Table of Contents

COOL Performance Tests

A performance test suite has been developed over time to test and validate the performance of COOL queries against Oracle database servers.

This test suite can be useful for instance to exclude performance regressions when moving to a more recent version of the COOL client software or of the Oracle server software (e.g. for the port to Oracle 11g servers described in task #23366 and more recently for the move to Oracle 12c servers as described in task #44885).

The most recent version of the test suite (built around the script createReport.sh) systematically tests the most important use cases for retrieving conditions data using COOL. The use cases that were considered in the 10g to 11g migration are the following:

- **SV_R**. Single version retrieval (with inline payload).
- **SP_R**. Single version retrieval (with a separate payload table).
- **MVUR**. Multi version retrieval from a user tag (with inline payload).
- **MPUR**. Multi version retrieval from a user tag (with a separate payload table).
- **MVHR**. Multi version retrieval from the HEAD (with inline payload).
- **MPHR**. Multi version retrieval from the HEAD (with a separate payload table).
- **MVTR**. Multi version retrieval from a normal tag (with inline payload).
- **MPTR**. Multi version retrieval from a normal tag (with a separate payload table).
- **SC_R**. Single version retrieval (with inline CLOB payload).

As described in the proceedings of the NSS 2008 conference in Dresden, the test suite aims at validating two specific aspects of COOL query performance:

- **Scalability**. Rather than testing query times in absolute terms, the test suite focuses on estimating the scalability of COOL query performance as the data volumes increase. The basic idea is that the retrieval of IOVs valid in a given time range should be optimized so that query response time is not only as low as possible, but it also remains essentially constant when retrieving IOVs from very old or very recent time ranges. In each test, query times are measured for different values of the validity time T around which IOVs are retrieved; these query times are then plotted as a function of T.
- **Stability of query execution plans**. Having identified *statistics* reliability and *bind variable peeking* as the two main factors for SQL query stability, their effect is now systematically studied in all COOL performance tests. For each test, query times are measured and plotted as a function of T in six different configurations: for two scenarios of bind variable peeking (low or high values of T) and for three scenarios for statistics (reliable statistics, unreliable statistics computed on empty tables, no statistics). Generally speaking, good performance can be expected 'out-of-the-box' (without adding optimizer hints) only for the scenario with reliable statistics and high values of peeked bind variables, while the other 5 cases can all lead to non-scalable execution plans. The test suite produces two such sets of six curves, both without hints (out of the box from Oracle) and with hints (the COOL mechanism for stabilizing the execution plans). If all is ok, the six curves with hints should all be flat.

The test suite is also designed to ensure the *reproducibility* of test results. When creating the full performance report, test tables are created and populated from scratch before query times are measured. When debugging a specific problem observed in the test suite, it is however more efficient to create the test data only once and then concentrate only on testing query times. A few more issues have been observed in the past to lead to confusing results and should be considered when interpreting surprising results (some of these are now taken care of inside the test suite and/or inside the COOL code).

- **Oracle cache and I/O**. The script by default launches a dry run of the test (in the 6 scenarios without hints, which need to scan more data) to fill all relevant data caches before executing the 12 reference tests (6 with hints and 6 without hints). This should remove all I/O effects and ensure a better

reproducibility of the measured CPU performance in the tests.

