DATA CURATION

QUERY LANGUAGES AND OPERATIONS TO SPECIFY AND TRANSFORM DATA

Justin Lopes
D093

Arushi Singhi
D109

TIMELINE

DATA CURATION

QUERY LANG.

OPERATIONS

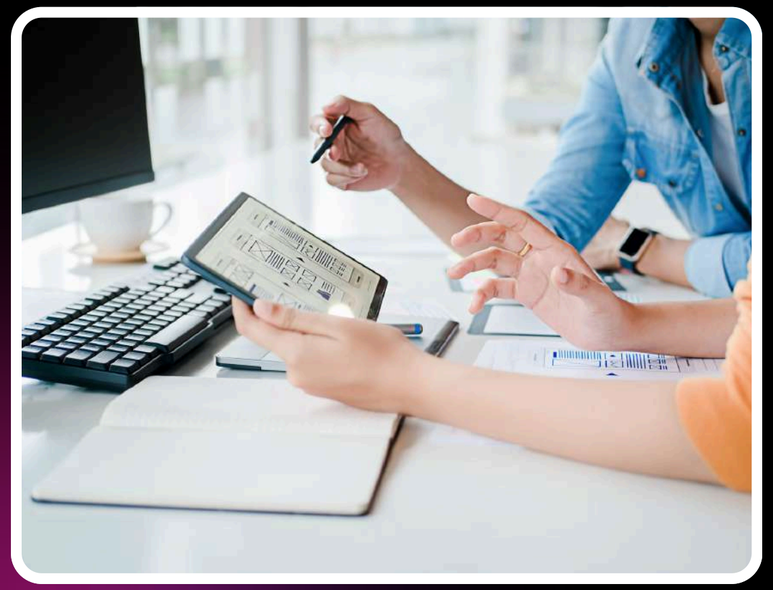
BEST PRACTICES

DATA CURATION

01. INTRODUCTION

02. BENEFITS/CHALLENGES

03. APPLICATIONS



In today's world, data is an important resource for businesses, researchers, and governments. But raw data is often messy, inconsistent, or overwhelming.

Data curation helps make sense of this raw data by organizing, cleaning, and preparing it so it can be used for analysis and decision-making.

It involves sorting, labeling, storing, and improving data to keep it valuable over time.

Data curation ensures that the data stays easy to find, reliable, and useful for things like research, business analysis, and making public policy decisions.

INTRODUCTION

BENEFITS

1. **Improved Data Quality**
   - Ensures accuracy, consistency, and completeness.
   - Reduces errors and enhances the reliability of analysis.
2. **Enhanced Accessibility**
   - Makes data discoverable and usable through clear organization and metadata tagging.
   - Reduces time spent searching for relevant information.
3. **Facilitated Decision-Making**
   - Provides high-quality data that leads to accurate insights and informed strategies.

4. **Streamlined Data Integration**
    - Simplifies combining datasets from diverse sources for comprehensive analysis.
5. **Regulatory Compliance**
    - Helps organizations meet legal and ethical data governance standards, avoiding potential fines and reputational damage.
6. **Cost Efficiency**
    - Reduces redundant storage costs by eliminating unnecessary or duplicate data.
    - Saves time and resources by enabling faster data preparation.
7. **Data Preservation**
    - Protects valuable datasets from degradation or loss over time.

CHALLENGES

- **Volume of Data**
  - Managing vast amounts of data, especially in big data environments, can overwhelm resources.
- **Diversity of Data Types**
  - Handling structured, semi-structured, and unstructured data adds complexity.
- **Metadata Management**
  - Creating and maintaining metadata is time-intensive but critical for data discoverability.
- **Data Quality Issues**
  - Addressing inconsistencies, missing values, or errors can be labor-intensive.
- **Resource Limitations**
  - Skilled personnel, advanced tools, and sufficient funding are often in short supply.
- **Governance and Compliance**
  - Ensuring adherence to evolving data regulations requires ongoing monitoring and expertise.
- **Technological Integration**
  - Combining disparate systems and tools into a seamless data curation workflow can be challenging.

