IBM Information On Demand 2008

IBM

Managing Dynamic SQL Performance

Billy Sundarrajan

Lead Database Administrator

Fifth Third Bank

June 10, 2008

IBM INFORMATION ON DEMAND 2008
October 26 – 31, 2008
Mandalay Bay
Las Vegas, Nevada

Agenda

- Dynamic SQL Performance tuning challenges
- DB2 V8 Enhancements
- DB2 9 Enhancements
- SQL Performance Warehouse
- Summary

Dynamic SQL Performance Tuning Challenges

Dynamic SQL performance tuning challenges (addressed by DB2 v8 enhancements)

- Identifying the “actual” end user performing a dynamic SQL – Most dynamic SQL get executed by an application server/common authorization ID
- Ability to drill down accounting information to a more granular level than just the authorization ID
  - Middleware such as Business Objects, Websphere perform the connect to DB2 using common authorization/RACF ID
- Identifying the contents of the dynamic SQL cache and the access paths being used
- Reducing the number of accounting (SMF 101) records for DISTSERV

Dynamic SQL performance tuning challenges (addressed by DB2 9 Enhancements)

- Using Resource Limit Facility to control the service units for dynamic SQL – at a more granular level than AUTHID/PLAN/PACKAGE
- Using DB2 traces for dynamic SQL applications – How to control the unit of work being traced
- Optimizing prepared dynamic SQL where the host variables change
- Securing the application server user ID

Identifying the “actual” user in a 3-tier architecture

Identifying the “actual” end user:

- Most dynamic SQL intensive applications use a functional ID/ common Authorization ID
  - Single ID avoids the creation and maintenance of a large number of RACF IDs in a web-based application
  - DBA does not have additional information about the “actual” end-user
  - In most instances, the IP address in the DB2 monitor is the IP address of the DB2 Connect gateway

Identifying the “actual” end user:

- DB2 v5-7 provided ability to pass “client accounting” / “client identification” information to SMF 101 records.
- Information that could be passed:
  - Client User ID, Client Workstation name, Client application (transaction) name, Client Accounting Information
- V5-7 –Enhancements to supply client accounting information to the SMF 101 records and to the monitor
  - SQLESETI
  - RRSAF
  - JDBC Universal driver
- What does V8 provide?

Client Identification registers –V8:

- V8 - provides four new registers
  - CURRENT CLIENT_USERID
  - CURRENT CLIENT_WRKSTNNAME
  - CURRENT CLIENT_APPLNAME
  - CURRENT CLIENT_ACCTNG
- Allows applications to intercept and use the information supplied/set by a web interface
- Client identifiers can be used to throttle DB2 DDF workloads in WLM

Client Userid register – V8:

- CURRENT CLIENT_USERID

  QWHCEUID – CHAR(16)

  contains the value of the client user ID from the client information that is specified for the connection. Set using:
  - sqleseti (SQLE_CLIENT_INFO_USERID)
  - RRS DSNRLI
  - DB2Connection.setDB2ClientUser(String info)
- In a distributed environment, if the value set by the API exceeds 16 bytes, it is truncated to 16 bytes.
- WLM identifier – SPM (bytes 1 through 16)

Client Workstation register – V8:

- CURRENT CLIENT_WRKSTNNAME
  QWHCEUWN – CHAR(18)

  contains the value of the workstation name from the client information that is specified for the connection. Set using:
  - sqleseti (SQLE_CLIENT_INFO_WRKSTNNAME)
  - RRS DSNRLI
  - DB2Connection.setDB2ClientWorkstation(String info)
- In a distributed environment, if the value set by the API exceeds 18 bytes, it is truncated to 18 bytes.
- WLM identifier – SPM (bytes 17 through 34)

Client Application Identifier register – V8:

- CURRENT CLIENT_APPLNAME

  QWHCEUTX - CHAR(32)

  contains the value of the application name from the client information that is specified for the connection. Set using:
  - sqleseti (SQLE_CLIENT_INFO_APPLNAME)
  - RRS DSNRLI
  - DB2Connection.
    setDB2ClientApplicationInformation(String info)
- In a distributed environment, if the value set by the API exceeds 32 bytes, it is truncated to 32 bytes.
- WLM identifier - PC

