IDERA® | Precise Case Study

PRECISE FOR ORACLE DATABASE

Telecomm Enterprise Improves Database Performance by Almost 400%

INTRODUCTION

A large enterprise telecommunications company that provides wireless services and internet services in the United States of America uses a third-party application to gather configuration and performance data on their servers. However, a batch job that runs at midnight each night to gather data from these servers and load it into the service reporter database was taking far too much time to run. By analyzing the long history of performance data within Precise, it appeared that a particular statement was slowly increasing in execution time.

THE APPLICATION

The company uses a third-party application to manage the Information Technology infrastructure. The company runs a batch job at midnight each night to gather configuration and performance data from 2,000 of their servers and load the data into the Service Reporter database of the third-party application. Data analysts begin at 8:30 AM to review the data gathered during the previous evening for capacity planning.

THE PROBLEM

The batch job that loads the data was taking 12 hours to run. The consultants for the third-party application do not have access to modifying the source code of the third-party application. The company was expecting to increase the number of processed servers from 2,000 to 6,000. The team for the third-party application was planning on upgrading their hardware to handle this additional data. Data analysts in the group for capacity planning were unable to start their analysis until as late as noon.

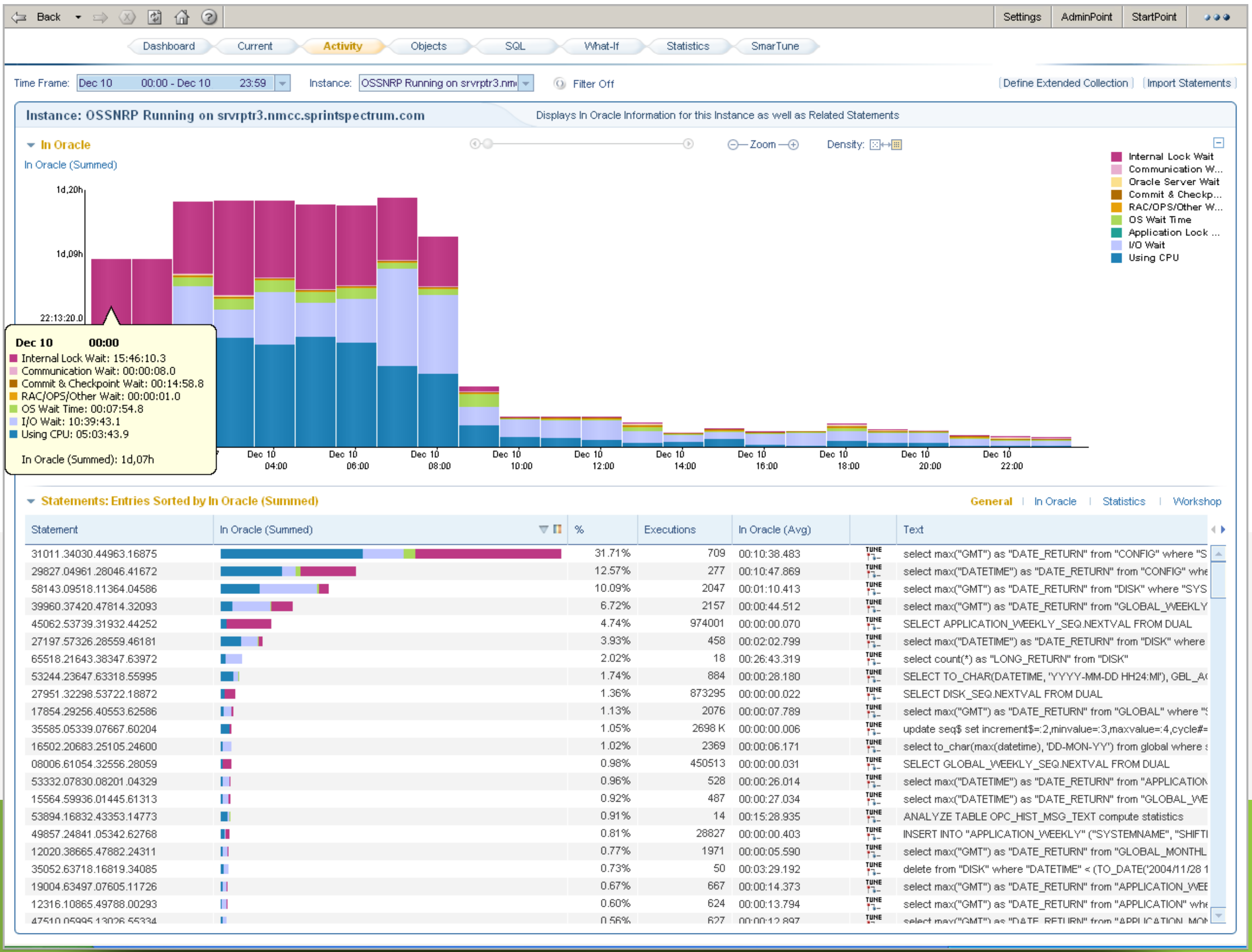
THE PRECISE INSTALLATION

The company installed monitors for Precise on the Oracle Database server for the Service Reporter database. The Precise agents were collecting data continuously for over nine months leading up to the problem.

