

LCG Persistency Framework

CORAL, COOL, POOL – Status and Outlook

R.Trentadue (CERN IT-ES)

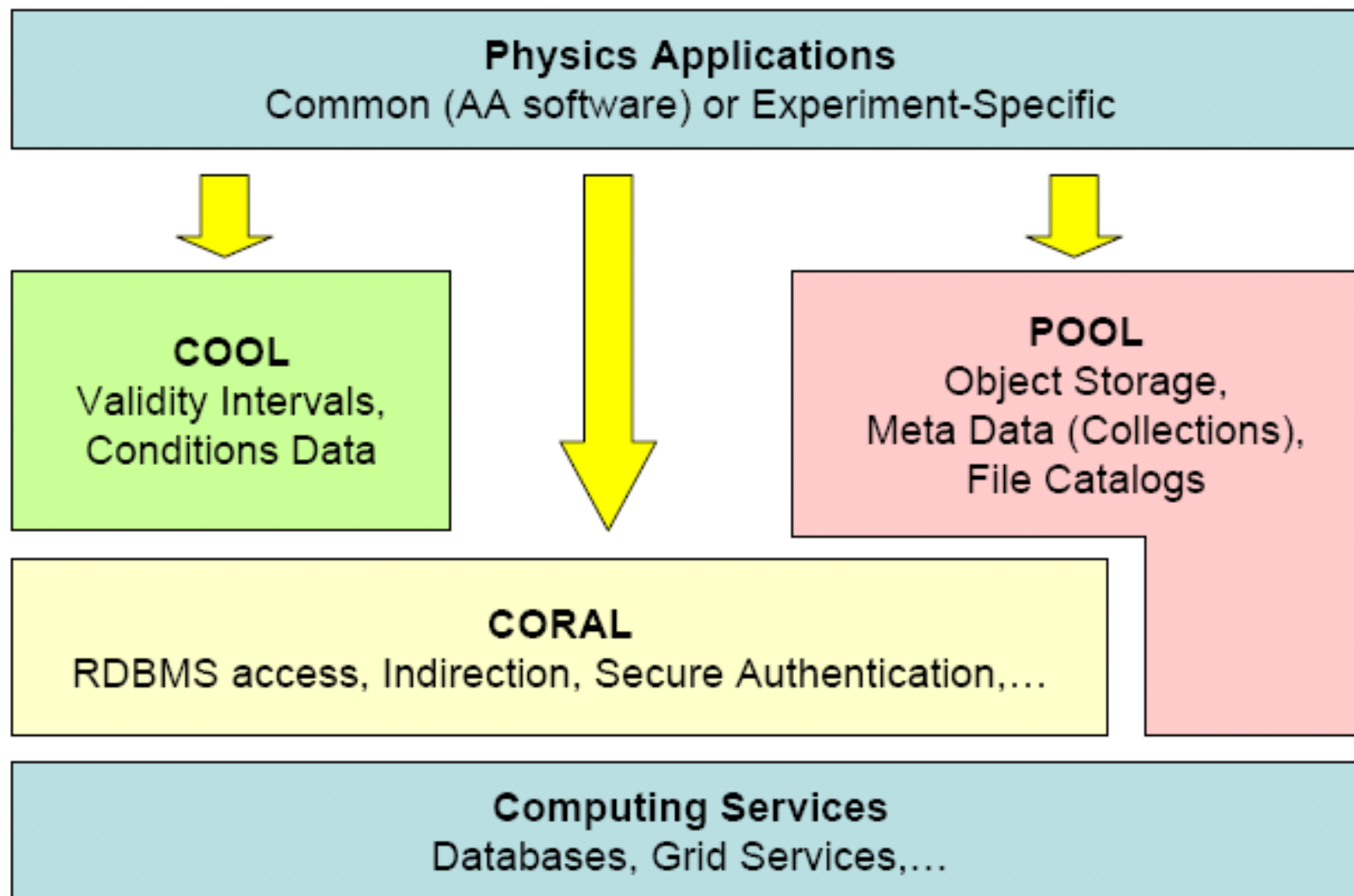
On behalf of the Persistency Framework team

CHEP2012 (New York), 22nd May 2012




- **Introduction**
 - Persistency Framework (PF) components
- **Usage in the experiments**
 - Evolution since CHEP 2010
- **Recent achievements**
 - CORAL handling of network instabilities
 - COOL performance validation on Oracle 11g
- **Outlook and conclusions**



- **CORAL**
 - Abstraction of access to relational databases
 - Support for Oracle, MySQL, SQLite, FroNtier
 - Used directly or indirectly via COOL/POOL
- **COOL**
 - Conditions data management
 - Conditions object metadata (interval of validity, version)
 - Conditions object data payload (user-defined attributes)
- **POOL**
 - Technologically-neutral hybrid data storage
 - Streaming of objects (e.g. to ROOT or relational DBs)
 - Object metadata catalogs (e.g. in XML or relational DBs)



