Goods: Organizing Google's Datasets

Written by Alon Halevy, Flip Korn, Natalya F. Noy, Christopher Olston, Neoklis Polyzotis, Sudip Roy, Steven Euijong Whang, as published in the 2016 International Conference on Management of Data

Presented by: Avi Kapadia, Avi Athota, and Dhruv Sharma

Introduction and Background

- A project designed to organize structured datasets at Google on an unprecedented scale
- Enables services such as dataset discovery, monitoring, annotations, and relationship analysis among datasets
- Enterprises, especially tech giants like Google, are experiencing an explosion in the number of datasets produced, driven by rapid R&D and innovation cycles
- Despite advanced tools for managing source code, similar systems for datasets are lacking, leading to inefficient data siloing and potential loss of valuable data insights.

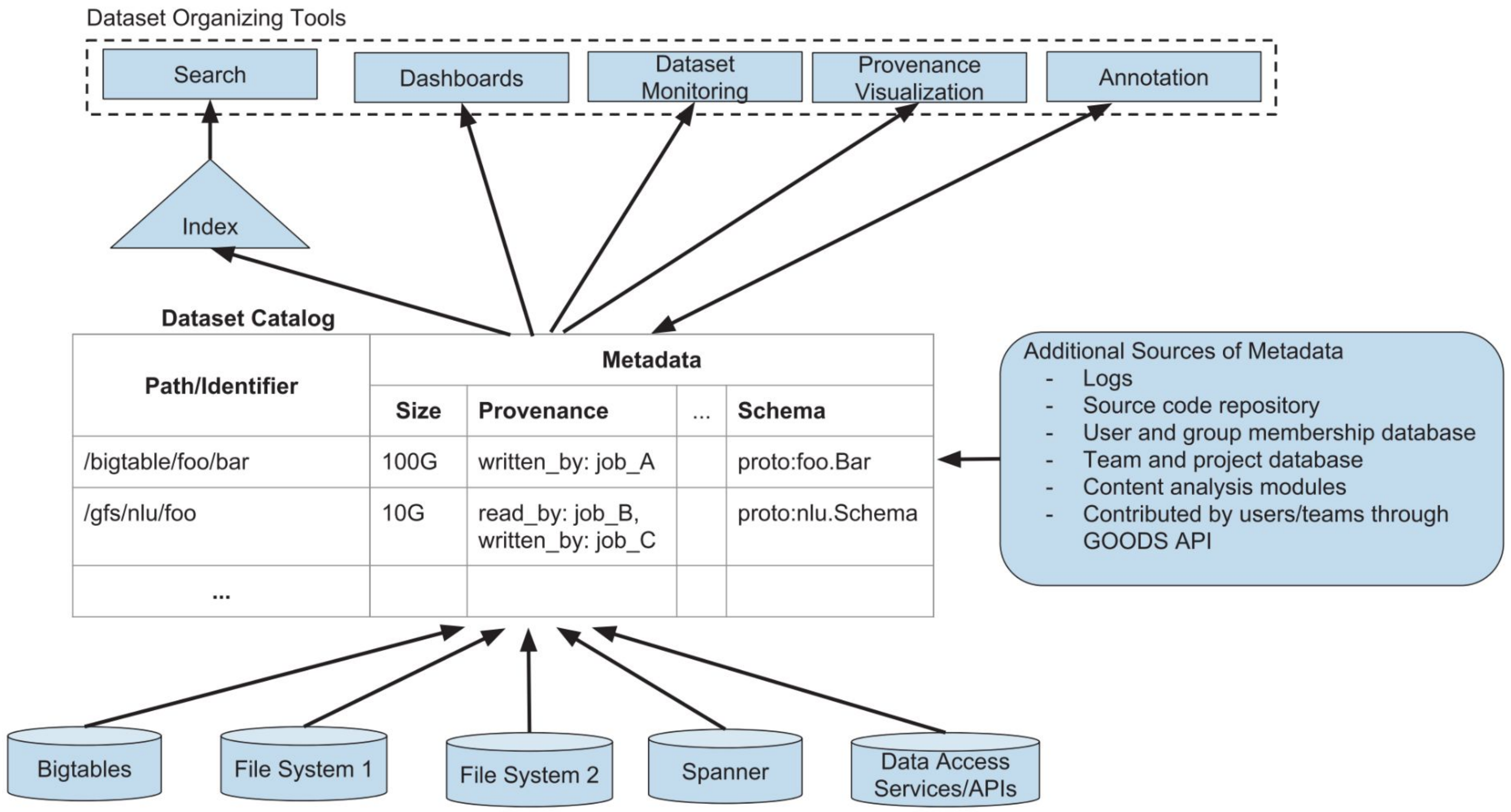
Image from https://zeenea.com/google-goods-the-management-and-data-democratization-tool-of-google/

