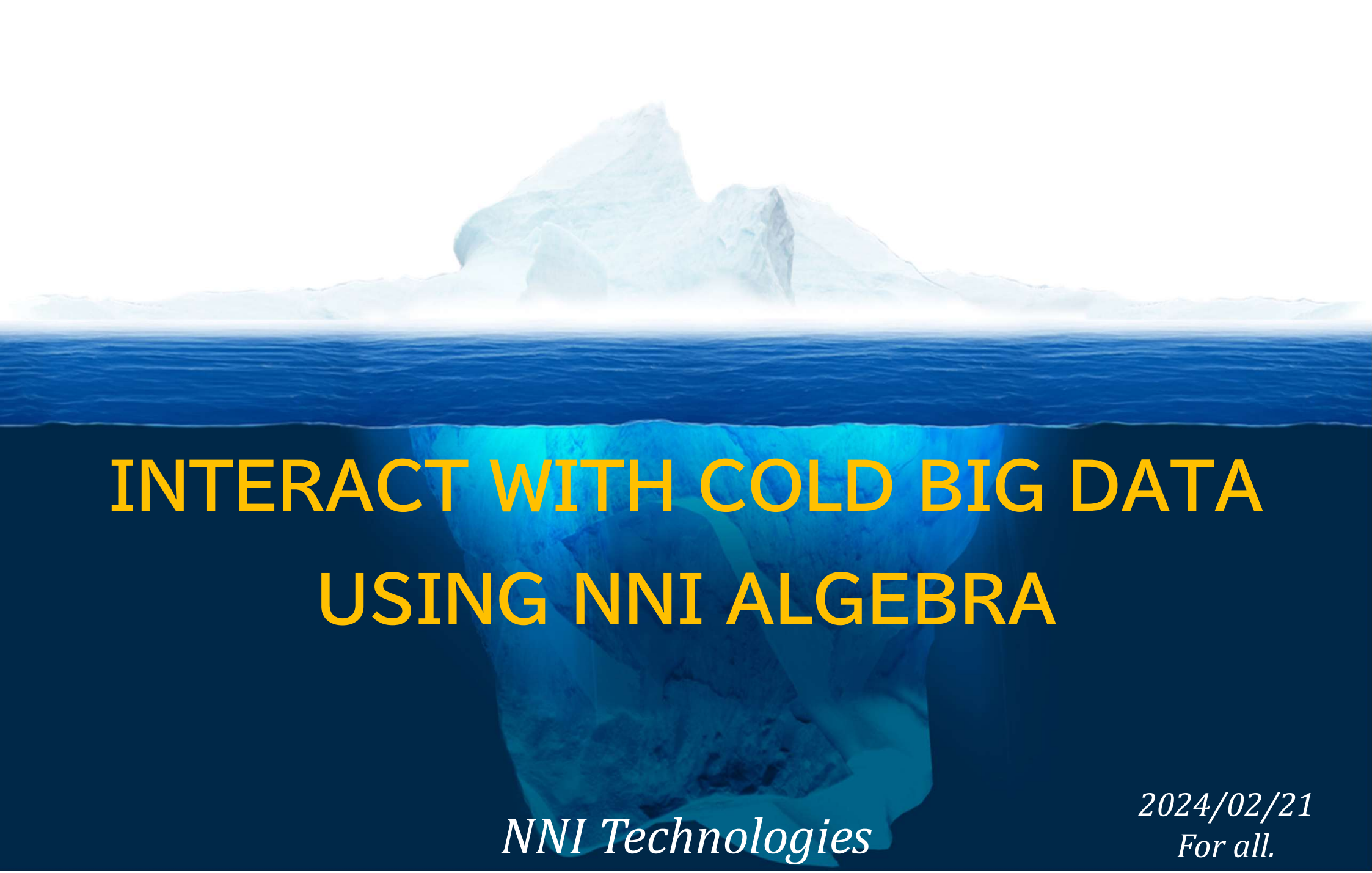


INTERACT WITH COLD BIG DATA USING NNI ALGEBRA

NNI Technologies

2024/02/21
For all.

What is the Natural Number Index (NNI)?

For databases, indexes such as hash and B-Tree are important mechanisms for speeding up searches. However, existing indexes had to be built separately from the data being searched. This resulted in two problems.