Client Accounting Information register –V8:

- CURRENT CLIENT_ACCTNG
  QMDASUFX – VARCHAR(200)
  contains the value of the user specified portion of accounting string. Set using:
  - sqleseti (SQLE_CLIENT_INFO_ACCTSTR)
  - RRS DSNRLI
  - DB2Connection.
    setDB2ClientAccountingInformation(String info)
- QMDAAINF (Documented in DSNQMDA) contains the entire accounting string
- In a distributed environment, if the value set by the API exceeds 200 bytes, it is truncated to 200 bytes.
- WLM identifier – AI (Starting position 56)

EXCSQLSET – DRDA Applications – V8:

- DRDA Applications can use EXCSQLSET
  - SET CLIENT USERID ‘my_userid’
  - SET CLIENT WRKSTNNAME ‘my_wrkstn’
  - SET CLIENT APPLNAME ‘my_applname’
  - SET CLIENT ACCTNG ‘my_accounting’
- Additional information allows the ability to store client identifier in tables (such as who last updated by data)
- EXCSAT EXTNAM/RRS is used for passing the correlation ID – stored in QWHCCV (WLM identifier – CI)
- DSNDQWAS SUBTYPE=ALL provides layout of SMF101 record

Websphere Applications

- WAS 6.1 - setClientInformation API can be used to pass client registers
  - API is defined on the WSConnection class
  - Part of the com.ibm.websphere.rsadapter package
  - Values that can be passed include
    - WSConnection.CLIENT_ID
    - WSConnection.CLIENT_LOCATION
    - WSConnection.CLIENT_APPLICATION_NAME
    - WSConnection.CLIENT_ACCOUNTING_INFO
  - Client Information can be reset by calling method with a NULL parameter

Drilling down on accounting information

- V7 – No easy mechanism to drill down information at a more detailed level than the authorization ID
- V8 –
  - SMF/Accounting information can be drilled down by Client User ID/Transaction/Workstation providing a more granular view of CPU/resource utilization
  - Breaking down the CPU consumption information beyond just the authorization ID provides insight into the areas that need tuning
  - With additional CLIENT registers that are available to the application, additional client information can be easily externalized into a DB2 table

What is in my Dynamic SQL cache?

What's in my Dynamic SQL cache?

- Dynamic SQL cache (DSC) contains statements that are prepared and inserted into the global cache (except SQLs using GTTs)
  - CACHEDYN=YES
- With IFCID 318 (Monitor trace) turned on, cache contains information about SQL execution, CPU time, GETPAGES etc
- Until V8, information on cached statements and access paths could not be easily externalized
  - Required a tool such as DB2 PE, Omegamon etc
  - Reading IFCID 22 (mini plan)/63 (SQL text)
  - Reading IFCID 316, 317
- What does V8 provide?

What's in my Dynamic SQL cache –V8

- Unlike static SQL, DBAs do not have easy access to the dynamic SQL statements run by an application
- V8 provides additional enhancements to EXPLAIN to view information in the DSC
- New EXPLAIN options:
  - EXPLAIN STMTCACHE ALL
  - EXPLAIN STMTCACHE STMTID
  - EXPLAIN STMTCACHE STMTTOKEN
    - Differs from traditional EXPLAIN – Does not re-explain to obtain access path
    - Can be run using SPUFI, DSNTEP2, Visual Explain, OSC

