

ORACLE®

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

SQL Performance Tuning in the Autonomous Database Era

Andrew Holdsworth
Christine Qu
Cary Dong

Real-World Performance

November 17, 2017

ORACLE

ORACLE
REAL-WORLD PERFORMANCE

Introductions



Introductions

Andrew Holdsworth

- 28 Years at Oracle
- Vice President Real World Performance
 - Good performance is rarely an accident
 - Most people get the systems they deserve
 - Good enough rarely is, aspire for excellence not good enough.



Introductions

曲卓 (Christine Qu)

- 12 Years at Oracle
- Manage Real-World Performance education in China
- Learn to analysis from top down, make sure you are on the right direction
- Be open and positive, aim high



Introductions

董志平(Cary Dong)

- 15 years of Oracle experience
- 9 years in RWP
- Manage Real-World Performance projects in China



Real-World Performance

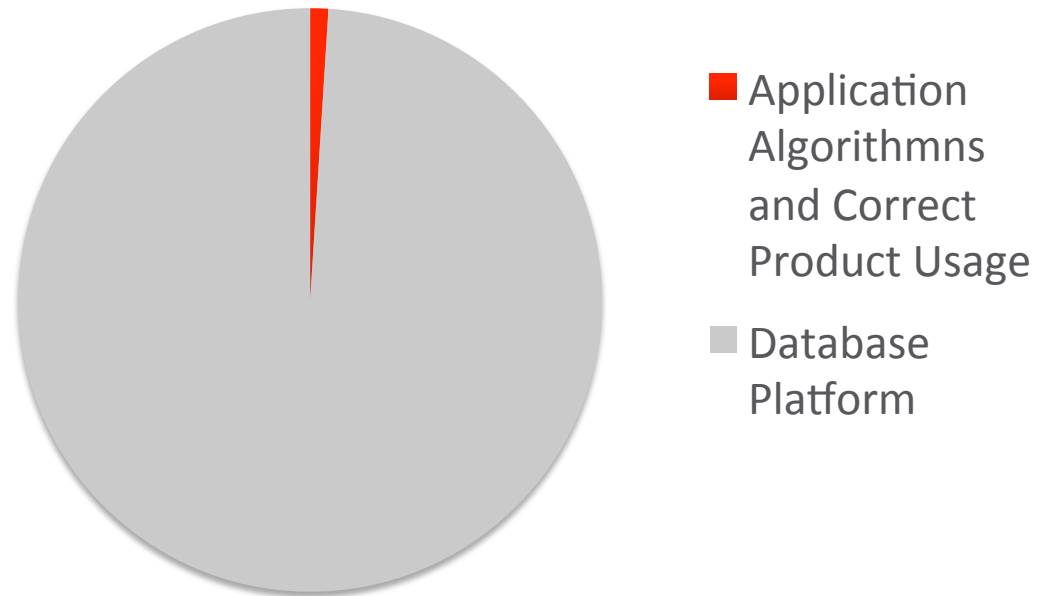
Who We Are

- Part of the Database Development Organization
- Global Team located in USA, Europe, Asia
- 350+ combined years of Oracle database experience
- Innovate to achieve exceptional Database Performance
- Our methods:
 - Use the product as it was designed to be used
 - Numerical and logical debugging techniques
 - Educate others about the best performance methods and techniques
 - Avoid and eliminate “tuning” by hacking/guessing/luck

The Real World Performance Perception Problem

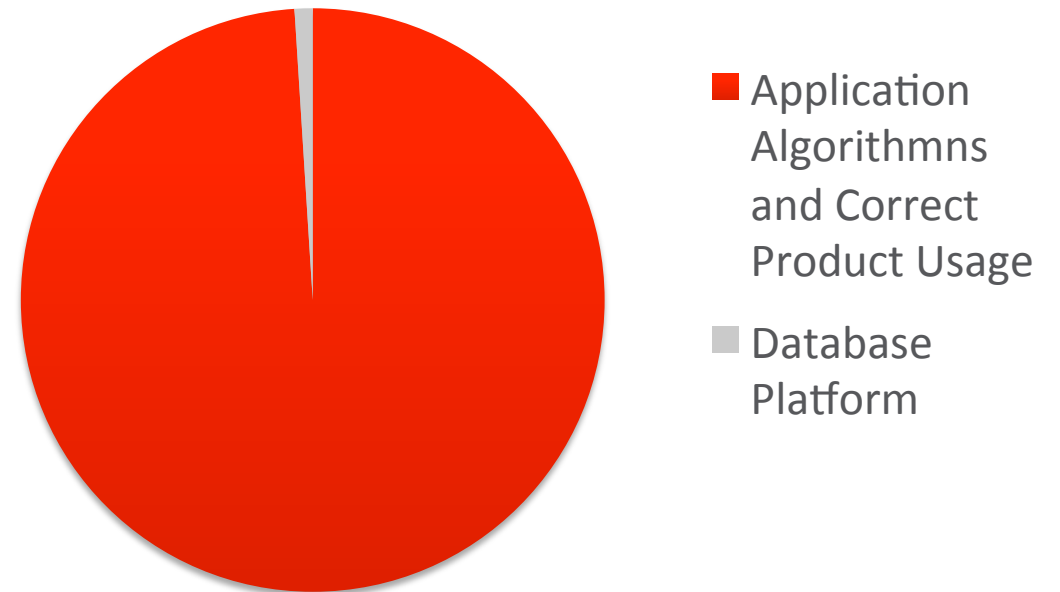
Where database user look for performance improvements

Perception



The best place to look for performance Improvements

Reality



Program Agenda

- 1 Why Autonomous Database?
- 2 SQL Performance Tuning Strategy
- 3 Makes Tuning Smarter
- 4 Panel

Program Agenda

- 1 Why Autonomous Database?
- 2 SQL Performance Tuning Strategy
- 3 Makes Tuning Smarter
- 4 Panel

Oracle Vision for Autonomous Database

Goal - Eliminate all human labor

No human labor means lower cost

No human error means better reliability and security

Oracle's Vision for Autonomous Database

How we do it

- **Self-Driving**
 - User defines workloads and policies, database makes them happen
- **Self-Securing**
 - Protection from both external attacks and internal users
- **Self-Repairing**
 - Automated protection from all downtime

Automated vs. Autonomous

Automated

- The car simplifies operations by automating tasks:
 - Cruise control
 - Emergency stopping
 - Warnings for lane changes
- The database simplifies operations:
 - Automatic storage management, automatic storage management, ...
 - Dozens of other features

Autonomous

- The car drives itself
 - No need to use the steering wheel or brake.
 - Simply tell the car where you are going.
- The database manages itself
 - All features automatically implemented
 - Simply tell the database your goals

Automatically Diagnoses Performance

- Autonomous Database includes Oracle's industry leading diagnostics automation
- Automatic Database Diagnostic Monitor (ADDM)
 - Automatically diagnoses root cause of performance issues
 - A.I. (Expert System)
- Active Workload Repository (AWR)
 - Automatically keeps detailed performance and resource utilization history
- Real-Time SQL Monitoring
 - Automatically diagnoses how resources are used in SQL statements

Automatically Optimizes Itself

- Autonomous Database includes Oracle's industry leading database tuning automation
- Many database algorithms self optimize – caching, locking, storage indexes, offload, etc.
- Optimizer is now further automated by gathering statistics as new data is loaded
- Automatic SQL (Re)Tuning
 - Machine learning technology that is constantly re-evaluating SQL plans based on the latest statistics and recommending/implementing better plans
- Tuning is workload dependent – e.g. OLTP vs analytics so we specialize services
- Tuning is an extremely difficult problem
 - Even scheduling a fleet of trucks to optimally make deliveries is incredibly complex
 - Database has many degrees of freedom and tradeoffs that must be considered

Database Administrator Questions and Fears

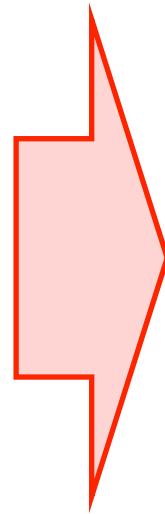
- Will my job go away?
- Will my job change?
- Will I lose control?