PF usage in the experiments

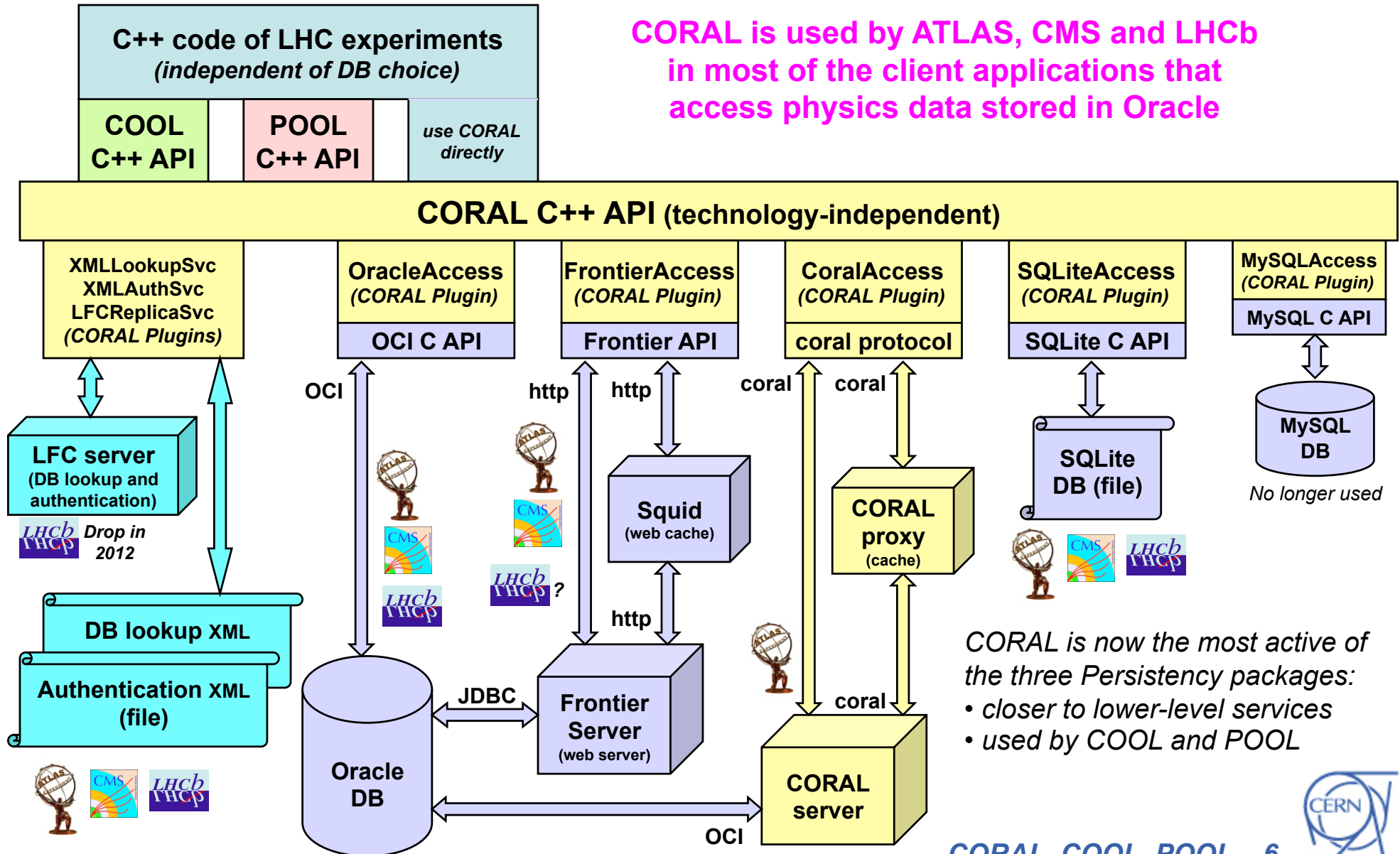
Persistency Framework in the LHC experiments	 ATLAS	 CMS	 LHCb
CORAL (Oracle, SQLite, XML authentication and lookup)	Conditions data (COOL) Geometry data (detector descr.) Trigger configuration data Event collections/tags (POOL)	Conditions data Geometry data (detector descr.) Trigger configuration data	Conditions data (COOL)
CORAL + Frontier (Frontier/Squid)	Conditions, Geometry, Trigger (R/O access in Grid, Tier0)	Conditions, Geometry, Trigger (R/O access in Grid, HLT, Tier0)	— <i>(will be tested in 2012)</i>
CORAL Server (CoralServer/CoralServerProxy)	Conditions, Geometry, Trigger (R/O access in HLT)	—	—
CORAL + LFC (LFC authentication and lookup)	—	—	Conditions data (authentication/lookup in Grid) <i>(will be dropped in 2012)</i>
COOL	Conditions data	—	Conditions data
POOL (ROOT storage service)	Event data Event collections/tags Conditions data (payload)	—	Event data <i>(dropped in 2011)</i>
POOL (Collections – ROOT and Relational)	Event collections/tags	—	—