New EXPLAIN options – V8

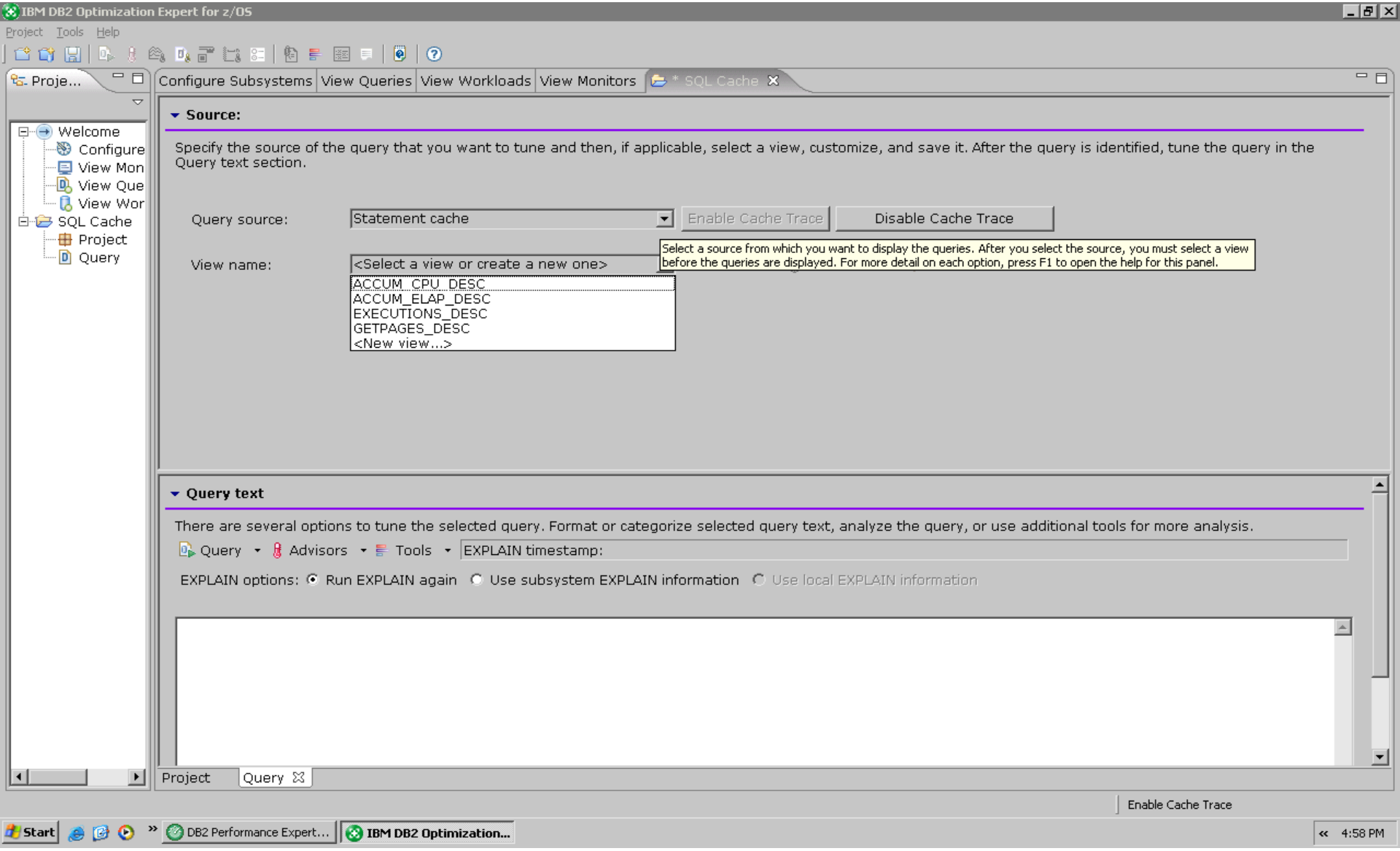
- EXPLAIN STMTCACHE ALL
  - Dumps the SQL statement cache –
  - Populates a table <current sqlid>. DSN_STATEMENT_CACHE _TABLE
  - DB2 V8 NFM required
  - Uses LOBs
  - Member specific – In a Data sharing environment should be run on all members