- **Load on the test server and spurious processes**. You should possibly execute the tests on a server with negligible load from other database users. If the plots you obtain still show some amount of randomness, check the CPU load on the server; in the past, one such observation was caused by an Oracle Enterprise Manager process running wild, which had to be killed and restarted.
- **Overhead from creating the 10053 trace file**. In each of the 12 scenarios, the test forces Oracle to execute a hard parse (to recompute the appropriate execution plan) and produce a 10053 trace file. This is achieved by executing a dummy DDL statement on the relevant tables. This hard parse is time consuming and causes the test client to observe a very long retrieval time. To be independent of this overhead, this first test is discarded from the plots.
- **Automatic computation of statistics**. When executing the script for the first time to create the test databases, the statistics of all relevant tables are now locked by the test script. This should ensure that the reference tables with no statistics or bad statistics permanently remain in that state, even if statistics are automatically collected during the night every 24h.
- **Oracle 11g SQL plan baselines**. The Oracle 11g "SQL plan baseline" feature is disabled in CORAL, and a fortiori in COOL, by default (by issuing the relevant "alter session" statement at logon in CORAL), because it was found to lead to unstable and unexpected execution plans during the tests. *(Should this be done also at the COOL level using an OPT_PARAM hint?)*
- **Oracle 11g adaptive cursor sharing**. The Oracle 11g "adaptive cursor sharing" feature is disabled in COOL for the main IOV queries (by adding a NO_BIND_AWARE hint), because it was found to lead to unstable and unexpected execution plans during the tests.
- **Oracle 11g cardinality feedback**. The Oracle 11g "cardinality feedback" feature was briefly tested in May 2012 (well after the validation of COOL performance on 11g servers) because it was suspected to be the cause of 11g server memory fragmentation leading to ORA-04031 errors in the COOL and CORAL nightlies (see bug #94270). However, it was eventually decided to keep this feature untouched (i.e. enabled by default in 11g) in CORAL and COOL.
- **Oracle 11g and 12c dynamic sampling**. The Oracle 11g "dynamic sampling" feature has been disabled by default in the CORAL and COOL functional test suite using qmtest in May 2012 (well after the validation of COOL performance on 11g servers) because it was found to be the cause of 11g server memory fragmentation, through extra queries with OPT_DYN_SAMP hints, leading to ORA-04031 errors in the COOL and CORAL nightlies (see bug #94270). However this feature is only disabled for qmtest and is kept enabled by default for all other use cases, including the performance tests. In 12c, in particular, some tests on 12c in August 2013 (described below) indicated that this feature does help choosing better execution plans when table statistics exist but are not up-to-date.
- **Oracle 12c adaptive optimization**. The Oracle 11g "adaptive optimization" feature has been disabled by default in COOL (using OPT_PARAM query hints) in August 2013 during the COOL query performance validation on Oracle 12c (bug #102272). This was causing extra queries with OPT_ESTIMATE hints, as well as extra STATISTICS COLLECTOR entries in plan tables, to appear in the trace files of the COOL performance test suite. It was also observed to lead to worse performance and less reproducible results in the absence of COOL hints.
- **"Good" execution plans may differ in their access to the channels table**. No INDEX hints are included in COOL for the access to the channels table, and it is not even clear a priori that index scans shuld be better than full table scans (bug #103740). The existence of different execution plans with hints, which used to be marked as red, is now marked as orange to indicate that these cases require visual inspection of the plots.

Performance validation for the Oracle 10g to 11g migration

The test script has been significantly improved to produce an automated performance report. This shows the numerical results of the tests on a plot, as well as selected information extracted from the 10053 trace file. The following are the plots produced in December 2011 during the comparison of COOL performance on Oracle 10g and Oracle 11g (see task #23366).

For each use case, there are essentially only three points that need to be checked:

- **The 6 plots with hints should display a constant retrieval time**. This ensures scalability of COOL performance as the tables get bigger. This must be visually inspected (there is no automatic fit for the data in the plots).
- **The 6 trace files with hints should all use the same execution plan**. This is displayed in the summary table for each use case. The relevant information is surrounded by a green box if all is ok. Note that this is correlated with the observation of the plots: if the execution plans are different, you may expect that one of the plot will disaply a different pattern, most likely with an increasing query time (non-scalable behaviour).
- **There should be no unused hints in the 6 trace files with hints**. If this should happen (quite unlikely), it would be an indication that the COOL hints need to be changed because they are no longer appropriate.

The following are the reports for the SV_R use case only, comparing Oracle 10g (10.2.0.5) with two different versions of Oracle 11g (11.2.0.2 and 11.2.0.3). The report for 11.2.0.2 clearly shows some problems, which are due to Oracle bug 10405897, as described in task #23366.

- SV_R-10.2.0.5-full.pdf: Performance report for the SV_R use case, against Oracle 10.2.0.5. Full version (all execution plans).
- SV_R-11.2.0.2-full.pdf: Performance report for the SV_R use case, against Oracle 11.2.0.2. Full version (all execution plans).
- SV_R-11.2.0.3-full.pdf: Performance report for the SV_R use case, against Oracle 11.2.0.3 Full version (all execution plans).

The following are the reports for all 9 use cases, comparing the reference versions of Oracle 10g (10.2.0.5) and 11g (11.2.0.3). Two versions of each report are attached: a short one showing only the best execution plan (that used by the 6 scenarions with hints) for each use case, and the full one showing all execution plans for each use case (i.e. all the different bad execution plans which would be observed if there were no hints in COOL).

- ALL-10.2.0.5.pdf: Performance report for all 9 use cases, against Oracle 10.2.0.5. Short version (only the best execution plan for each use case).
- ALL-10.2.0.5-full.pdf: Performance report for all 9 use cases, against Oracle 10.2.0.5. Full version (all execution plans for each use case).
- ALL-11.2.0.3.pdf: Performance report for all 9 use cases, against Oracle 11.2.0.3 Short version (only the best execution plan for each use case).
- ALL-11.2.0.3-full.pdf: Performance report for all 9 use cases, against Oracle 11.2.0.3. Full version (all execution plans for each use case).