- **(Being) adopted** – Frontier for ATLAS T0, geometry, trigger and in LHCb
 - ATLAS reco jobs at T0 read the ~same conditions from Squid cache, avoiding direct Oracle access
- **(Being) dropped** – POOL and LFCReplicaSvc in LHCb



CORAL usage in the experiments

CORAL is used by ATLAS, CMS and LHCb in most of the client applications that access physics data stored in Oracle



CORAL is now the most active of the three Persistency packages:

- closer to lower-level services
- used by COOL and POOL





- **Interact with AA projects and IT services**
 - SPI (external software, CMT config, nightly builds)
 - ROOT (object streaming in POOL, PyCool)
 - GRID middleware (for CORAL authentication)
 - CERN IT and site services (especially Oracle)
- **Release plan agreed with the experiments**
 - Monthly Architect Forum meetings
 - Including PF and all other LCG AA projects
 - Following monthly meetings with SPI and librarians
 - Approximately one release per month
 - Releases are built and now fully validated by SPI team

- **Software releases and maintenance**
 - 14 new releases (including external upgrades, e.g. ROOT)
 - Port to new platforms: gcc46 and clang on SLC6
 - Port to gcc47 and c++11 is ongoing
 - Windows platforms have been dropped
- **Service operation and user support**
- **Review of support model for the future (TEGs)**
 - POOL is being dropped
 - Experiments request continued CORAL and COOL support



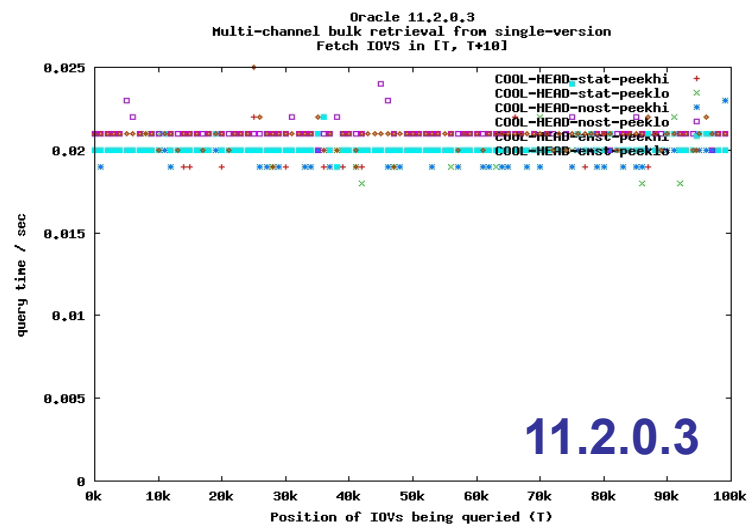
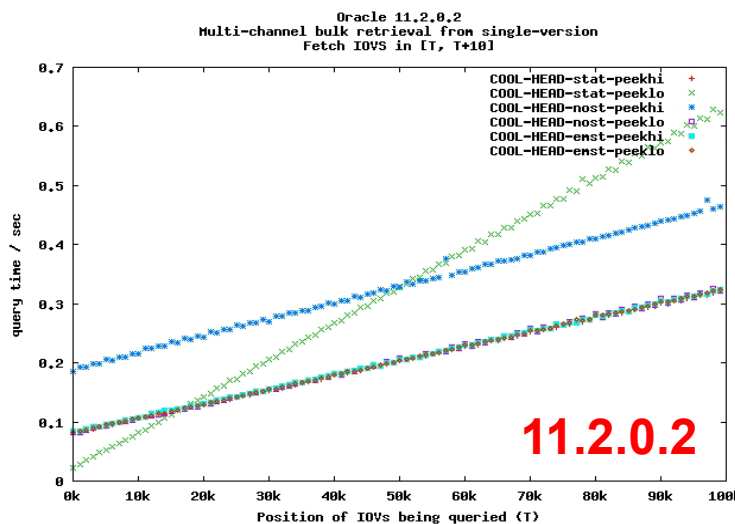
- **Oracle client software selection and installation**
 - New version of 11g client with configuration workaround for Oracle bug (redefined Kerberos symbols clash with O/S)
- **Follow-up of complex service incidents**
 - e.g. Kerberos KDC request flood from POOL tools (actually due to a combination of bugs in ROOT/xrootd and O/S)
- **User support**
 - help ATLAS to use Frontier for T0 (discrepancy with Oracle), and to study the spikes of high load observed on DB server
 - CMS support for issues related to Oracle error ORA-25408