New EXPLAIN options – V8

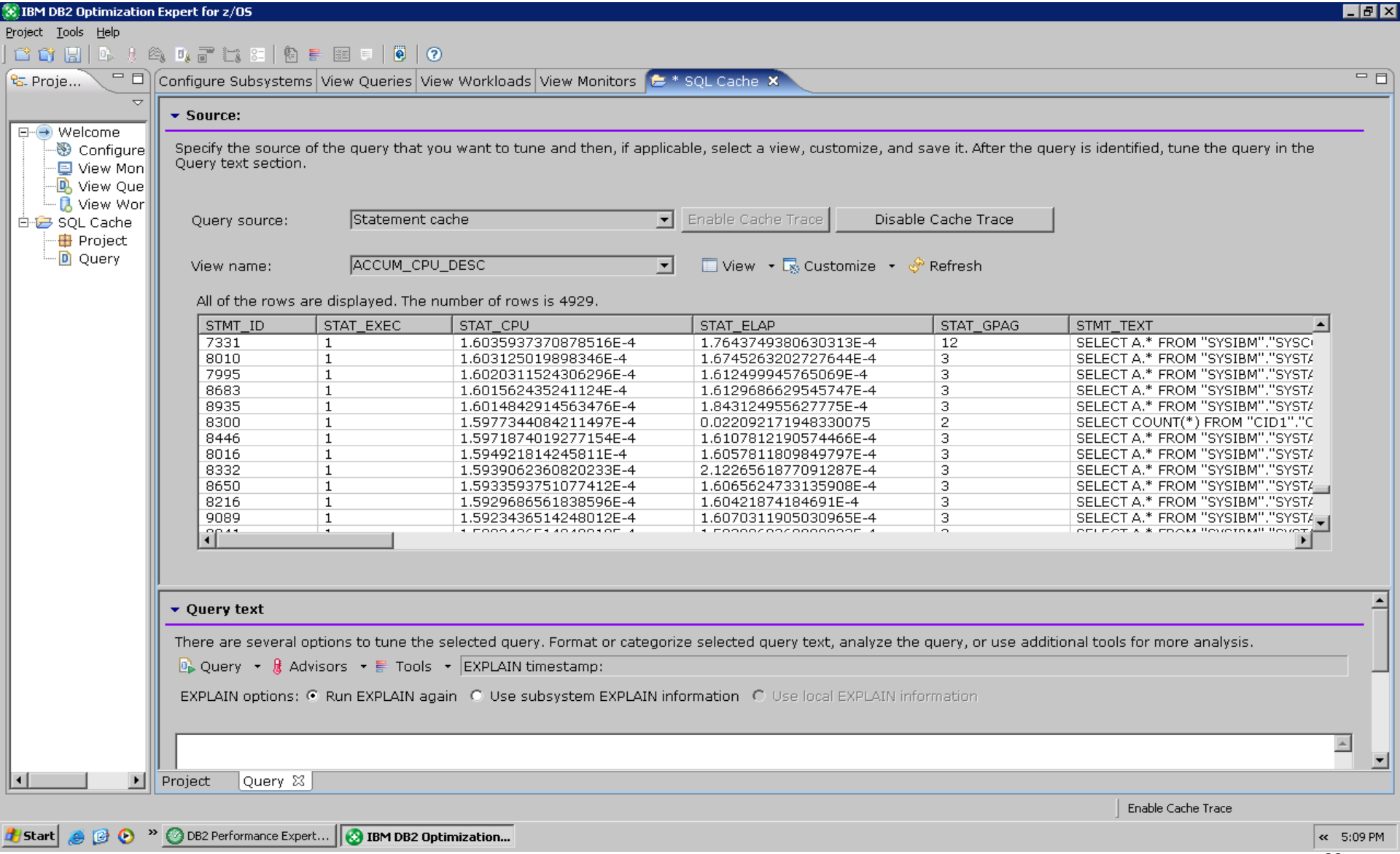
- EXPLAIN STMTCACHE ALL
  - Information identical to information from IFCID 316 and IFCID 317
  - Collection and reset of statistics is controlled by IFCID 318
  - Visual Explain/Optimization Service Center (OSC) provides easy to use interface to perform EXPLAIN command
  - OSC can be downloaded from

    http://www-306.ibm.com/software/data/db2/zos/downloads/osc.html

EXPLAIN STMTCACHE ALL using DB2 OSC



EXPLAIN STMTCACHE ALL using DB2 OSC



New EXPLAIN options – V8

- EXPLAIN STMTCACHE STMTID
  - Specifies that the cached statement associated with the statement ID (STMT_ID column in DSN_STATEMENT_CACHE_TABLE) is to be explained.
  - Statement ID is an integer that uniquely determines a statement that has been cached in dynamic statement cache. The statement ID can be retrieved through IFI monitor facilities from IFCID 316 or 124 and is shown in some diagnostic IFCID trace records such as 172, 196, and 337.