What does Autonomous Database mean for the DBA?

Less time on Administration

- Less time on infrastructure
- Less time on patching, upgrades
- Less time on ensuring availability
- Less time on tuning



More time on Innovation

- More time on database design
- More time on developing new apps
- More time on data analytics
- More time on securing data

Challenge: There are more data management tasks than humans to do the work

Reality vs. Fears

- Your job will not go away - there is a **shortage of skilled database experts**
 - Database automation has been improving for decades
- But your job will change, so you must **change**
- You will spend less time on **generic maintenance**, more time **innovating**
- More time with the **business**
 - Executing more projects, reducing backlog, getting more value from data
 - Cloud's fast provisioning and pay-as-you-go enables rapid experimentation
- More time with **developers**
 - Optimizing data access, improving end-user experience
- More time on new techniques like **Machine Learning**

Database Administrator Must Ensure End-to-End Service Levels

- Focus on optimizing how applications and databases work together:
 - Define good data model, and good SQL
 - Avoid row at a time processing, and repeated logins/parsing
 - Understand where time is being spent
 - Understand tradeoffs in parallelism, plans, indexes, partitions, etc.
 - Ensure that **sensitive data is kept secure end-to-end**
 - **Beware what an application asks for – the database will do exactly that**
- Need new skills in Cloud Systems, Cloud Networking, Cloud Storage
 - DBA sizes, monitors, ensures full stack works as expected
- Database Administrator will gain more control
 - Will be in charge of, and in control of, end-to-end service levels

Program Agenda

- 1 Why Autonomous Database?
- 2 SQL Performance Tuning Strategy
- 3 Makes Tuning Smarter
- 4 Panel

SQL Performance

- Elapsed time: 1 hour → 1 min
 - Will you stop working?
- Could it be possible to make it down to 1 sec?

Aim High

SQL Performance

- Is it a valid SQL?
- Do you know the business logic the SQL represents for?
- Is it a well constructed SQL?
- Any mistakes in the SQL?
 - N-1 join conditions for a N table join?
 - Implicit data type conversion?

Aim High

Work on Good SQL

SQL Performance

- Is your database software correctly patched?
- Are you running with default init.ora parameter settings? If not, why?

Aim High

Work on Good SQL

Environment settings

SQL Performance

- Stats
 - Stats on raw data
 - System stats
- Constraints
 - NOT NULL, PK, FK, UK
- Schema design
 - Index
 - Partitioning
 - Compression
 - Clustering

Aim High

Work on Good SQL

Environment settings

Optimization

SQL Performance

- Access method
- Join method
- Join order
- Distribution method
- Skew?

Aim High

Work on Good SQL

Environment settings

Optimization

Execution

SQL Performance

- Diagnosing
 - SQL Monitor Report
 - Find the leverage



Aim High

Work on Good SQL

Environment settings

Optimization

Execution

Program Agenda

- 1 Why Autonomous Database?
- 2 SQL Performance Tuning Strategy
- 3 Makes Tuning Smarter
- 4 Panel

Introduction to SQL Monitor

- Released in Oracle Database 11g
- Enables in depth performance monitoring of a SQL statement
 - Always on, enabled out of the box
 - Single execution of the SQL statement
 - Includes currently executing statements
- Monitored statements
 - Serial statements with 5 seconds of total CPU/IO time
 - All parallel statements
 - /*+ monitor */ hint
 - Queries / DML / DDL

Introduction to SQL Monitor

- Formats available
 - Text
 - HTML
 - Active (use this one!)
- Available from
 - Monitored SQL screen
 - command line
 - EM, EM Express
 - PerfHub

How we use the SQL Monitor report

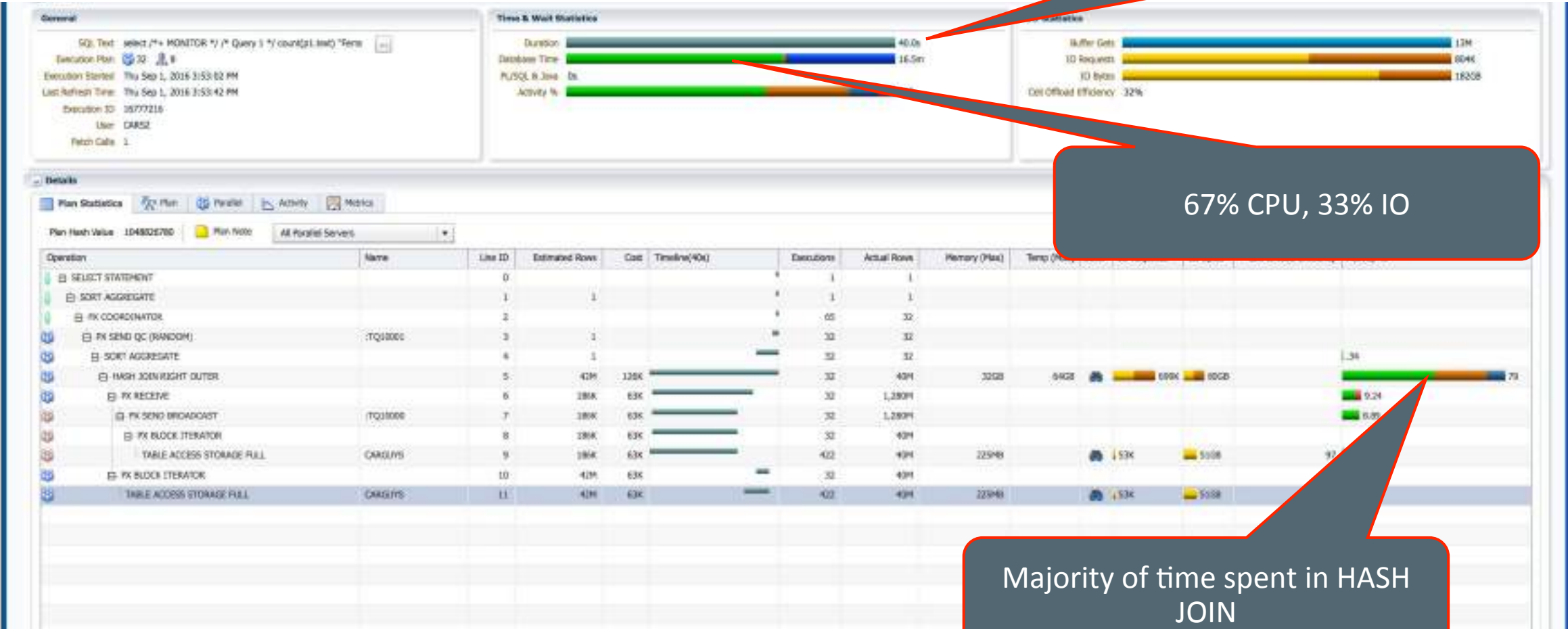
- Top down
 - Where is time spent?
 - Which row sources?
 - Estimated vs actual row cardinalities?
 - Executions
 - Parallel server executions
 - Nested loop iterations
 - Partition wise operations
 - Skew

