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.
Advanced Database Performance Analysis Using Metric Extensions and SPA

Waleed Ahmed, Oracle
John Beresniewicz, Oracle
Duane Lupinski, Lead DBA CSX
Program Agenda

- Database Performance Analysis
  - Challenges
- Advanced Use Cases
  1. Detecting Run-away Queries Using Metric Extensions
  2. Identifying High Risk SQL in Growing Data Volume Environment
  3. Upgrading Oracle E-Business Suite Using SQL Performance Analyzer
- Appendix
Database Performance Analysis

Challenges

Find?
- Reported by end users
- Discovered by daily checks
- Reactive to end-users

Fix?
- Relate symptom to possible cause
- Relate possible cause to probable metric
- Implement change

Validate?
- Did it work?
- End-user feedback?
- Run reports on metric

Track?
- Will it occur again?
- Custom script to watch for the metric
- Complex alerting and notification
1 – Find: Diagnostics

- For problem diagnosis use ADDM, EM Performance

- ADDM reveals significant problems including problematic SQL details and recommendations improve to performance

- ADDM family – Compare Period, Real-Time ADDM
2 – Fix: Tuning

- Seamless integration between Find and Fix
- In-depth analysis and recommendation of the fix
- Gather statistics for this example...
- How would we validate this changes?
3 – Validate: Real Application Testing

SQL Performance Analyzer (SPA)

- Test and predict impact of system changes on SQL query performance
- Analyze performance changes for improvements and regressions
- Comprehensive performance analysis and reporting
- Re-execute SQL queries in the given environment
- End-to-end solution: STS, SQL Plan Baselines, and SQL Tuning Advisor
3 – Validate: SPA

- End-to-end SPA workflow using Optimizer Statistics
- Seamless integration from Find to Fix to Validate to Implement with confidence
- No more guesswork
- Reference OOW 2011 slides here
3 – Validate: When to use SPA?

SPA Provides Broad Testing Coverage

- Optimizer Statistics
- Database parameter changes
- Database schema changes (e.g., add/drop indexes)
- Implementation of tuning recommendations
- I/O subsystem changes (e.g., ASM, Database Machine)
- For consolidation (schema, CDB/PDB)
- On test, standby, and production environments
- Extended to home-grown scripts, application specific database changes, third-party testing tools using STS compare analysis
- Supports most applications - EBS, SAP, Siebel, home-grown, etc

SPA Provides Testing in Various Database Releases

- My Oracle Support (MOS) Note: 560977.1
- 9.2/10.1 → 10.2 or 11g or 12 releases
- 10.2.0.x → 10.2.0.y or 11g or 12 releases
4 – Track: Monitoring in Enterprise Manager (EM)

- Lights-out data center monitoring
  - Manage by exception
  - Notifications
  - Integrated with My Oracle Support

- Complete and integrated across stack
  - Entire Oracle stack
  - Heterogeneous infrastructure monitoring via plug-ins
  - Extensible

- Integrates with third-party systems
  - Helpdesks and other management systems

- Metric Extensions (12c)
  - Next generation User Defined Metrics (UDM)
  - Can be defined for any target type, including applications
4 – Track: Metric Extensions

EM Metrics for Application Monitoring

Problem

- Application specific database events that may cause adverse business impact, examples:
  - Sudden burst of orders increases application order queues
  - Application specific locking in database resulting in sessions waiting

Solution

- Utilize Metric Extensions to extend Enterprise Manager monitoring
- Metric Extension periodically monitors application orders queue length in the database
- Any violations of thresholds triggers an incident
4 – Track: Orders Queue Length Metric Extension

- Create a Metric Extension that monitors ORDERS_QUEUE table
- Set Warning and Critical limits for New and Processing status
- Set Incident Rules – notify when warning or critical limits are violated

