Collecting Cached SQL Data and Its Related Analytics

Gerald Hodge

HLS Technologies, Inc.

Agenda

- Quick Review of SQL Prepare
  - **CACHEDYN=YES** and **KEEPDYNAMIC=YES**
  - **CACHEDYN=YES** and **KEEPDYNAMIC=YES with COMMIT**
  - **CACHEDYN=YES** and **KEEPDYNAMIC=NO with COMMIT**
- Other Presentations
- Objective
- Trace Requirements
- DSNZPARM Settings
- KEEPDYNAMIC(YES) Bind Option
- CURRENT EXPLAIN MODE
- Data Sources
- Data Extraction

HLS TECHNOLOGIES INC.

Agenda

- Analytic Examples
- Optimization Differences
- Explain Differences
- Measurement Patterns
- Verschlimbesserung

HLS
TECHNOLOGIES INC.

Quick Review of SQL Prepare

- CACHEDYNAMIC is a ZPARM. It activates saving both Dynamic SQL and its prepare in the EDM Pool Cache.
- KEEPDYNAMIC is a Bind Parameter. Its default is NO.
- A Full Prepare is an optimization for a given SQL. If Cache is active then a copy of the SQL and its optimization is saved in the Cache.
- A Short Prepare is where an exact copy of the SQL is in the Cache. Its optimization is then copied into Thread Storage and executed.

HLS TECHNOLOGIES INC.

CACHEDYN=YES and KEEPDYNAMIC=YES

Dynamic SQL without COMMIT

- Prepare S
- Execute S
- Execute S

DB2 Actions Store Statement in Cache

And EXECUTE from Cache

- Full Prepare of S
- Prepared Statement S

- First Program
- Overhead of prepare only once

HLS TECHNOLOGIES INC.