- **Validation of releases**
 - Fully automated and outsourced to SPI (thanks!)
- **Various fixes and optimizations**
 - Reduce number of Oracle data dictionary queries
- **Test server maintenance**
 - fully “quattorized” CORAL server and MySQL server in CC

- **Crashes before 2011**
 - new session; new query; delete session; execute query CRASHES
 - Query keeps reference to SessionProperties that goes out of scope
 - Fix: replace references by SessionProperties shared pointers
 - more complex crashes reproduced and fixed in multi-threaded mode
- **Crashes before May 2012**
 - new session; new query; delete session; delete query CRASHES
 - Query owns OCIDescribe that keeps reference to deleted OCISvcCtx
 - This is really an Oracle bug (they should do their bookkeeping...)
 - Fix: defer deletion of OCISvcCtx to SessionProperties destructor (SessionProperties is last to be deleted thanks to previous patch)
- **Correlated to “network glitch” analysis**
 - some of the above crashes were triggered (and first observed) by old mechanism for reconnecting after a network glitch
 - broken old Session was deleted and replaced by a new Session
 - new mechanism keeps same Session instances and “refreshes” OCI
 - but the above crashes were also observed due to peculiar (but legitimate) user code, e.g. delete session before executing a query

- **Motivation?**
 - several reports of errors (e.g. ORA-24327, ORA-03113...)
- **First task was to solve hangs and crashes**
 - hangs accompanying ORA-24327 due to CORAL bug, fixed first
 - crashes were triggered by old reconnection mechanism
 - broken old Session was deleted and replaced by a new Session
 - new mechanism keeps same Session instances and “refreshes” OCI
 - i.e. CORAL reconnection implementation was heavily bugged!
 - CORAL was reconnecting, but was doing it in the wrong way
- **Another task was to understand interference of TAF**
 - triggered originally by CMS reports of ORA-25408 errors
 - after a lot of analysis, summary is that TAF does not help here
 - designed by Oracle for another use case (instance crash)
- **Final task was to tackle network glitch itself**
 - when should CORAL reconnect and when should it not?
 - analysis of transaction status (R/W, serializ. R/O, non serializ. R/O)
 - how to reconnect?
 - refresh OCI pointers rather than delete and recreate Session instance
 - logic has been completely moved to Oracle-specific plugin

- **Validation of COOL performance has been completed**
 - Motivation: CERN server upgrade from Oracle 10.2.0.5 to 11g
 - Completed in Q1 2012 – all servers are now running Oracle 11.2.0.3
 - Initially delayed by performance issues observed for COOL in Oct 2011
 - Problem seen in Oct 2011 is now understood to be caused by a **bug in Oracle 11.2.0.2 server** (Oracle bug 10405897)
 - Confirmed by enabling/disabling Oracle patch in a private 11.2.0.2 DB
 - Bug was introduced in 11.2.0.2 – it was not there in 11.2.0.1!
 - Early COOL 11g tests in 2010 had seen no issue as they used 11.1.0.7
 - *Bad news: even a minor server patch can break performance!*
 - Fixed in 11.2.0.3 – good news: we should no longer worry now
 - Performance on 11.2.0.3 is as good as on 10.2.0.5
 - Execution plans look a bit different, but algorithm is probably the same



- **Test script was improved and procedure was documented**
 - See <https://twiki.cern.ch/twiki/bin/view/Persistence/CoolPerformanceTests>
 - A detailed performance report can be created with one command
 - Covering 9 common use cases for querying COOL data
 - Showing queries, hints, performance plots and execution plans
 - e.g. <https://twiki.cern.ch/twiki/pub/Persistence/CoolPerformanceTests/ALL-11.2.0.3-full.pdf>