Example 1

Observations

- Query currently runs for 40 seconds
- Needs to run under 5 seconds
- Examination of the SQL Monitor report shows
 - 67% CPU
 - 33% IO
- The most expensive row source is HASH JOIN RIGHT OUTER (line 5)
 - 45% of the CPU
 - Large amount of read/write from TEMP

Example 1: Observations



Duration 40s

67% CPU, 33% IO

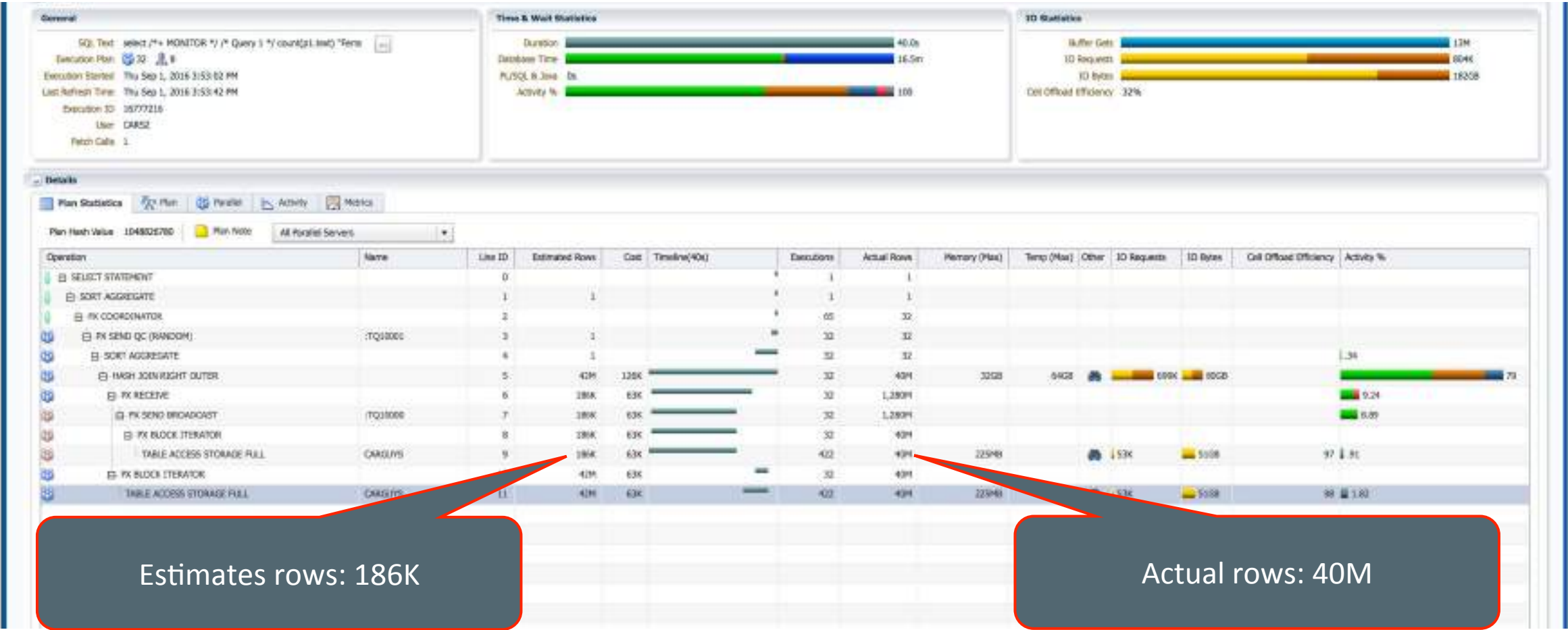
Majority of time spent in HASH JOIN

Example 1

Diagnosis

- However...
 - Is the HASH JOIN itself the problem?
- Examination of the Cardinality estimates
 - The scan of the CARGUYS table at line 11
 - estimate of 42M rows vs actual 40M rows
 - estimate is very accurate
 - The scan of the CARGUYS table at line 9
 - estimate of 186K rows vs actual 40M
 - more that 200x underestimated

Example 1: Diagnosis



Example 1

Diagnosis

- What is the cause of the big mis-estimate in cardinality?
 - Examining the predicate information:

Filter Predicates ("MODEL"='458 Italia' AND "MAKE"='Ferrari' AND "COUNTRY"='Italy')

- All three filter predicates are highly correlated
 - Ferrari's are only made in Italy
 - Only Ferrari makes a model called the '458 Italia'
- The optimizer has multiplied the selectivity of the individual predicates and underestimated the number of rows that will be retrieved

Example 1: Diagnosis

The screenshot displays the Oracle SQL Developer interface. At the top, it shows 'User: CARSL' and 'Fetch Calls: 1'. Below this, the 'Details' pane is active, showing a query plan for Plan Hash Value 1048026790. The plan includes several operations, with 'TABLE ACCESS STORAGE FULL' on the table 'CARGLYS' highlighted in blue. A callout box with a red border and a pointer to the highlighted row contains the text 'Highly correlated predicates'. A pop-up window titled 'TABLE ACCESS STORAGE FULL' provides detailed statistics for this operation.

Operation	Object	Line ID	Predicate	Pruning	Operation Cost	Estimated Rows	Estimated Bytes
SELECT STATEMENT		0					
SORT AGGREGATE		1					
PX COORDINATOR		2					
PX SEND QC (RANDOM)	:TQ10001	3					
SORT AGGREGATE		4					
HASH JOIN RIGHT OUTER		5					2,067M
PX RECEIVE		6				190K	7,813K
PX SEND BROADCAST	:TQ10000	7				190K	7,813K
PX BLOCK ITERATOR		8		1 - 13		190K	7,813K
TABLE ACCESS STORAGE FULL	CARGLYS	9		1 - 13	63K / 95%	63K / 95%	7,813K
PX BLOCK ITERATOR						62M	917M
TABLE ACCESS STORAGE FULL					63K	93M	917M

TABLE ACCESS STORAGE FULL

- Sub-tree Cost: 63K (41% CPU)
- Operation Cost: 63K (41% CPU)
- Sub-tree Time: 00:00:03
- Rows: 188K
- Bytes: 7,813K
- Object: CARGLYS
- Access Predicates: <math>2 >= 2 \text{ AND } 2 <= 2 \text{ AND } ('MODEL' = '452 Italia' \text{ AND } 'MAKE' = 'Ferrari' \text{ AND } 'COUNTRY' = 'Italy')</math>
- Filter Predicates: ('MODEL' = '452 Italia' \text{ AND } 'MAKE' = 'Ferrari' \text{ AND } 'COUNTRY' = 'Italy')
- Partition Start: 1
- Partition Stop: 13
- Query Block Name/Object Alias: SEL\$ASCF08M/DAGL\$YS08L\$1