CACHEDYN=YES and KEEPDYNAMIC=YES

Dynamic SQL with COMMIT

- Prepare S
- Execute S
- Commit
- Execute S
- Commit
- Execute S

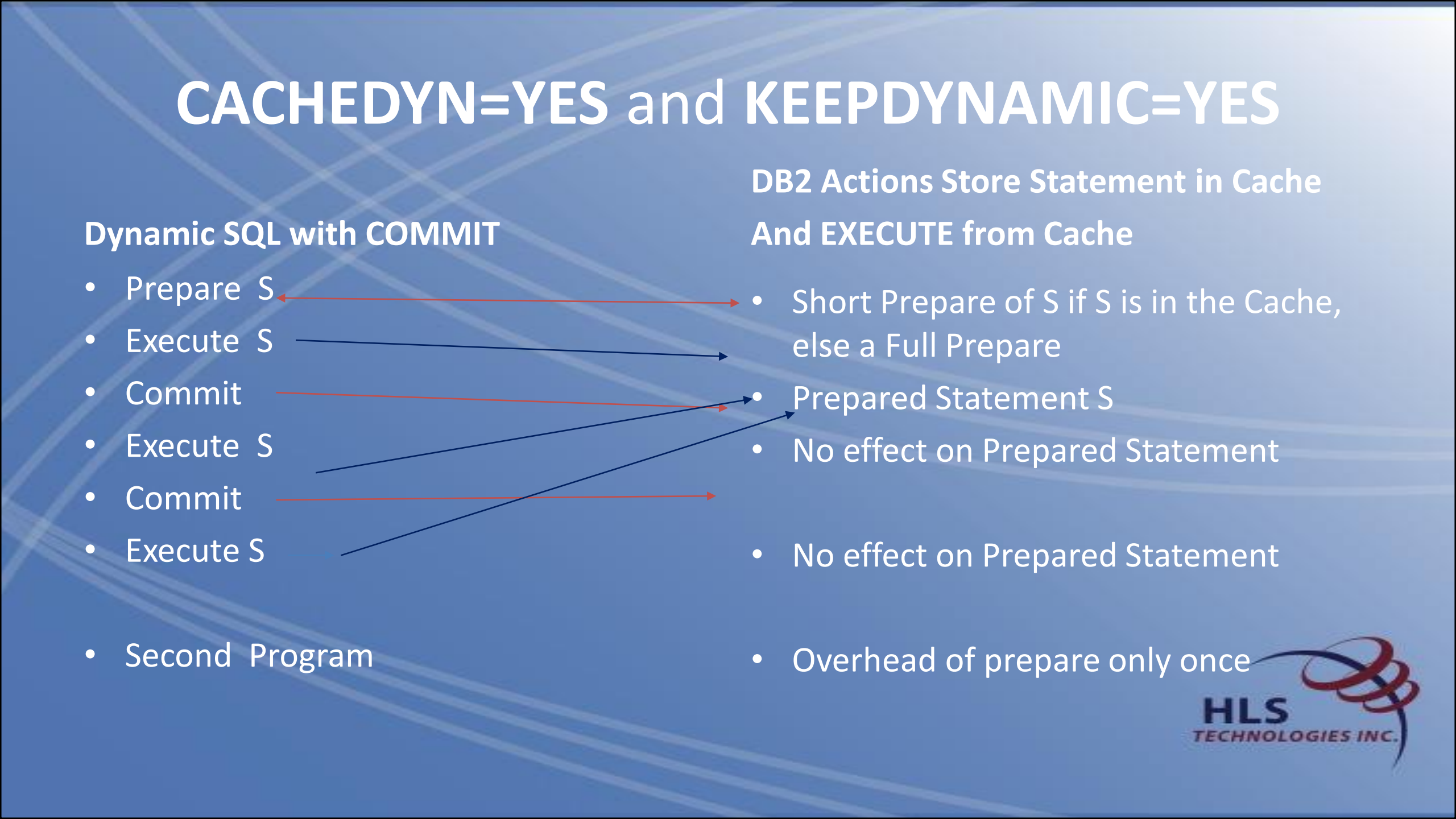
- Second Program

DB2 Actions Store Statement in Cache And EXECUTE from Cache

- Short Prepare of S if S is in the Cache, else a Full Prepare
- Prepared Statement S
- No effect on Prepared Statement

- No effect on Prepared Statement

- Overhead of prepare only once



CACHEDYN=YES and KEEPDYNAMIC=NO

Dynamic SQL with COMMIT

- Prepare S
- Execute S
- Commit
- Execute S
- Commit
- Execute S

- Second Program

DB2 Actions Store Statement in Cache And EXECUTE from Cache

- Full Prepare of S
- Prepared Statement S
- Full Prepare of S
- Prepared Statement S
- Full Prepare of S
- Prepared Statement S
- Overhead of prepare repeated after each commit

HLS TECHNOLOGIES INC.

Use of Global Dynamic Statement Cache

- Enabled by setting CACHEDYN=YES.
- With a Full Prepare both Statement Text and its optimized Prepare are saved in the Cache for use across all threads.
- The First Prepare is a Full Prepare, a Short Prepare is a copy from the global cache into Thread Storage.
- KEEPDYNAMIC=YES (a bind option) keeps the prepare across Commits.
- KEEPDYNAMIC=NO removes the statement and the prepare from the Thread Storage.

HLS TECHNOLOGIES INC.

Other Presentations

All You Need to Know About
Monitoring and Tuning of Dynamic SQL
Session Number: 2483 Insight 2015
Namik Hrle IBM Oct 27, 2015

Finding Ca$h in the Cache:
Leveraging DB2's Dynamic
Statement Cache
Rita Bowers Shelter Insurance
Companies
Session E14 IDUG 2014

Objective

- Collect interval measurement data from the DB2 Cache.
- Use this data to identify:
  - The actual SQL being executed from the Cache after the DB2 Optimization.
  - The actual Explain from the Cache.
  - The SQL performance data from the Cache.
  - Preform Analytics to find performance opportunities.

HLS TECHNOLOGIES INC.

Trace Requirements

- Traces that need to be on
  - 316 EXTERNALIZES EXECUTIONS STATS
  - 317 EXTERNALIZES STATEMENT TEXT
    - This is the actual SQL executed after Optimization
  - 318 CONTROLS TYPE OF STATISTICS

  Cost is estimated at 1 to 2 percent of CPU cost of SQL execution.