SQL Query:
```sql
SELECT decode(status, 1, 'NEW', 2, 'PROCESSING', 3, 'COMPLETE') as STATUS, count(status) as ORDER_STATUS
FROM oe.order_queue
GROUP BY status
HAVING STATUS IN (1, 2)
```

Do not hallucinate.
4 – Track: Monitoring

- Your custom metric integrated natively into OOB metrics
- Historical metric view in one pane shows past trend of orders queues, stored in EM repository
- Alert history summary with actions taken
- Email notification sent when violations occur
Program Agenda

- Database Performance Analysis
  - Challenges

- Advanced Use Cases
  1. Detecting Run-away Queries Using Metric Extensions
  2. Identifying High Risk SQL in Growing Data Volume Environment
  3. Upgrading Oracle E-Business Suite Using SQL Performance Analyzer

- Appendix
Case Study 1: Detecting Run-Away Queries Using Metric Extensions

John Beresniewicz, Oracle
What are EM 12c Metric Extensions?

- User extensions to EM target monitoring metrics
  - Replacement for EM 10/11 User Defined Metrics (UDM)
- First-class EM Metrics: Metric Group of Metric Columns
  - Multiple measure columns with alert threshold capability
  - Key column support, including key-specific thresholds
- Supports SQL, OS, and JMX fetchlets
  - Whatever the target type already uses
- Lifecycle support: Develop, Test, Deploy
Metric Extension (ME) Lifecycle

- Develop and unit test privately
- Deploy and system test on actual targets
- Publish for general use in EM
- Export/import to propagate to other EM environments
Find Metric Extensions
Manage Metric Extensions

Import Metric Extension

- File Name
- Name ME$

Actions
- Create
- Create Like
- View
- Edit
- Create Next Version
- Delete
- Manage Access
- Import
- Export
- Deploy To Targets
- Manage Target Deployments
- View Template Associations
- Save As Deployable Draft
- Publish Metric Extension
“Runaway Query”

- SQL that consumes too much CPU or executes too long
  - BUGS: Need to find during Development and Testing contexts
- Real-time SQL Monitoring introduced in DB 11g
  - Monitors long-running and parallel query (PQ) SQL executions
  - Rich interactive user interface and Active Report
- GV$SQL_MONITOR
  - Not really intended for user-written queries
Design Metric Extension

- **GV$SQL_MONITOR query**
  - **SQL_ID**: key column
  - **TotalCPUSecs**
  - **TotalElapsedSecs**
- Execute every 15-minutes (to not miss anything)
- Set alert thresholds on CPU and Elapsed times
Create Metric Extension

- Guided wizard makes it easy
  - Well organized
  - Excellent explanations

- Develop and test on any database before deploying to production
Monitoring

- Email notification with custom subject line
- Informative email with context-sensitive hyperlinks into EM
- Your metric is a first-class EM metric
Metric Extensions are Cool!

- First-class metrics of your own design
- Easy-to-use user interface makes it a snap
- Lifecycle support and export/import important value-adds
- Monitor targets for application-specific issues
  - OOB metrics are for the general case
- RunawaySQL ME has found many bugs
  - Using EM to improve EM
Program Agenda

- Database Performance Analysis
  - Challenges

- Advanced Use Cases
  1. Detecting Run-away Queries Using Metric Extensions
  2. Identifying High Risk SQL in Growing Data Volume Environment
  3. Upgrading Oracle E-Business Suite Using SQL Performance Analyzer

- Appendix
Case Study 2: Identifying High Risk SQL in Growing Data Volume Environment
Identify High Risk SQL with Growing Data Volumes

**Problem**
- Data stores and sources are expanding at a rate greater than ever
- The fast increase of data volume commonly results in performance slowdown
- Performance slow down can lead to unplanned downtime

**Reactive Fix**
- Adjust system or db configuration
- Tune storage system or SQL reactively
- Provision more hardware
- Reduce or restrict user access