1. They require build time and memory allocation, so can only be added to specific columns.
2. They are lost when multiple data are combined, such as by UNION or JOIN operations.

Natural number index (NNI) is an index that uses the inherent properties of the table-structured data being operated to speed up search, etc. It does this by decomposing the data into multiple simple components with easy-to-handle properties, and then combining those components according to the purpose of use. NNI automatically appears on all columns and is not lost when combined, which solves the two problems of existing indexes.

Existing Indexes			Natural Number Index
Only in limited columns	✕	Are they in all columns?	○ Yes
lost	✕	Not lost even if combined	○ Not lost

What is COLD BIG DATA?

COLD BIG DATA is stored in a location that is not easy to access from CPUs, mainly due to cost constraints. It accounts for 80-90% of big data.

Much of the data for fraud detection, failure prediction, credit evaluation, and AI and ML is in this COLD BIG DATA.

However, COLD BIG DATA is difficult to use. As a result, much of it remains underutilized.

COLD BIG DATA residing in various locations

Energy

e.g., at a Nuclear Power Plant

Challenge: For predictive maintenance, efficiency improvement, etc.

Transportation

e.g., Connected Car Data

Challenge: Utilizing the massive archived data in multiple ways

Civil Engineering

e.g., BEMS(Building Energy Management System)

Challenge: Multiple ways big data utilization from diverse equipment

Communication

e.g., Call/Communication/Movement Logs

Challenge: Massive data is generated but unable to fully utilize

Sustainable

CO_2

e.g., Carbon Trading

Challenge: Interconnecting Big Data of the world is underutilized

Why COLD BIG DATA is UNUSABLE and UNSEEN?

RESISTS BEING STRUCTURED

Columns and metrics change

- But semi-structured data is inefficient

WITHOUT INDEXES

- Unindexed data is too time-consuming and costly

TOO MANY

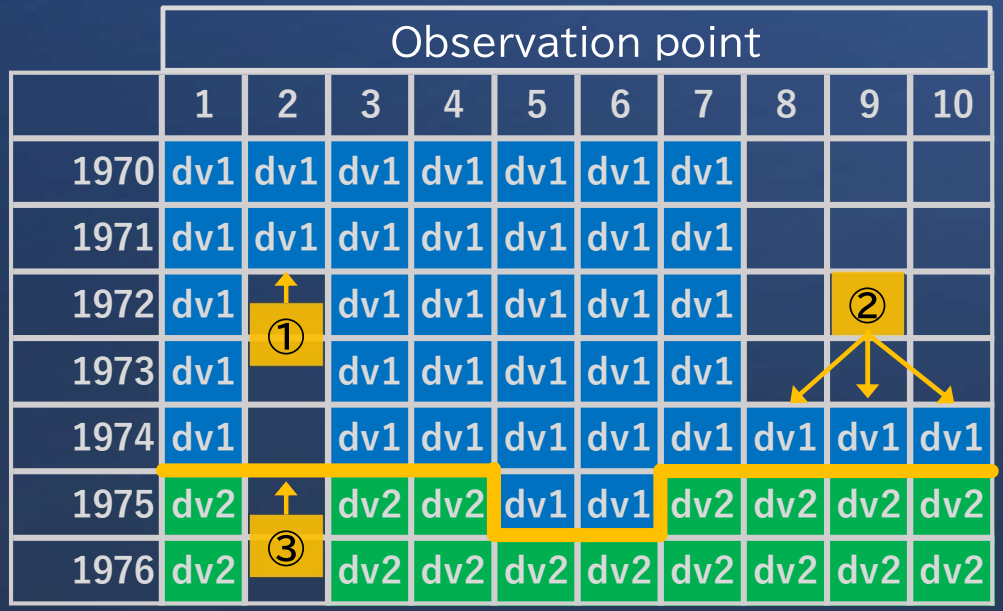
- 10-year data is made up of 3,652 daily data sets

TOO HUGE

- Trillions of Records
- Tens of Thousands of Columns

Reasons Why Big Data Resists Structure

Inevitable Schema, Metrics Evolution



① ② Discontinuation and Addition of Observation Points

③ Turnover of Observation Instruments

Issues Associated with the Turnover of Observation Instruments

Parameters	dv1	dv2
t	10 ms sampling	2 ms sampling
x	-	10 bit
y	-	10 bit
z	8 bit	10 bit

To Ensure Structured, It Is Necessary to:

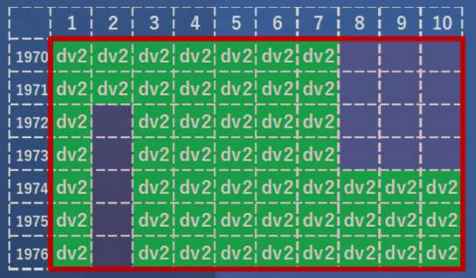
1. Change ‘**t**’ from 10 ms to 2 ms and Implement Latching
2. Add Columns ‘**x**’ and ‘**y**’
3. Convert ‘**z**’ from 8 bits to 10 bits

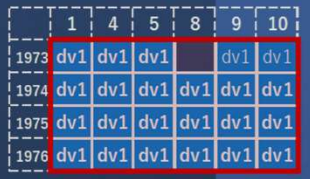
Yet, Unresolved Issues Persist!

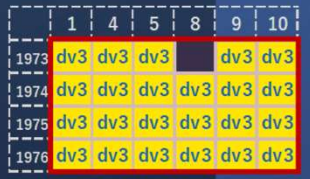
Creating Purpose-Specific Data can solve problems

Source data sets

Purpose-specific data sets
with desired schema and metrics







Generate purpose-specific data sets on demand

Multi-Source Composer(MSC)

The Multi-Source Composer (MSC) is a system that utilizes NNI to enable on-demand integration of diverse big data files stored in the D5A file format distributed worldwide.
The system exhibits flexibility in adapting to changes in schema or metrics, making it a scalable solution for creating purpose-specific data sets from a variety of data sources.

Additionally, all columns in the generated data sets are automatically indexed, enabling real-time search, aggregation, and sorting on any column.
These capabilities make MSC a powerful solution that simultaneously addresses both the un-usability and invisibility issues associated with COLD BIG DATA.

Mapped Table-Structured Data for purpose-specific data

Development of NNI Algebra

NNI algebra was developed through a joint research project with JAXA started in 2019 on ”High-speed Access to Large Time-series Data.“

Invention of Mapped Table-Structured Data

NNI algebra enables the realization of Mapped Table-Structured Data, which can be described as mappings from a set of source tabular data. When a cell requiring a value for display or other purposes, the inverse mapping is used to identify the source of the mapping for that cell and retrieve the value from there.

Compact even with trillions of records

Mapped Table-Structured Data remains compact even with trillions of records because it only needs to store the mapping definitions, not the values themselves.

Combination of thousands is instant

Mapping can be performed in multiple stages, enabling hierarchical combinations. Additionally, thousands of combinations can be easily defined. Since the combinations are pre-defined, they can be completed almost instantaneously.

Virtual indexes automatically appear on all columns

Virtual indexes can be formed in a wide variety of combinations

Source table-formatted data

0	A		0	X
1	B		1	Y
2	C		2	Z

Examples of scenarios where virtual indexes can be successfully implemented

Performance Testing of MSC

Demo. movie

https://youtu.be/oJ5d1JsSLNw

ETL

Combining 1,100 big data and generating 3.15 trillion records in mapped table-structured data takes 0.3 seconds.

Overviewing

The sorting of the above 3.15 trillion records in mapped table-structured data is instantaneous, and the search takes 0.03 seconds.