  I have yet to find any production site that can measure cost.

  Warning: Many places have objections to any traces!!

Trace Requirements

- The Traces should be started to a Monitor Class.
- This is not a post processor for SMF / IFCID Records!
- These traces cause data to be placed in the STATEMENT_CACHE that are then externalize with the EXPLAIN STMTCACHE ALL.
- Because of the 2G potential size of SQL Statements they are externalized differently than in an SMF Record.

HLS TECHNOLOGIES INC.

Trace Requirements

- While an EXPLAIN may be externalized from the CACHE there is no IFCID to provide the EXPLAIN.
- You will have to execute specific code to obtain the EXPLAIN from the Cache.
- DO NOT WRITE THIS EXPLAIN DATA TO YOUR COMMON EXPLAIN TABLES!
- Think of the volume.
- The SQL will need to be post processed to make it readable.

HLS TECHNOLOGIES INC.

DSNZPARM Settings

- CACHEDYN Yes causes EDM Pool to save prepared Dynamic SQL to be saved in EDM Pool. LIFO is used for purging.
- EDMSTMTC specify EDM statement pool size.
- MAXKEEPD Maximum # of prepared statements in Cache.
- CACHEDYN_FREELOCAL Only effects KEEPDYNAMIC(YES). Default is ENABLED in V10.
- CONTSTOR If YES unused thread storage released at commit.
- MINSTOR If YES DB2 reduces thread's local storage. In V10 recommendation is No.

HLS TECHNOLOGIES INC.

EDMPOOL Warning!

- Check that you have no Program Products that do not release Dynamic SQL.
- If you do then turning Cache on will eventually slow DB2.
  - DB2 will attempt to use the Cache only to find out it cannot.
- You lose both the advantage of the Program Product and Caching.
- I have not found any way to avoid this problem.

HLS TECHNOLOGIES INC.

KEEPDYNAMIC(YES) Bind Option

- Bind Option KEEPDYNAMIC(YES) saves the prepare in the Cache after a Commit.
- KEEPDYNAMIC(NO) will cause a full prepare after each commit.
- This is a matter of Religious discussion both on the Net and within some organizations.
- If set to no then you lose the ability to do a short prepare.
- This can have a substantial CPU cost.

HLS
TECHNOLOGIES INC.

CURRENT EXPLAIN MODE

- SET CURRENT EXPLAIN MODE = NO | YES | EXPLAIN
- NO means that an EXPLAIN is available from the Cache.
- Yes means that when an Optimization occurs an EXPLAIN is written to the standard EXPLAIN tables.
- Can you afford added clutter in your standard EXPLAIN tables?
- If YES you have an EXPLAIN, but if SQL Rewrite occurred you do not have the matching SQL!

HLS TECHNOLOGIES INC.

Data Sources

Run an extract of data from the Cache at least every hour.

- The extract takes only a few seconds and is non-disruptive.
- Delta the data to provide interval information.
- The extracted data is processed to provide:
  - The actual SQL executed as a result of the DB2® Optimization.
  - The actual Explain data as present in the Cache for the interval.
  - The measurement data associated with the execution of the SQL
    - The measurement data is cumulative from interval to interval and is modified to show the activity within the interval.

HLS TECHNOLOGIES INC.

Data Extraction

- **EXPLAIN STMTCACHE ALL**
  - Populates table DSN_STATEMENT_CACHE_TABLE
  - Qualified by current SQLID
    - If executed by SYSADM or SQLADM then all data is extracted
    - Otherwise only data allowed by the current SQLID
  - One row for each cached statement
  - An EXPLAIN has to be done separately otherwise there is no EXPLAIN for the SQL from the cache! (See Pragmatics below.)

HLS TECHNOLOGIES INC.

DSN_STATEMENT_CACHE_TABLE

- Has a copy of the unique SQL.
- SELECT A,B from… does not equal SELECT B,A from…
- WHERE CLAUSE with Literals makes the statement unique.
- If lots of Literals then consider “concentrate with literals”.
  - A DB2 Connect parameter.
  - A PREPARE parameter.
- Contains performance data.
- Explain information has to be extracted with a different statement.