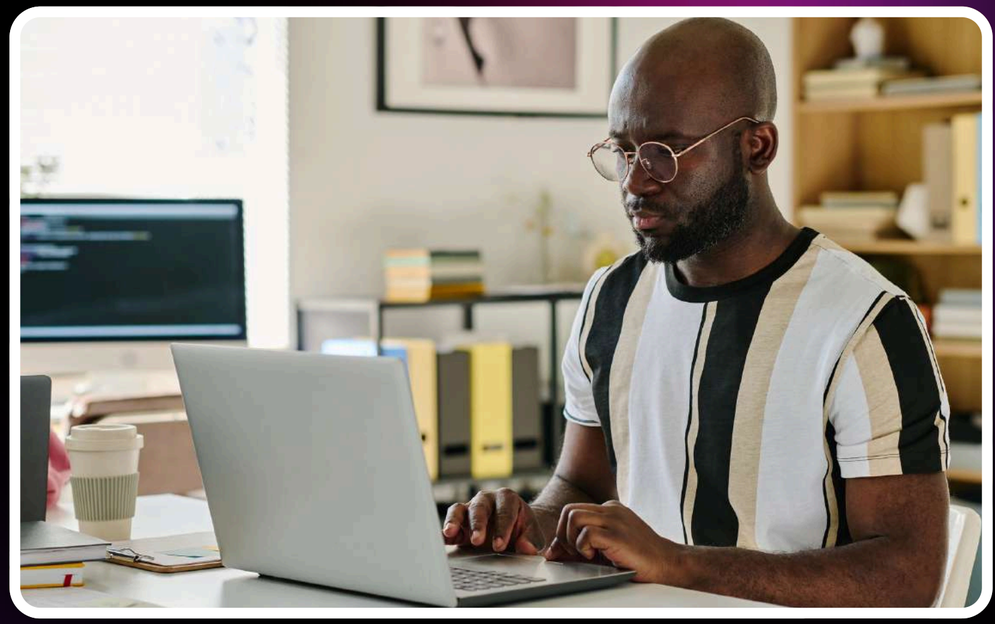
APPLICATIONS

Data curation makes data more useful by making sure it's easy to find, understand, and use. Here are some examples:

In Research:
Researchers organize and keep data in a way that others can use it to repeat their studies or build on them.

In Business:
Data curators clean up and label customer data so marketing teams can use it to create better ads and campaigns.

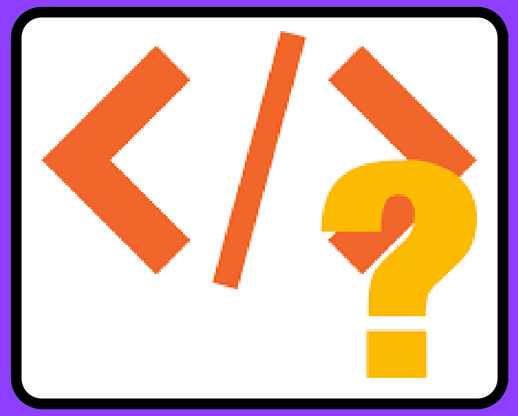
In Public Policy:
Curated census data helps policymakers make decisions based on clear, accurate information about the population.



QUERY LANGUAGES

SQL

NoSQL



XQuery

SPARQL

01. SQL

- Standard language for managing relational databases.
- Allows operations like querying, updating, inserting, and deleting data.
- Used in popular DBMSs: MySQL, PostgreSQL, SQL Server, SQLite.