THE PRECISE ANALYSIS – PHASE 1

By analyzing the long history of performance data within Precise, it appeared that fewer than six SQL statements were accounting for the majority of time spent in the Oracle Database. A single SQL statement was slowly increasing in execution time. However, this problem was not apparent unless they looked at six months of data. The ‘SQL’ workspace showed that several key indexes were missing. Analysis of individual SQL statements showed that adding indexes would have a huge positive impact with minimal risk. The ‘What-If’ workspace showed that adding indexes would not adversely affect other SQL statements.

The ‘In Oracle’ area at the top of the ‘Activity’ workspace for the relevant instance of Oracle Database (Figure 1), the overall performance showed an excessive amount of internal lock wait and input/output wait (with almost 16 hours and almost 11 hours, respectively, at midnight). The “Statements: Entries Sorted by In Oracle (Summed)” area at the bottom of the ‘Activity’ workspace shows that the top three statements together account for more than 50% of all of the time in Oracle Database (with some 32%, 13%, and 10%, respectively).



Time Frame: Dec 10 00:00 - Dec 10 23:59

Instance: OSSNRP Running on srvrptr3.nm

Filter Off

Instance: OSSNRP Running on srvrptr3.nmcc.sprintspectrum.com

Displays In Oracle Information for this Instance as well as Related Statements

In Oracle

In Oracle (Summed)

🟥 Internal Lock Wait
🟪 Communication W...
🟨 Oracle Server Wait
🟫 Commit & Checkp...
🟧 RAC/OPS/Other W...
🟩 OS Wait Time
🟩 Application Lock ...
🟦 I/O Wait
🟦 Using CPU

Dec 10 00:00
🟥 Internal Lock Wait: 15:46:10.3
🟪 Communication Wait: 00:00:08.0
🟫 Commit & Checkpoint Wait: 00:14:58.8
🟧 RAC/OPS/Other Wait: 00:00:01.0
🟩 OS Wait Time: 00:07:54.8
🟦 I/O Wait: 10:39:43.1
🟦 Using CPU: 05:03:43.9

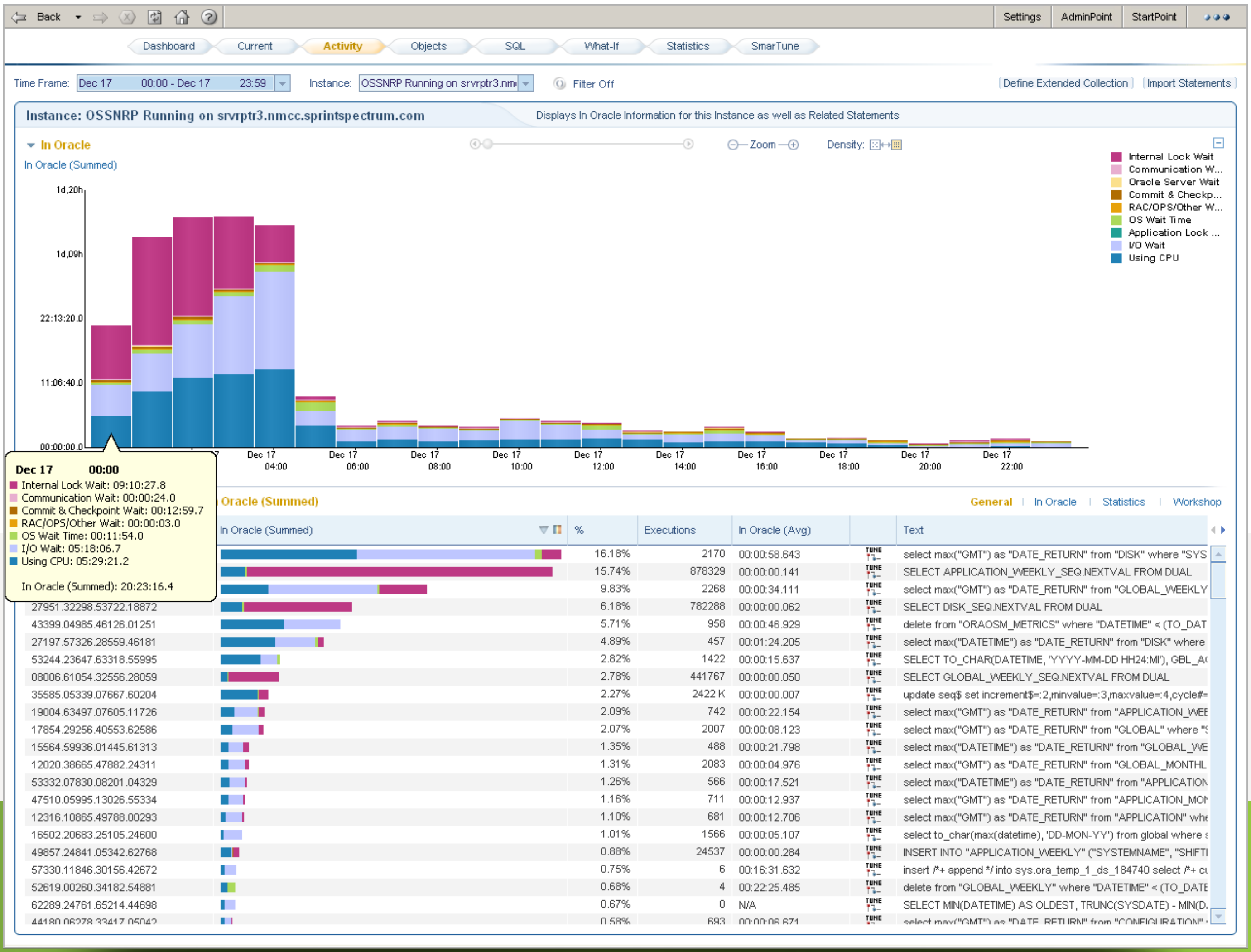
In Oracle (Summed): 1d,07h

Statements: Entries Sorted by In Oracle (Summed)