HLS TECHNOLOGIES INC.

Data Extraction

- **EXPLAIN STMTCACHE STMTID**
  - **You provide an integer constant or a host variable to select a single row from the cache. QW0316ID in CACHE_TABLE.**
  - **In your PLAN_TABLE the QUERYNO is the STMT_ID that is the value.**
  - PLAN_TABLE, DSN_STATEMNT_TABLE, DSN_FUNCTION_TABLE are the tables populated. The data also remains in the Cache, which also has the execution statics. You still have to get the executed SQL from the CLOB in your externalized CACHE table.
  - This Data is also available from IFCIDs 172, 196, 124.

HLS TECHNOLOGIES INC.

Data Extraction

- **EXPLAIN STMTCACHE STMTTOKEN**
  - **UP to 240 character string constant or Host Variable**
  - **Assigned by an application at prepare RRASF function for local app and** sqlseti API for remote applications.
  - Value is reported in PLAN_TABLE's STMTTOKEN column QW0316UI.
  - PLAN_TABLE, DSN_STATEMNT_TABLE, DSN_FUNCTION_TABLE are the tables populated. The data also remains in the Cache, which also has the execution statics. You still have to get the executed SQL from the CLOB in your externalized CACHE table.

HLS TECHNOLOGIES INC.

Data Extraction

- EXPLAIN STMTCACHE ALL offers your best chance not to miss data.
- Interval processing gives you the best chance to collect data that might be purged if the Cache becomes full.
- Interval processing allows the best chance to get EXPLAIN data that matches the actual execution.
- Interval processing allows compare and contrast across intervals.
- Interval processing presents challenges that we deal with in the Pragmatics section.
- You need to decide how available you want the data to be.

HLS TECHNOLOGIES INC.

Analytical Examples

- Identify SQL Optimization differences across time.
- Identify Explain differences.
- Identify measurement patterns of interest for both the System and the individual Dynamic SQL.
- From the measurement patterns find tuning opportunities.
  - Two real life examples.

HLS TECHNOLOGIES INC.

Optimization Differences

- It is known that the Dynamic SQL coming from the Optimizer may not match the Dynamic SQL as written
- The Optimizer will try to turn subselects into Joins, etc.
- If there are noticeable differences between interval performance for an SQL Statement then there is a simple SPUFI check to see if the executed SQL statement matches for both intervals and to check the EXPLAIN.

HLS TECHNOLOGIES INC.

Explain Differences

- As with the SQL result from the Optimizer the Explain for different intervals can very.
- The Optimizer is influenced by CPU usage, Buffer Pool measurement and other factors at the time of optimization.
- This means that the same Dynamic SQL may have a different Access Path in different intervals if the SQL is brought out and back in.
- A SPUFI is provided to check the Explain data between two intervals for a given SQL statement.

HLS TECHNOLOGIES INC.

Verschlimmbesserung

- Definition: Worsening with each improvement.
- First used to describe a medical condition.
- Then used to describe a military condition.
- Now use to describe the German Tax Code.
- Can be used to describe many attempts at Performance Tuning.

HLS
TECHNOLOGIES INC.

Measurement Patterns

- Instead of providing SQL measurements across a complete day information is provided at an hourly interval, this interval can be shorten for finer analysis.
- A SPUFI is provided to provide a daily summary for key SQL.
- By viewing the interval data it can be determined if specific SQL uses more elapsed time and / or CPU time for intervals across the day.

HLS TECHNOLOGIES INC.

Measurement Patterns: A Real Life Example

- A EU Bank ran one set of SQL in their morning with satisfactory results.
- In the late Morning and Afternoon the results were much slower.
- There was similar set of SQL coming active during the slow down.
- There were Legal reasons for the differences between the sets of SQL.
- There were Legal / Political reasons that limited the tuning opportunities.
- It was identified that the slowdown was a result of contention on a specific index.

HLS TECHNOLOGIES INC.