Example 1

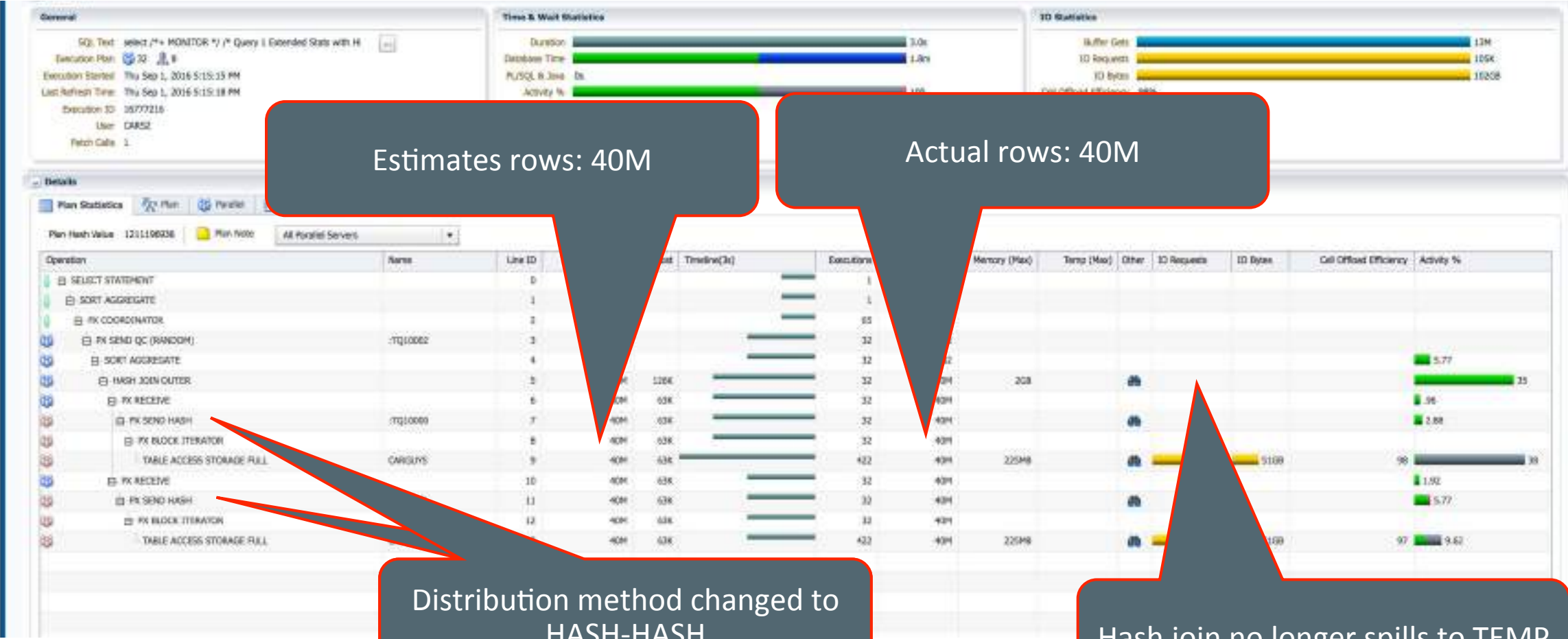
- Solution

- Use extended statistics
 - Build a column group on MAKE, MODEL, COUNTRY
- The optimizer can now determine the selectivity of the three predicates combined

- Result

- The cardinality estimate has now changed to 40M rows.
- Changes distribution method from BROADCAST to HASH-HASH
- This results in a more efficient HASH JOIN, which does not spill to TEMP
- Query now runs in 3 seconds.

Example 1: Solution



Estimates rows: 40M

Actual rows: 40M

Distribution method changed to HASH-HASH

Hash join no longer spills to TEMP

Example 2

Observations

- Query takes 42 minutes
 - Almost all CPU
- Complex SQL statement
- Parallel execution requested
- Time spent in nested full table scans
- Optimizer estimated it would perform one table scan
 - Actually performed 5776 table scans

Example 2: Overview

The screenshot displays the Oracle Enterprise Manager interface for 'Monitored SQL Execution Details'. The 'Overview' section shows the following information:

- SQL ID: 1F50K37Y30
- Parallel: 8
- Execution Started: Wed Mar 23 2016 12:15:25 PM
- Last Refresh Time: Wed Mar 23 2016 12:54:39 PM
- Execution ID: 14778188
- User: APLIAP
- Fetch Calls: 0

The 'Time & Wait Statistics' section shows a bar chart for Duration (41.2m) and Database Time (41.2m). The 'IO Statistics' section shows Buffer Gets (9 GB), IO Requests (4), and IO Bytes (32GB).

The 'Details' section shows the execution plan with the following operations:

Operation	Time (s)	CPU	IO	Wait
INSERT STATEMENT				
LOAD TABLE CONVENTIONAL				
PX COORDINATOR				
PX SEND QC (RANDOM)	5.051	1	0	0
WINDOW SORT	5.051	1	0	0
PX RECEIVE	5.049	1	0	0
PX SEND HASH	5.049	1	0	0
HASH JOIN OUTER	5.049	1	0	0
PX RECEIVE	5.022	1	0	0
PX SEND HASH	5.022	1	0	0
HASH JOIN OUTER	5.023	1	0	0
PX RECEIVE	5.784	1	0	0
PX SEND HASH	5.784	1	0	0
VIEW	5.764	1	0	0
HASH JOIN BUFFERED	5.764	1	0	0
PX RECEIVE	5.708	1	0	0
PX SEND HASH	5.706	1	0	0
HASH JOIN BUFFERED	5.706	1	0	0
BUFFER SORT				
PX RECEIVE	5.647	1	0	0
PX SEND HASH	5.647	1	1	327

Annotations on the screenshot:

- A red circle highlights the 'Parallel: 8' value in the Overview section.
- A red circle highlights the 'Duration: 41.2m' and 'Database Time: 41.2m' bars in the Time & Wait Statistics section.
- A red circle highlights the 'PX COORDINATOR' operation in the execution plan.
- A red circle highlights the 'PX SEND QC (RANDOM)' operation in the execution plan.

Callout boxes provide the following information:

- Requested Parallel:** Points to the 'Parallel: 8' value.
- Duration 41.2 minutes All CPU:** Points to the 'Duration' and 'Database Time' bars.
- Executed serially:** Points to the 'PX SEND QC (RANDOM)' operation.

Example 2: SQL

The screenshot displays the Oracle Enterprise Manager interface for monitoring SQL execution. On the left, the 'Overview' section shows the SQL ID '1f89537314', execution status 'Parallel', and start time 'Wed May 4, 2017 16:78:00'. The 'Details' section shows the execution plan with steps like 'INSERT STATEMENT', 'LOAD TABLE CONVENTIONAL', and 'PK COORDINATOR'. The main area shows the SQL text, which is a complex query involving multiple joins and subqueries. A callout box with a red border and a pointer highlights the SQL text, containing the text 'Complex SQL statement'. On the right, the 'IO Statistics' section shows 'Buffer Gets' at 96M, 'IO Requests' at 4, and 'IO Bytes' at 2040. Below this, a table shows execution statistics for various steps, including 'IO Requests' and 'CPU Activity %'.