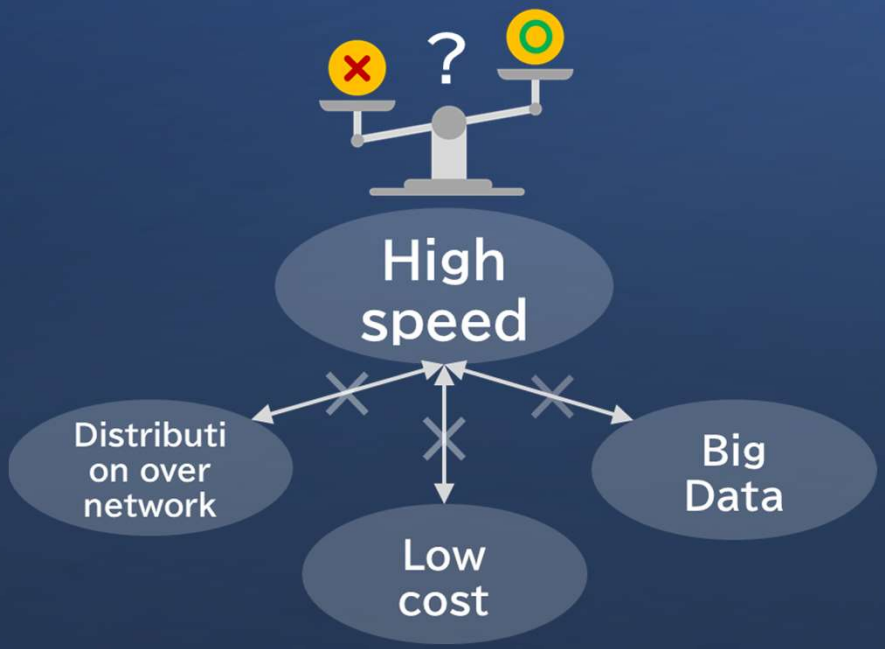
Store / Transfer

The size of the above mapped table-structured data is typically less than 1MB. Therefore, the transfer is easy.

Unparalleled Superiority

(solving three dilemmas and enabling three impossibilities)

Overcoming three dilemmas that conflict with high speed



Enable three impossibilities

1. ETL

Time consuming ⇒ Instant

2. Indexes on generated

No ⇒ Every column

3. Store / Transfer

Difficult ⇒ Easy

Multi Source Composer (MSC)

2023.11.26

https://youtu.be/oJ5d1JsSLNw

MSC v1.0 (to be released in December 2024)

All Columns are Indexed

Exabyte-scale Mapped Table-Structured Data Editor

1. All Columns are Indexed
2. Column Copy and Paste
3. Column Insertion and Deletion
4. Cell Rearrangement
5. Data Reinterpretation
6. Save
7. Transfer

Complementing Existing Technologies

Data Lake House/
Cloud DB

Multi-Source Composer
(MSC)

Comparison with Google BigQuery

The reason for comparing with BigQuery

- A general big data solution.
- Provides an overview of purpose-specific databases.

Big data used for comparison

3.15 trillion records, 4 columns (demo video below)

https://youtu.be/oJ5d1JsSLNw

Search time for the Integer column above

BigQuery: 10 to 25 seconds(estimated)

MSC: 0.03 seconds

The search cost for the Integer column mentioned above

BigQuery: $126.0 per search

MSC: less than $0.05 per search

Calculation of search cost

BigQuery:

Based on Google's pricing table, the cost is $5.0 per 1TB scan.

3.15 trillion records of integers is 25.2TB.

Therefore, 5.0 * 25.2 = $126.0

MSC:

Calculation is based on on-premises PCs and NAS environments with SSDs.

Search like operations are expected to be performed more than 1,000 times per day.

MSC: A Complementary Solution to Existing DBs

Data Lake House/Cloud DB			Multi-Source Composer	
OK	○	Full Managed	✕	Not Yet
OK	○	Rich Database Functionality	✕	No
OK	○	Peripheral Tools & Services	✕	Not Yet
Expensive	✕	Cost	○	Low Cost
PB is tough	✕	Ultra-large Big Data	○	Practically infinite*1
Difficult	✕	Purpose-specific DB	○	Easy
Difficult	✕	Dedicated Databases for Each User	○	Easy
Difficult	✕	Support for Distributed Databases over Networks	○	Easy

*1. 756PB View (in Japanese): https://youtu.be/wkH49DmSPSQ

Integration of existing DB and MSC

DBs can access
COLD BIG DATA
anywhere in the world

Conclusion

- COLD BIG DATA becomes much more accessible and useful
- MSC complements to existing DBs and enables them to access COLD BIG DATA anywhere in the world

NNI Technologies

Home Page

https://www.nni-tech.com/english-top

Demo video
(3.15 trillion records, 4 columns)

https://youtu.be/oJ5d1JsSLNw

756PB of video views, in Japanese

https://youtu.be/wkH49DmSPSQ

Appendix

1. MSC Targets: Structured Data and Semi-structured Data
2. How to integrate data from multiple sources
3. D5A Format

MSC Targets: Structured Data and Semi-structured Data

Structured Data

Both are targets of MSC

Semi-structured Data

- Table formatted data (target)