Past Work

Authors	[IBM Research Team]	[Cloud Services Teams]	Franklin et al.	Howe et al.	Open Knowledge Foundation	Microsoft	Researchers in [Webtables]	Spyglass developers	Propeller Team
Year	N/A	N/A	2005	2008-2010	2006	N/A	Various	N/A	N/A
Contribution	Developed an early data-lake management system focusing on structured enterprise data storage and retrieval.	Offered data-lake solutions integrated into cloud services, facilitating scalable data storage and access but lacking post-hoc metadata aggregation.	Introduced Dataspaces, an attempt to provide flexible integration of data from multiple sources without upfront integration costs.	Created DataHub, a version-control system for datasets inspired by Git, allowing for collaborative dataset management.	Developed CKAN, a management system for open data repositories that enables data sharing and discovery.	Launched Azure Marketplace, focusing on commercial data exchange and standardized dataset formats.	Focused on extracting and indexing HTML tables from the web to create searchable data repositories, yet not addressing structured dataset needs fully.	Designed a metadata-search system enhancing file management through complex queries over file metadata, not adapted to varied dataset formats.	Implemented a fast file-search service updating indices in real-time, yet primarily for file systems, not diverse datasets.
Limits	Scalability issues; lacking tools for metadata management	Tunnel vision on focus; lacking tools for metadata management	Manual effort needed for integration; lacked automated metadata inference	Designed for data scientists but not necessarily enterprise level customers	Focused on public data sharing but not for complex and private data	Mostly commercial, lacked focus on the diversity of in-house enterprise datasets	Did not reference structured datasets; limited to publicly available web data	Focused on file metadata rather than rich dataset metadata; limited query power	Mostly for file systems, not for large-scale dataset management

Overview - the Big Idea of Google Goods

- Revolutionary approach to dataset management
- Core features:
    - Aggregates metadata after datasets have been created
    - Operates transparently in the background
    - Offers dataset search and discovery based on metadata
    - Real time monitoring of metadata
- Main contributions:
    - Successfully manages metadata for tens of billions of datasets.
    - Provides a scalable, flexible platform that enhances the discoverability and usability of datasets across diverse environments.
    - Balances the freedom in dataset creation with structured access and management, fostering an environment conducive to innovation and efficient data use


Dataset Organizing Tools
Search
Dashboards
Dataset Monitoring
Provenance Visualization
Annotation
Index
Dataset Catalog
Additional Sources of Metadata
- Logs
- Source code repository
- User and group membership database
- Team and project database
- Content analysis modules
- Contributed by users/teams through GOODS API
Bigtables
File System 1
File System 2
Spanner
Data Access Services/APIs

Path/Identifier	**Metadata**			
	Size	**Provenance**	...	**Schema**
/bigtable/foo/bar	100G	written_by: job_A		proto:foo.Bar
/gfs/nlu/foo	10G	read_by: job_B, written_by: job_C		proto:nlu.Schema
...				

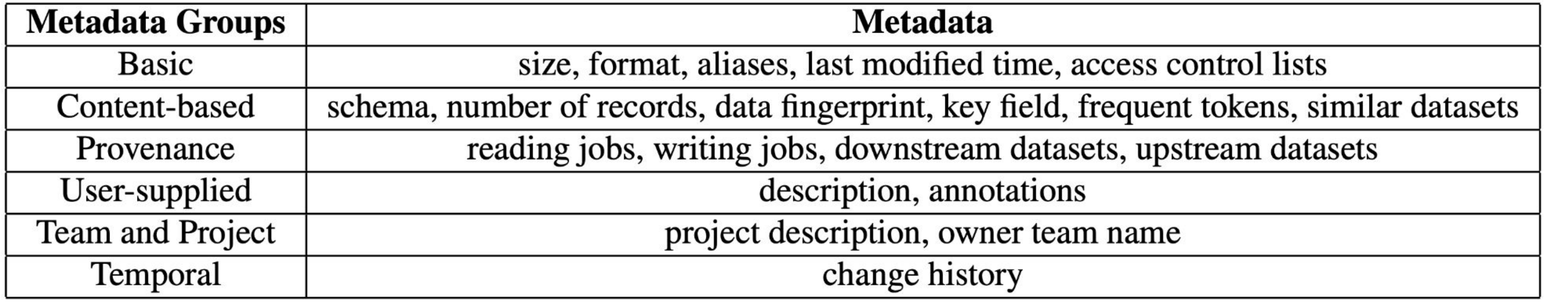
Figure 1: Overview of Google Dataset Search (Goods)