COMPONENTS

- **Data Query Language (DQL) :** Retrieves data from databases.
  - Example: SELECT first_name, last_name FROM employees WHERE department = 'Sales';
- **Data Definition Language (DDL) :** Manages database structure (tables, schemas).
  - **CREATE, ALTER, DROP, TRUNCATE**
  - Example (CREATE): CREATE TABLE employees (...);
- **Data Manipulation Language (DML) :** Manipulates data in the database.
  - **INSERT, UPDATE, DELETE**
  - Example (INSERT): INSERT INTO employees (id, first_name, ...) VALUES (1, 'John', 'Doe');
- **Data Control Language (DCL):** Manages access permissions.
  - **GRANT, REVOKE**
  - Example (GRANT): GRANT SELECT, INSERT ON employees TO user1;
- **Transaction Control Language (TCL) :** Manages transactions for data integrity.
  - **COMMIT, ROLLBACK, SAVEPOINT**
  - Example (COMMIT): COMMIT;

02. NOSQL

- Standard language for managing relational databases. Allows operations like querying, updating, inserting, and deleting data. Used in popular DBMSs: MySQL, PostgreSQL, SQL Server, SQLite.

- **Key Features**:
  - **Schema-less/Flexible Schema**: No predefined schema required; data can change dynamically.
  - **Horizontal Scalability**: Distributes data across multiple servers to handle large datasets and high traffic.
  - **High Availability**: Built-in replication and fault tolerance for data redundancy.
  - **Distributed Architecture**: Optimized for performance, resilience, and fault tolerance.

Feature	NoSQL	SQL
Data Model	Flexible (key-value, documents, graphs)	Structured (tables and rows)
Scalability	Horizontal (across servers)	Vertical (single server)
Schema	Schema-less or dynamic	Predefined schema
Query Language	Custom (varies by database)	SQL (Standardized)

03. XQUERY

- A powerful query and functional programming language for querying and manipulating XML data.
- Enables extraction, transformation, and creation of XML documents.
- Standardized by the W3C and used in web services, document management, and data integration.
- Tailored for hierarchical and semi-structured data, similar to SQL for relational databases.

KEY FEATURES

- **XML Data Querying**
  - Allows extracting data from XML documents while respecting their hierarchical nature.
- **XPath Integration**
  - Supports XPath for navigating and selecting XML elements and attributes.
  - Example: /library/book/title to get book titles.
- **FLWOR Expressions**
  - For, Let, Where, Order By, and Return for complex queries.
  - Example: Find books published after 2020 with <bookInfo> structure
- **Handling Sequences**
  - Operations on ordered collections (sequences) of XML data.
  - Example: Combine titles and authors into a single result.
- **Joining Data**
  - Perform joins by combining multiple sequences.

- **Data Transformation**
  - Reshape XML data by rearranging, adding, or removing elements.
- **Using Functions in XQuery**
  - Built-in functions for string manipulation, date handling, and more.
  - Example: string-length($book/title) to calculate title length.
- **XQuery and XML Databases**
  - XQuery is widely used in XML databases for efficient querying and transformation.
  - Databases like BaseX, eXist-db, and MarkLogic use XQuery for XML data handling.
- **XQuery and XSLT**
  - XQuery can be used alongside XSLT for data extraction and transformation.
  - XSLT focuses on document transformation, while XQuery focuses on querying and extracting data.

SYNTAX

- It is case-sensitive
- XQuery elements, attributes, and variables must be valid XML names (should start with letter or underscore)
- An XQuery string value can be in single or double quotes
- An XQuery variable is defined with a $ followed by a name, e.g. $bookstore
- XQuery comments are delimited by (: and :), e.g. (: XQuery Comment :)
- Basic Query:

  ```
  doc("library.xml")//book/title
  ```
- Variable Declaration:

  ```
  let $books := doc("library.xml")//book
  ```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>

<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>

<book category="web">
  <title lang="en">XQuery Kick Start</title>
  <author>James McGovern</author>
  <author>Per Bothner</author>
  <author>Kurt Cagle</author>
  <author>James Linn</author>
  <author>Vaidyanathan Nagarajan</author>
  <year>2003</year>
  <price>49.99</price>
</book>