New EXPLAIN options – V8

- EXPLAIN STMTCACHE STMTID
  - Column QUERYNO is given the value of the statement ID in every row inserted into the plan table, statement table, or function table by the EXPLAIN statement. (i.e., STMT_ID column copied to QUERYNO column)
  - BIND_TIME is given the value of CACHED_TS
  - Writes information to the DSN_STATEMENT_CACHE_TABLE and PLAN_TABLE

New EXPLAIN options – V8

- EXPLAIN STMTCACHE STMTTOKEN
  - Explains all cached statements associated with a statement token (statement token up to 240 bytes)
  - STMTTOKEN is associated with the cached statement by the application program that originally prepares and inserts the statement into the cache.

New EXPLAIN options – V8

- EXPLAIN STMTCACHE STMTTOKEN
  - Application programs use RRSAF SET_ID function, or sqleseti API from a remotely-connected program.
  - Column STMTTOKEN is given the value of the statement token, and the column QUERYNO is given the value of the statement ID for the cached statement with the statement token, BIND_TIME is given the value of CACHED_TS.
  - Writes information to the DSN_STATEMENT_CACHE_TABLE and PLAN_TABLE

EXPLAIN versus EXPLAIN STMTCACHE

- Unlike EXPLAIN, EXPLAIN STMTCACHE does not go through the EXPLAIN process
- DB2 writes the current access path to the PLAN_TABLE; re-explaining the statement can provide different results
- SQLCODE -20248 if statement no longer in cache

PLAN_TABLE – How do I identify a cached statement?

- COLLID Column in PLAN_TABLE, and DSN_STATEMENT_CACHE_TABLE will contain the value DSNDYNAMICSQLCACHE for cached statements
- New column STMTTOKEN in PLAN_TABLE contains the statement token value for the Statement being explained

PLAN_TABLE – How do I identify a cached statement?

DSN_STATEMENT_CACHE_TABLE

(Key columns)

- STMT_ID
- STMT_TOKEN
- COLLID
- CACHED_TS
- EXPLAIN_TS
- STMT_TEXT

PLAN_TABLE

(Key Columns)

- QUERYNO
- STMT_TOKEN
- COLLID
- BIND_TIME

STMT_ID – Token identifying a cached statement
STMT_TOKEN –Identification token set by user (null if not populated)
COLLID – DSNDYNAMICSQLCACHE to indicate cached statement
CACHED_TS – Timestamp when statement was inserted to SQL cache
EXPLAIN_TS – Timestamp when EXPLAIN STMTCACHE ALL/STMTID was run
(Multiple EXPLAIN_TS possible for the same Statement)
STMT_TEXT – SQL Text

QUERYNO – Contains STMT_ID from the dynamic SQL Cache
STMT_TOKEN – Contains STMT_TOKEN from the dynamic SQL cache
COLLID – DSNDYNAMICSQLCACHE to indicate statement being explained is from the dynamic SQL cache
BIND_TIME – CACHED_TS from dynamic SQL cache
(Duplicate rows possible when EXPLAIN STMTCACHE STMTID executed multiple times)

How do I reduce the volume of DISTSERV SMF101s

Too many SMF101 records for DISTSERV?

- Most installation of DB2 use type 2 inactive connection support (i.e., CMTSTAT=INACTIVE)
- In DB2 v7, with CMTSTAT=INACTIVE, SMF101 record is cut every time a thread becomes Inactive
- The number of SMF records could exceed several million (Mid-large size shops cut upwards of 35 million+ DB2 SMF101 records in a day for DISTSERV without ROLLUP)

Too many SMF101 records for DISTSERV?

- Large volumes inhibit the ability to extract useful information for DDF/RRSAF threads
- Large volumes result in increased batch run times (for processing SMF records)
- What does V8 provide?