Measurement Patterns: A Solution

- A duplicate index was created.
- For Auditing / Legal reason the two indexes had to be the same.
- The new index was located in a Buffer Pool that was not busy during the business day.
- A Hint was made to the first set of SQL to choose the new index.
- The result was that the first set of SQL's performance stabilized across the day to what it was in the morning.
- The second set of SQL ran as before with a performance improvement.

HLS
TECHNOLOGIES INC.

Query Rewrite

- A Dynamic SQL Statement that used subselects had a performance issue.
- The Performance DBA was not allowed to change the SQL so he created a Hint.
- The Hint took, but had no effect?!
- Query Rewrite had occurred and the EXPLAIN was vary different.
- There was some difficulty in accepting the situation.

HLS TECHNOLOGIES INC.

Simple Examples of System Analytics

Examples on Website

- Global Cache Hit Ratio
- Local Cache Hit Ratio

HLS TECHNOLOGIES INC.

Local Dynamic Cache vs Global Dynamic Cache

- Local Dynamic Cache is in DBM1 Storage.
- Entry is removed when an application ends, there is an explicit prepare, a rollback occurs, the statement is remove because it has not been recently used.
- The global dynamic statement cache is allocated in the dynamic statement cache pool (EDMSTMTC).

  It is allocated above the 2GB bar.

HLS TECHNOLOGIES INC.

Local Dynamic Cache Hit Ratio

- QXSTNPRP / (QXSTNPRP + QXSTIPRP)
- QXSTIPRP Implicit Prepares. SQL not in Local Cache and not in the Global Cache and a Prepare is done.
- QXSTNPRP avoided Prepares. The SQL is found in the Cache.
- Statements Invalidated: QXSTDEXP+QXSTDINV.
- QXSTDEXP Avoid Prepares.
- QXSTDINV Invalidation of SQL Statement.

HLS TECHNOLOGIES INC.

Global Dynamic Cache Hit Ratio

- (QISEDSG – QISEDSI) / QIESDSG
- Goal is between 90% to 95%.
- QSIEDSG Cache search requests.
- QISEDSI Full Prepares.
- QXSTNFND SQL not found in Cache.
- QISEDSG – QISEDSI Short Prepares.

HLS TECHNOLOGIES INC.

CPU Use

STMT_ID	PROGRAM	STAT_CPU	SQL
57	ADBMAIN	10.05	T.NAME,T.TSNAME,T.RECLENGTH
105	DSNESM68	.10	SELECT T.NAME,T.TSNAME,

HLS TECHNOLOGIES INC.

Pragmatics

- A quick course in the pragmatics of interval processing for the DSN_STATEMENT_CACHE_TABLE.
- SYSDBA, SYSOPR issues.
- Preparing for glitches.
- Interval Processing.
- Testing in the System Area and Development.
- Questions for access in Production.
- What to look for…

HLS TECHNOLOGIES INC.

Pragmatics
Collecting Cached SQL Data and Its Related Analytics

Gerald Hodge

HLS Technologies, Inc.

An Approach to tuning

- Ubi desinit philosophus, ibi incipit medicus.
- Justinian restating Aristotle.
- Begin with and stay with the doable.
- Try to demonstrate benefit using simple actions.
- Make sure that you do show and tells about what you discover and the results.
- If something does not cause improvement state that and you will get credibility.

Agenda

- Determining Status of Caching on your System.
- Create your DSN_STATEMENT_CACHE.
- Create your DSN_STATEMENT_CACHE_HISTORY.
- The basics of processing the STATEMENT_CACHE data.
  - SYSOPER / SYSDBA Issues.
- Processing Cycle.
- EXPLAIN SQL Cycle.
- Getting your SQL Statement.

HLS TECHNOLOGIES INC.

Agenda

- A brief review of where we are.
- Your Friend Data Studio.

HLS TECHNOLOGIES INC.

Check Caching Parameters

- Ask the SYSDBA or SYSPROG if Cache is set on you DB2 Systems.
- If it is not then ask why?
- Do you have a ISV Application that interferes with the Cache?
- These are part of your DB2 applications that handle data access.
- They are usually a replacement for Caching.
- If you are using one STOP!