Statement	In Oracle (Summed)	%	Executions	In Oracle (Avg)		Text
31011.34030.44963.16875		31.71%	709	00:10:38.483	TUNE	select max("GMT") as "DATE_RETURN" from "CONFIG" where "S
29827.04961.28046.41672		12.57%	277	00:10:47.869	TUNE	select max("DATETIME") as "DATE_RETURN" from "CONFIG" whe
58143.09518.11364.04586		10.09%	2047	00:01:10.413	TUNE	select max("GMT") as "DATE_RETURN" from "DISK" where "SYS
39960.37420.47814.32093		6.72%	2157	00:00:44.512	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL_WEEKLY
45062.53739.31932.44252		4.74%	974001	00:00:00.070	TUNE	SELECT APPLICATION_WEEKLY_SEQ.NEXTVAL FROM DUAL
27197.57326.28559.46181		3.93%	458	00:02:02.799	TUNE	select max("DATETIME") as "DATE_RETURN" from "DISK" where
65518.21643.38347.63972		2.02%	18	00:26:43.319	TUNE	select count(*) as "LONG_RETURN" from "DISK"
53244.23647.63318.55995		1.74%	884	00:00:28.180	TUNE	SELECT TO_CHAR(DATETIME, 'YYYY-MM-DD HH24:MI'), GBL_A(
27951.32298.53722.18872		1.36%	873295	00:00:00.022	TUNE	SELECT DISK_SEQ.NEXTVAL FROM DUAL
17854.29256.40553.62586		1.13%	2076	00:00:07.789	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL" where "S
35585.05339.07667.60204		1.05%	2698 K	00:00:00.006	TUNE	update seq$ set increment$=:2,minvalue=:3,maxvalue=:4,cycle#=
16502.20683.25105.24600		1.02%	2369	00:00:06.171	TUNE	select to_char(max(datetime), 'DD-MON-YY') from global where s
08006.61054.32556.28059		0.98%	450513	00:00:00.031	TUNE	SELECT GLOBAL_WEEKLY_SEQ.NEXTVAL FROM DUAL
53332.07830.08201.04329		0.96%	528	00:00:26.014	TUNE	select max("DATETIME") as "DATE_RETURN" from "APPLICATION
15564.59936.01445.61313		0.92%	487	00:00:27.034	TUNE	select max("DATETIME") as "DATE_RETURN" from "GLOBAL_WE
53894.16832.43353.14773		0.91%	14	00:15:28.935	TUNE	ANALYZE TABLE OPC_HIST_MSG_TEXT compute statistics
49857.24841.05342.62768		0.81%	28827	00:00:00.403	TUNE	INSERT INTO "APPLICATION_WEEKLY" ("SYSTEMNAME", "SHIFTI
12020.38665.47882.24311		0.77%	1971	00:00:05.590	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL_MONTHL
35052.63718.16819.34085		0.73%	50	00:03:29.192	TUNE	delete from "DISK" where "DATETIME" < (TO_DATE('2004/11/28 1
19004.63497.07605.11726		0.67%	667	00:00:14.373	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION_WEE
12316.10865.49788.00293		0.60%	624	00:00:13.794	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION" whe
47510.05995.13026.55334		0.56%	627	00:00:12.897	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION_MON

Figure 1:
The ‘In Oracle’ area at the top of the ‘Activity’ workspace for the relevant instance of Oracle Database.

The ‘In Oracle’ area at the top of the ‘Activity’ workspace for the relevant instance of Oracle Database (Figure 2) shows that the run-time of the batch job reduced to 4½ hours (that is, refer to the clock time of the sixth vertical bar) by creating efficient indexes with the ‘SQL’ workspace.



Time Frame: Dec 17 00:00 - Dec 17 23:59
Instance: OSSNRP Running on srvrptr3.nm

Instance: OSSNRP Running on srvrptr3.nmcc.sprintspectrum.com
Displays In Oracle Information for this Instance as well as Related Statements

Dec 17 00:00
Internal Lock Wait: 09:10:27.8
Communication Wait: 00:00:24.0
Commit & Checkpoint Wait: 00:12:59.7
RAC/OPS/Other Wait: 00:00:03.0
OS Wait Time: 00:11:54.0
I/O Wait: 05:18:06.7
Using CPU: 05:29:21.2

In Oracle (Summed): 20:23:16.4

Oracle (Summed)