Complex SQL statement

Step	Estimate	Actual	Temp (M)	IO Requests	CPU Activity %	Wait Activity %
1	3,651					
2	3,651					
3	5,049					
4	8,849					
5	5,049		0	0		
6	3,822		0	0		
7	3,822		0	0		
8	3,822		0	0		
9	3,764		0	0		
10	5,764		0	0		
11	3,764		0	0		
12	3,764		0	0		
13	3,706		0	0		
14	5,706		0	0		
15	3,706		0	0		
16	3,647		0	0		
17	5,647		1	337		

Example 2: Plan Statistics screen shows estimate and actual

Operation	Cost	Cardinality	Bytes	Temp Space	IO
PK COORDINATOR		1	1,110		
PK SEND QC (RANDOM)	5.681	0			
WINDOW SORT	5.681	0			
PK RECEIVE	5.649	0			
PK SEND HASH	5.649	0			
HASH JOIN OUTER BUFFERED	5.649	0			
PK RECEIVE	5.623	0			
PK SEND HASH	5.623	0			
HASH JOIN OUTER BUFFERED	5.623	0			
PK RECEIVE	5.764	0			
PK SEND HASH	5.764	0			
VIEW	5.764	0			
HASH JOIN BUFFERED	5.764	0			
PK RECEIVE	5.705	0			
PK SEND HASH	5.705	0			
HASH JOIN BUFFERED	5.705	0			
BUFFER SORT	5.647	0			
PK RECEIVE	5.647	0			
PK SEND HASH	5.647	0			
VIEW	5.647	0			
SORT UNIQUE	5.647	0			
NESTED LOOPS	7.238	7,238			
NESTED LOOPS OUTER	5.776	5,776			
FILTER	5.172	5,172			
NESTED LOOPS OUTER	6.343	6,343			
NESTED LOOPS OUTER	6.042	6,042			
TABLE ACCESS BY GLOBAL INDEX R...	6.342	6,342			
INDEX RANGE SCAN	6.636	6,636			
MAT_VIEW ACCESS BY INDEX ROWID	5.343	5,115			
INDEX	6.343	5,115			
MAT_VIEW	6.343	3,486			
INDEX	6.343	5,629			
MAT_VIEW	5.775	5,627			
INDEX KE	5.775	5,627			
VIEW	4.212	5,775	7,238		
UNION ALL PUSHED PREDICATE	5.776	7,238			
MERGE JOIN ANTI NA	5.776	4,752			
NESTED LOOPS OUTER	5.775	7,238			

Optimizer estimate for index scan is close

But only expects 1 row after table filters

Example 2

Diagnosis

- Poor cardinality estimate caused by SUBSTR() function

Example 2: SQL Monitor Plan Screen Shows Predicates

Optimizer estimates filters are highly selective

But substring filter actually removes few rows

Operation	Sub-Plan	Sub-Plan Cost	Sub-Plan CPU	Rows	Bytes	Object
TABLE ACCESS BY GLOBAL INDEX ROWID	4.474	(02% CPU)	4,418	(02% CPU)	00.00.04	Rows: 1 Bytes: 45 Object: LWRN_GLNAMES

Filter Predicates: ('GLN'.GLNSET3_05_SAF='N' AND SUBSTR('GLN'.VAR_LEVEL_DISP,1,4)<>'0010')

Projection: 'GLN'.ROWID[ROWID.ID], 'GLN'.COMPANY[NUMBER.ZX], 'GLN'.VAR_LEVEL_DISP[CHARACTER.L]

