


# Fabric - naming conventions

 SYSM-353 - Jira project doesn't exist or you don't have permission to view it.



- Naming Conventions for Tables, Columns and Data Structures
- Platform Context: Microsoft Fabric (Lakehouse / Warehouse)

Version	Date	Description	Contributor
V0.1	16 Mar 2026	Initial document	COLOMBANI Théo
V0.2	29 Apr 2026	Revised document: <ul style="list-style-type: none"><li>• added new suffixes for columns naming</li><li>• added section 9 and 10</li></ul>	COLOMBANI Théo
V0.3	07 Apr 2026	Added to the wiki	COLOMBANI Théo
V0.4			

- 1. Purpose and Scope
- 2. Core Platform Principles
  - 2.1 Medallion Data Architecture
  - 2.2 One Object = One Concept
  - 2.3 Naming Style
- 3. Platform Naming Structure
- 4. Table Naming Standards
  - 4.1 Application - Tables
  - 4.2 Platform - Tables
  - 4.3 Enterprise - Tables
- 5. Enterprise Table - Types
  - Dimension Tables
  - Fact Tables
  - Bridge Tables
  - Reference Tables
  - Aggregate Tables
- 6. Column Naming Standards
  - 6.1 Identity & Keys
  - Decision Rules: `_id` vs `_key` vs `_cd`
  - 6.2 Codes & Classifications
  - 6.3 Names & Descriptions
  - 6.4 Dates & Time
  - 6.5 Financial & Monetary
  - 6.6 Quantities & Measures
  - 6.7 Percentages & Ratios
  - 6.8 Boolean Fields
  - 6.9 Status & Lifecycle
  - 6.10 Audit & Metadata
  - 6.11 Technical Fields
  - 6.12 Event / Log Data
  - 6.13 Geography
- 7. Column Type Standards
- 8. Mandatory Metadata Columns
  - Application Layer
  - Platform Layer
  - Enterprise Layer
  - Others Technical standards
- 9. Error and Data Quality Tables
  - Naming Patterns
  - Key Differences
  - When to Use Each Type
    - `reject_<object>`

- quarantine\_<object>
  - error\_<object>
  - Columns
  - 10. Mapping Tables
    - Naming Pattern
    - Common Use Cases
      - Code Standardization
      - Entity Alignment
      - Category Harmonization
      - Hierarchy Standardization
    - Typical Structure
- 

# 1. Purpose and Scope

This document defines the enterprise standards for naming tables, columns, schemas and data structures within the organization's Data Platform implemented on Microsoft Fabric.

The goal of this standard is to ensure:

- Consistent data structures across the platform
- Improved discoverability of datasets
- Simplified onboarding of data engineers and analysts
- Clear lineage between data layers
- Support for BI dimensional models and analytics workloads

This standard focuses specifically on naming conventions for:

- Tables
  - Columns
  - Column types
  - Data layers
  - Domain schemas
  - Supporting technical tables
- 

# 2. Core Platform Principles

## 2.1 Medallion Data Architecture

The data platform follows a layered Medallion architecture where data quality increases progressively across three layers:

application platform enterprise

- application contains raw ingested data.
- platform contains cleaned and standardized data.
- enterprise contains business-ready datasets optimized for analytics.

## 2.2 One Object = One Concept

Each table must represent one logical business concept or process.

Examples:

- customer
- sales\_order
- invoice\_line

Tables should never represent multiple unrelated concepts, whatever the layer.

## 2.3 Naming Style

All names must follow the same style:

- lowercase only
- snake\_case format
- singular nouns
- explicit descriptive names
- no spaces
- no special characters

Example:

- sales\_order
  - customer\_account
  - inventory\_movement
- 

## 3. Platform Naming Structure

The general naming structure of the platform is: *<grouping>.<object>*

Examples:

- application : sap.vbak
  - platform : sap.sales\_order\_header
  - entreprise : sales.fact\_sales\_order
- 

## 4. Table Naming Standards

### 4.1 Application - Tables

application tables store raw source data exactly as ingested.

Naming pattern: *<source\_system>.<source\_object>*

Examples:

- sap.vbak
- sap.vbap
- salesforce.account
- workday.worker

Rules:

- table names follow the source system object names
- minimal transformation

### 4.2 Platform - Tables

Platform tables contain cleaned and standardized datasets aligned to source systems.