HLS TECHNOLOGIES INC.

DSNZPARMs for Caching

- CACHEDYN Yes causes EDM Pool to save prepared Dynamic SQL to be saved in EDM Pool. LIFO is used for purging.
- EDMSTMTC specify EDM statement pool size.
- MAXKEEPD Maximum # of prepared statements in Cache.
- CACHEDYN_FREELOCAL Only effects KEEPDYNAMIC(YES). Default is ENABLED in V10.
- CONTSTOR If YES unused thread storage released at commit.
- MINSTOR If YES DB2 reduces thread's local storage. In V10 recommendation is No.

HLS TECHNOLOGIES INC.

Create your DSN_STATEMENT_CACHE

- DB2 Samplib contains a member DSNTESC.
- DSNTESC contains the DDL to create DSN_STATEMENT_CACHE.
- Cut and Paste this member to another PDS.
- You will need to create two copies to your DB2 Subsystem.
- Create each with a different Creator ID. We use HLS1 and HLS2.

HLS TECHNOLOGIES INC.

Create your DSN_STATEMENT_CACHE_HISTORY

- DB2 SAMPLIB contains a member DSNTESC.
- DSNTESC contains the DDL to create DSN_STATEMENT_CACHE.
- Cut and Paste this member to another PDS.
- Change the DDL entries to reflect a name ending HISTORY.
- Add a column to the end of the DDL:
- “ CAPTURE_TS” TIMESTAMP NOT NULL
- You will use this column to set the interval for the capture.

HLS TECHNOLOGIES INC.

SYSOPER / SYSDBA Issue

- The batch job that issues the EXPLAIN STATEMENT_CACHE ALL
  - Should be run under SYSOPER or SYSADM authority.
  - If not then only the data associated with the authority will be externalized:
    - SYSOPER / SYSDBA will externalize all CACHE data.
    - APPIDxx will only externalize data associated with it authority.
    - Your HISTORY Table should be available to all appropriate staff.

HLS TECHNOLOGIES INC.

Processing the STATEMENT_CACHE data

- 1st Time you do an EXPLAIN STATEMENTCACHE ALL both tables will be empty.
- After that you will alternate the target table.
- You need to decrement the values in the target table using the other table as the base.
- You do this as you are updating the HISTORY table.
- This allows you to alternate the target for the EXPLAIN and use the older entry to decrement the values in the newer to do the update.

HLS TECHNOLOGIES INC.

Processing Cycle

- If you do hourly updates, most sites copy the HISTORY table data to a weekly table and clear the HISTORY table.
- Few sites want more than a rolling amount of data in the weekly table.
- Usually, sites run a set of reports at the conclusion of a day's processing.
- Other reports consist of running SPUIFs or jobs you create to run for questions during the day.

HLS TECHNOLOGIES INC.

EXPLAIN SQL Cycle

- Remember EXPLAIN STMTCACHE ALL does not explain the SQL.
- The results of a Full Prepare are in the CACHE associated with the SQL
- They are not externalize with the EXPLAIN ALL. Why would an EXPLAIN ALL include EXPLAIN SQL data?
- You need to do a SELECT of the STMTID of every SQL that is new in the CACHE.

HLS TECHNOLOGIES INC.

EXPLAIN SQL Cycle

- You then programmatically execute an

  EXPLAIN STMTCACHE STMTID

  for each of the STMTIDs you selected.
- PLAN_TABLE, DSN_STATEMNT_TABLE, DSN_FUNCTION_TABLE are the tables populated. The data also remains in the Cache, which also has the execution statics. You still have to get the executed SQL from the CLOB in your externalized CACHE table.
- DO NOT use your production Plan Tables because of volume

HLS TECHNOLOGIES INC.

Getting your SQL Statement

- There is a CLOB associated with each row in the DSN_STATEMENT_CACHE.
- The data in the CLOB is in the format that DB2 sees. It is not formatted to be readable by us.
- The CLOB is externalized by the EXPLAIN STMTCACHE ALL.
- We provide a SPUFI example to assist in formatting.