Query Block Name/Object Alias: SEL00417C3A/GLN@SEL01C

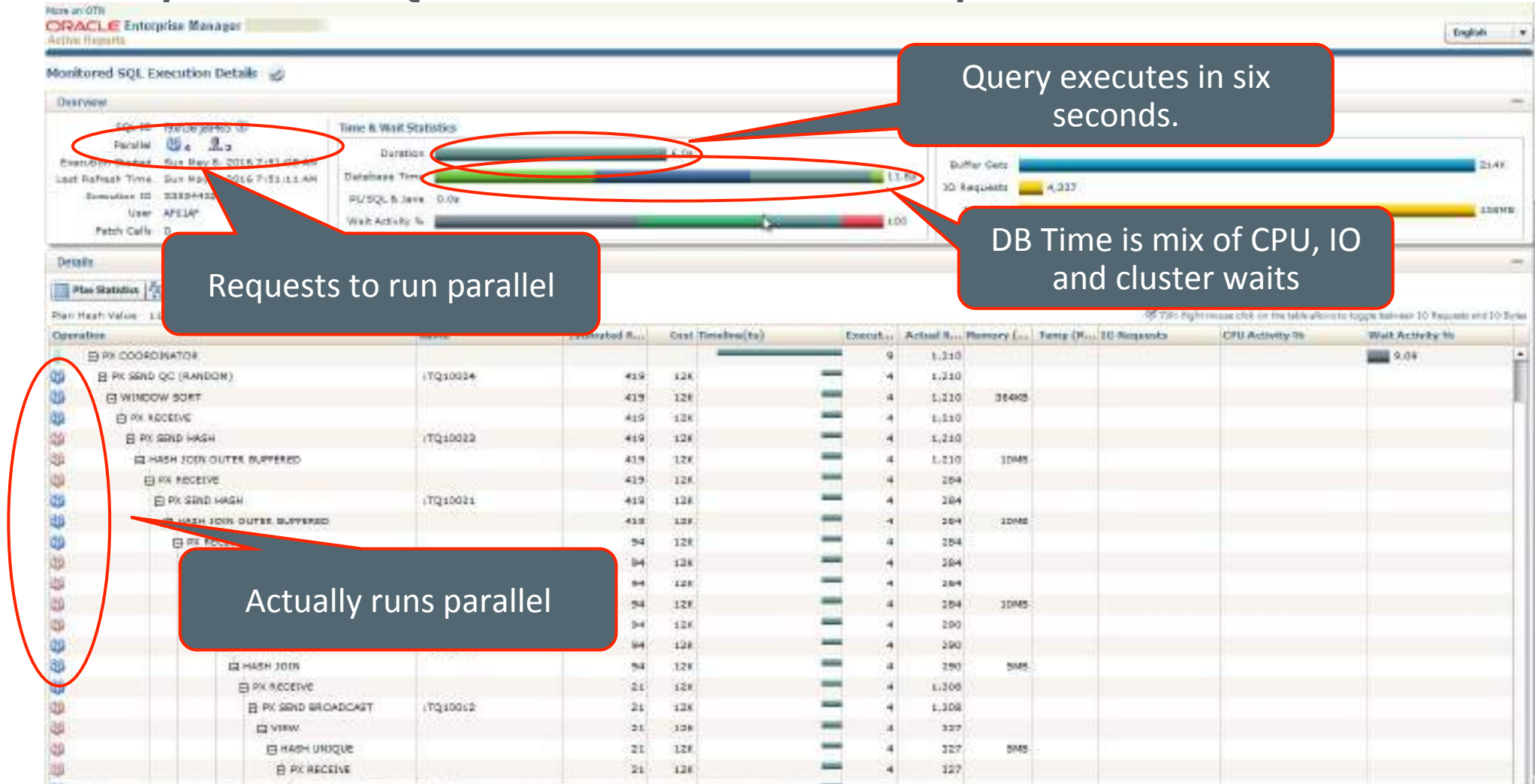


Example 2

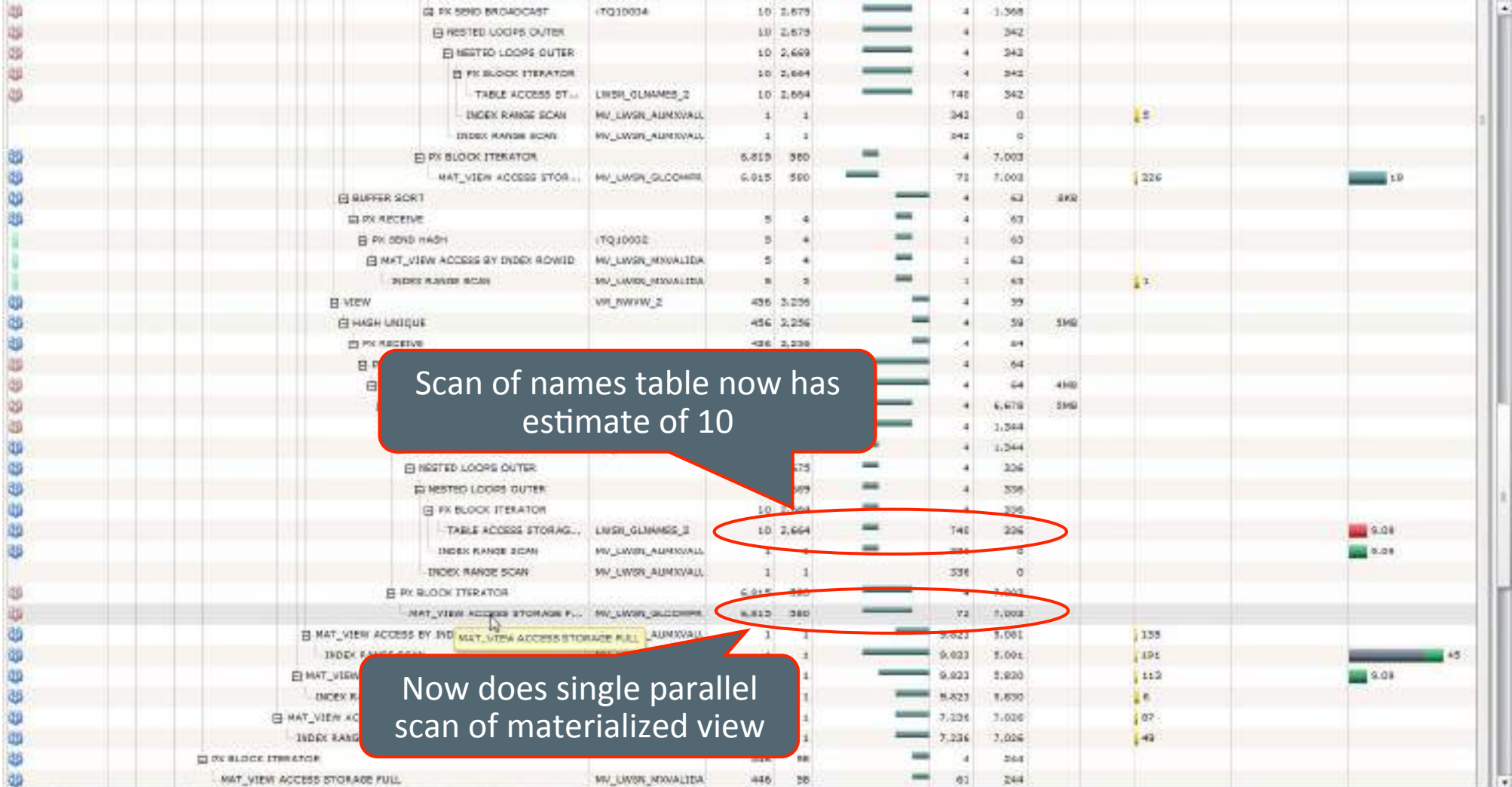
Solution

- Extended statistics on SUBSTR() function
- Optimizer now estimated 10 rows rather than 1
- Optimizer uses hash join rather than nested loop
- Accesses table with single scan
- Elapsed time: 6 seconds