Naming pattern: *<source\_system>.<object\_name>*

Examples:

- sap.sales\_order\_header
- sap.sales\_order\_item
- salesforce.account

Rules:

- normalized column naming
- deduplicated records
- standardized data types
- still aligned with source system entities

### 4.3 Enterprise - Tables

Enterprise tables contain enterprise analytics datasets organized by business domain.

Naming pattern: *<domain>.<object>*

Examples:

- sales.fact\_sales\_order
  - sales.dim\_customer
  - finance.fact\_invoice
- 

## 5. Enterprise Table - Types

## Dimension Tables

Pattern: *dim\_<entity>*

Examples:

- dim\_customer
- dim\_product
- dim\_employee
- dim\_date

## Fact Tables

Pattern: *fact\_<xxxx>*

Examples:

- fact\_sales\_order
- fact\_invoice
- fact\_inventory\_movement

## Bridge Tables

Pattern: *brg\_<entity>\_<entity>*

Examples:

- brg\_customer\_segment
- brg\_product\_category

## Reference Tables

Pattern: *ref\_<reference\_object>*

Examples:

- ref\_currency
- ref\_country
- ref\_status

## Aggregate Tables

Pattern: *agg\_<metric>\_<grain>*

Examples:

- agg\_daily\_sales
  - agg\_monthly\_revenue
- 

# 6. Column Naming Standards

Column names must be explicit, consistent, and semantically meaningful. All columns must follow this structure:

*<business\_term>\_<suffix>*

Examples:

- customer\_id
- order\_amt
- payment\_status\_cd

## 6.1 Identity & Keys

Suffixes:

- **\_id** Unique business identifier coming from a source system or representing a real-world entity. It is stable and meaningful outside the data platform.  
Example: *customer\_id = "C12345"*
- **\_key** Surrogate key generated within the data platform for internal joins and modeling purposes (typically in dimensional models).  
Example: *customer\_key = 102938*
- **\_ref** External reference identifier used to link to external systems, documents, or transactions.  
Example: *order\_ref = "EXT-99821"*

## Decision Rules: **\_id** vs **\_key** vs **\_cd**

Use **\_id** when:

- The identifier exists in the source system
- It represents a real business entity
- It can be understood by business users

Use **\_key** when:

- The identifier is generated inside the data platform
- It is used for joins between tables
- It supports dimensional modeling (star schema)

Use **\_cd** when:

- The value represents a classification or coded value
- It belongs to a controlled list or reference table

Summary:

- **\_id** business identifier
- **\_key** technical surrogate key
- **\_cd** classification code

## 6.2 Codes & Classifications

Suffixes:

- **\_cd** Short coded value representing a classification (often linked to a reference table).  
Example: *status\_cd = "SHIPPED"*
- **\_type** High-level classification describing the nature or category of an entity.  
Example: *customer\_type = "B2B"*
- **\_category** Business grouping used to organize entities into categories.  
Example: *product\_category = "ELECTRONICS"*
- **\_segment** Business segmentation (e.g., customer segmentation, marketing segmentation).  
Example: *customer\_segment = "PREMIUM"*
- **\_level\_cd** Code representing a hierarchical level or classification tier.  
Example: *risk\_level\_cd = "HIGH"*

## 6.3 Names & Descriptions

Suffixes:

- **\_name** Human-readable name of an entity or object.  
Example: *customer\_nm = "John Smith"*
- **\_desc** Detailed textual description providing more context or explanation.  
Example: *status\_desc = "Order has been shipped"*
- **\_label** Display-friendly label used for reporting or UI purposes.  
Example: *product\_label = "Laptop - Premium Range"*

## 6.4 Dates & Time

Suffixes:

- **\_dt** Business date without time component, used for events like orders or deliveries.  
Example: *order\_dt = "2025-03-12"*
- **\_ts** Timestamp including date and time, typically used for technical or event tracking.  
Example: *created\_ts = "2025-03-12 14:25:32"*
- **\_time** Time-only value (rare use cases such as time-of-day analysis).  
Example: *event\_time = "14:25:32"*

## 6.5 Financial & Monetary

Suffixes:

- **\_amt** Monetary amount representing a financial value.  
Example: *total\_amt = 1250.75*
- **\_cost\_amt** Cost associated with a product, service, or operation.  
Example: *unit\_cost\_amt = 45.20*
- **\_price\_amt** Price of a product or service.  
Example: *unit\_price\_amt = 60.00*
- **\_rev\_amt** Revenue generated from transactions.  
Example: *net\_rev\_amt = 1200.00*
- **\_tax\_amt** Tax amount applied to a transaction.  
Example: *tax\_amt = 250.75*
- **\_disc\_amt** Discount amount applied to a transaction.  
Example: *disc\_amt = 50.00*

## 6.6 Quantities & Measures

Suffixes:

- **\_qty** Measured quantity of items or units.  
Example: *order\_qty = 3*
- **\_cnt** Integer count of occurrences or records.  
Example: *item\_cnt = 5*
- **\_vol** Volume measurement (e.g., shipment or storage volume).  
Example: *shipment\_vol = 1.5*
- **\_wt** Weight measurement of an item or shipment.  
Example: *product\_wt = 2.3*
- **\_len** Length or size measurement.  
Example: *cable\_len = 1.2*

## 6.7 Percentages & Ratios

Suffixes:

- **\_pct** Percentage value expressed between 0 and 100.  
Example: *margin\_pct = 35.5*
- **\_rate** Ratio or rate, often expressed between 0 and 1 or as a fraction.  
Example: *conversion\_rate = 0.12*
- **\_ratio** Relationship between two values (e.g., debt ratio).  
Example: *debt\_ratio = 0.45*
- **\_idx** Indexed value used for benchmarking or comparison.  
Example: *price\_idx = 105*

## 6.8 Boolean Fields

Suffixes:

- **is\_<condition>** Boolean flag indicating whether a condition is true or false.  
Example: *is\_active = true*
- **has\_<condition>** Boolean flag indicating the presence of something (less preferred).  
Example: *has\_discount = false*

Rule:

- Always prefer is\_

## 6.9 Status & Lifecycle

Suffixes:

- **\_status\_cd** Code representing the current status of an entity or process.  
Example: *order\_status\_cd = "DELIVERED"*
- **\_status\_desc** Description of the status for readability and reporting.  
Example: *order\_status\_desc = "Order successfully delivered"*

## 6.10 Audit & Metadata

Columns:

- **created\_ts** Timestamp when the record was first created in the system.  
Example: *created\_ts = "2025-03-12 10:00:00"*
- **updated\_ts** Timestamp of the most recent update applied to the record.  
Example: *updated\_ts = "2025-03-12 12:30:00"*

- **load\_ts** Timestamp when the data was loaded into the current layer.  
Example: *load\_ts* = "2025-03-12 13:00:00"
- **pipeline\_run\_id** Identifier of the pipeline execution that produced the data.  
Example: *pipeline\_run\_id* = "run\_20250312\_01"
- **record\_source** Source system or process that generated the record.  
Example: *record\_source* = "sap"
- **row\_hash** Hash value used to detect changes in the record.  
Example: *row\_hash* = "a94f8c1d2b"

## 6.11 Technical Fields

Suffixes:

- **\_hash** Hash value used for change detection or deduplication.  
Example: *row\_hash* = "a94f8c1d2b"
- **\_version** Version number of a model, record, or logic.  
Example: *model\_version* = "v1.2"
- **\_seq** Sequence number representing ordering or position.  
Example: *line\_seq* = 1

## 6.12 Event / Log Data

Suffixes:

- **\_event** Name or type of an event.  
Example: *event\_type* = "LOGIN"
- **\_ts** Timestamp when the event occurred.  
Example: *event\_ts* = "2025-03-12 09:15:00"
- **\_id** Unique identifier of the event.  
Example: *event\_id* = "EVT123456"

## 6.13 Geography

Suffixes:

- **\_country\_cd** Country code based on a standard reference (e.g., ISO).  
Example: *country\_cd* = "FR"
- **\_region\_cd** Region or state code.  
Example: *region\_cd* = "IDF"
- **\_city** City name.  
Example: *city* = "Paris"
- **\_postal\_cd** Postal or ZIP code.  
Example: *postal\_cd* = "75001"

# 7. Column Type Standards

Suffix	Expected Type
<b>_id</b>	string or integer
<b>_key</b>	integer
<b>_ref</b>	string
<b>_cd</b>	string
<b>_type</b>	string
<b>_category</b>	string
<b>_segment</b>	string
<b>_level_cd</b>	string
<b>_name</b>	string
<b>_desc</b>	string
<b>_label</b>	string