1. Use case SV_R (COOL-preview on Oracle 11.2.0.3.0)

Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit Production
Instance intR1 (Linux x86_64 on i386) on 2011-12-13

```

Primary SQL statement (hint options=""):
SELECT /*+ NO_BIND,AWARE_QR_NAME(MAIN) INDEX@MAIN COOL_J@MAIN (CHANNEL_ID)@SV_SINCE@IOV_UNTIL@
LEADING@MAIN COOL_C2@MAIN COOL_J@MAIN) USE_M@MAIN COOL_J@MAIN) INDEX@MAXI COOL_J@MAXI
(CHANNEL_ID)@SV_SINCE@IOV_UNTIL)@COOL_J3 OBJECT_ID AS "OBJECT_ID", COOL_J3.CHANNEL_ID AS "CHANNEL_ID",
COOL_J3.IOV_SINCE AS "IOV_SINCE", COOL_J3.IOV_UNTIL AS "IOV_UNTIL", COOL_J3.USER_TAG_ID AS "USER_TAG_ID",
COOL_J3.SYS_INSTM AS "SYS_INSTM", COOL_J3.LASTMOD_DATE AS "LASTMOD_DATE", COOL_J3.ORIGINAL_ID AS
"ORIGINAL_ID", COOL_J3.NEW_HEAD_ID AS "NEW_HEAD_ID", COOL_J3.I AS "I", COOL_J3.S4 AS "S4" FROM
AVALASSI "XSV_RCWD_F0001_CHANNELS" "COOL_C2", AVALASSI "XSV_RCWD_F0001_JOVS" "COOL_J3" WHERE
COOL_J3.CHANNEL_ID=COOL_C2.CHANNEL_ID AND COOL_J3.IOV_SINCE=COALESCE(
SELECT /*+ QR_NAME(MAXI)*/
MAX(COOL_J1.IOV_SINCE) FROM AVALASSI "XSV_RCWD_F0001_JOVS" COOL_J1 WHERE
(COOL_J1.CHANNEL_ID=COOL_C2.CHANNEL_ID AND COOL_J1.IOV_SINCE<="since1" & "since2") AND
COOL_J3.IOV_SINCE<="until3" AND COOL_J3.IOV_UNTIL>="since3" ORDER BY COOL_J3.CHANNEL_ID ASC,
COOL_J3.IOV_SINCE ASC
Main hint in alternative SQL statement (hint options="nohint"):
/*+ NO_BIND,AWARE_QR_NAME(MAIN) */
Identified 3 execution plan(s) from 12 trace files (in /tmp/avalassi/ALL/11.2.0.3/SV_R on lxmrta5001):

```

Trace file (stat)-(peek)-(hint)	Exec plan	Bind variables	Hints
		since1 since3 until3 since3u	Used Unused
stat-peekhi	█	99000 99000 99010 99000	7 █
stat-peeklo	█	0 0 10 0	7 █
nost-peekhi	█	99000 99000 99010 99000	7 █
nost-peeklo	█	0 0 10 0	7 █
emst-peekhi	█	99000 99000 99010 99000	7 █
emst-peeklo	█	0 0 10 0	7 █
stat-peekhi-nohint	#1	99000 99000 99010 99000	3 0
stat-peeklo-nohint	#2	0 0 10 0	3 0
nost-peekhi-nohint	#1	99000 99000 99010 99000	3 0
nost-peeklo-nohint	#2	0 0 10 0	3 0
emst-peekhi-nohint	#3	99000 99000 99010 99000	3 0
emst-peeklo-nohint	#3	0 0 10 0	3 0

SV_R / Execution plan#1 (COOL-preview on Oracle 11.2.0.3.0)

Id	Operation	Name
0	SELECT STATEMENT	
1	SORT ORDER BY	
2	NESTED LOOPS	
3	NESTED LOOPS	
4	INDEX FULL SCAN	XSV_RCWD_F0001_CHANNELS_PK
5	SORT AGGREGATE	
6	FIRST ROW	
7	INDEX RANGE SCAN (MIN/MAX)	XSV_RCWD_F0001_JOVS_CSU3_INDX
8	INDEX RANGE SCAN	XSV_RCWD_F0001_JOVS_CSU3_INDX
9	TABLE ACCESS BY INDEX ROWID	XSV_RCWD_F0001_JOVS

```

4 - filter(COALESCE(:since3)<=:until3)
7 - access("COOL_J1"."CHANNEL_ID"=:B1 AND "COOL_J1"."IOV_SINCE"=:since1)
8 - access("COOL_J3"."CHANNEL_ID"=:COOL_C2"."CHANNEL_ID" AND
"COOL_J3"."IOV_SINCE"=:COALESCE(:since3) AND "COOL_J3"."IOV_UNTIL">=:since3u AND
"COOL_J3"."IOV_SINCE"=:until3 AND "COOL_J3"."IOV_UNTIL" IS NOT NULL)
8 - filter("COOL_J3"."IOV_UNTIL">=:since3u)
/*+
BEGIN_OUTLINE_DATA
IGNORE_OPTIM_EMBEDDED_HINTS
OPTIMIZER_FEATURES_ENABLE('11.2.0.3')
DB_VERSION('11.2.0.3')
ALL_ROWS
OUTLINE_LEAF(@"MAXI")
OUTLINE_LEAF(@"MAIN")
OUTLINE(@"MAXI")
OUTLINE(@"MAIN")
INDEX(@"MAIN" "COOL_C2"@"MAIN" ("XSV_RCWD_F0001_CHANNELS"."CHANNEL_ID"))
INDEX(@"MAIN" "COOL_J3"@"MAIN" ("XSV_RCWD_F0001_JOVS"."CHANNEL_ID"))
"XSV_RCWD_F0001_JOVS"."IOV_SINCE"="XSV_RCWD_F0001_JOVS"."IOV_UNTIL")
LEADING(@"MAIN" "COOL_C2"@"MAIN" "COOL_J3"@"MAIN")
USE_M@"MAIN" "COOL_J3"@"MAIN")
NL_JOIN(@"MAIN" "COOL_J3"@"MAIN")
PUSH_SUBQ(@"MAXI")
INDEX(@"MAXI" "COOL_J1"@"MAXI" ("XSV_RCWD_F0001_JOVS"."CHANNEL_ID"))
"XSV_RCWD_F0001_JOVS"."IOV_SINCE"="XSV_RCWD_F0001_JOVS"."IOV_UNTIL")
END_OUTLINE_DATA
*/

