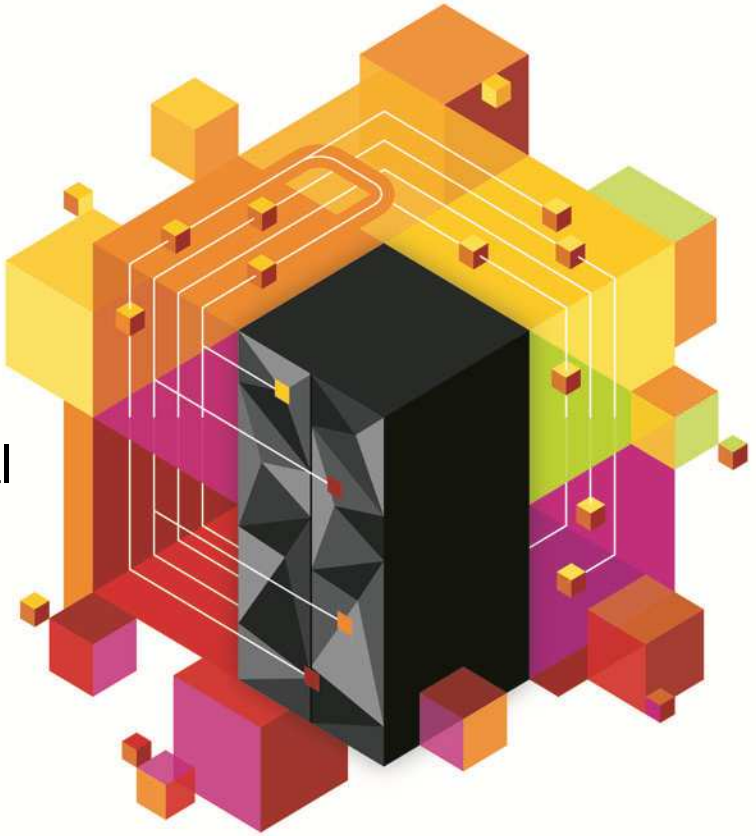
DB2 Stored Procedures Monitoring, Analysis, and Tuning on System z