<b>_dt</b>	date
<b>_ts</b>	timestamp
<b>_time</b>	time
<b>_amt</b>	decimal
<b>_cost_amt</b>	decimal
<b>_price_amt</b>	decimal
<b>_rev_amt</b>	decimal
<b>_tax_amt</b>	decimal
<b>_disc_amt</b>	decimal
<b>_qty</b>	decimal or integer
<b>_cnt</b>	integer
<b>_vol</b>	decimal
<b>_wt</b>	decimal
<b>_len</b>	decimal
<b>_pct</b>	decimal
<b>_rate</b>	decimal
<b>_ratio</b>	decimal
<b>_idx</b>	decimal or integer
<b>is_</b>	boolean
<b>has_</b>	boolean
<b>_status_cd</b>	string
<b>_status_desc</b>	string
<b>_hash</b>	string
<b>_version</b>	string
<b>_seq</b>	integer
<b>_event</b>	string
<b>_country_cd</b>	string
<b>_region_cd</b>	string
<b>_city</b>	string
<b>_postal_cd</b>	string
<b>created_ts</b>	timestamp
<b>updated_ts</b>	timestamp
<b>load_ts</b>	timestamp
<b>pipeline_run_id</b>	string
<b>record_source</b>	string
<b>row_hash</b>	string

---

## 8. Mandatory Metadata Columns

## Application Layer

The application layer captures raw ingestion information to ensure traceability and replay capability.

Column Name	Type	Description	Example
_ingestion_ts	timestamp	Timestamp when the record was ingested into the platform. Used to track ingestion time and support replay or debugging of ingestion pipelines.	2025-03-12 14:25:32
_batch_id	string	Identifier of the ingestion batch or pipeline execution that loaded the record. Useful for pipeline monitoring and troubleshooting.	batch_20250312_01
_source_system	string	Name of the source system from which the record originated. Helps identify data lineage and upstream systems.	sap
_source_file (or source object)	string	Name or path of the file used to ingest the record (when file-based ingestion is used). Enables traceability back to the original data file.	Sap_sales_20250312.csv
_record_hash	string	Hash value representing the content of the record. Used to detect changes between loads or identify duplicates efficiently.	a9f4b12c89d2

## Platform Layer

Column Name	Type	Description	Example
created_ts	timestamp	Timestamp when the record was first created in the platform layer. Helps identify when the data became available in the curated layer.	2025-03-12 14:25:32
updated_ts	timestamp	Timestamp of the latest update applied to the record in the platform layer. Useful for incremental processing and debugging transformations.	2025-03-12 16:02:01
row_hash	string	Hash calculated from the business columns of the row. Used to detect if the data has changed between loads.	8fa21a3b5d
record_source	string	Source system or pipeline that generated the record in the platform layer. Used for lineage and audit.	sap_sales_pipeline
is_deleted	boolean	Logical deletion flag used when the source record is removed or marked as inactive. Enables soft deletion instead of physical removal.	false

## Enterprise Layer

Column Name	Type	Description	Example
created_ts	timestamp	Timestamp when the record was first created in the enterprise dataset. Useful for lineage and auditing.	2025-03-12 17:30:10
updated_ts	timestamp	Timestamp of the latest modification applied to the record. Helps identify refresh cycles of the dataset.	2025-03-13 02:00:00

## Others Technical standards

Column Name	Type	Description	Example
load_ts	timestamp	Timestamp when the record was loaded into the table.	2025-03-13 02:00:00
pipeline_run_id	string	Identifier of the pipeline or job execution that produced the object.	pipeline_run_84721

## 9. Error and Data Quality Tables

Error and data quality tables are used to **manage invalid, incomplete, or inconsistent data** during ingestion and transformation processes.