```

1. Use case SV_R (COOL-preview on Oracle 11.2.0.2.0)

Oracle Database 11g Enterprise Edition Release 11.2.0.2.0 - 64bit Production
Instance intR1 (Linux x86_64 on i386) on 2011-12-13

```

Primary SQL statement (hint options=""):
SELECT /*+ NO_BIND,AWARE_QR_NAME(MAIN) INDEX@MAIN COOL_J@MAIN (CHANNEL_ID)@SV_SINCE@IOV_UNTIL@
LEADING@MAIN COOL_C2@MAIN COOL_J@MAIN) USE_M@MAIN COOL_J@MAIN) INDEX@MAXI COOL_J@MAXI
(CHANNEL_ID)@SV_SINCE@IOV_UNTIL)@COOL_J3 OBJECT_ID AS "OBJECT_ID", COOL_J3.CHANNEL_ID AS "CHANNEL_ID",
COOL_J3.IOV_SINCE AS "IOV_SINCE", COOL_J3.IOV_UNTIL AS "IOV_UNTIL", COOL_J3.USER_TAG_ID AS "USER_TAG_ID",
COOL_J3.SYS_INSTM AS "SYS_INSTM", COOL_J3.LASTMOD_DATE AS "LASTMOD_DATE", COOL_J3.ORIGINAL_ID AS
"ORIGINAL_ID", COOL_J3.NEW_HEAD_ID AS "NEW_HEAD_ID", COOL_J3.I AS "I", COOL_J3.S4 AS "S4" FROM
AVALASSI "XSV_RCWD_F0001_CHANNELS" "COOL_C2", AVALASSI "XSV_RCWD_F0001_JOVS" "COOL_J3" WHERE
COOL_J3.CHANNEL_ID=COOL_C2.CHANNEL_ID AND COOL_J3.IOV_SINCE=COALESCE(
SELECT /*+ QR_NAME(MAXI)*/
MAX(COOL_J1.IOV_SINCE) FROM AVALASSI "XSV_RCWD_F0001_JOVS" COOL_J1 WHERE
(COOL_J1.CHANNEL_ID=COOL_C2.CHANNEL_ID AND COOL_J1.IOV_SINCE<="since1" & "since2") AND
COOL_J3.IOV_SINCE<="until3" AND COOL_J3.IOV_UNTIL>="since3" ORDER BY COOL_J3.CHANNEL_ID ASC,
COOL_J3.IOV_SINCE ASC
Main hint in alternative SQL statement (hint options="nohint"):
/*+ NO_BIND,AWARE_QR_NAME(MAIN) */
Identified 9 execution plan(s) from 12 trace files (in /tmp/avalassi/SV_R/11.2.0.2/SV_R on lxmrta5001):

```

Trace file (stat)-(peek)-(hint)	Exec plan	Bind variables	Hints
		since1 since3 until3 since3u	Used Unused
stat-peekhi	█	99000 99000 99010 99000	7 █
stat-peeklo	█	0 0 10 0	7 █
nost-peekhi	█	99000 99000 99010 99000	7 █
nost-peeklo	█	0 0 10 0	7 █
emst-peekhi	█	99000 99000 99010 99000	7 █
emst-peeklo	█	0 0 10 0	7 █
stat-peekhi-nohint	#5	99000 99000 99010 99000	3 0
stat-peeklo-nohint	#6	0 0 10 0	3 0
nost-peekhi-nohint	#7	99000 99000 99010 99000	3 0
nost-peeklo-nohint	#8	0 0 10 0	3 0
emst-peekhi-nohint	#9	99000 99000 99010 99000	3 0
emst-peeklo-nohint	#9	0 0 10 0	3 0

- **With hints: only one (good) plan on 10.2.0.5 and 11.2.0.3**
 - Plan looks different but is probably exactly the same algorithm
 - Using INDEX RANGE SCAN (MIN/MAX)
- **With hints: 3 (bad) plans on 11.2.0.2**
 - Depending on statistics and bind variables (no exec plan stability!)
 - All of them are bad and involve INDEX FULL SCAN

SV_R / Execution plan #1 (COOL-preview on Oracle 11.2.0.3.0)