Challenges Addressed

- Scale and Size of Datasets
  - Catalog contains over 26 billion datasets; anticipated to double with additional data
  - This doesn’t even represent data with restricted access
- Variety of Data
  - Datasets vary in format (text files, csv, Bigtables) and storage systems (GoogleFS, database servers)
- Churn of Catalog Entries
  - Churn: Rate at which datasets are added or deleted daily.
  - Time-to-Live (TTL): Designated lifespan of datasets before deletion.
  - Statistics: Approximately 1 billion datasets (~5%) are deleted daily.

Challenges Addressed

- Uncertainty of Metadata Discovery
  - Difficulty in determining complete and accurate metadata post-hoc
- Computing Dataset Importance
  - Importance varies based on user interaction and dataset connectivity
- Rediscovering Dataset Semantics
  - Identifying meaningful data interpretations from raw data is complex



Metadata Groups	Metadata
Basic	size, format, aliases, last modified time, access control lists
Content-based	schema, number of records, data fingerprint, key field, frequent tokens, similar datasets
Provenance	reading jobs, writing jobs, downstream datasets, upstream datasets
User-supplied	description, annotations
Team and Project	project description, owner team name
Temporal	change history

Table 2: Metadata in the Goods catalog

Metadata and Provenance

1. Basic Metadata
2. Provenance
3. Schema
4. Content Summary
5. User-Provided Annotations
6. Semantics
7. Additional Metadata

Organizing datasets into clusters (part 1)

- Rationale for clustering
  - Identifying natural clusters reduces computational overhead.
  - Clusters group similar datasets, providing logical-level abstraction.
  - Saves on metadata extraction by propagating metadata across cluster members.
- Clustering Techniques:
  - Cheap clustering techniques essential to balance computational savings.
  - Utilize dataset paths for hinting on clustering, e.g., timestamps, versions.
  - Compose hierarchies to construct granularity semi-lattice structures.
- Granularity Semi-Lattice:
  - Semi-lattice structure represents different levels of dataset granularity.
  - Each node corresponds to a choice for grouping datasets into clusters.
  - Table 3 lists abstraction dimensions used for constructing the semi-lattice.

Organizing datasets into clusters (part 2)

- Stability and User Experience:
  - Frequent catalog churn requires stable clustering.
  - Simple solution: Create entries only for top-most elements of each semi-lattice.
  - Ensures low number of cluster entries and stable clusters over time.
- Metadata Propagation:
  - Aggregate metadata for clusters by summarizing individual members' metadata.
  - Propagate shared metadata like schema across cluster members.
  - Application-specific decision on materializing or computing metadata on-demand.
- Figure 3: Distribution of Datasets within Clusters:
  - Visualization of the number of datasets within each cluster.
  - Clustering compresses physical datasets into logical datasets, facilitating catalog inspection.
  - Significant computational savings achieved, especially for large clusters.

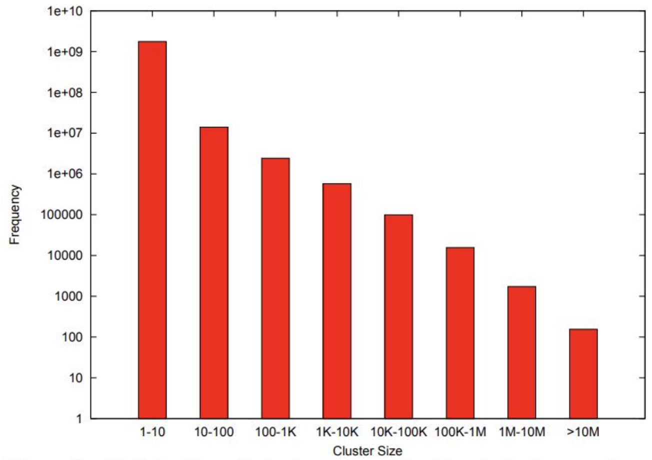


Figure 3: Distribution of cluster sizes. The X axis is the number of datasets in a cluster (cluster size). The Y axis is the number of clusters of the corresponding cluster size.