	In Oracle (Summed)	%	Executions	In Oracle (Avg)		Text
		16.18%	2170	00:00:58.643	TUNE	select max("GMT") as "DATE_RETURN" from "DISK" where "SYS
		15.74%	878329	00:00:00.141	TUNE	SELECT APPLICATION_WEEKLY_SEQ.NEXTVAL FROM DUAL
		9.83%	2268	00:00:34.111	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL_WEEKLY
27951.32298.53722.18872		6.18%	782288	00:00:00.062	TUNE	SELECT DISK_SEQ.NEXTVAL FROM DUAL
43399.04985.46126.01251		5.71%	958	00:00:46.929	TUNE	delete from "ORAOSM_METRICS" where "DATETIME" < (TO_DAT
27197.57326.28559.46181		4.89%	457	00:01:24.205	TUNE	select max("DATETIME") as "DATE_RETURN" from "DISK" where
53244.23647.63318.55995		2.82%	1422	00:00:15.637	TUNE	SELECT TO_CHAR(DATETIME, 'YYYY-MM-DD HH24:MI'), GBL_A
08006.61054.32556.28059		2.78%	441767	00:00:00.050	TUNE	SELECT GLOBAL_WEEKLY_SEQ.NEXTVAL FROM DUAL
35585.05339.07667.60204		2.27%	2422 K	00:00:00.007	TUNE	update seq$ set increment$=:2,minvalue=:3,maxvalue=:4,cycle#=
19004.63497.07605.11726		2.09%	742	00:00:22.154	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION_WEE
17854.29256.40553.62586		2.07%	2007	00:00:08.123	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL" where "
15564.59936.01445.61313		1.35%	488	00:00:21.798	TUNE	select max("DATETIME") as "DATE_RETURN" from "GLOBAL_WE
12020.38665.47882.24311		1.31%	2083	00:00:04.976	TUNE	select max("GMT") as "DATE_RETURN" from "GLOBAL_MONTHL
53332.07830.08201.04329		1.26%	566	00:00:17.521	TUNE	select max("DATETIME") as "DATE_RETURN" from "APPLICATION
47510.05995.13026.55334		1.16%	711	00:00:12.937	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION_MON
12316.10865.49788.00293		1.10%	681	00:00:12.706	TUNE	select max("GMT") as "DATE_RETURN" from "APPLICATION" whe
16502.20683.25105.24600		1.01%	1566	00:00:05.107	TUNE	select to_char(max(datetime), 'DD-MON-YY') from global where s
49857.24841.05342.62768		0.88%	24537	00:00:00.284	TUNE	INSERT INTO "APPLICATION_WEEKLY" ("SYSTEMNAME", "SHIFTI
57330.11846.30156.42672		0.75%	6	00:16:31.632	TUNE	insert /*+ append */ into sys.ora_temp_1_ds_184740 select /*+ cu
52619.00260.34182.54881		0.68%	4	00:22:25.485	TUNE	delete from "GLOBAL_WEEKLY" where "DATETIME" < (TO_DATE
62289.24761.65214.44698		0.67%	0	N/A	TUNE	SELECT MIN(DATETIME) AS OLDEST, TRUNC(SYSDATE) - MIN(D.
44180.06278.33417.05042		0.58%	693	00:00:06.671	TUNE	select max("GMT") as "DATE_RETURN" from "CONFIGURATION"

Figure 2:
The ‘In Oracle’ area at the top of the ‘Activity’ workspace for the relevant instance of Oracle Database before and after adding efficient indexes.