Id	Operation	Name
0	SELECT STATEMENT	
1	SORT ORDER BY	
2	NESTED LOOPS	
3	NESTED LOOPS	
4	INDEX FULL SCAN	XSV_RCDWF0001_CHANNELS_PK
5	SORT AGGREGATE	
6	FIRST ROW	
7	INDEX RANGE SCAN (MIN/MAX)	XSV_RCDWF0001_JOVS_CSU_3INDX
8	INDEX RANGE SCAN	XSV_RCDWF0001_JOVS_CSU_3INDX
9	TABLE ACCESS BY INDEX ROWID	XSV_RCDWF0001_JOVS

```

4 - filter(COALESCE(:sinc3s)<=:until3)
7 - access("COOL_I1","CHANNEL_ID"=:B1 AND "COOL_I1"."IOV_SINCE"<=:since1)
8 - access("COOL_I3","CHANNEL_ID"=:COOL_C2,"CHANNEL_ID" AND
"COOL_I3"."IOV_SINCE"<=:COALESCE(:sinc3s) AND "COOL_I3"."IOV_UNTIL">=:sinc3u AND
"COOL_I3"."IOV_SINCE"<=:until3 AND "COOL_I3"."IOV_UNTIL" IS NOT NULL)
8 - filter("COOL_I3"."IOV_UNTIL">=:sinc3u)
  
```

```

/*
BEGIN_OUTLINE_DATA
IGNORE_OPTIM_EMBEDDED_HINTS
OPTIMIZER_FEATURES_ENABLE('11.2.0.3')
ALL_ROWS
OUTLINE_LEAF(@"MAXI")
OUTLINE_LEAF(@"MAIN")
OUTLINE(@"MAXI")
OUTLINE(@"MAIN")
INDEX(@"MAIN" "COOL_C2"@"MAIN" ("XSV_RCDWF0001_CHANNELS"."CHANNEL_ID"))
INDEX(@"MAIN" "COOL_I3"@"MAIN" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE"<=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
LEADING(@"MAIN" "COOL_C2"@"MAIN" "COOL_I3"@"MAIN")
USE_NL(@"MAIN" "COOL_I3"@"MAIN")
NL_BATCHEING(@"MAIN" "COOL_I3"@"MAIN")
PUSH_SUBQ(@"MAXI")
INDEX(@"MAXI" "COOL_I1"@"MAXI" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE"<=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
END_OUTLINE_DATA
*/
  
```

11.2.0.3 (good)

SV_R / Execution plan #1 (COOL-preview on Oracle 10.2.0.5.0)

Id	Operation	Name
0	SELECT STATEMENT	
1	SORT ORDER BY	
2	TABLE ACCESS BY INDEX ROWID	XSV_RCDWF0001_JOVS
3	NESTED LOOPS	
4	INDEX FULL SCAN	XSV_RCDWF0001_CHANNELS_PK
5	SORT AGGREGATE	
6	FIRST ROW	
7	INDEX RANGE SCAN (MIN/MAX)	XSV_RCDWF0001_JOVS_CSU_3INDX
8	INDEX RANGE SCAN	XSV_RCDWF0001_JOVS_CSU_3INDX
9	SORT AGGREGATE	
10	FIRST ROW	
11	INDEX RANGE SCAN (MIN/MAX)	XSV_RCDWF0001_JOVS_CSU_3INDX

```

4 - filter(COALESCE(:sinc3s)<=:until3)
7 - access("COOL_I1","CHANNEL_ID"=:B1 AND "COOL_I1"."IOV_SINCE"<=:since1)
8 - access("COOL_I3","CHANNEL_ID"=:COOL_C2,"CHANNEL_ID" AND
"COOL_I3"."IOV_SINCE">=:COALESCE(:sinc3s) AND "COOL_I3"."IOV_UNTIL">=:sinc3u AND
"COOL_I3"."IOV_SINCE"<=:until3 AND "COOL_I3"."IOV_UNTIL" IS NOT NULL)
8 - filter("COOL_I3"."IOV_UNTIL">=:sinc3u)
11 - access("COOL_I1","CHANNEL_ID"=:B1 AND "COOL_I1"."IOV_SINCE"<=:since1)
  
```

```

/*
BEGIN_OUTLINE_DATA
IGNORE_OPTIM_EMBEDDED_HINTS
OPTIMIZER_FEATURES_ENABLE('10.2.0.5')
ALL_ROWS
OUTLINE_LEAF(@"MAXI")
OUTLINE_LEAF(@"MAIN")
OUTLINE(@"MAXI")
OUTLINE(@"MAIN")
INDEX(@"MAIN" "COOL_C2"@"MAIN" ("XSV_RCDWF0001_CHANNELS"."CHANNEL_ID"))
INDEX(@"MAIN" "COOL_I3"@"MAIN" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE">=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
LEADING(@"MAIN" "COOL_C2"@"MAIN" "COOL_I3"@"MAIN")
USE_NL(@"MAIN" "COOL_I3"@"MAIN")
PUSH_SUBQ(@"MAXI")
INDEX(@"MAXI" "COOL_I1"@"MAXI" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE">=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
END_OUTLINE_DATA
*/
  