Backend Implementation: Catalog Storage

Catalog Storage:

- Utilization of Bigtable as the storage medium for the Goods catalog.
- Each row represents a dataset or dataset cluster, with the dataset path or cluster path serving as the key.

Bigtable Features:

- Per-row transactional consistency suits most processing, allowing independent handling of datasets.
- Ability to infer schema and analyze dataset content without needing to access entries for other datasets.

Physical Structure:

- Bigtable comprises independent column families, with separate families optimized for batch processing.
- Storage of metadata categorized into datasets and status information about module outcomes.

Status Metadata:

- Lists modules that processed each dataset entry, along with timestamps, success status, and error messages.
- Crucial for coordinating module execution and system inspection, facilitating debugging and performance analysis.

Batch Job Performance + Scheduling

- Two main types of jobs
  - large # of diverse batch-processing jobs
  - small # of jobs that serve frontend / API

- Key features
  - Performance
  - Independence
  - Scheduling
  - Prioritization
  - Crawler Operations

Fault Tolerance

Per-Dataset Error Handling:

- Modules record errors in status metadata.
- Bounded retries triggered for errors.

Job-wide Status Metadata:

- Job start time and success indication recorded.
- Consistent approach ensures correct provenance graph.

Content Analysis Module Handling:

- Sandbox potentially dangerous jobs in separate processes.
- Watchdog thread prevents long stalls.

Geographical Replication:

- Catalog replicated at multiple locations.
- Asynchronous background replication from master.

Garbage Collection of Metadata

Challenges and Constraints:

- Declarative predicates for removal conditions.
- Prevention of "dangling rows."
- Independence of other modules during garbage collection.

Constraint Details:

- Predicates utilize metadata and module statuses.
- Prevent creation of "dangling rows."
- Modules must run independently during garbage collection.

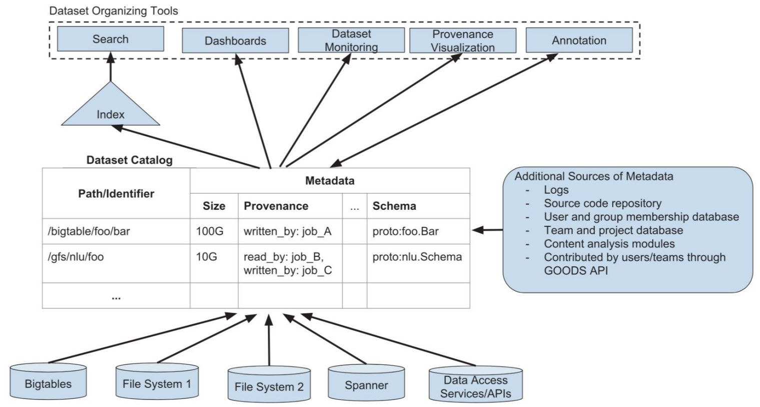
Garbage Collection Process:

- Two-phase approach: tombstone marker and deletion.
- Modules ignore rows with tombstone marker.
- Module iterations limited to 24 hours.

Front End: Serving the Catalog

Dataset Profile Pages

- Export metadata for specific datasets into easy-to-view profile pages.
- Enable editing of metadata for augmentation or correction.
- Balance comprehensive presentation with manageable size.
- Cross-link metadata with specialized tools for enhanced context.


Dataset Organizing Tools
Search
Dashboards
Dataset Monitoring
Provenance Visualization
Annotation
Index
Dataset Catalog
Additional Sources of Metadata
- Logs
- Source code repository
- User and group membership database
- Team and project database
- Content analysis modules
- Contributed by users/teams through GOODS API
Bigtables
File System 1
File System 2
Spanner
Data Access Services/APIs

Path/Identifier	Metadata			
	Size	**Provenance**	...	**Schema**
/bigtable/foo/bar	100G	written_by: job_A		proto:foo.Bar
/gfs/nlu/foo	10G	read_by: job_B, written_by: job_C		proto:nlu.Schema
...				

Figure 1: Overview of Google Dataset Search (Goods). The figure shows the Goods dataset catalog that collects metadata about datasets from various storage systems as well as other sources. We also infer metadata by processing additional sources such as logs and information about dataset owners and their projects, by analyzing content of the datasets, and by collecting input from the Goods users. We use the information in the catalog to build tools for search, monitoring, and visualizing flow of data.