<book category="web" cover="paperback">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>

</bookstore>
```

FLWOR EXPRESSIONS

```
for $x in doc("books.xml")/bookstore/book
where $x/price>30
order by $x/title
return $x/title
```

```xml
<title lang="en">Learning XML</title>
<title lang="en">XQuery Kick Start</title>
```

CONDITIONAL EXPRESSIONS

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>

<book category="cooking">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
</book>

<book category="children">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>

<book category="web">
  <title lang="en">XQuery Kick Start</title>
  <author>James McGovern</author>
  <author>Per Bothner</author>
  <author>Kurt Cagle</author>
  <author>James Linn</author>
  <author>Vaidyanathan Nagarajan</author>
  <year>2003</year>
  <price>49.99</price>
</book>

<book category="web" cover="paperback">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>

</bookstore>
```

```
for $x in doc("books.xml")/bookstore/book
return if ($x/@category="children")
then <child>{data($x/title)}</child>
else <adult>{data($x/title)}</adult>
```

```
<adult>Everyday Italian</adult>
<child>Harry Potter</child>
<adult>XQuery Kick Start</adult>
<adult>Learning XML</adult>
```

BENEFITS

- **Powerful Data Manipulation**
  - Complex querying and data transformations, ideal for XML data.
- **Efficient XML Querying**
  - Optimized for querying hierarchical data in XML documents.
- **Seamless Integration with XML Databases**
  - Natively supported by XML databases for efficient processing.
- **Extensibility**
  - User-defined functions and advanced transformations enhance flexibility.

04. SPARQL

- A query language specifically for RDF data.
- Allows flexible querying, including the retrieval, insertion, deletion, and modification of RDF data.
- **Key Features:**
  - **Pattern Matching**: Queries are built around **graph patterns**, looking for triples in the RDF graph.
  - **Data Manipulation**: Besides retrieving data, SPARQL supports updating RDF data.

WHAT IS RDF?

- **Key Features of RDF**:
  - **Triples**: The fundamental unit of RDF is the triple, consisting of:
    - **Subject**: The resource being described (e.g., a person or a book).
    - **Predicate**: The relationship (e.g., "hasName" or "writtenBy").
    - **Object**: The value or related resource (e.g., "John Doe" or "Semantic Web Explained").
  - **Graph Representation**: RDF triples are visualized as a graph:
    - **Nodes** represent resources (Subjects/Objects).
    - **Edges** represent predicates (relationships).

- **Example RDF Data**:
  - Book1 → hasTitle → "Semantic Web Explained"
  - Book1 → writtenBy → Author1
  - Author1 → hasName → "Jane Doe"

BASIC QUERY STRUCTURE

1. **PREFIX**: Define namespaces to simplify URIs.
   - Example: PREFIX foaf: <http://xmlns.com/foaf/0.1/>
2. **SELECT**: Specify which variables to return.
   - Example: SELECT ?name ?email
3. **WHERE**: Define the graph pattern to match.
   - Example:

```
WHERE {
  ?person foaf:name ?name .
  ?person foaf:mbox ?email .
}
```

4. **FILTER**: Optional conditions to refine results.
   - Example:

```
FILTER regex(?name, "John")
```

EXAMPLE

Retrieve names and emails of people:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?email
WHERE {
  ?person foaf:name ?name .
  ?person foaf:mbox ?email .
}
```

Example Dataset:

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix ex: <http://example.org/> .

ex:person1 foaf:name "Jane" .
ex:person1 foaf:mbox "jane@example.com" .
ex:person2 foaf:name "John" .
ex:person2 foaf:mbox "john@example.com" .

ex:person1 foaf:knows ex:person2 .
```

Result:

Name	Email
John Doe	john.doe@example.com
Jane Smith	jane.smith@example.org

USE CASES

1. **Data Integration**:
   - Combine data from multiple sources (e.g., integrating healthcare and geographic data).
2. **Linked Data Exploration**:
   - Explore relationships across datasets (e.g., linking books to authors).
3. **Ontology-Based Searches**:
   - Navigate knowledge representations (ontologies) to retrieve data.
4. **Recommendation Systems**:
   - Generate recommendations based on linked data, such as suggesting books based on shared authors.