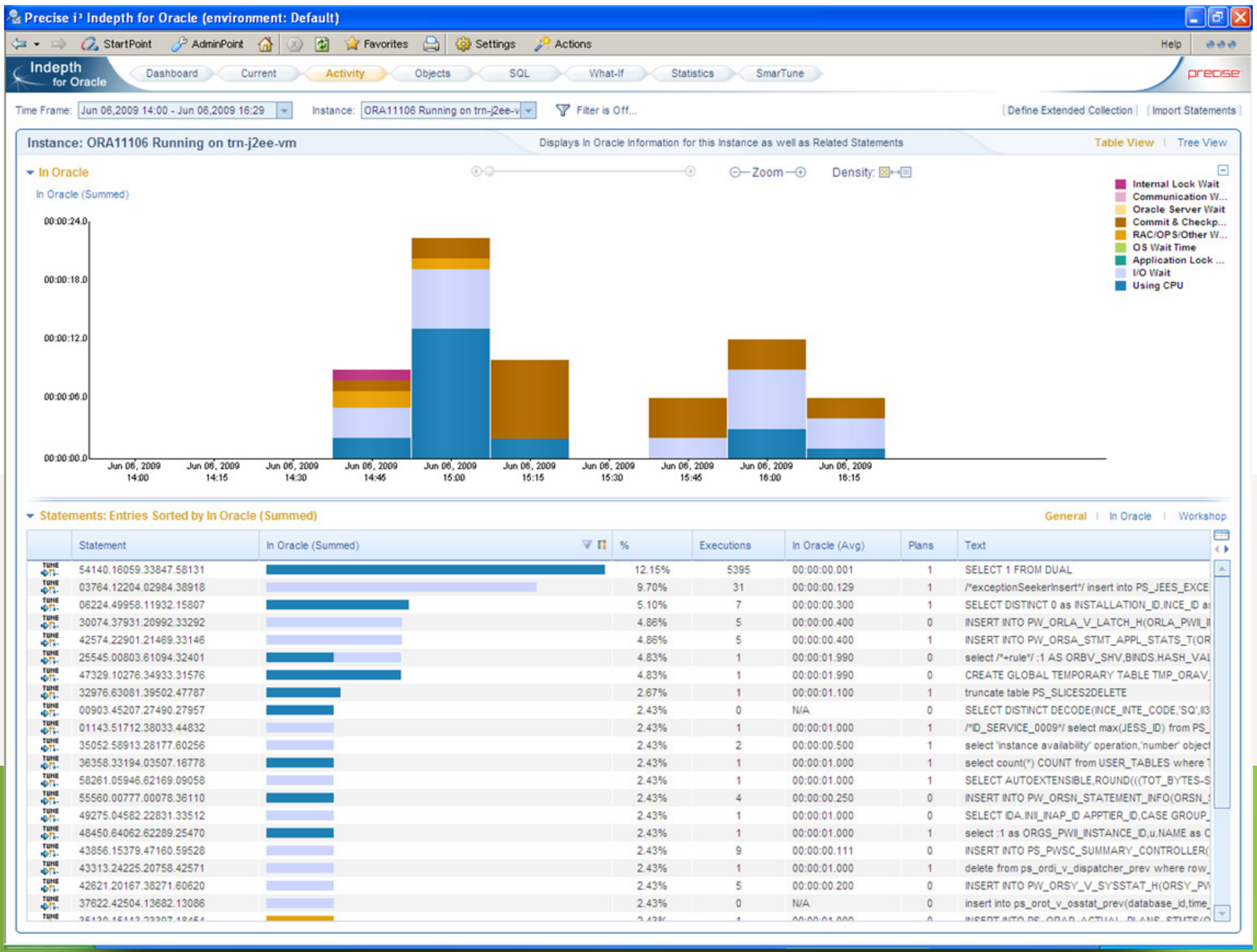
THE PRECISE ANALYSIS – PHASE 2

Indexes were further refined using the 'SQL' workspace. It appears that sequence values of Oracle Database, used extensively in the batch process, were not being cached that resulted in a lot of internal lock wait states. The company changed all sequences that were not using the cache.

It also appeared that SQL statements had many hardcoded values. The third-party application set the parameter 'CURSOR_SHARING' that determines what kind of SQL statements can share the same cursors to 'EXACT' which only allows statements with identical text to share the same cursor.

This setting resulted in tens of thousands of hard parses. The company changed this setting to 'SIMILAR' that causes statements that may differ in some literals but are otherwise identical to share a cursor unless the literals affect either the meaning of the statement or the degree to which the plan is optimized. After these additional changes, the company reduced the run-time of the batch job from 4½ hours to 2½ hours.

The 'Overview' area at the top of the 'Objects' workspace for the relevant instance of Oracle Database (Figure 3) shows the cumulative positive effects of all changes. That is, compare the final bar and its informational balloon of this screen capture image with the first bar and its informational balloon from Figure 1.



Time Frame: Jun 06,2009 14:00 - Jun 06,2009 16:29

Instance: ORA11106 Running on trn-j2ee-v

Filter is Off...

Instance: ORA11106 Running on trn-j2ee-vm

Displays In Oracle Information for this Instance as well as Related Statements

Statements: Entries Sorted by In Oracle (Summed)

Statement	In Oracle (Summed)	%	Executions	In Oracle (Avg)	Plans	Text
54140.16059.33847.58131		12.15%	5395	00:00:00.001	1	SELECT 1 FROM DUAL
03764.12204.02984.38918		9.70%	31	00:00:00.129	1	/*exceptionSeekerInsert*/ insert into PS_JEES_EXCE
06224.49958.11932.15807		5.10%	7	00:00:00.300	1	SELECT DISTINCT 0 as INSTALLATION_ID,INCE_ID as
30074.37931.20992.33292		4.86%	5	00:00:00.400	0	INSERT INTO PW_ORLA_V_LATCH_H(ORLA_PWII_I
42574.22901.21469.33146		4.86%	5	00:00:00.400	1	INSERT INTO PW_ORSA_STMT_APPL_STATS_T(OR
25545.00803.61094.32401		4.83%	1	00:00:01.990	0	select /*+rule*/ :1 AS ORBV_SHV,BINDS.HASH_VAL
47329.10276.34933.31576		4.83%	1	00:00:01.990	0	CREATE GLOBAL TEMPORARY TABLE TMP_ORAV_
32976.63081.39502.47787		2.67%	1	00:00:01.100	1	truncate table PS_SLICES2DELETE
00903.45207.27490.27957		2.43%	0	N/A	0	SELECT DISTINCT DECODE(INCE_INTE_CODE,'SQ',II3
01143.51712.38033.44832		2.43%	1	00:00:01.000	1	/*ID_SERVICE_0009*/ select max(JESS_ID) from PS_
35052.58913.28177.60256		2.43%	2	00:00:00.500	1	select 'instance availability' operation,'number' object
36358.33194.03507.16778		2.43%	1	00:00:01.000	1	select count(*) COUNT from USER_TABLES where T
58261.05946.62169.09058		2.43%	1	00:00:01.000	1	SELECT AUTOEXTENSIBLE,ROUND(((TOT_BYTES-S
55560.00777.00078.36110		2.43%	4	00:00:00.250	0	INSERT INTO PW_ORSN_STATEMENT_INFO(ORSN_:
49275.04582.22831.33512		2.43%	1	00:00:01.000	0	SELECT IDA.INII_INAP_ID APPTIER_ID,CASE GROUP_
48450.64062.62289.25470		2.43%	1	00:00:01.000	1	select :1 as ORGS_PWII_INSTANCE_ID,u.NAME as C
43856.15379.47160.59528		2.43%	9	00:00:00.111	0	INSERT INTO PS_PWSC_SUMMARY_CONTROLLER(
43313.24225.20758.42571		2.43%	1	00:00:01.000	1	delete from ps_ordi_v_dispatcher_prev where row_
42621.20167.38271.60620		2.43%	5	00:00:00.200	0	INSERT INTO PW_ORSY_V_SYSSTAT_H(ORSY_PW
37622.42504.13682.13086		2.43%	0	N/A	0	insert into ps_orot_v_osstat_prev(database_id,time_