**Proactive Optimization**
- Identify SQL with potential volatile plans in increasing data volume DBs
- Preempt SQL performance reliably
- Assess and plan for data growth
Identify High Risk SQL with Growing Data Volumes

1. Run SPA to establish baseline in test of a given workload*
2. Modify statistics to simulate data scale-up factor increase
3. Run SPA to identify high risk SQL
4. Periodically run high risk SQL in production to monitor performance

* Workload captured into SQL Tuning Sets (STS)
1 – SPA Baseline Trial in Test

- Setup test environment either full scale or scaled-down, data volume-wise
- Ensure table and index statistics are same as that of production
- Run SPA trials in explain plan mode for the workload
2 – Simulate Data Growth using Statistics

Backup existing table and index statistics

• Create table to hold original table and index statistics
  
  ```sql
  dbms_stats.create_stat_table()
  ```

• Export current table and index statistics to this table
  
  ```sql
  dbms_stats.export_table_stats()
  dbms_stats.export_index_stats()
  ```

Simulate scale up of data volume

• Assuming column data is uniformly distributed: …For all tables increase number of rows and blocks by a factor of 1.1 (10%) using
  
  ```sql
  dbms_stats.set_table_stats()
  ```

• For all indexes increase number of rows, number of leaf blocks, number of distinct keys and clustering factor by a factor of 1.1 (10%), using
  
  ```sql
  dbms_stats.set_index_stats()
  ```

Refer to slide notes section for example commands
3 – Run SPA Trial Analysis

- Run SPA Trial with the scaled-up optimizer statistics to assess the impact of data volume growth
- Identify High Risk SQLs due to data volume growth
4 – Periodically Test Regressed SQL in Production

- Subset Regressed SQL into a separate SQL Tuning Set

  - Create a new empty SQL Tuning Set:
    
    ```sql
    dbms_sqltune.CREATE_SQLSET()
    ```

  - Load regressed SQL into the new SQL Tuning set from the SPA task
    
    ```sql
    dbms_sqltune.load_sqlset()
    ```

- Using this STS, run SPA trials periodically in production to monitor and tune volatile SQL Plans
4 – Periodically Test Regressed SQL in Production

- On a routine basis run SPA Analysis on regressed SQLs having volatile plans
- Preempt performance degradation and fix proactively in rapidly growing data volume environments
Predict Data Growth Effect on SQL Performance

Advanced Database Performance Analysis using SPA

Using Optimizer Statistics and SPA, test incrementally, SQL Plan volatility with data volume growth

- TPCH workload containing 34 queries
- Run SPA tests at increments of 1.1, 1.5, 2, 3, 4 and 5 times the original data size using dbms_stats to simulate this data volume increase
- Tests were done assuming uniform column data distribution
- Optimizer Cost was measured for the workload
- SPA Trials were run in explain plan mode
• Using TPCH, captured 34 SQL and ran SPA Analysis on increasing data volumes

• Focused on queries with plan changes

• Unchanged performance with new plans also requires testing to minimize risk

• Regressed SQL represents only 11% of total workload (worst case scenario at 2X)

• Significant Cost changes past 2X, investigate partitioning or additional access structures at this point (SQL Access Advisor)
Plan Sensitivity to Data Volume Growth

- Plan Sensitivity Index (PSI) = Optimizer Cost Change % by data volume growth factor
- Linear optimizer cost increase observed with data volume scale increase up until 2X
- When data volume is double the original size, significant regression can be predicted
- For OLTP workload, increasing PSI index signals regressing application performance
- Use SQL Tuning Advisor and/or Plan Baseline to remediate regression(s)
SPA with Growing Data Volume

Summary

SPA can be used to identify high risk SQL with minimal effort in environments with:

- Large workloads and increasing data volumes

Prevent unexpected performance degradation proactively:

- Test high risk SQL in production and tune
Program Agenda

- Database Performance Analysis
  - Challenges

- Advanced Use Cases
  1. Detecting Run-away Queries Using Metric Extensions
  2. Identifying High Risk SQL in Growing Data Volume Environment
  3. Upgrading Oracle E-Business Suite Using SQL Performance Analyzer

- Appendix
Case Study 3: Upgrading Oracle E-Business Suite using SQL Performance Analyzer

Duane Lupinski
Lead Database Administrator, CSX
CSX

Customer

Project

Challenges

Solution

Success
CSX

- CSX (the company)
  - One of the nation’s leading transportation suppliers
  - Encompasses 21,000 miles of tracks in 23 states from Florida to Ontario
  - Access to 70 ports, nationwide transloading and warehousing services
  - More than two thirds of Americans live within CSX’s territory
  - 450+ Oracle databases (200+ RAC); 60+ TB storage footprint; mostly Oracle 11gR2; supporting all facets of business operations

- Duane Lupinski (me)
  - Oracle DBA for 11 years
  - OCP – Oracle Database Administration 8i, 9i, 10g, 11g
  - Expertise in upgrades, migrations, performance tuning, backup/recovery, high availability and disaster recovery solutions
CSX

- Migrate and concurrently upgrade the 12.1.3 EBS database
  - From Oracle EE 10.2.0.4 to Oracle EE 11.2.0.2
  - From a single node on Linux platform to a new two node RAC on new database hardware (also Linux platform)

- EBS implementation consists of the following products: AP, AR, Cash management, PO, INV, OE (order entry partially), XTR (treasury), CDH, OKC (core contracts), Project Accounting PA along with numerous customizations

- Database is approx. 2.8TB in size

- Critical component of finance operations within CSX
CSX

- Limited downtime to perform the upgrade due to business impact
- Time constraints due to competition with other internal business priorities
- Availability of testing resources
- Determining performance impact given change in database version and architecture
CSX

- Used Real Application Testing:
  - Setup Test environment
    - Duplicate production db using Data Guard
    - Upgrade to 11g and convert to RAC
  - Capture Production workload
    - Captured workloads from AWR into SQL Tuning Sets
    - This allowed us to assess workload at multiple periods i.e. month-end, peak and high batch activity
  - Replay each workload in test using **SQL Performance Analyzer**
    - Using SPA guided workflow in Enterprise Manager
  - Assess SPA Reports
    - Identified regression and tuned
Capture Production Workload

- Captured historical workload from AWR Snapshot to build SQL Tuning Sets - Analyze SQL at various periods of time (i.e. month-end peak, other times of high batch activity, etc.)
- Used STS filter options to limit only those SQL on source database that met a certain threshold
- Moved captured workload using Enterprise Manager Copy to Database feature
CSX

- Execute SPA to validate changes
  - Tuned regression using SQL Tuning Advisor, Access Advisor etc.
  - 79% Improvement, Zero regression *(One of many runs)*
  - Upgraded and migrated to RAC with assurance
CSX

- **SQL Performance Analyzer**
  - **Reliably** upgraded and migrated to RAC, critical finance EBusiness suite database using SQL Performance Analyzer
  - **Significantly reduced** man hours spent on addressing performance issues proactively
  - **Minimal** resources required from application testing group
  - **Improved** quality of testing by using actual production workload
  - **Accurate** representation of SQL executed for varying workloads (OLTP, batch) during a given window
  - **Seamless** integration with SQL Tuning Advisor simplifies fixing of regression
Advanced Databases Performance Analysis

Solution

Find
- Built-in, self-diagnostics engine: Automatic Database Diagnostics Monitor (ADDM)
  (Diagnostics Pack)

Fix
- Automates complex and time consuming task of application tuning
  (Tuning Pack)
- Validates tuning and “what-if” activities
  (Real Application Testing)

Validate
- Extend EM out-of-box metrics - Can be defined for any target type, including applications
  (Metric Extensions)

Track
Hardware and Software
Engineered to Work Together