

Teradata spool errors

A practical diagnosis and fix guide

A failed query does not automatically need a larger spool allowance. First determine whether it processes too much data, concentrates work on a few AMPs, or competes with other requests for the same user allowance. The remedy depends on that distinction.

This guide helps Teradata developers and database administrators collect useful evidence, identify the likely cause, and test a correction. It combines the spool and skew explanations in Teradata Query Performance Tuning with a reproducible DWHPro lab.

Start with the failing request

Record the exact error, user, execution time, SQL and parameter values. Keep the execution plan and the relevant table definitions. If the request previously succeeded, note what changed: data volume, value distribution, SQL, statistics, concurrency or system configuration.

A low account-level spool reading after failure does not prove that spool was available when the request failed. Intermediate spool may already have been released.

How to use this guide

Pages 2 and 3 explain the mechanism and the first checks. Pages 4 and 5 help interpret skew and execution plans. Page 6 presents the controlled experiment. Pages 7 and 8 provide a validation checklist and an assessment brief.

Adapted from Teradata Query Performance Tuning by Roland Wenzlowsky and Artemios Vogiatzis, copyright 2020. DWHPro companion guide, 24 October 2026. The controlled experiment used Teradata Database 20.00.28.81; broader advice must be assessed against your release and configuration.

Understand what uses spool

Teradata uses spool to hold intermediate results and query output. Volatile tables also consume spool. A request may therefore compete with other requests and retained volatile data under the same user. A shared batch account is particularly important to investigate.

Why free capacity can be misleading

Parallel execution distributes work across AMPs. Under a simple evenly divided spool limit, one AMP can exhaust its allowance while other AMPs still have headroom. Aggregate free space is therefore not sufficient evidence that a request should succeed.

For example, a 16,000,000-byte limit across four AMPs gives a simple nominal allowance of 4,000,000 bytes per AMP. This describes the isolated lab in this guide. Production limits can be affected by profiles, space-management settings and skew allowances; confirm the effective settings with your DBA.

Separate three possible causes

Excess volume: too many rows or unnecessarily wide rows survive into intermediate steps. Examine filtering, projection and join cardinality.

Uneven distribution: a redistribution key sends a disproportionate number of rows to one AMP. Examine frequent values, NULLs and the distribution after filtering.

Shared demand: concurrent queries or retained volatile tables consume the same allowance. Correlate the failure with other sessions rather than investigating the SQL alone.

A larger allowance may be appropriate for a legitimate workload. It does not remove skew, correct an accidental many-to-many join, or guarantee enough physical capacity.

Preserve the exact error

The controlled example in this guide reproduces error 2646. Do not classify every spool-related failure from a generic description alone. Retain the complete error text and ask the DBA to distinguish a user limit from a system capacity or accounting problem.

Collect the first evidence

Start with the account and the incident time. The following per-AMP snapshot query was executed in the controlled lab. It requires access to DBC.DiskSpaceV; visibility and permissions can differ in your environment.

```
SELECT Vproc, CurrentSpool, MaxSpool
FROM DBC.DiskSpaceV
WHERE DatabaseName = USER
ORDER BY Vproc;
```

CurrentSpool is a point-in-time account reading. MaxSpool is a configured limit field, not the peak spool reached by this request. Compare AMPs and correlate readings with the workload; do not interpret these columns as query-level measurements.

Check the configured database value

```
SELECT SpoolSpace
FROM DBC.DatabasesV
WHERE DatabaseName = USER;
```

This query is adapted from the book and was validated on Teradata 20.00.28.81. It reads the value for the current database user. Ask the DBA whether a profile or other settings affect the effective allowance. A dictionary value alone does not describe every enforcement rule.

Capture the incident together

Keep the query or session identifier, timestamp and time zone, error text, SQL with representative parameters, EXPLAIN, table definitions and statistics information. Note concurrent requests under the same user and volatile tables still needed by the session.

Where existing DBQL and monitoring permit, collect request and step evidence around the failure. Missing logging is an evidence gap, not proof of a healthy distribution. Agree any additional logging with the DBA before enabling it.

Do not rerun an expensive failing production query solely to obtain a screenshot. Use existing monitoring or an approved, bounded reproduction.

Distinguish stored skew from execution skew

Static skew describes uneven storage of a table across AMPs. Execution skew arises while a request filters, redistributes, joins or aggregates rows. A well-distributed base table can still produce a heavily skewed intermediate result.

Inspect the key used by the suspect step

Read the execution plan to identify the redistribution or grouping key. Then examine that key in the rows that actually reach the step. A distribution check over the entire base table may hide a concentration created by a filter.

```
-- Template: replace table, key and filter.  
SELECT TOP 10 join_key, COUNT(*) AS row_count  
FROM your_database.your_table  
WHERE <same filter as the suspect input>  
GROUP BY join_key  
ORDER BY row_count DESC;
```

Replace every placeholder. This query pattern was tested on a synthetic volatile table, including NULLs and an excluded row. For a multi-column key, group by the complete combination. Review the cost before running this aggregation over a large table; it can itself require substantial resources. Frequent values suggest where to investigate but do not prove the failing step or AMP.

Investigate the shape of the data

Look for default values, missing identifiers, a dominant customer or category, and NULL concentrations. Check whether both sides of a join contain multiple rows for the same key. If one key has m rows on one side and n matching rows on the other, an equality join can produce m times n rows for that key.

Some large joins are correct and necessary. Others indicate an incomplete join condition or an incorrect assumption about uniqueness. Establish which case applies before reducing rows.

Do not remove NULLs, add DISTINCT, or pre-aggregate solely to make a query finish. Each change can alter the required result. Validate the business meaning as well as resource usage.