Charles Lewis, DB2 Advisor

IBM System z Software Technical Professional

September 11, 2013



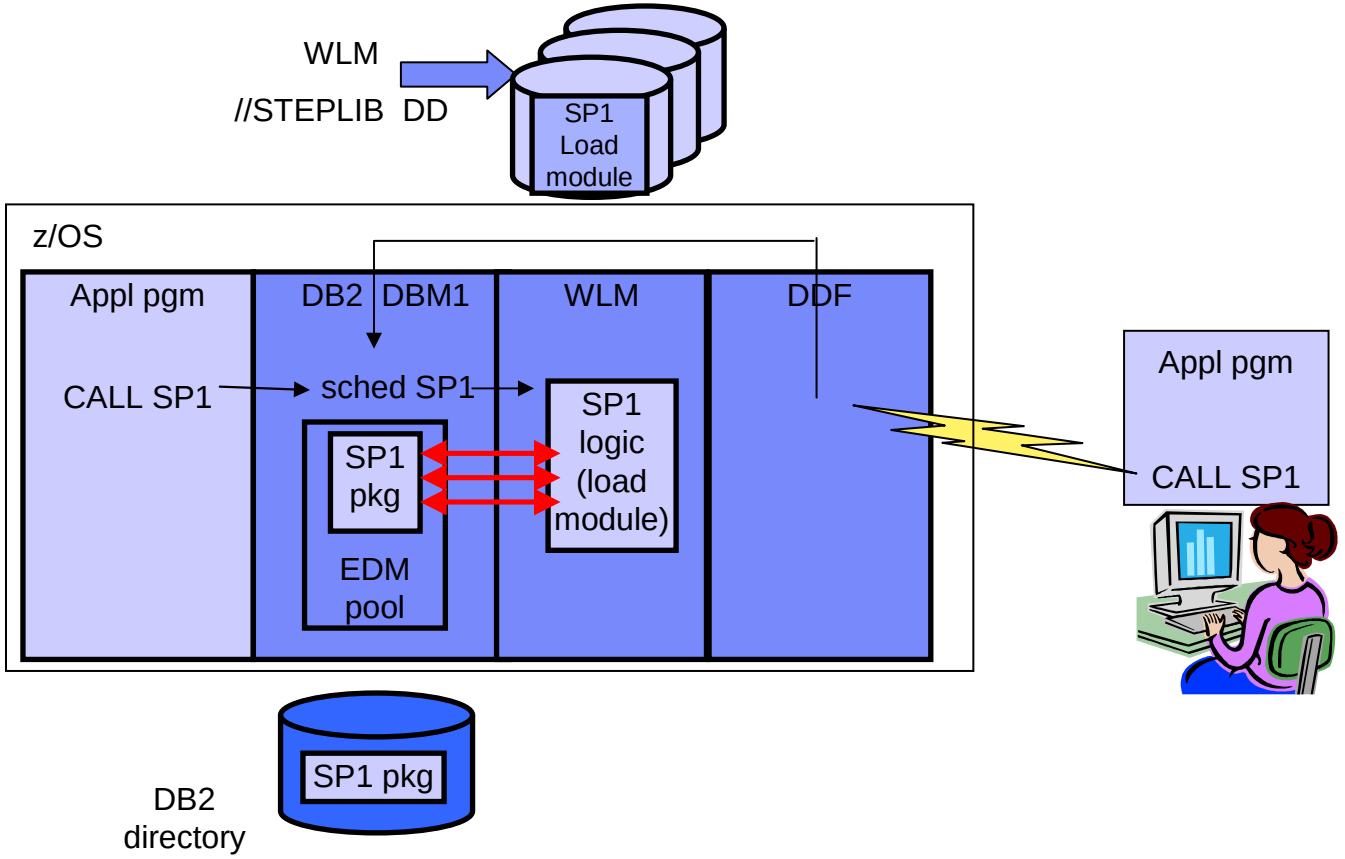
Agenda

- **What are stored procedures?**
  - Benefits of stored procedures
  - Stored procedure analysis – Issues and solutions
- **Monitoring stored procedures**
- **Tuning stored procedures**
- **Summary / resources for more information**