HLS TECHNOLOGIES INC.

Preparing for Glitches

- If caching is not in use be prepared for falling back your DSNZPARM.
  - SET SYSPARM LOAD(module name)
- If your Production EXPLAIN Tables fill up check your EXPLAIN program to insure it is using the your Explain Tables.
- Be prepared to stop your extraction program because of volume problems.

HLS TECHNOLOGIES INC.

A brief review of where we are

- We know how to externalize the Cache Data.
- We know how to externalize the SQL.
- We know how to obtain the EXPLAINs.
- The obvious and the not so obvious.
  - Unnecessary commits.
  - Wait Times.
  - Heavy GetPages.
  - SQL Examples

HLS TECHNOLOGIES INC.

Find Time Intervals

- SELECT DISTINCT CAPTURE_TS
  FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
  ;
- Displays the distinct time intervals in your History Table.
- Use this to select data from specific intervals.

HLS TECHNOLOGIES INC.

Select Top 10 CPU Users

```
SELECT STMT_ID,
    SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
    CACHED_TS AS TIME_CACHED,
    SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
    STAT_CPU,
    STAT_ELAP,
    STAT_GPAG,
    CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
    CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
    CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
    CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
    CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
    CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
    VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
WHERE STAT_CPU > 0
ORDER BY STAT_CPU DESC
FETCH FIRST 10 ROWS ONLY
;
```

HLS TECHNOLOGIES INC.

Select Top 10 Elapsed Time

```
SELECT STMT_ID,
    SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
    CACHED_TS AS TIME_CACHED,
    SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
    STAT_CPU,
    STAT_ELAP,
    STAT_GPAG,
    CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
    CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
    CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
    CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
    CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
    CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
    VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
WHERE STAT_ELAP > 0
ORDER BY STAT_ELAP DESC
FETCH FIRST 10 ROWS ONLY
;
```

HLS TECHNOLOGIES INC.

Top 10 SQL Number of Rows Examined

```
SELECT STMT_ID,
      SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
      CACHED_TS AS TIME_CACHED,
      SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
      STAT_CPU,
      STAT_EROWB,
      STAT_PROWB,
      CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
      CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
      CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
      CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
      CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
      CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
      VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_EROWB > 0
 ORDER BY STAT_EROWB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL for GET PAGES

```
SELECT STMT_ID,
    SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
    CACHED_TS AS TIME_CACHED,
    SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
    STAT_CPU,
    STAT_GPAGB,
    STAT_EROWB,
    CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
    CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
    CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
    CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
    CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
    CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
    VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
WHERE STAT_GPAGB > 0
ORDER BY STAT_GPAGB DESC
FETCH FIRST 10 ROWS ONLY
;
```

HLS TECHNOLOGIES INC.

Top 10 SQL for Index Scans

```
SELECT STMT_ID,
       SUBSTR(PROGRA_NAME, 1, 9) AS PROGRAM,
       CACHED_TS AS IME_CACHED,
       SUBSTR(PRIMAUH, 1, 9) AS AUTH_ID,
       STAT_CPU,
       STAT_INDXB,
       STAT_RSCNB,
       CAST(STAT_SUSSYNIO AS DECIMAL(18, 5)) AS SYNCIO,
       CAST(STAT_SUSLOCK AS DECIMAL(18, 5)) AS LOCKS,
       CAST(STAT_SUSOTHR AS DECIMAL(18, 5)) AS OTHER,
       CAST(STAT_SUSOTHW AS DECIMAL(18, 5)) AS OTHERW,
       CAST(STAT_SUSLATCH AS DECIMAL(18, 5)) AS LATCH,
       CAST(STAT_SUSPLATCH AS DECIMAL(18, 5)) AS PLATCH,
      VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
WHERE STAT_INDXB > 0
ORDER BY STAT_INDXB DESC
FETCH FIRST 10 ROWS ONLY
;
```

HLS TECHNOLOGIES INC.

Top 10 SQL for Pages Processed