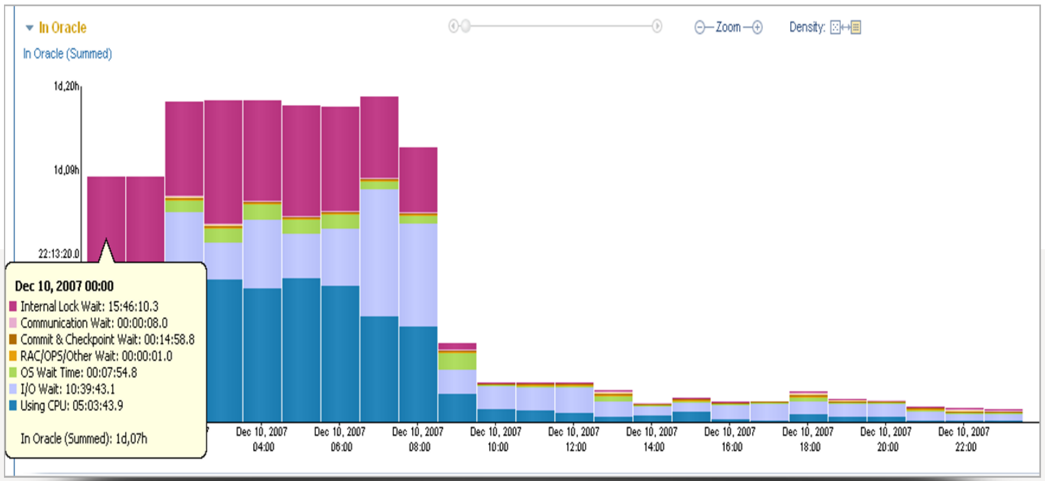
Figure 3:
The "Overview" area at the top of the 'Objects' workspace for the relevant instance of Oracle Database after the changes.

THE RESOLUTION

The run-time of the batch job processing 2,000 servers was reduced from 12 hours to 2½ hours without modifying any source code. The team for the third-party application is now able to process the required 6,000 servers in less than 8½ hours. The batch job now spends only 20% of its time in the Oracle Database, as compared to 90% earlier.

The evolution of the performance tuning with Precise shows the initial performance (Figure 4a), the performance after the first phase of the analysis that resulted in adding useful indexes (Figure 4b), and the performance after the second phase of the analysis that resulted in modifying a parameter (Figure 4c).

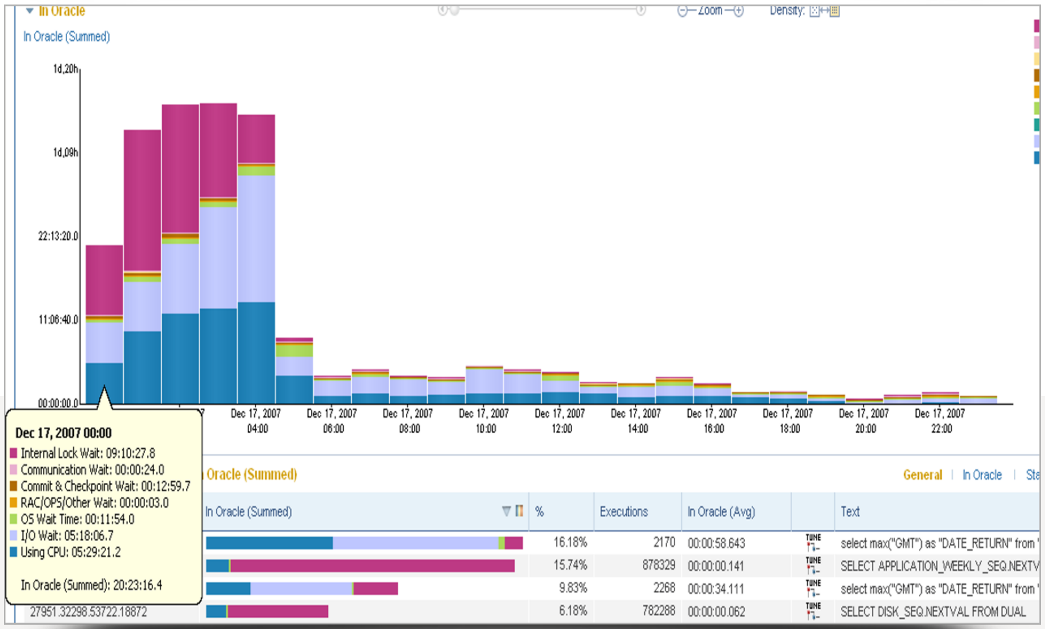
The evolution of performance tuning with Precise



Dec 10, 2007 00:00
- Internal Lock Wait: 15:46:10.3
- Communication Wait: 00:00:08.0
- Commit & Checkpoint Wait: 00:14:58.8
- RAC/OPS/Other Wait: 00:00:01.0
- OS Wait Time: 00:07:54.8
- I/O Wait: 10:39:43.1
- Using CPU: 05:03:43.9

In Oracle (Summed): 1d,07h

Figure 4a: Initial performance.