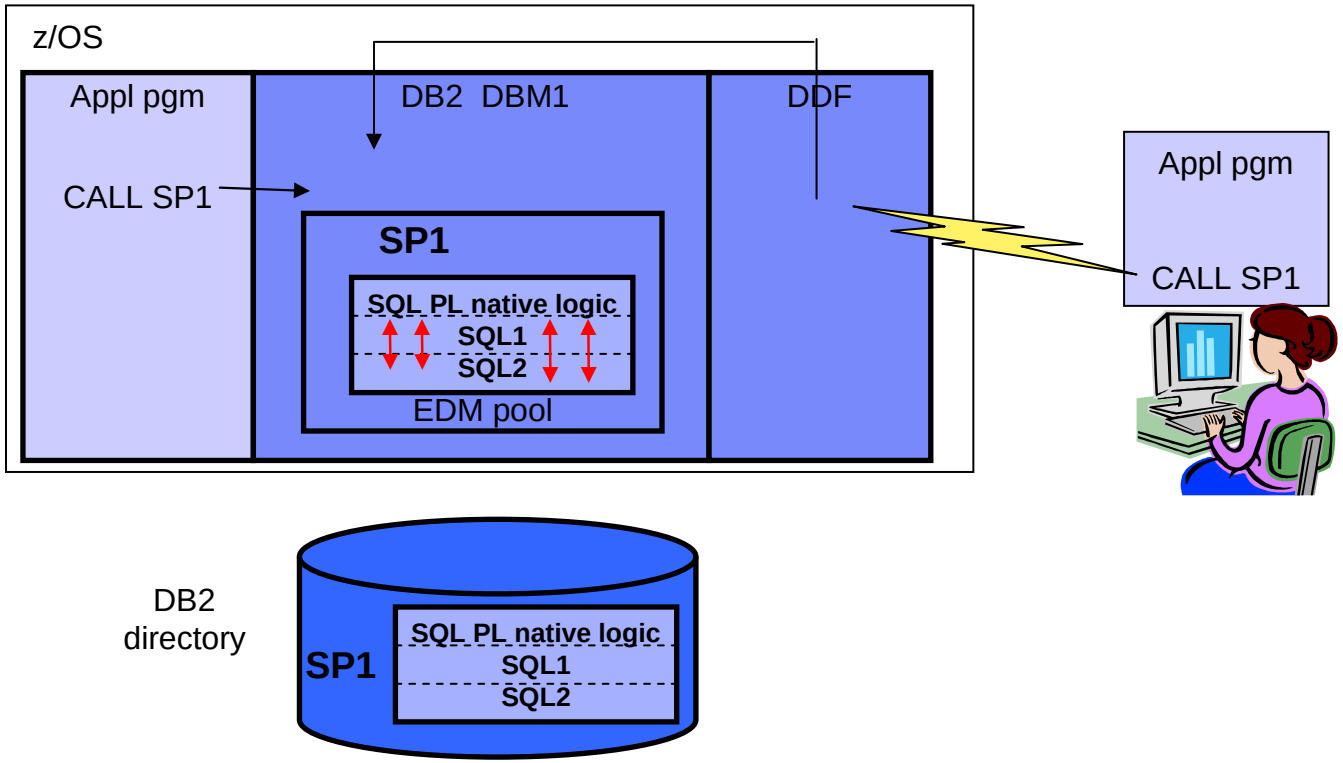
What are Stored Procedures?

- **A stored procedure is a user-written program that can be called by an application with an SQL CALL statement.**
- **It is a compiled program that is stored at a DB2 server**
- **It can execute business logic and SQL statements**
- **Stored procedure types**
  - External high level language procedures COBOL, PL/I, C, C++, Assembler, REXX, and Java
  - External SQL procedures
  - Native SQL procedures introduced by DB2 9 for z/OS

DB2 z/OS Stored Procedure Processing (External)



Native SQL Procedure Processing (Internal)



Programming Benefits of Stored Procedures

- **Modularity in application development**
- **Data will be processed always in a consistent way according to the rules defined in the stored procedure**
- **Enforcement of business rules**
  - You can use stored procedures to define business rules that are common to several applications.
  - can be an alternative to using constraints and triggers.
- **Improved application security**
  - Sensitive business logic runs on the DB2 server
  - End users are authorized to execute a stored procedure,
    they do not need table privilege -> similar to static authorization model
- **Application integration solutions**
  - can access non-DB2 resources
    e.g. VSAM files, MQ queues, IMS or CICS transactions
  - Stored procedures can have access to commands that run only on the server.