Note that, **for each use case, the best execution plan is different in 10g and 11g**, even if both Oracle versions lead to the same good performance in the plots. I actually believe that the core of the algorithm is the same in 10g and 11g (based on the **FIRST ROW** and **INDEX RANGE SCAN (MIN/MAX)** steps within the correlated subquery), whereas what differs in 11g from 10g are only minor details of the xecution plans, or even only the way the same plan is described in the plan table. I actually find the execution plan description in 11g more understandable than the one in 10g: in my opinion only *one* MIN/MAX scan is needed in the query, as described in the 11g plan, because there is only one MAX clause (in the correlated subquery), whereas the plan table description in 10g mentions two separate MIN/MAX scans.

Performance validation for the Oracle 11g to 12c migration

After several few modifications to the performance test toolkit and to the CORAL/COOL code, the performance report has also been produced for Oracle 12c servers. Some of the most important changes to the toolkit and code are the following (see task #44885 for some additional details),

- Note that the very first performance tests of COOL on Oracle 12c were affected by a bug (lcgcool:18504), with the effect that full table statistics were available in the 'emst" plots, while no statistics was available in the "stat" and "nost" plots (so that these two sets of plots were actually identical). All reports included in this twiki page have been re-created after fixing this bug.
- For the MVUR use case, without hints Oracle was initially suggesting a plan almost identical to that triggered by COOL hints, except that a simpler 3-D index was used instead of the hinted 5-D index. It was understood that this is due to the presence of too few user tags in the test schema, making the two plans essentially equivalent. This was fixed by improving the test schema, adding several more user tags (see task #4485 and lcgcool:18643).
- By default, the "adaptive optimization" new feature of Oracle 12c is disabled in COOL. As described in task #44885, this was eventually implemented using the OPT_PARAM hint to modify in COOL at the SQL query level two Oracle 12c parameters.
    - The OPT_PARAM('OPTIMIZER_ADAPTIVE_REPORTING_ONLY','TRUE') hint makes sure that, when an "adaptive" plan exists for a query, the "default" plan is used instead of any supposedly better plan re-optimized by Oracle: in practice, this ensures that the trace file produced for a given query in the COOL performance report only contain one "default" plan for it, while the additional plans with OPT_ESTIMATE hints that would otherwise appear are absent.
    - The OPT_PARAM('OPTIMIZER_ADAPTIVE_FEATURES','FALSE') hint, in addition, disables the computation of these additional re-optimizations: in practice, this gets rid of the additional "STATISTICS COLLECTOR" entries in the plan tables, which would otherwise be observed. Note that this second hint alone would not be enough: in particular, the additional OPT_ESTIMATE queries would still appear in the trace files if the first hint was removed and only this second one was included.
    - In the initial implementation of the port of CORAL and COOL to Oracle 12c, the 'OPTIMIZER_ADAPTIVE_REPORTING_ONLY' parameter had been modified by default at the CORAL level using an ALTER SESSION statement. In the latest software versions, CORAL does not modify any of these two parameters by default, but allows users to switch off adaptive optimization (by modifying *both* parameters) if the environment variable CORAL_ORA_DISABLE_ADAPTIVE_OPT is set.
    - Note also that setting these parameters via SQL hints in COOL, rather than relying on CORAL 'alter session' statements, also provides the advantage that these settings are automatically propagated for Frontier access (where no 'alter session' on Oracle are possible).
- The performance toolkit was in any case modified to be able to produce reports also with adaptive optimization enabled (allowing several 'MAIN' queries in the trace files and parsing only the relevant ones). Some of these reports are attached and listed below.
- The performance toolkit was modified to highlight in red any unexpected bind variables. In particular, such warnings are used when different bind values are used within each of the "peeklo" and "peekhi" sets of trace files, or when the same bind values are used by these two sets of trace files.
- The analysis and fix for the statistics bug, the addition of extra user tags in the test database and the addition of a hint to get rid of 'STATISTICS COLLECTOR' entries were initially motivated by the observation that the plans used with hints were in some cases different from the best plans identified by Oracle without hints, in the most favorable case of "peekhi" bind variables with up-to-date table statistics. All these issues have now disappeared, as noted below.

Results using COOL defaults (Oracle 12c adaptive optimization disabled and dynamic sampling enabled)

The following is the report for all 9 use cases against Oracle 12c (12.1.0.1), using the COOL default where adaptive optimization is disabled. All trace files and temporary files used to prepare this report are available on trac.