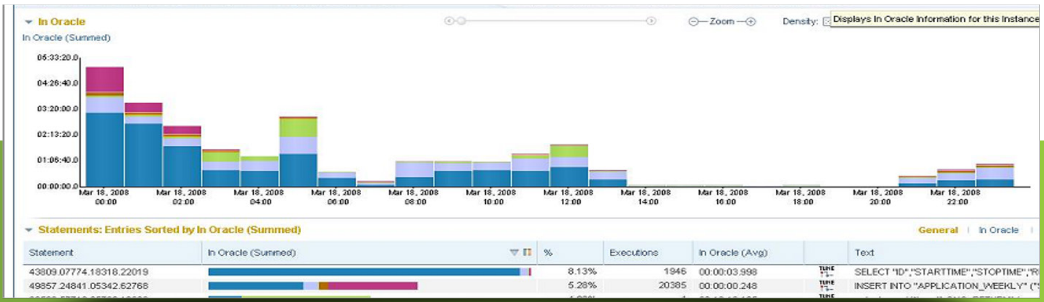
Dec 17, 2007 00:00
- Internal Lock Wait: 09:10:27.8
- Communication Wait: 00:00:24.0
- Commit & Checkpoint Wait: 00:12:59.7
- RAC/OPS/Other Wait: 00:00:03.0
- OS Wait Time: 00:11:54.0
- I/O Wait: 05:18:06.7
- Using CPU: 05:29:21.2

In Oracle (Summed): 20:23:16.4

Oracle (Summed)

	In Oracle (Summed)	%	Executions	In Oracle (Avg)		Text
		16.18%	2170	00:00:58.643	TUNE	select max("GMT") as "DATE_RETURN" from '
		15.74%	878329	00:00:00.141	TUNE	SELECT APPLICATION_WEEKLY_SEQ.NEXTV
		9.83%	2268	00:00:34.111	TUNE	select max("GMT") as "DATE_RETURN" from '
27951.32298.53722.18872		6.18%	782288	00:00:00.062	TUNE	SELECT DISK_SEQ.NEXTVAL FROM DUAL

Figure 4b: performance after the first phase of the analysis.



Statements: Entries Sorted by In Oracle (Summed)

Statement	In Oracle (Summed)	%	Executions	In Oracle (Avg)		Text
43809.07774.18318.22019		8.13%	1946	00:00:03.998	TUNE	SELECT "ID","STARTTIME","STOPTIME","R
49857.24841.05342.62768		5.28%	20385	00:00:00.248	TUNE	INSERT INTO "APPLICATION_WEEKLY" ("

Figure 4c: performance after the second phase of the analysis.

THE SAVINGS

The company can now postpone a necessary hardware upgrade indefinitely. A particular consultant for the third-party application who had been focusing on the performance of the application can now focus on the functional aspects of the application. The business consultants can now have the data ready for their analysis when they arrive in the morning. The manager of the group for capacity planning told another vendor that Precise saved his group “hundreds of thousands if not millions of dollars” by avoiding unnecessary costs for hardware and software upgrades.