Read the plan and choose one intervention

EXPLAIN describes the optimizer plan and its estimates. It does not measure actual peak spool. Read it alongside available execution evidence, especially where estimated and observed row counts diverge.

Follow rows through the plan

Identify where rows are filtered, redistributed, duplicated, sorted and joined. Check the columns carried into large intermediate results. Follow the spool identifiers so you can connect a large estimated result to the operation that creates and consumes it.

A redistribution on a frequent key can concentrate work. Duplication can increase the volume stored across participating AMPs. A sort may add pressure while other intermediate results remain live. None of these operations is automatically a defect; assess their scale and purpose.

Match the change to the evidence

If unnecessary rows survive early steps, test a semantically equivalent filter closer to the input. Be careful with outer joins: moving a predicate can change which unmatched rows are retained.

If intermediate rows are unnecessarily wide, project only required columns. If cardinality estimates are poor, inspect statistics on relevant filters and joins and agree targeted collection rather than collecting everything.

If a staging or volatile table concentrates rows, test an appropriate alternative primary index in isolation. Evaluate downstream access and joins as well as the immediate insert. A more even distribution is not the only design objective.

If concurrency is the main contributor, test scheduling or workload changes. Release a volatile table when its session no longer needs it. Do not change transaction semantics merely to clear rows.

Change one factor where possible. Save the original plan and result checks, then compare the new plan, correctness and resource measurements under comparable conditions.

A controlled spool and skew experiment

The DWHPro lab used Teradata Database 20.00.28.81, BTEQ 20.00.00.11, BTET mode and four AMPs. A dedicated, otherwise idle user had a 16,000,000-byte spool allowance with no profile override. No customer data was used.

The source contained 10,000 synthetic rows: integer IDs from 0 to 9,999, a hot_key column set to 1 for every row, and a CHAR(200) payload. Two empty volatile targets had the same columns but different primary indexes.

Target primary index	Observed result across three attempts
hot_key	Error 2646 on all three attempts
id	All three inserts succeeded with 10,000 rows

The spool allowance stayed unchanged. Changing the target primary index changed both the distribution and the execution plan. This demonstrates a successful correction in this setup; it does not isolate the contribution of each optimizer effect.

What the plans showed

The skewed path included redistribution and sorting, with an estimated redistribution spool of 2,210,000 bytes. The balanced path used a direct all-AMP merge without that redistribution and sort step. The estimate is not a measured peak.

What the timings do not show

Both failed and successful requests reported one second per attempt at BTEQ whole-second resolution in the confirmation run. These measurements support a failure-versus-success comparison, not a speedup percentage. Post-insert account snapshots include volatile tables and measurement overhead; they are not per-request peaks.

[Read the worked example and download the SQL and results](#)

The downloadable script deliberately reproduces failures. Run it only in a DBA-provisioned isolated test account. Results can differ with release, topology, allocation state and space settings.

Validate the correction

A query that finishes is only the first check. The corrected request must preserve the intended result and behave acceptably in the workload where it will run.

Check correctness

Compare row counts and business totals, then check key-level differences, duplicates and NULL handling. Matching counts alone do not prove identical results. Where feasible, compare the full result as a multiset so duplicate multiplicity is preserved. Investigate every difference.

Check resource use

Use the same data, parameters, user settings and comparable workload conditions for the baseline and candidate. Record successful completion, elapsed time, available CPU and I/O metrics, and relevant per-AMP or step evidence. Label any missing measurement explicitly.

Repeat the test enough to distinguish a consistent outcome from an isolated observation. Account for caching and competing workload. If the baseline always fails, report the change from failure to completion; do not invent a runtime improvement for a request that never completed.

Check the production context

Include representative busy periods and data distributions. A change that helps one query may increase the cost of another. For a primary-index change, assess loading, access paths, joins and downstream processing before deployment.

Keep a short change record

Record the original symptom, evidence, proposed cause, exact change, correctness checks, test environment, observed results and rollback approach. Distinguish a measured result from an estimate or hypothesis.

Avoid system-wide settings changes as an unmeasured first response. In particular, disabling protection, dropping indexes or using administrative space-repair procedures requires a separate DBA assessment of the actual problem.

Get help with a recurring failure

If the problem returns, several requests compete for spool, or a correction creates new performance problems, a workload review can help establish what to change first.

Discuss a Teradata performance assessment

Tell DWHPro which workload is affected, when the failure occurs, and what it prevents your team from delivering. We can discuss an assessment of an agreed set of queries and the available execution evidence, with priorities and next steps defined around your environment.

For the initial conversation, prepare a short description of the problem, the Teradata release, whether the issue is new or recurring, and the business impact. Keep credentials and confidential query contents out of the initial enquiry. Agree a suitable private channel before sharing detailed evidence.

[Contact DWHPro about a performance assessment](#)

Continue learning

[Teradata Query Performance Tuning by Roland Wenzlofsky and Artemios Vogiatzis](#)

This companion adapts the book sections on data demographics and static skew, spool space for execution steps, optimizer statistics, execution plans and spool-space considerations. It adds the October 2026 controlled lab and a practical incident and validation checklist.

Evidence and scope

The lab archive contains benchmark.sql, chronological BTEQ output and a README describing prerequisites and limitations. The per-AMP snapshot query appears in that executed script. The allowance query and frequency-query pattern were also validated on Teradata 20.00.28.81. The frequency check returned counts of 3, 2 and 1 for two keys and NULL, correctly excluding a row outside the filter. Templates still require adaptation to your objects and workload.

The guide focuses on the AMP-based Teradata SQL environment illustrated by the source book and lab. It is not a claim that every storage model, deployment or release enforces spool in exactly the same way. Use current platform documentation and your DBA configuration when applying the guidance.