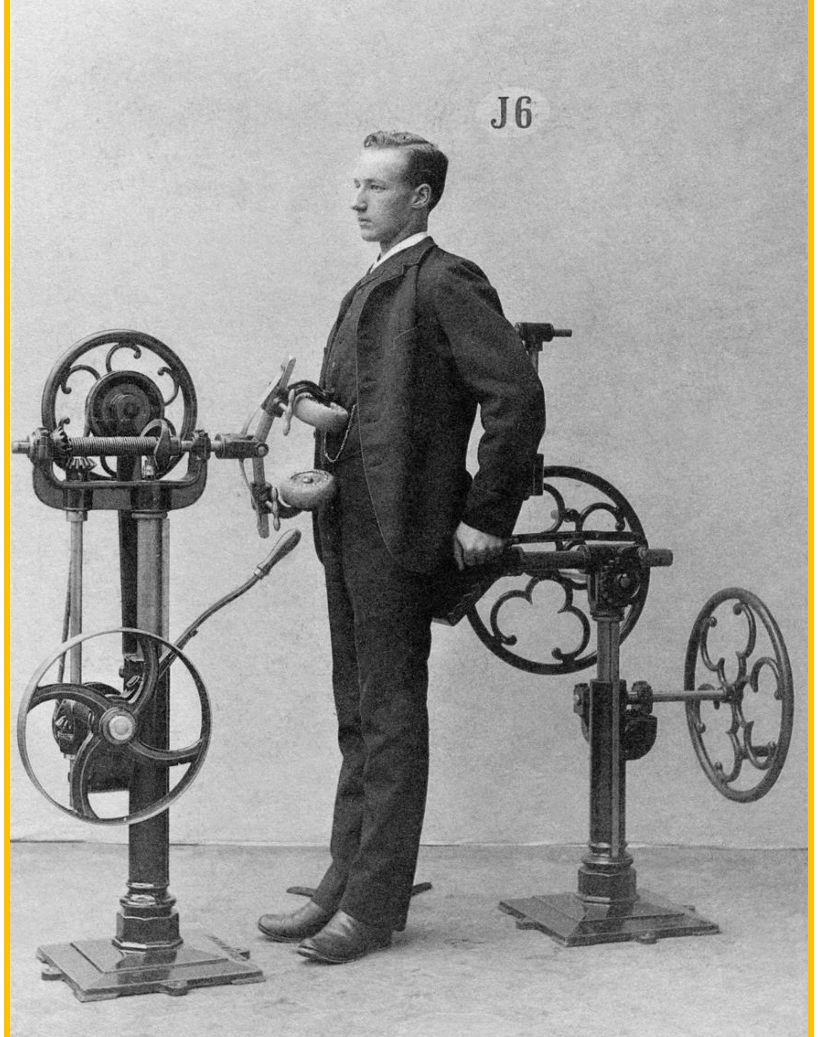


Linked Data

Descriptive

Preservation

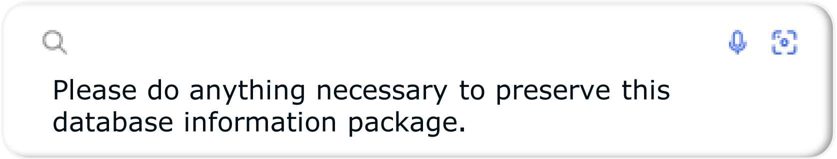
Structural



Lean back and feel the benefits of a circular massage to the lower abdomen. by Okänd - Swedish National Museum of Science and Technology, Sweden - Public Domain, Retrieved from Eureopeana

Option 1: Lean back

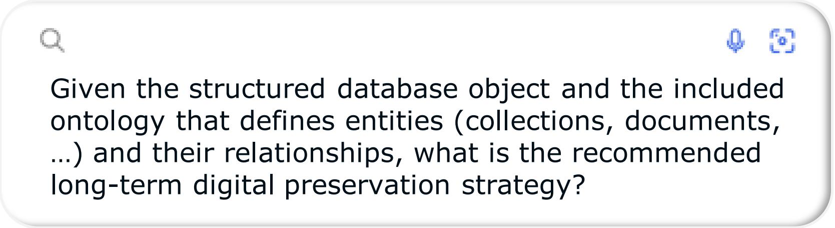
- **Focus on other activities and let AI do the work**


Please do anything necessary to preserve this database information package.

© Gerhard Vetter, Educational Use Permitted, Deutsche Fotothek, Retrieved from Europeana

Option 2: Lean in

- **Actively guide digital preservation using domain knowledge**



Given the structured database object and the included ontology that defines entities (collections, documents, …) and their relationships, what is the recommended long-term digital preservation strategy?

Thank you for your attention!

Questions?