SUMMARY

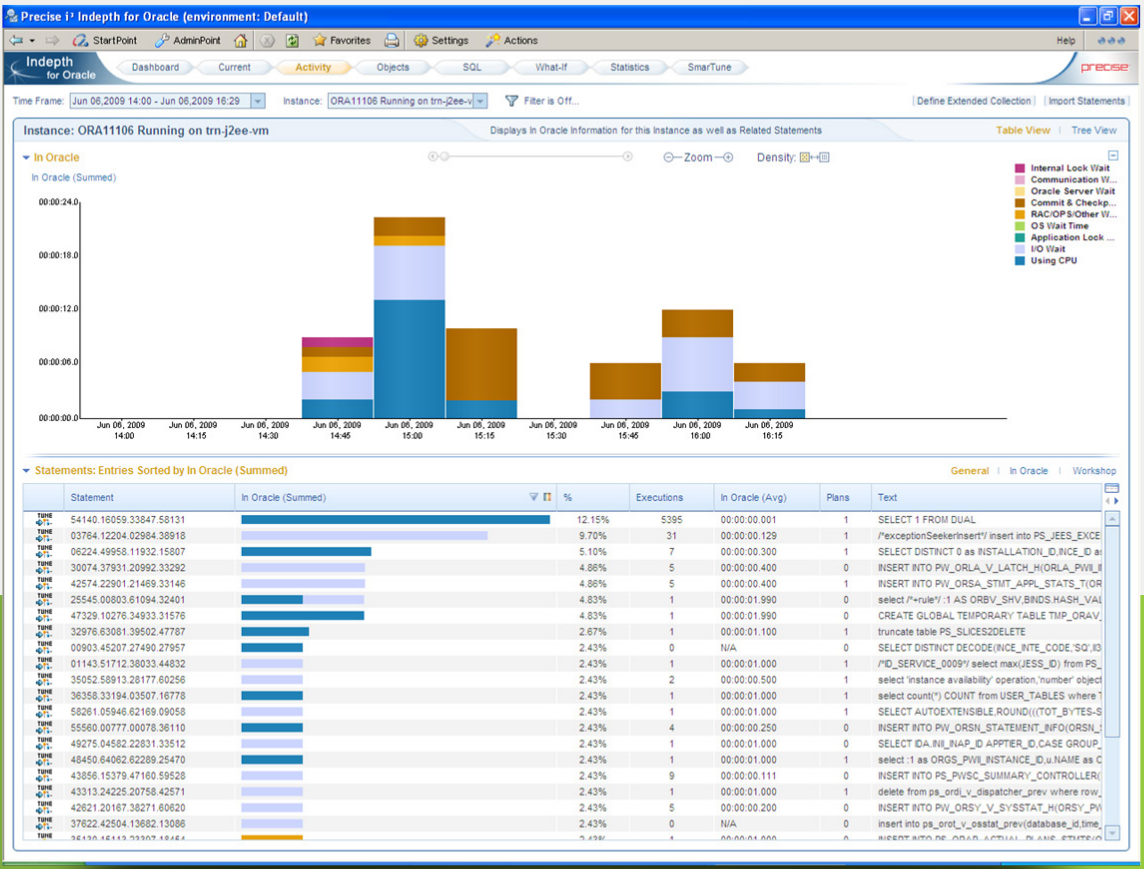
The business-critical third-party application that gathers configuration and performance data from 2,000 servers had a batch job that was running in 12 hours. The expected throughput of the application was going to triple. Precise showed how to make changes to indexes, objects, and initialization parameters. The company reduced the run-time of the batch job to 2½ hours. The analysts of the company now have the data they need when they need it. The company achieved an almost 400% improvement in job throughput without having access to any source code to the third-party application by only changing the database.

PRECISE FOR ORACLE DATABASE

ACCELERATE BUSINESS PERFORMANCE

- Database Performance Fuels Company Performance
- Multiple Platform Database Monitoring and Alerting
- Performance Management Database
- Root Cause Identification
- Tuning Recommendations
- What-if Analysis
- Capacity Planning

SEE IT IN ACTION



Statement	In Oracle (Summed)	%	Executions	In Oracle (Avg)	Plans	Text
54140.16059.33847.58131		12.15%	5395	00:00:00.001	1	SELECT 1 FROM DUAL
03764.12204.02984.38918		9.70%	31	00:00:00.129	1	/*exceptionSeekerInsert*/ insert into PS_JEES_EXCE
06224.49958.11932.15807		5.10%	7	00:00:00.300	1	SELECT DISTINCT 0 as INSTALLATION_ID,INCE_ID a
30074.37931.20992.33292		4.86%	5	00:00:00.400	0	INSERT INTO PW_ORLA_V_LATCH_H(ORLA_PWII_I
42574.22901.21469.33146		4.86%	5	00:00:00.400	1	INSERT INTO PW_ORSA_STMT_APPL_STATS_T(OR
25545.00803.61094.32401		4.83%	1	00:00:01.990	0	select /*+rule*/ :1 AS ORBV_SHV,BINDS.HASH_VAL
47329.10276.34933.31576		4.83%	1	00:00:01.990	0	CREATE GLOBAL TEMPORARY TABLE TMP_ORAV_
32976.63081.39502.47787		2.67%	1	00:00:01.100	1	truncate table PS_SLICES2DELETE
00903.45207.27490.27957		2.43%	0	N/A	0	SELECT DISTINCT DECODE(INCE_INTE_CODE,'SQ',I3
01143.51712.38033.44832		2.43%	1	00:00:01.000	1	/*ID_SERVICE_0009*/ select max(JESS_ID) from PS_
35052.58913.28177.60256		2.43%	2	00:00:00.500	1	select 'instance availability' operation,'number' object
36358.33194.03507.16778		2.43%	1	00:00:01.000	1	select count(*) COUNT from USER_TABLES where T
58261.05946.62169.09058		2.43%	1	00:00:01.000	1	SELECT AUTOEXTENSIBLE,ROUND(((TOT_BYTES-S
55560.00777.00078.36110		2.43%	4	00:00:00.250	0	INSERT INTO PW_ORSN_STATEMENT_INFO(ORSN_
49275.04582.22831.33512		2.43%	1	00:00:01.000	0	SELECT IDA.INII_INAP_ID APPTIER_ID,CASE GROUP_
48450.64062.62289.25470		2.43%	1	00:00:01.000	1	select :1 as ORGS_PWII_INSTANCE_ID,u.NAME as C
43856.15379.47160.59528		2.43%	9	00:00:00.111	0	INSERT INTO PS_PWSC_SUMMARY_CONTROLLER(
43313.24225.20758.42571		2.43%	1	00:00:01.000	1	delete from ps_ordi_v_dispatcher_prev where row_
42621.20167.38271.60620		2.43%	5	00:00:00.200	0	INSERT INTO PW_ORSY_V_SYSSTAT_H(ORSY_PW
37622.42504.13682.13086		2.43%	0	N/A	0	insert into ps_orot_v_osstat_prev(database_id,time_

IDERA

IDERA.com