```

10.2.0.5 (good)

SV_R / Execution plan #1 (COOL-preview on Oracle 11.2.0.2.0)

Id	Operation	Name
0	SELECT STATEMENT	
1	SORT ORDER BY	
2	NESTED LOOPS	
3	NESTED LOOPS	
4	VIEW	VW_SQ_1
5	FILTER	
6	HASH GROUP BY	
7	INDEX FULL SCAN	XSV_RCDWF0001_JOVS_CSU_3INDX
8	INDEX RANGE SCAN	XSV_RCDWF0001_JOVS_CSU_3INDX
9	TABLE ACCESS BY INDEX ROWID	XSV_RCDWF0001_JOVS

```

5 - filter(COALESCE(MAX("COOL_I1"."IOV_SINCE"),:sinc3s)<=:until3)
7 - access("COOL_I1","IOV_SINCE"<=:since1)
7 - filter("COOL_I1"."IOV_SINCE"<=:since1)
8 - access("ITEM_ID"=:COOL_I3,"CHANNEL_ID" AND
"COOL_I3"."IOV_SINCE">=:COALESCE(MAX("COOL_I1"."IOV_SINCE"),:sinc3s) AND
"COOL_I3"."IOV_UNTIL">=:sinc3u AND "COOL_I3"."IOV_SINCE"<=:until3 AND "COOL_I3"."IOV_UNTIL"
IS NOT NULL)
8 - filter("COOL_I3"."IOV_UNTIL">=:sinc3u AND "COOL_I3"."CHANNEL_ID" IS NOT NULL)
  
```

```

/*
BEGIN_OUTLINE_DATA
IGNORE_OPTIM_EMBEDDED_HINTS
OPTIMIZER_FEATURES_ENABLE('11.2.0.2')
DIAGNOSTICS_ENABLE('11.2.0.2')
ALL_ROWS
OUTLINE_LEAF(@"SEL$009E028F")
OUTLINE_LEAF(@"SEL$693E4321")
ELIMINATE_JOIN(@"SEL$DC3943C0" "COOL_C2"@"MAIN")
OUTLINE(@"MAXI")
OUTLINE(@"SEL$DC3943C0")
UNNEST(@"MAXI")
OUTLINE(@"SEL$B2D7668E")
OUTLINE(@"MAIN")
NO_ACCESS(@"SEL$693E4321" "VW_SQ_1"@"SEL$B2D7668E")
INDEX(@"SEL$693E4321" "COOL_I3"@"MAIN" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE">=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
LEADING(@"SEL$693E4321" "VW_SQ_1"@"SEL$B2D7668E" "COOL_I3"@"MAIN")
USE_NL(@"SEL$693E4321" "COOL_I3"@"MAIN")
NL_BATCHEING(@"SEL$693E4321" "COOL_I3"@"MAIN")
INDEX(@"SEL$009E028F" "COOL_I1"@"MAXI" ("XSV_RCDWF0001_JOVS"."CHANNEL_ID"))
"XSV_RCDWF0001_JOVS"."IOV_SINCE">=:XSV_RCDWF0001_JOVS"."IOV_UNTIL")
USE_HASH_AGGREGATION(@"SEL$009E028F")
END_OUTLINE_DATA
*/
  
```

11.2.0.2 (bad) – 1st of 3 plans

- **POOL ownership is being moved to ATLAS**
 - Dropped by LHCb in Q4 2011 (replaced by direct ROOT)
 - Still being maintained by PF team for ATLAS in \leq LCG61
 - LCG61 is ATLAS prod release for 2012 LHC data taking
 - Built internally by ATLAS as of LCG62 (current dev release)
- **Several enhancements in POOL collection packages**
 - Fully developed by ATLAS for usage exclusively in ATLAS



- **CORAL and COOL will be the main focus**
 - POOL supported only as long as its transfer to ATLAS is completed
- **Highest load comes from software and service support**
 - Regular software releases, port to new platforms and externals
 - Oracle client installation and support
 - Debugging of complex service issues
 - Individual user support
- **A few new developments, in parallel**
 - e.g. CORAL monitoring
- **Maintenance tasks**
 - Move from CVS to SVN, add support for cmake
 - Improve documentation, also to simplify future support



- **CORAL and COOL are an essential and successful common solution for data taking and analysis in the LHC experiments**
 - ATLAS and LHCb need CORAL and COOL – CMS needs CORAL
 - Mainly for conditions data, but also for geometry, trigger and more
 - The LHC experiments expect that CORAL and COOL will continue to be supported by CERN or WLCG, as discussed in the DB TEG
- **Main recent achievements for CORAL and COOL**
 - COOL performance validation on Oracle 11g servers
 - Improved CORAL handling of network and database instabilities
- **POOL is no longer a common project**
 - Dropped by LHCb, will now be used and maintained only by ATLAS