Example 2: SQL Monitor After Expression Statistics Added



Example 2: SQL Monitor After Expression Statistics Added

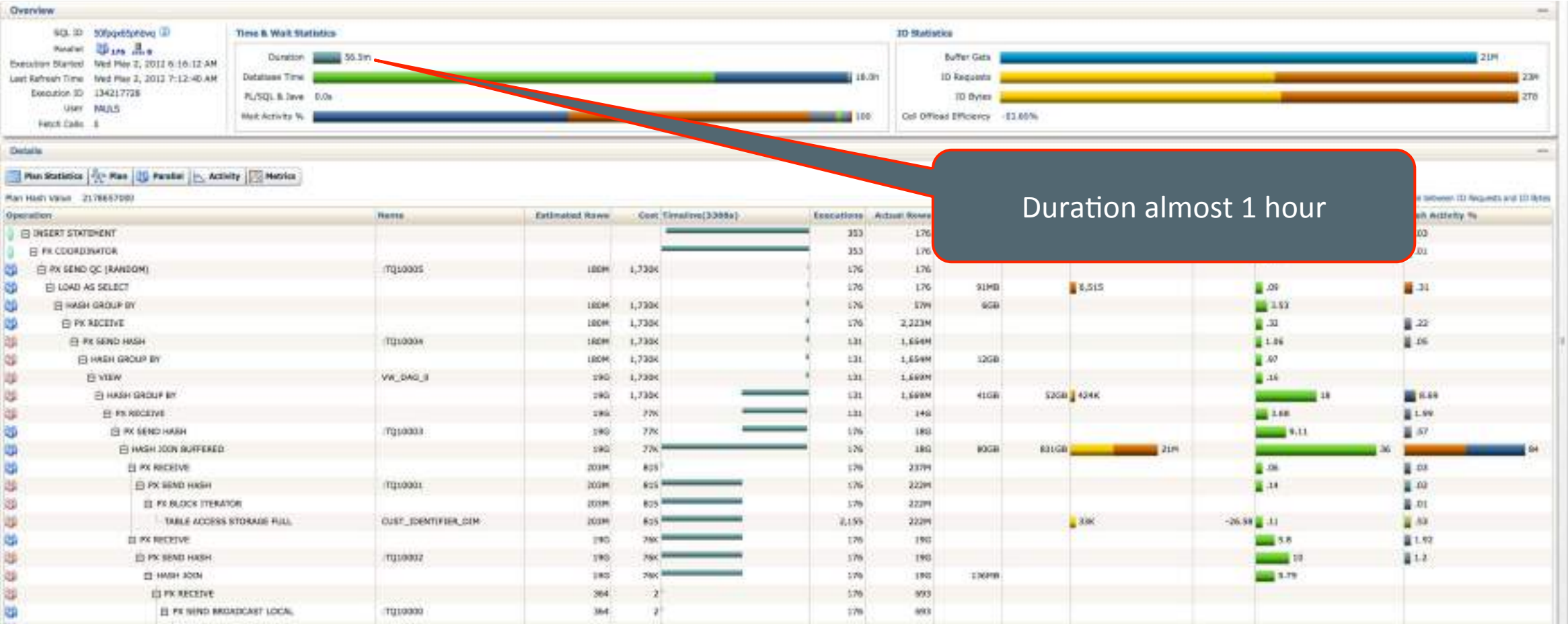


Example 3

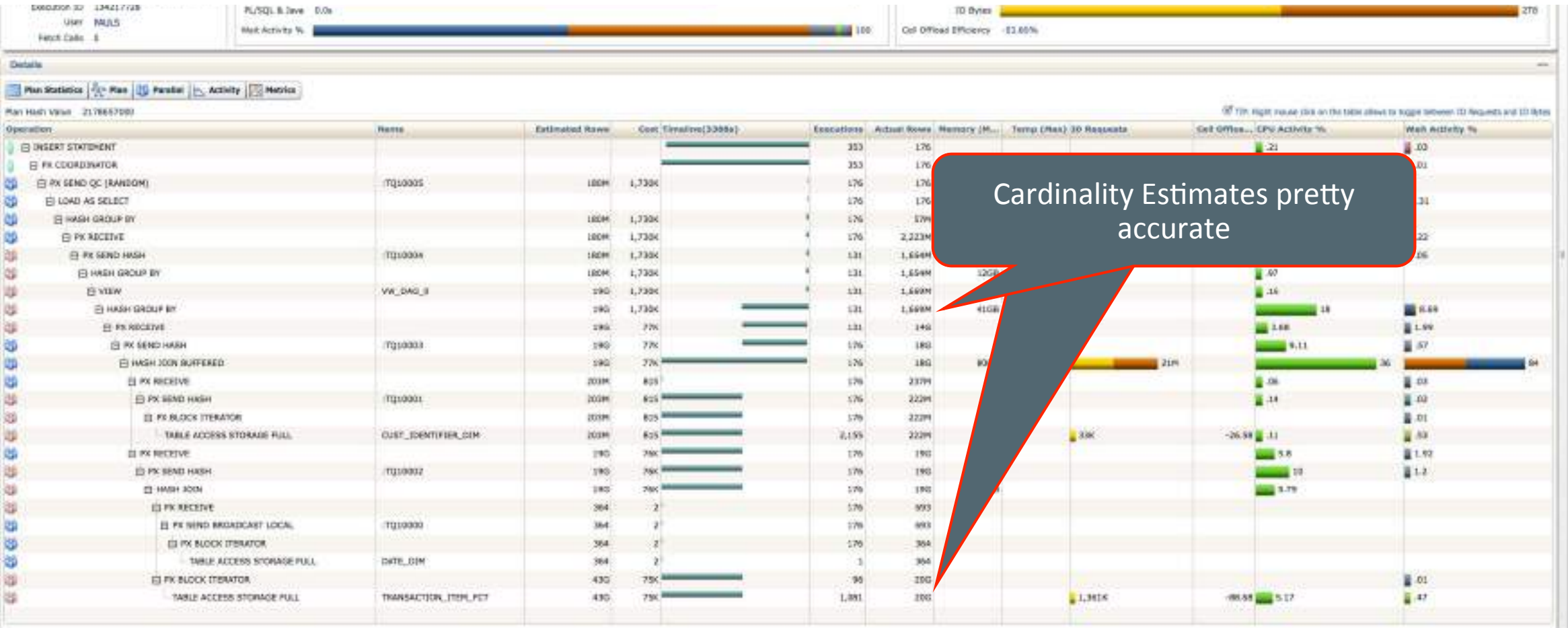
Observations

- Query takes 56 minutes
- Time spent in HASH JOIN BUFFERED
- Cardinality estimates are accurate
- Large amount of TEMP IO
- Table Scan of 1TB table ran for 1800's.
 - Scan rate for the platform is much higher than that
 - Scan is constrained by the buffering of the HASH JOIN

