# Data Model Design Strategy: Financial Data Lake

Context: Harmonizing Sybase (Legacy) and Postgres (Modern) into Snowflake for Agentic Consumption.

Pattern: Medallion Architecture (Bronze -> Silver -> Gold).

Core Constraint: T-1 Daily Batch Loading.

## Part 1: The Design Process (Step-by-Step)

Do not rush into writing DDL. Follow this 4-step process to ensure the model supports both traditional reporting and AI Agent queries.

### Step 1: Source Profiling (The "Truth" Hunt)

*Objective: Understand the physical reality of the legacy data.*

* **Action:** Run profiling queries on Sybase Trades and Positions tables.
* **Check for:**
  + **Implicit Nulls:** Does Sybase use empty strings "" or magic numbers -99 instead of NULL?
  + **Date Formats:** Are dates stored as VARCHAR (e.g., "20241025") or integers?
  + **Precision:** What is the max scale of price and quantity? (e.g., Crypto might need 18 decimal places).

### Step 2: The Silver Layer Design (Normalization & Cleaning)

*Objective: Make disparate systems look like one system.*

* **Action:** Define the standard data types and naming conventions.
* **The Rules:**
  + **Naming:** Snake\_case everywhere (trade\_date, not TradeDate).
  + **Booleans:** Convert Sybase 1/0 and Y/N to Snowflake BOOLEAN (TRUE/FALSE).
  + **Timestamps:** All timestamps must be cast to TIMESTAMP\_NTZ (No Timezone) and normalized to **UTC**.
  + **Strings:** TRIM() all text fields.

### Step 3: The Gold Layer Design (Business Logic & History)

*Objective: Structure data for the Agent's questions ("How did risk change?").*

* **Action:** Select the History Strategy for each entity.
  + **Transactions (Trades):** Immutable Fact Table. Append-only.
  + **States (Positions/Risk):** Periodic Snapshot Fact Table. (Add snapshot\_date).
  + **Entities (Clients/Products):** Type 2 Slowly Changing Dimension (SCD). (Add valid\_from, valid\_to).

### Step 4: The Semantic Layer (Views)

*Objective: Protect the Agent from complexity.*

* **Action:** Define the Views that filter out "expired" history.
* **Rule:** The Agent should *never* have to write WHERE valid\_to IS NULL. Provide a \_current view that does it for them.

## Part 2: Critical Issues & "Bear Traps"

In financial data modeling, these are the most common failure points.

### 1. The Symbology Problem (The "Tower of Babel")

* **Issue:** Sybase calls it IBM US, Postgres calls it IBM, Bloomberg calls it IBM.N.
* **Risk:** The Agent fails to join market data to trades, or reports split inventory.
* **Solution:** Build a dim\_instrument\_master table immediately.
  + Create an internal surrogate\_key (e.g., UUID) for every instrument.
  + Map all incoming source IDs to this key in the **Silver** layer.
  + *Never* let raw ticker strings propagate to the Gold layer.

### 2. Restatements (The "Changed Past")

* **Issue:** A trade from last Tuesday is "busted" (cancelled) and re-booked today with a new price, effective last Tuesday.
* **Risk:** If you use a simple "Daily Snapshot," you might miss the correction to history.
* **Solution:** **Bi-Temporal Modeling** (Optional but recommended).
  + valid\_time: When the trade happened (Business Time).
  + transaction\_time: When we learned about it (System Time).
  + *Simplified T-1 Approach:* In the nightly batch, reload the *entire* active window (e.g., past 30 days) of trades to catch back-dated corrections, using MERGE logic.

### 3. The "Zombie" Dimension Rows

* **Issue:** A new Trade arrives referencing Client\_ID: 999, but the Client master table hasn't loaded Client 999 yet.
* **Risk:** The INNER JOIN fails, and the trade disappears from reports.
* **Solution:** **Late Arriving Dimensions**.
  + In the Silver transformation, if a trade references a missing client, insert a "Stub" record into dim\_client (ID: 999, Name: "Unknown").
  + Update the Stub with real details when the Client batch runs later.

### 4. Numerical Precision (The "Floating Point" Trap)

* **Issue:** Using FLOAT for financial calculations.
* **Risk:** 0.1 + 0.2 resulting in 0.30000000000000004.
* **Solution:** Always use NUMBER(p, s) in Snowflake.
  + Prices: NUMBER(18, 8) (to handle FX/Crypto).
  + Amounts: NUMBER(38, 2) (for large notionals).

## Part 3: Proposed Gold Schema (Starter)

### Table: fact\_trades\_unified

*Goal: A single table for all trades, regardless of source.*

| **Column Name** | **Data Type** | **Note** |
| --- | --- | --- |
| trade\_sk | VARCHAR | Surrogate Key (Hash of Source + Source\_ID) |
| trade\_date | DATE | The business date of the trade |
| instrument\_sk | VARCHAR | FK to dim\_instrument\_master |
| client\_sk | VARCHAR | FK to dim\_client |
| side | VARCHAR | 'BUY' or 'SELL' (Normalized) |
| quantity | NUMBER(18,6) |  |
| price | NUMBER(18,8) |  |
| exec\_venue | VARCHAR |  |
| source\_system | VARCHAR | 'SYBASE' or 'POSTGRES' |
| load\_timestamp | TIMESTAMP\_NTZ | Audit trail |

### Table: fact\_risk\_daily\_snapshot

*Goal: Daily immutable record of risk.*

| **Column Name** | **Data Type** | **Note** |
| --- | --- | --- |
| snapshot\_date | DATE | The "As Of" date (Partition Key) |
| client\_sk | VARCHAR | FK to dim\_client |
| instrument\_sk | VARCHAR | FK to dim\_instrument\_master |
| net\_exposure | NUMBER(38,2) | Pre-calculated USD exposure |
| var\_95 | NUMBER(18,4) | Value at Risk |
| delta | NUMBER(18,4) |  |