SMF 101 records rollup- V8

- New dynamic ZPARM ACCUMACC, ACCUMUID rolls-up accounting information from DDF/RRSAF threads
- Can be turned on/off dynamically
- Allows the DBA to obtain detailed information if needed without disruption
- Accounting data accumulated by either, or combination of
  - User ID
  - Transaction name
  - Workstation name

SMF 101 records rollup- V8

- How do I set the ZPARMS and register used for roll-up?
  - ACCUMACC
  - ACCUMID
  - User ID
  - Transaction name
  - Workstation name

SMF 101 records rollup- V8

- ZPARM– “DDF/RRSAF Accum” ACCUMAC (DSNTIPN)
- Controls whether DB2 accounting data should be accumulated for DDF and RRSAF threads. The type of accumulation is controlled by ZPARM ACCUMUID.

SMF 101 records rollup- V8

- ZPARM– “DDF/RRSAF Accum” ACCUMAC (DSNTIPN)
  - If “10” is specified (the default), then DB2 continues to write an accounting record for ’10’ occurrences of the “Aggregation field”
  - If “NO” is specified, an accounting record is cut when a DDF thread is made inactive, or when signon occurs for an RRSAF thread.
  - If 2-65535 is specified, then DB2 writes an accounting record every 'n' occurrences of the “Aggregation field” on the thread, where 'n' is the number specified for this parameter.
  - “Aggregation field” defined by ACCUMUID

SMF 101 records rollup- V8

- ACCUMUID – ZPARM – Defines the “Aggregation Fields” to be used for DDF and RRS accounting record rollup.
- Aggregation is based on the following three fields that are provided to the application:
  – ID of the end user (QWHCEUID, 16 bytes)
  – End user transaction name (QWHCEUTX, 32 bytes) contains the name of the executable
  – End user workstation name (QWHCEUWN, 18 bytes)

SMF 101 records rollup- V8

- Client accounting fields can be set using
  - "Server Connect" and "Set Client" (SQLESETI) calls
  - RRSAF threads via the RRSAF SIGN, AUTH SIGNON, and CONTEXT SIGNON functions
  - Java programs when using the new Java Universal Driver

SMF 101 records rollup- V8

- ACCUMUID values:
  - 0 : End user ID, AND end user transaction name, AND end user workstation name
  - 1 : End user ID
  - 2 : End user transaction name
  - 3 : End user workstation name
  - 4 : End user ID AND end user transaction name
  - 5 : End user ID AND workstation name
  - 6 : End user transaction name AND workstation name
- The default value is 0 (zero). The ACCUMUID value is ignored if ACCUMACC=NO (no DDF/RRS rollup).

SMF 101 records rollup- V8

- DB2 can override ACCUMACC in certain situations:
  - Storage threshold is reached for the accounting rollup blocks
  - No updates have been made to the rollup block for 10 minutes
  - Not all of the fields specified in the aggregation criteria are supplied
- May provide relief to the “SMF flooding” issue
- May want to check CPU changes
- ACCUMUID value with a extremely large number of combinations is usually not a good option

SMF 101 records rollup

- Enhances the ability to exploit accounting information
- Example:
  - Significantly reduces the amount of time taken to run performance reports (Up to 90% reduction in processing time for DISTSERV plans)
  - Rolling up/drilling down information by “client identifiers” allows the DBA to drill down and obtain CPU utilization information at a more granular level than just the AuthID
- IFCID 239 provides significant amount of package level data (buffer manager, lock manager, and SQL statistics accumulated at the package level)
  - Overhead associated with collection
  - Accounting Class(10) controls collection of additional data (PK28561)

DB2 v8 and DB2 9 Enhancements

Other DB2 v8 Enhancements

- Dynamic SQL statements – like static SQL was limited to 32KB – Version 8 – limit changed to 2 MB
- “Execute Immediate” can use a LOB/CLOB
- Entire statement information is provided by IFCID 350 –similar to IFCID 63
- Invalidate dynamic SQL cache - RUNSTATS UPDATE NONE REPORT NO, CREATE INDEX
- New ZPARM EDMSTMTC – Controls size of dynamic SQL cache