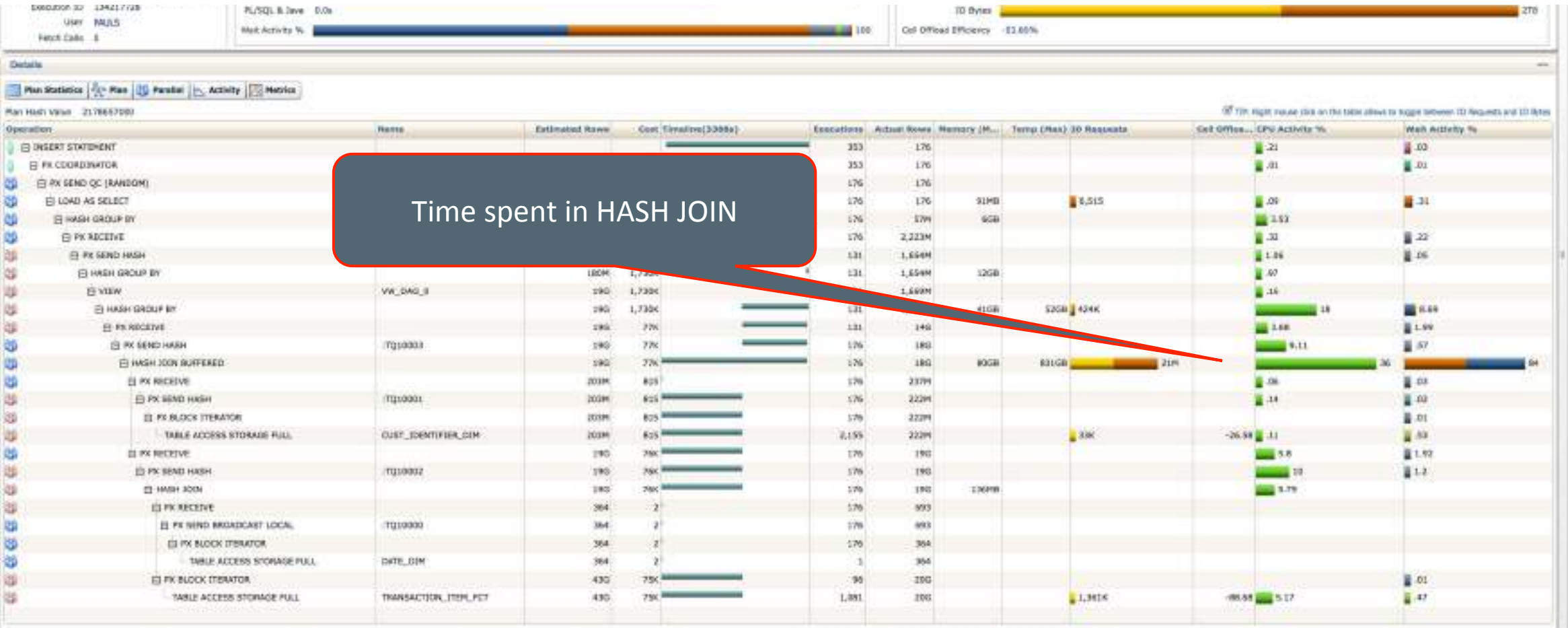
Example 3: Observations



Example 3: Observations



Example 3: Observations

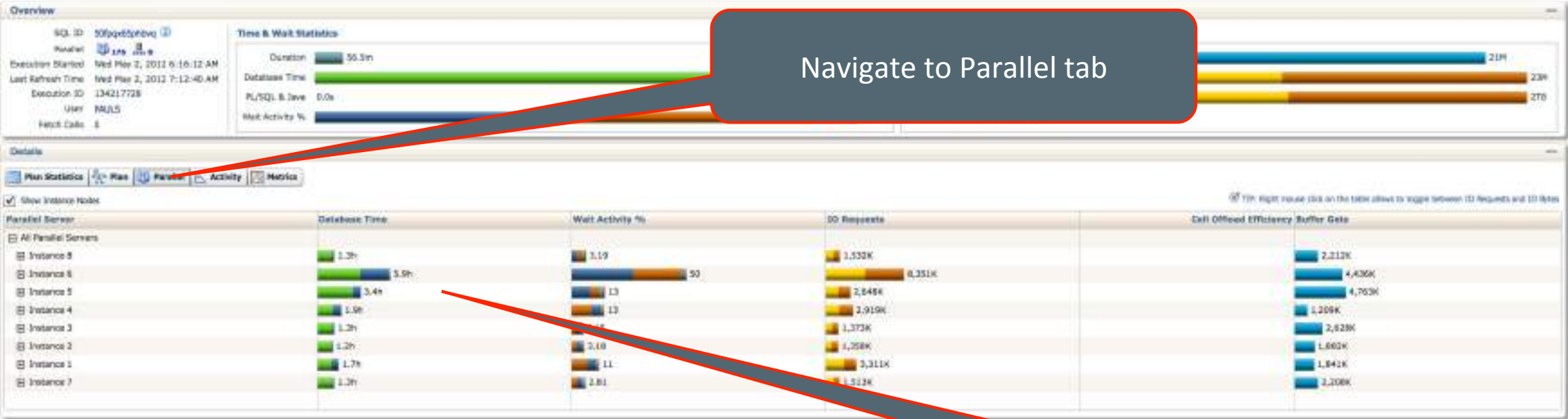


Example 3

Diagnosis

- Examining the Parallel Tab
 - Database time for PX servers on instance 6 is much higher
 - Expanding the Instance 6 node and expanding Parallel Set 1, we observe one PX server with a DB Time of 53 mins
 - A large part of the query was executed by a single PX server.

Example 3: Diagnosis

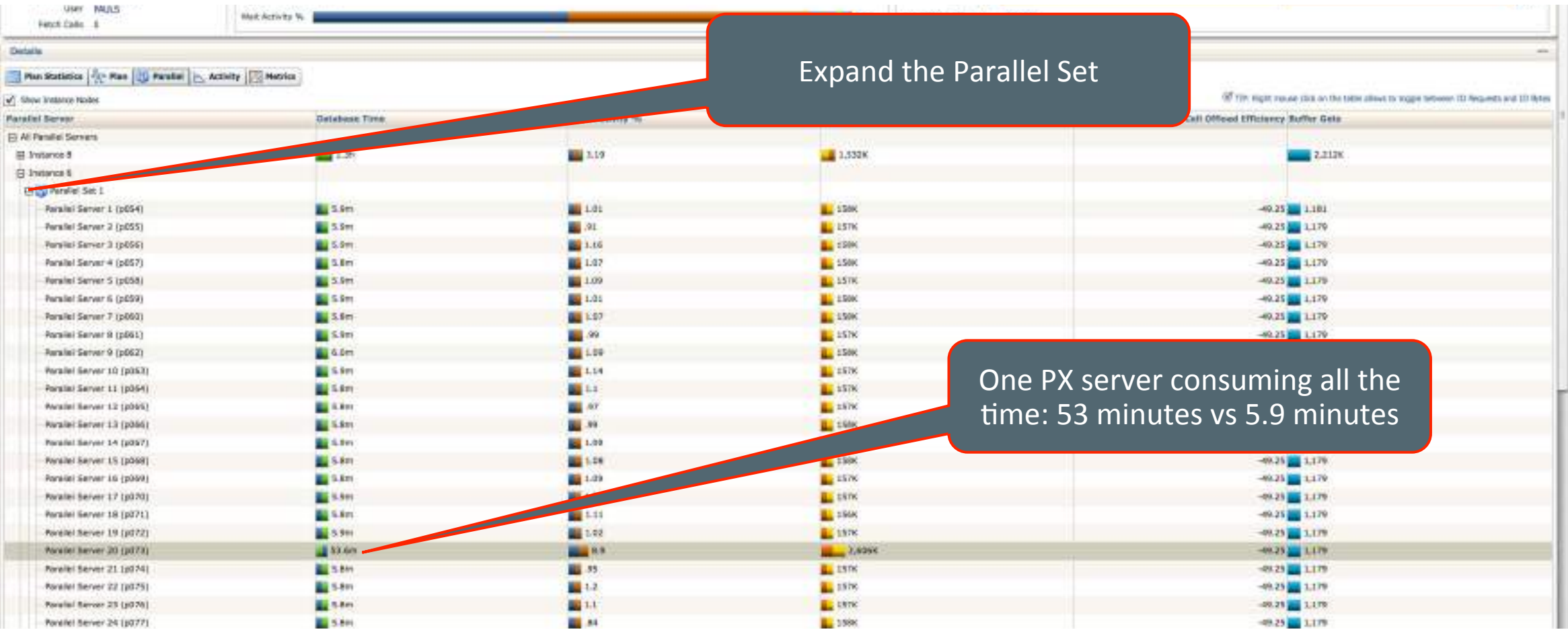


Navigate to Parallel tab

Skew in database time

Copyright © 1996, 2016, Oracle and/or its affiliates. All rights reserved. Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

Example 3: Diagnosis



Expand the Parallel Set

One PX server consuming all the time: 53 minutes vs 5.9 minutes

Example 3

Diagnosis

- Some domain knowledge was applied to this problem
 - The application supports loyalty cards
 - Majority of the customers ids are unknown or not identifiable
 - Implemented as a userID = -1
 - This value from the CUST_IDENTIFIER_DIM table hashes to a single hash bucket and so the HASH JOIN for that value is executed by a single PX slave

Example 3

Diagnosis

- Some domain knowledge was applied to this problem
 - The application supports loyalty cards
 - Majority of the customers ids are unknown or not identifiable
 - Implemented as a UserID = -1
 - UserID= -1 hashes to a single hash bucket
 - HASH JOIN for UserID= -1 is executed by a single PX slave

Example 3

Solution(s)

- Have the unknown customers be identified by a negative sequence number
- Run two separate queries
 - A query for known customers, ie `CUST_IDENTIFIER_ID > 0`
 - This query executes with HASH HASH distribution, like the original
 - A query for unknown customers `CUST_IDENTIFIER_ID = -1`
 - This query executes with a BROADCAST distribution
- Similar to above
 - Write the query as a UNION-ALL of the above
- In 12c, SKEW detection may generate a HYBRID plan that will automatically broadcast popular values, and Hash distribute non-popular values

SQL Monitor gotcha's

- Beware of cardinality estimates of 1 except for unique index lookup
- Execution count
 - Parallel operations
 - Partitioned operations
 - Nested loop
 - Or combination!
- For Nested loop plan
 - Estimated rows is for a single execution
 - Actual rows is for all executions (so far)
- Parallel server elapsed time bars not a good indication of time
- Fixed size in memory buffer means statements age out
- Plan lines limit defaults to 300
 - `_sqlmon_max_planlines` to change

SQL Monitor Conclusions

SQL Monitor is the best tool to understand why SQL is running slowly

- Always on
- Detailed data on where the time is going
- Shows cardinality estimates vs actuals
- Top SQL captured into AWR in 12c

Program Agenda

- 1 Why Autonomous Database?
- 2 SQL Performance Tuning Strategy
- 3 Makes Tuning Smarter
- 4 Panel

PANEL



Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Integrated Cloud

Applications & Platform Services

ORACLE®