## **Unraveling Views in a Trino-Nessie Lakehouse: A Research Journey (Simplified & Sourced)**

Building a modern data pipeline with dbt (Data Build Tool) on a Trino-Nessie-Iceberg lakehouse is powerful. However, we hit a roadblock: an error saying createView is not supported for Iceberg Nessie catalogs. This essay explains our research to understand this error and find the best solution.

**1. The Initial Problem: A "View" Error**

* Our investigation started when dbt tried to create a staging model as a view.
* Trino (our query engine) reported that its Nessie-configured Iceberg catalog couldn't do this.
* Was it our setup, or a deeper issue?

**2. What Trino's Developers Said (Source: Trino GitHub Issue #17768)**

* This official Trino issue was a key piece of evidence.
* **Key Finding:** Trino's special connector for Nessie (when you set iceberg.catalog.type=nessie) is designed to work with version-controlled Apache Iceberg *TABLES*.
* **The Limitation:** This connector mode **does not support creating standard SQL VIEW objects.**
* **What about "Iceberg Views"?**
  + The Iceberg specification *does* define a modern, versionable "Iceberg View."
  + However, the Trino developers confirmed in this issue that Trino **has not yet implemented the feature** to create or use these advanced Iceberg Views with the native Nessie connector.

**3. What Nessie's Developers Said (Source: Nessie GitHub Issue #8574)**

* Looking at the Nessie project itself, this issue showed the problem wasn't just with Trino.
* **Key Finding:** The Nessie catalog service, which provides Git-like versioning for tables, **does not yet have a built-in way to store or manage view definitions** (neither standard SQL views nor the new Iceberg Views).
* **Important Insight:** For views to work perfectly here, both Trino (the engine) and Nessie (the catalog) would need to add this feature.

**4. Understanding the "Iceberg REST Catalog" (Source: General Iceberg Documentation & Community Discussions)**

* We learned that the "Iceberg REST Catalog" is not a specific product, but an **open standard**—a common language—for how catalogs should talk using web APIs.
* **Nessie's Role:** Project Nessie is a powerful catalog that *implements* this REST standard and then adds its special "Git-for-Data" features.
* **A Potential Workaround?** Some advice suggested setting Trino's connector to the more generic iceberg.catalog.type=rest mode when talking to Nessie.
* **Our Conclusion:** While a valid connection method, this **wouldn't magically fix the view problem**. The Nessie server itself (the backend) still can't store view definitions, no matter how Trino is configured to talk to it.

**5. Finding the Solution in dbt (Source: Official dbt Documentation)**

* Since creating views in our lakehouse was a current limitation, we looked for a solution within dbt itself.
* **The ephemeral Materialization:** This was the perfect answer for our staging models.
  + An ephemeral model is **not actually created** in the data warehouse as a table or view.
  + Instead, dbt takes its SQL code and directly inserts it as a subquery (or Common Table Expression - CTE) into any models that use it.
  + This **completely avoids the CREATE VIEW command** and the error.
* **The views\_enabled: false Flag:** The dbt documentation also showed a global project setting:
  + Adding flags: views\_enabled: false to dbt\_project.yml tells dbt to **never try to create views for any reason**, even for its internal operations. This is a great safety net.

**6. What About Materialized Views?**

* The same core issue applies: The Trino-Nessie connector is focused on Iceberg tables and **does not support CREATE MATERIALIZED VIEW**.
* **dbt's Alternative:** A dbt model set to materialized: 'table', when run regularly (e.g., every hour or day using dbt run), achieves the same goal. It rebuilds a physical table with fresh data, acting like a materialized view but managed by dbt.

**7. Our Final Understanding and Solution**

* **The Root Cause:** The createView is not supported error is a **known feature limitation** of the current Trino-Nessie integration, confirmed by official GitHub issues from both projects. It's not a mistake in our setup.
* **The Best Solution with dbt:**
  + Use the ephemeral materialization for staging models.
  + Globally set flags: views\_enabled: false in dbt\_project.yml.
* This research shows how important it is to understand the specific features of each tool when building with the powerful, but always evolving, modern data stack.