DB2 9 Enhancements

- Enhancement to DSNRLST based on client variables
- REOPT AUTO enhancement
- -START TRACE enhancement
- Trusted Context – Prevent the application server ID from getting used outside of the app server

Using RLF enhancements – DB2 9

- DB2 9 allows the RLF to be set at a more granular level then AUTHID/PLAN/PACKAGE
  - For dynamic SQL PLAN is always DISTSERV which does not provide adequate granularity
  - Utilizes a table DSNRLMTxx – Middleware resource limit facility
    - Allows the option to specify
      - RLFEUAN – End User application name
      - RLFEUWN – End user workstation name
      - RLFEUID – End User ID
      - RLFIP – IP from which dynamic SQL originated
    - START RLIMIT starts RLST as well as the RLMT tables
    - Only one RLMT table can be active at any point in time

REOPT(AUTO) – DB2 9

- Enhancement to REOPT – REOPT(AUTO)
  - DB2 determines if a new access path based on the parameter values
  - DB2 saves the access path in the dynamic SQL Cache
  - Combines the advantages of REOPT(ALWAYS) and REOPT(ONCE)

-START TRACE enhancements (DB2 9)

- -START TRACE has new keywords
  - USERID - End Client UserID
    APPNAME - Application Name
    WRKSTN - Client Workstation Name
    CONNID - Connection
    CORRID - Correlation ID
  - Constraint block modified to include specific end client User ID, Application name, workstation name, connection, or correlation ID
  - Filtering block allows XUSERID, XAPPNAME, XWRKSTN, XCONNID, XCORRID – To exclude threads matching specific end clients
  - X option allowed with wildcards but a x(*) is not allowed

TRUSTED CONTEXT

- Addresses security/audit challenges using a common middleware/functional ID to connect to DB2
  - Database entity which allows for a unique set of interaction based on Authorization ID and connection attributes
  - Two types of trusted context
    - Implicit – No application changes are needed
    - Explicit – Application changes are needed
  - Allows Authorization IDs to connect from a specific IP address without authentication and assign default roles
  - Explicit trusted context allows for user ID switching
  - Switching User IDs allows the actual user to be used for logging all database changes

TRUSTED CONNECTION

- Addresses challenges using a common middleware/functional ID to connect to DB2
  - ROLE is a Database entity to which authorities can be granted
  - TRUSTED CONNECTION is a database connection established in a TRUSTED CONTEXT
  - Misuse of application IDs can be alleviated by granting security on DB2 objects to roles
  - Common application/functional IDs can be granted the ROLE under a TRUSTED CONTEXT

Performance Warehouse

Performance Warehouse

- Performance warehouse is a collection of DB2 performance data
  - Performance information (Summary and exception) at a Plan, Package, SQL level stored in custom tables serves as a tuning repository
  - Plan/Package Information can be extracted from SMF 101 (IFCID 3 and IFCID 239 records)
  - Information from DB2 augmented with user specific information
  - Tables for Top “n” dynamic and static SQL
  - SQL level data extracted from SQL level monitoring tools such as IBM DB2 Query Monitor, Dynamic SQL cache, DB2/OSC monitor profiles (DB2 9 onwards)

Performance Warehouse

- Using Enhancements in DB2 v8/9 for collecting SQL level performance data
  - DB2 v8 – Top ‘n’ SQL (CPU, elapsed time, GETPAGES) - EXPLAIN STMTCACHE ALL output can be parsed to populate custom built SQL cache tables
  - DB2 9 – Using profile monitor - Statements which leave the cache or statements (static/dynamic) which exceed thresholds are written to DSN_STATEMENT_RUNTIME_INFO table

Summary

- V8/9 – Provides a plethora of tools to “tame” dynamic SQLs
  - Special registers
  - SQLESETI/JDBC/RRS Signon to set client variables
  - ACCUMAC/ACCUMID to reduce SMF records
  - IFCID 350
  - RUNSTATS REPORT NO/UPDATE NONE to invalidate dynamic SQL cache
  - EXPLAIN STMTCACHE
  - ZPARM EDMSTMTC
  - -START TRACE, DSNRLMT, REOPT(AUTO), Profile Monitor