- ALL-12.1.0.1-full.pdf: Performance report for all 9 use cases, against Oracle 12.1.0.1. Full version (all execution plans).

The conclusions from this report and from its comparison to the corresponding report for Oracle 11g are the following:

- Performance for SV_R on 12c is always good when hints (those developed on 10g and 11g) are provided, independently of statistics and bind variables.
- When hints are not provided, performance is generally good only in the most favorable case "stat-peekhi" (with up-to-date statistics and peeking of high values of bind variables). In other words, Oracle 12c (with adaptive optimization disabled) finds a good execution plan in this case even if COOL hints are not provided. In all 9 use cases, this execution plan is exactly the same as that imposed by COOL hints (plan #1 is used in all 9 "stat-peeki-nohint" entries).
- For all 9 use cases, the best execution plan is almost identical in 11g and 12c. The only differences shown by the reports are that **TABLE ACCESS BY INDEX ROWID BATCHED** is sometimes used in 12c instead of the **TABLE ACCESS BY INDEX ROWID** used in 11g (in the SP_R, MPUR, MPHR, MVTR and MPTR use cases). This is also reflected in the additional **BATCH TABLE ACCESS BY ROWID** hint shown in the 12c outlines, which is missing in the corresponding 11g outlines.

Tests using Oracle 12c defaults (Oracle 12c adaptive optimization enabled and dynamic sampling enabled)

For comparison, the following is the report for all 9 use cases against Oracle 12c (12.1.0.1), using the Oracle 12c default where adaptive optimization (normally disabled in COOL) is enabled. All trace files and temporary files used to prepare this report are available on trac.

- ALL-12.1.0.1-withAdaptiveOptimization-full.pdf: Performance report for all 9 use cases, against Oracle 12.1.0.1, with adaptive optimization. Full version (all execution plans).

The conclusions from this report are the following:

- Performance for SV_R on 12c is still good when hints are provided, even if adaptive optimization is enabled, independently of statistics and bind variables.
- When COOL hints are not provided, Oracle adaptive optimization leads to execution plans that are often sub-optimal and seem untable and not clearly reproducible.
    - In particular, it seems that the bind variables appropriate to the query being performed are ignored and Oracle adaptive optimization determines an alternative plan using different bind variable values. In some cases (e.g. use case SC_R in this report), Oracle 12c adaptive optimization finds a bad execution plan in all 6 combinations of statistics and bind variable values, because it uses "peeklo" bind values all the time.
    - Moreover, the precise bind variable values used to determine this plan seem to vary from case to case (e.g. 2000 is used for SV_R instead of the "peeklo" 0 or the "peekhi" 9900), without a clearly understandable or reproducible pattern.
    - In summary, Oracle adaptive optimization for these queries seem to lead to bad performance and unstable/unreproducible results, which justifies the decision to disable it in COOL.

Tests of dynamic sampling on 12c (Oracle 12c adaptive optimization disabled and dynamic sampling disabled)

For comparison, the following is the report for all 9 use cases against Oracle 12c (12.1.0.1), with adaptive optimization disabled (as is the default in COOL) and dynamic sampling also disabled. This is achieved by setting the environment variable CORAL_ORA_DISABLE_OPT_DYN_SAMP (originally introduced as a workaround for ORA-04031 bug #94270). All trace files and temporary files used to prepare this report are available on trac.

- ALL-12.1.0.1-withoutDynamicSampling-full.pdf: Performance report for all 9 use cases, against Oracle 12.1.0.1, without dynamic sampling. Full version (all execution plans).

The conclusions from this report are the following:

- Performance for SV_R on 12c is still good when hints are provided, even if dynamic sampling is disabled, independently of statistics and bind variables.
- However, in 8 use cases out of 9, the "emst" tests with hints follow a slightly different execution plan with respect to the "stat" and "nost" tests. The difference is that access to the COOL channels table proceeds via a full table scan rather than via a full index scan. We could in principle add a hint forcing the use of the channels table index, but it is not obvious that this would be the best choice in general. Also, no problems concerning the access to the channels table have been reported in recent years. In summary, it seems more appropriate to keep things as they are and let the optimizer choose the most appropriate access method on the channels table. It is interesting that the only difference is seen for the "emst" tests, i.e. those where statistics are present but stale. When dynamic sampling is enabled, a different plan is chosen, which is probably an indication that this feature does help recovering more up-to-date statistics.

-- AndreaValassi - 24-Aug-2013

This topic: Persistency > CoolPerformanceTests
Topic revision: r22 - 2014-01-28 - AndreaValassi