OPERATIONS

01. TO SPECIFY DATA

02. TO TRANSFORM DATA

03. EXAMPLES

FILTERING

IDENTIFICATION

SORTING

GROUPING

OPERATIONS TO SPECIFY DATA

01. DATA IDENTIFICATION

- Definition : Involves recognizing and cataloging data within an organization to understand its nature, location, and sensitivity.
- Purpose : Essential for data protection, compliance, and effective data management.
- Processes Involved:
  - Data Discovery
  - Data Classification
  - Data Mapping
- Benefits:
  - Enhances data security by identifying sensitive information.
  - Facilitates compliance with data protection regulations.
  - Improves data governance and management.

02. DATA FILTERING

- Definition : The process of selecting a subset of data from a larger dataset based on specific criteria.
- Purpose : Helps in focusing on relevant data for analysis, improving efficiency and accuracy management.
- Methods:
  - Conditional Filtering
  - Range Filtering
  - Pattern Filtering
- Benefits:
  - Reduces data volume for analysis.
  - Enhances data quality by excluding irrelevant information.
  - Improves decision-making by focusing on pertinent data.

03. DATA SORTING

- Definition : Arranging data in a specific order, such as ascending or descending, based on one or more fields.
- Purpose : It organizes data to facilitate easier analysis and interpretation.
- Types:
  - Numerical Sorting
  - Alphabetical Sorting
  - Chronological Sorting
- Benefits:
  - Simplifies data analysis by providing structure.
  - Helps in identifying trends and patterns.
  - Facilitates efficient data retrieval.

04. DATA GROUPING

- Definition : Aggregating individual data points into categories or classes to simplify analysis.

- Purpose : It helps in summarizing large datasets and identifying patterns.

- Methods:
  - Class Intervals (e.g., age groups).
  - Categorical Grouping (e.g., product types).

- Benefits:
  - Simplifies complex data sets.
  - Facilitates the creation of frequency distributions.
  - Aids in statistical analysis and visualization.

AGGREGATION

CLEANING

NORMALIZATION

VALIDATION

INTEGRATION

OPERATIONS TO TRANSFORM DATA

01. DATA CLEANING

- Definition : The process of identifying and correcting errors, inconsistencies, and inaccuracies in datasets.

- Steps Involved:
  - Removing Duplicates: Eliminate redundant entries.
  - Handling Missing Values: Fill gaps using techniques like imputation.
  - Correcting Errors: Fix typos, invalid formats, or incorrect entries.
  - Standardizing Data: Ensure uniformity in data representation.

- Benefits:
  - Improves data quality and reliability.
  - Enables accurate analysis and decision-making.

02. DATA NORMALIZATION

- Definition : Adjusting data to fit within a standard range or distribution.

- Techniques:
  - Min-Max Scaling: Adjusts values to a scale of 0 to 1.
  - Z-Score Normalization: Converts values based on standard deviation.

- Purpose:
  - Removes scale differences in data features.
  - Enhances performance of algorithms like machine learning models.

- Benefits:
  - Simplifies comparisons between variables.
  - Reduces data redundancy.

03. DATA AGGREGATION

- Definition : Summarizing detailed data into a high-level format.

- Methods:
  - Summing Data: Total sales over time.
  - Averaging Data: Calculating average customer ratings.
  - Counting Records: Number of transactions in a day.

- Applications:
  - Simplifies large datasets for dashboards and reports.
  - Enables trend and pattern analysis.

04. DATA INTEGRATION

- Definition : Combining data from multiple sources into a unified view.

- Methods:
  - ETL (Extract, Transform, Load): Moves data from sources to a data warehouse.
  - Real-Time Integration: Combines live data streams.

