**Databricks SQL Issues with Mitigations**

**Databricks SQL Issues Comparison Table with Mitigations**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | SQL Issue | One-Liner Description | Where It Typically Arises | SQL Areas Most Affected | Mitigation Strategies |
| 1 | Query Performance Bottlenecks | Poorly optimized queries lead to slow execution and high compute costs. | Dashboards, ad hoc analytics | SQL Warehouses, Dashboards, Notebooks | Use EXPLAIN to analyze plans; optimize with Delta caching, partition pruning, and Z-ordering. |
| 2 | Data Freshness Delays | Queries return stale data when underlying tables aren’t refreshed as expected. | Materialized views, scheduled queries | Dashboards, Reports | Automate refresh schedules; validate pipeline SLAs; monitor update frequencies. |
| 3 | Permissions Errors | Users encounter access denied errors due to missing grants or incorrect workspace permissions. | Query execution, dashboard sharing | SQL Warehouses, Tables | Grant privileges explicitly; test access from user roles; use Unity Catalog for centralized governance. |
| 4 | Inconsistent Naming Conventions | Tables, views, and databases use inconsistent names, creating confusion across teams. | Schema design, collaborative development | Databases, Views | Establish and document naming conventions for all data assets. |
| 5 | Cost Sprawl from Warehouses | Idle SQL warehouses continue running, driving up unnecessary costs. | Ad hoc analytics, unattended sessions | SQL Warehouses | Set auto-stop timeouts; monitor utilization; enforce warehouse usage policies. |
| 6 | Ambiguous Column References | Joins and queries fail due to duplicate column names without explicit qualifiers. | Complex queries, views | SQL Notebooks, Views | Use fully qualified column names; alias tables clearly in joins. |
| 7 | Schema Drift in Views | Underlying table schema changes break dependent views. | Table evolution, view maintenance | Views, Dashboards | Validate schema compatibility; use versioned tables and clear change processes. |
| 8 | Inefficient Data Types | Suboptimal column data types increase storage and reduce performance. | Table creation, schema migrations | Tables, Query Performance | Optimize types (e.g., INT vs. BIGINT); review schemas before deployment. |
| 9 | Lack of Query Auditing | No visibility into who ran which queries and when, reducing accountability. | Production workloads, compliance environments | SQL Warehouses, Tables | Enable query history logging; review logs periodically for suspicious activity. |
| 10 | Limited Alerting and Monitoring | Dashboards fail or queries degrade without timely notification to users or admins. | Production dashboards, scheduled queries | Dashboards, Jobs | Set alerts on query failures; integrate monitoring tools for proactive notifications. |

**Quick Reference**

* **SQL Warehouse:** Compute resource for running SQL queries.
* **Dashboards:** Visual reports built on top of queries.
* **Views:** Virtual tables referencing other data.
* **Unity Catalog:** Centralized governance for data access.
* **Materialized Views:** Precomputed tables to improve performance.

**Example Mitigation Commands and Configurations**

**Analyze and Optimize Query Plans:**

sql

CopyEdit

EXPLAIN SELECT user\_id, COUNT(\*) FROM sales GROUP BY user\_id;

**Grant Access to Tables:**

sql

CopyEdit

GRANT SELECT ON TABLE main.sales.transactions TO `analyst\_group`;

**Set Auto-Stop for SQL Warehouse:**

* In the SQL Warehouse settings UI, set auto-stop timeout (e.g., 10 minutes idle).

**Alias Columns to Avoid Ambiguity:**

sql

CopyEdit

SELECT a.id AS order\_id, b.id AS customer\_id

FROM orders a

JOIN customers b

ON a.customer\_id = b.id;

**Enable Query History:**

* Accessible in the **SQL Editor** under *Query History*.

**Create Materialized View:**

sql

CopyEdit

CREATE MATERIALIZED VIEW sales\_summary AS

SELECT region, SUM(amount) AS total\_sales

FROM sales

GROUP BY region;

**Schedule View Refresh:**

* In the UI: Schedule a **refresh interval** for materialized views.
* Or via SQL:

sql

CopyEdit

ALTER MATERIALIZED VIEW sales\_summary SET SCHEDULE '5 MINUTES';