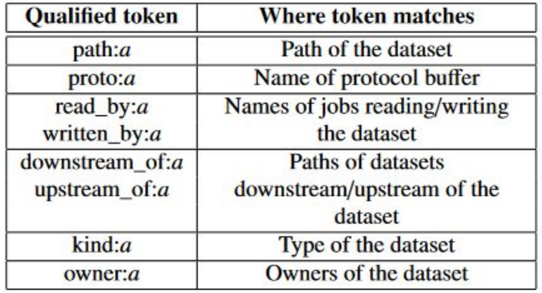
Front End: Serving the Catalog (continued)

Dataset Search

- Enables Googlers to find datasets using keyword queries.
- Backed by conventional inverted index for document retrieval.
- Supports partial matches on dataset paths and other metadata sections.

Team Dashboards

- Configurable one-stop shop for displaying team datasets and metadata.
- Provides health metrics, other dashboards, and storage system status.
- Automatically updates dashboard content as metadata updates.
- Enables dataset monitoring and alerts for expected property failures



Qualified token	Where token matches
path:*a*	Path of the dataset
proto:*a*	Name of protocol buffer
read_by:*a* written_by:*a*	Names of jobs reading/writing the dataset
downstream_of:*a* upstream_of:*a*	Paths of datasets downstream/upstream of the dataset
kind:*a*	Type of the dataset
owner:*a*	Owners of the dataset

Table 4: Examples of qualifying a search token *a* so that it matches different sections of the index. A search query may comprise several qualified and unqualified tokens.

Lessons Learned

Evolve as You Go

- Adapted Goods based on observed user trends and usage logs.
- Examples include audit protocol buffers, re-finding datasets, understanding legacy code, bookmarking and annotating datasets.

Use Domain-Specific Signals for Ranking

- Leveraged domain-specific signals like dataset type and lineage fan-out for ranking.
- Ongoing effort to tune scoring function based on user experience.

Lessons Learned (continue)

Expect and Handle Unusual Datasets

- Encountered unexpected scenarios during development.
- Implemented adhoc and system-level solutions to address issues.

Export Data as Required

- Exported catalog data as triples for visualization and complex queries.
- Utilized specialized engines for powerful query processing.

Ensure Recoverability

- Configured Bigtable for snapshot retention and built custom recovery schemes.
- Implemented early detection mechanisms for data corruption.

Related Work

- Data Lakes and Dataspaces
  - Goods organizes and indexes Google's dataset lake.
  - Similar efforts in other companies and academia focus on data lake management and dataspace concepts.
- Version Management and Data Repositories
  - Contrasts Goods with version management systems like DataHub and data repositories like CKAN and Microsoft Azure Marketplace.
- Search Systems
  - Goods challenges traditional file-search systems like Spyglass and Propeller.
  - Discusses indexing and searching structured datasets in the context of Goods.
- Provenance Management
  - Mentions previous works on provenance management systems like PASS and Trio.
  - Highlights Goods' reliance on weaker signals for provenance inference

Conclusions and Future Work

Challenges

- Ranking datasets effectively.
- Filling missing metadata.
- Understanding semantics of data.
- Improving catalog freshness.
- Instilling a "data culture" in enterprises.

Future Directions

- Integrate with larger knowledge graph.
- Explore combination of post-hoc and proactive dataset registration approaches.
- Develop systems to foster "data culture" in enterprises.

Further Work Elsewhere.

- Data Lake Management Systems:
    - Companies and research institutions are developing systems similar to Goods to organize and index large volumes of data in accessible repositories.
    - These systems aim to streamline data storage, retrieval, and analysis.
- Metadata Catalogs
    - Various platforms exist for metadata management, data discovery, and collaboration.
    - Features include metadata organization, lineage tracking, and dataset profiling.
- Version Control Systems for Data:
    - Similar to DataHub, these systems enable versioning and collaboration on datasets.
    - They help track changes, ensure data integrity, and support collaborative data projects.
- Search Systems for Structured Data:
    - Specialized search systems are designed to efficiently retrieve structured datasets based on metadata attributes and keywords.
    - They enable users to find relevant datasets quickly and accurately.
- Provenance Management Systems
    - Systems like PASS and Trio track data lineage and history, ensuring data quality, reproducibility, and compliance.
    - They provide insights into how data is generated, processed, and used over time.

Study Questions

1) How does the Goods project leverage clustering techniques to reduce computational overhead in metadata extraction for the vast number of datasets within the catalog, and what considerations are taken into account to ensure the efficiency and effectiveness of the clustering process?

2) What does the term “post-hoc” refer to and how does that relate to metadata extraction using Goods?