```
SELECT STMT_ID,
       SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
       CACHED_TS AS TIME_CACHED,
       SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
       STAT_CPU,
       STAT_PROWB,
       STAT_SORTB,
       CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
       CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
       CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
       CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
       CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
       CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
       VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_PROGB > 0
 ORDER BY STAT_PROWB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL Where RID Limit Exceeded

```
SELECT STMT_ID,
      SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
      CACHED_TS AS TIME_CACHED,
      SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
      STAT_CPU,
      STAT_RIDLIMTB,
      CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
      CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
      CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
      CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
      CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
      CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
      VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_RIDLIMTB > 0
 ORDER BY STAT_RIDLIMTB DESC
 FETCH FIRST 10 ROWS ONLY
```

HLS TECHNOLOGIES INC.

Top 10 SQL Where RID Storage Exceeded

```
SELECT STMT_ID,
      SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
      CACHED_TS AS TIME_CACHED,
      SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
      STAT_CPU,
      STAT_RIDSTORB
      STAT_RIDLMITB,
      CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
      CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
      CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
      CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
      CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
      CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
      VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_RIDSTORB > 0
 ORDER BY STAT_RIDSTORB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL Tablespace Scans

```
SELECT STMT_ID,
      SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
      CACHED_TS AS TIME_CACHED,
      SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
      STAT_CPU,
      STAT_RSCNB,
      STAT_RIDSTORB,
      CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
      CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
      CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
      CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
      CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
      CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
      VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_RSCNB > 0
 ORDER BY STAT_RSCNB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL with Sorts

```
SELECT STMT_ID,
       SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
       CACHED_TS AS TIME_CACHED,
       SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
       STAT_CPU,
       STAT_SORTB,
       STAT_INDXB,
       CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
       CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
       CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
       CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
       CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
       CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
       VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_SORTB > 0
 ORDER BY STAT_SORTB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL Synchronous Buffer Reads

```
SELECT STMT_ID,
       SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
       CACHED_TS AS TIME_CACHED,
       SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
       STAT_CPU,
       STAT_SYNRB,
       STAT_WRITB,
       CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
       CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
       CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
       CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
       CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
       CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
       VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_SYNRB > 0
 ORDER BY STAT_SYNRB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Top 10 SQL Buffer Writes

```
SELECT STMT_ID,
       SUBSTR(PROGRAM_NAME, 1, 9) AS PROGRAM,
       CACHED_TS AS TIME_CACHED,
       SUBSTR(PRIMAUTH, 1, 9) AS AUTH_ID,
       STAT_CPU,
       STAT_SYNRB,
       STAT_WRITB,
       CAST(STAT_SUS_SYNIO AS DECIMAL(18, 5)) AS SYNCIO,
       CAST(STAT_SUS_LOCK AS DECIMAL(18, 5)) AS LOCKS,
       CAST(STAT_SUS_OTHR AS DECIMAL(18, 5)) AS OTHER,
       CAST(STAT_SUS_OTHW AS DECIMAL(18, 5)) AS OTHERW,
       CAST(STAT_SUS_LATCH AS DECIMAL(18, 5)) AS LATCH,
       CAST(STAT_SUS_PLATCH AS DECIMAL(18, 5)) AS PLATCH,
       VARCHAR(SUBSTR(STMT_TEXT, 1, 3200)) AS SQL_STMT
 FROM IBMUSER.DSN_STATEMENT_HISTORY_TABLE
 WHERE STAT_WRITB > 0
 ORDER BY STAT_WRITB DESC
 FETCH FIRST 10 ROWS ONLY
 ;
```

HLS TECHNOLOGIES INC.

Where the Cache Fields are Documented

- https://www.ibm.com/support/knowledgecenter/SSEPEK_11.0.0/com.ibm.db2z11.doc.perf/src/tpc/db2z_dsnstatementcachetable.dita

HLS TECHNOLOGIES INC.

Your Friend Data Studio

- The Rita Bowers presentation shows how to use Data Studio to examine the Cache in real time.

HLS TECHNOLOGIES INC.

Find Me

- Gerald Hodge
- Cell 281 830 5214
- www.hlstechnologies.com

- If I can I will help.

HLS
TECHNOLOGIES INC.