Often convertible

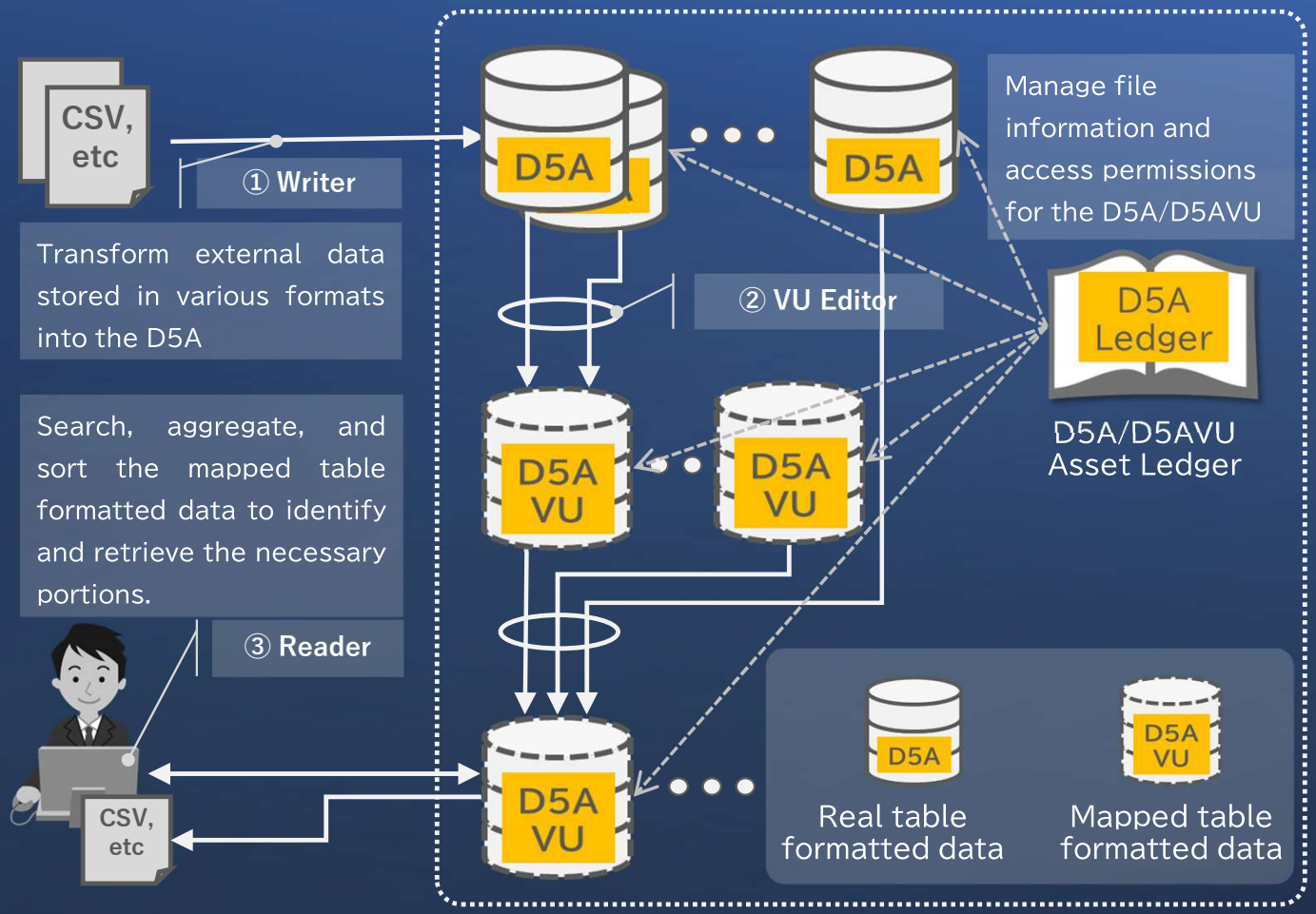
- Data formats such as XML and JSON (not direct targets)

NNI technology does not handle

Unstructured data

- Text Data: Books, Articles, Web Pages, Emails, etc.
- Image Data: Photos, Illustrations, Diagrams, etc.
- Audio Data: Music, Conversation, etc.
- Video Data: Movies, TV shows, YouTube videos, etc.

How to integrate data from multiple sources



Real table formatted data is stored in the D5A file format, while mapped table formatted data is stored in the D5AVU file format. A D5A file contains actual data values, while a D5AVU file contains mappings that define how the data in the mapped table format is derived from the source table format (whether real or mapped).