They are critical to:

- ensure data quality without blocking pipelines
- isolate problematic records, enable debugging and monitoring
- support data governance and audit

### Naming Patterns

- reject\_<object>
- quarantine\_<object>
- error\_<object>

Examples:

- reject\_sales\_order
- quarantine\_customer
- Error\_invoice

## Key Differences

- reject invalid data (cannot be used)
- quarantine questionable data (needs review)
- error technical processing issue

## When to Use Each Type

### reject\_<object>

Use this table when:

- records fail **hard validation rules**
- data is clearly invalid and cannot be processed
- records must be excluded from downstream layers

Typical cases:

- missing mandatory fields (e.g., customer\_id is null)
- invalid formats (e.g., wrong date format)
- broken schema

Example:

- reject\_sales\_order contains orders where order\_id is missing

### quarantine\_<object>

Use this table when:

- data is **potentially usable but requires review**
- validation rules are not strict blockers
- business validation fails but data may still be recoverable

Typical cases:

- unknown reference values (e.g., unknown country\_cd)
- inconsistent business logic
- partial data issues

Example:

- quarantine\_customer contains customers with missing segmentation

### error\_<object>

Use this table when:

- errors occur during **processing or transformation logic**
- the issue is technical rather than data quality related

Typical cases:

- transformation failures
- join issues
- pipeline execution errors

Example:

- error\_invoice contains records that failed during currency conversion

## Columns

Typical structure:

- object\_id Identifier of the record related to the error (usually a business identifier).  
Example: *object\_id = "ORD12345"*

- `source_object` Name of the data platform object (table or dataset) where the error was detected. This refers to the layer, domain, and table within the platform.  
Example: `source_object = "platform.sap.sales_order_header"`
  - `column_name` Name of the column that caused the error (when applicable). Helps pinpoint the issue precisely.  
Example: `column_name = "currency_cd"`
  - `invalid_value` Value that caused the error. Useful for debugging and data quality analysis.  
Example: `invalid_value = "EUROPE"`
  - `error_code` Standardized error code used to classify the type of issue. Enables monitoring and grouping of errors.  
Example: `error_code = "CURR_001"`
  - `error_message` Human-readable description of the error explaining what went wrong.  
Example: `error_message = "Invalid currency code"`
  - `error_type` Category of error indicating whether it is a data issue, validation issue, or technical issue.  
Example: `error_type = "DATA_VALIDATION"`
  - `pipeline_run_id` Identifier of the pipeline execution that generated the error. Used for traceability and debugging.  
Example: `pipeline_run_id = "run_20250312_01"`
  - `load_ts` Timestamp when the error record was written to the error table.  
Example: `load_ts = "2025-03-12 14:32:10"`
- 

## 10. Mapping Tables

Mapping tables are used to **translate, standardize, or align data between source-oriented datasets and enterprise business models.**

Especially for an architectures where:

- lower layers are source-system oriented
- upper layers are business-domain oriented

Mapping tables are used to:

- align source system values with enterprise standards
- resolve inconsistencies between systems
- standardize business definitions
- enable cross-system integration

### Naming Pattern

`map_{source_object}_{business_object}`

Example:

- `map_sap_customer_enterprise_customer`

## When to Use Mapping Tables

Use mapping tables when:

- multiple systems use different identifiers or codes
- business definitions differ across systems
- values need to be harmonized for analytics

### Common Use Cases

#### Code Standardization

Different systems use different formats for the same concept.

Example:

- `source_value = "FR"`
- `target_value = "FRA"`

#### Entity Alignment

Different systems represent the same entity differently.

Example:

- `SAP customer_id` differs from `CRM customer_id`

## Category Harmonization

Source categories do not match enterprise categories.

Example:

- source\_value = "ELEC"
- target\_value = "ELECTRONICS"

## Hierarchy Standardization

Different hierarchy structures across systems.

Example:

- local region vs global region



## Typical Structure

Mapping tables usually contain:

- source\_system Origin of the data
  - Example: *source\_system* = "sap"
- source\_value Value from the source system
  - Example: *source\_value* = "ELEC"
- target\_value Standardized enterprise value
  - Example: *target\_value* = "ELECTRONICS"
- mapping\_type Type of mapping applied
  - Example: *mapping\_type* = "CATEGORY\_MAPPING"
- valid\_from\_dt Start date of the mapping validity
  - Example: *valid\_from\_dt* = "2024-01-01"
- valid\_to\_dt End date of the mapping validity
  - Example: *valid\_to\_dt* = "9999-12-31"

## Workflow history

This view shows the 5 most recent entries. The complete workflow log is available from the 'Document Activity' menu item.

Apr 07, 2026	Actor	Type	Activity	Version
	 COLOMBANI-ext, Theo	Edit	updated the page at 9:32 am	
	 COLOMBANI-ext, Theo	Edit	created the page at 9:31 am	