- Benefits:
  - Provides a consolidated data view.
  - Enables cross-functional insights.
  - Supports interoperability between systems.

05. DATA VALIDATION

- Definition : Ensuring data accuracy, consistency, and completeness.

- Techniques:
  - Schema Validation: Checks structure and format.
  - Range Validation: Verifies values are within acceptable limits.
  - Cross-Validation: Compares related data points for consistency.

- Benefits:
  - Prevents errors in data-driven processes.
  - Improves compliance with regulations.

EXAMPLE

Name	Gender	Crime Committed	Frequency
Meyin	Male	Sleeping	0
Keyur	Male	Using Phone	3
Keyur	Male	Talking	9000
Meet	Male	Using Earphones	once
Vaibhav	Male	Exists	3
Nakul	Male	Talking	1
Rajat		Sitting with Mayin and Shrey	500

EXAMPLE

Name	Gender	Crime Committed	Frequency
Mayin	Male	Sleeping	0
Keyur	Male	Using Phone	3
Keyur	Male	Talking	9000
Meet	Male	Using earphones	once
Vaibhav	Male	Exists	3
Nakul	Male	Talking	1
Rajat		Sitting with Mayin and Shrey	500

CURATED RESULT

Name	Crime Committed	Frequency
Mayin	Sleeping	1000000
Nakul	Talking	10000
Keyur	Using Phone, Talking	9003
Rajat	Sitting with Mayin and Shrey	500
Vaibhav	Exists	3
Meet	Using Earphones	1

REAL-LIFE EXAMPLE

- I've been working on a Pokémon game that required extensive data curation operations and techniques.
- In the upcoming video demo, I invite you to observe and guess which aspects of the game involved data curation efforts.

TECHNIQUES USED

Data Collection

- Character Spritesheets
- Animated Sprites for Pokémon

Data Transformation

- Conversion of .gifs to .pngs
- Organization and Naming: For forms (male, female, shiny).

Data Integration

- Adding Mega Pokémon
- Pokedex Data: Converted from TypeScript dict to JSON.

Data Aggregation

- Moves Data
- Learnsets Data

Data Cleaning and Formatting

- Type Chart and Nature Effectiveness
- Custom Formatting

Q. Which of the following is the correct sequence of steps in Operations to Specify Data?

a) Data Filtering → Data Sorting → Data Identification → Data Grouping

b) Data Identification → Data Filtering → Data Sorting → Data Grouping

c) Data Sorting → Data Identification → Data Filtering → Data Grouping

d) Data Grouping → Data Identification → Data Sorting → Data Filtering

ANS : b

Q. What is the main purpose of Data Normalization in Data Transformation?

a) To remove duplicate records and fix incorrect data

b) To organize data into logical categories based on attributes

c) To ensure data is in a consistent format for comparison

d) To summarize data for high-level insights

ANS : c

Q. Which step in Data Transformation helps combine information from multiple sources into a unified system?

a) Data Cleaning

b) Data Aggregation

c) Data Integration

d) Data Validation

ANS : c

Q. Which of the following is NOT a key component of data curation?

a) Data Identification

b) Data Preservation

c) Data Mining

d) Metadata Creation

ANS : c

Q.What is one of the primary benefits of data curation in an organization?

a) It increases redundant storage costs.

b) It reduces time spent searching for relevant data.

c) It eliminates the need for data integration.

d) It removes the need for compliance with regulations.

ANS : b

Q.Which of the following best describes XQuery?

a) A query language used only for relational databases.

b) A functional programming language for querying and manipulating XML data.

c) A replacement for SQL in modern databases.

d) A markup language for defining XML structures.

ANS : b

Q.What is the main function of SPARQL?

a) To transform XML documents into relational tables.

b) To query and manipulate RDF data using pattern matching.

c) To convert JSON data into XML format.

d) To validate XML schemas and DTDs.

ANS : b

THANK YOU!