The ② VU Editor is a tool for defining and editing mappings that generate new mapped table formatted data from source D5A and D5AVU files.

The D5A Ledger manages information such as file information and access rights for D5A and D5AVU files. For example, if an organization is granted access to the D5A Ledgers of both JAXA and NASA, it can create and use new D5AVU files by combining the D5A and D5AVU files from both organizations.

D5A Format

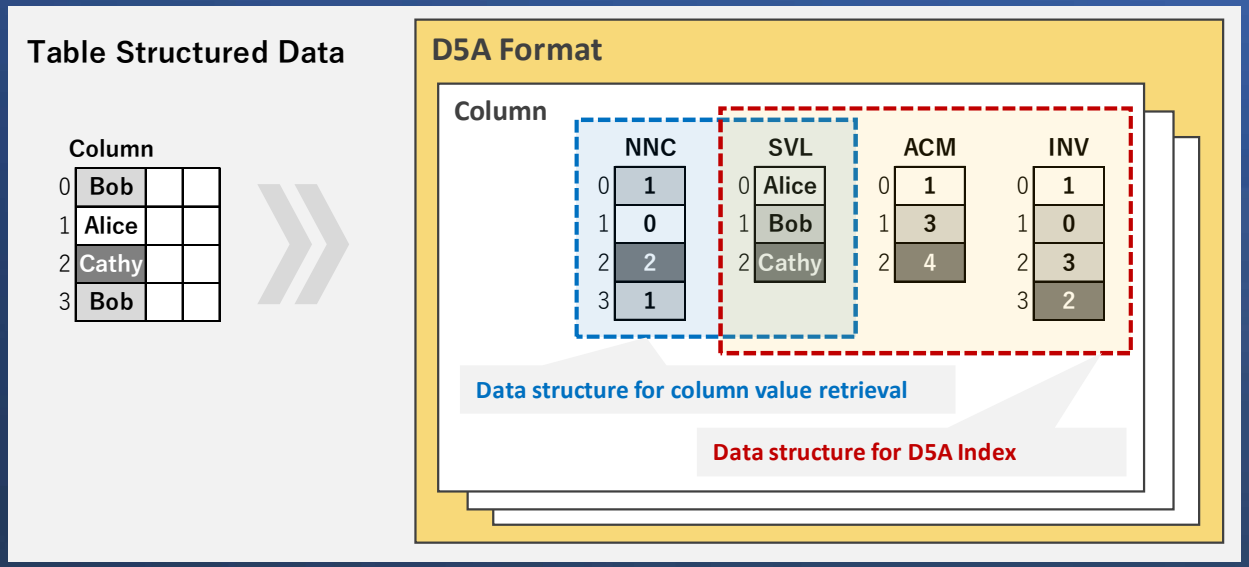


Figure 1. Table structured data and D5A Format

Each column in D5A consists of two juxtaposed structures described below.

The first structure is for retrieving column values, and is the part enclosed by the blue dotted line in Figure 1.

- **SVL** (Sorted Value List) is a structure that extracts the values appearing in the column and sorts them uniquely and in ascending order.
- **NNC** (Natural Numbered Column) is a structure that replaces the column values with their storage positions on the SVL.

The second structure is for the D5A Index, which is an index built on the D5A column. It is the part enclosed by the red dotted line in Figure 1.

This second structure consists of SVL, ACM, and INV.

ACM (ACcuMulation array) is a structure that maps values in SVL[i] to **INV** (INVerted record number) according to the following relationship:

$ACM[-1] \equiv 0$

$ACM[i] \Leftrightarrow (INV[ACM[i-1]], \cdots, INV[ACM[i]-1])$

For example, Bob on SVL has a storage position of 1, so it becomes the following:

$ACM[1] \Leftrightarrow (INV[ACM[1-1]], \cdots, INV[ACM[1]-1])$

$\Leftrightarrow (INV[ACM[0]], \cdots, INV[ACM[1]-1])$

$\Leftrightarrow (INV[1], \cdots, INV[3-1])$

$\Leftrightarrow (0, 3)$

The D5A Index allows you to easily identify the record numbers where a value v appears as follows:

1. Use binary search to find the storage position i of v in the SVL.
2. Use the formula above to identify the range of INV.

Thank you