Total Cost of Ownership Benefits of Stored Procedures

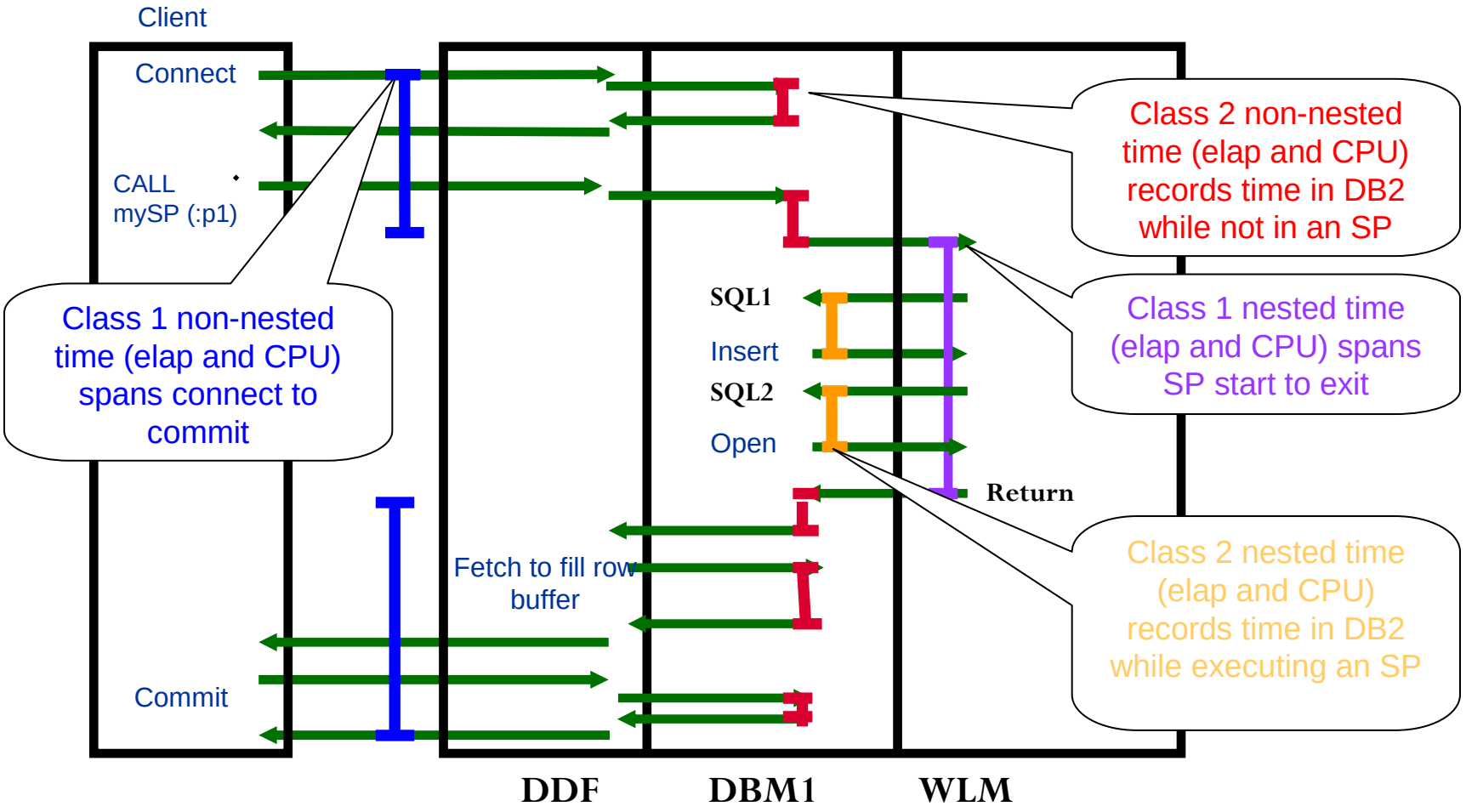
- **Reduced network traffic for distributed applications**
  - Grouping SQL statements into a stored procedure results in two trips across the network for each group of statement, resulting in better performance for applications
- **Cost of ownership reduction**
  - If stored procedure is called from distributed client via DRDA, a portion is eligible for zIIP redirect.
    - Including: Call statement processing; Result set processing; Commit processing
  - Stored procedures written in Java can take advantage of zAAP engines
  - Native SQL procedures run as enclave SRB in DBM1 address space and the Stored Procedure execution itself is zIIP off-loadable with DB2 9 for z/OS.
  - For WLM managed stored procedures:
    - SQL processing runs under a TCB hence not eligible for zIIP redirect
- **As of now, there is NO performance benefit for calling a Stored Procedure from a local application**

Stored Procedure Language / API CPU Cost comparison - Update

- **IRWW workload (OLTP workload consisting of 7 transactions)**
- **Called from distributed JCC type 4 client**

Language/API	Base CPU/Tran Cost	Billable CPU/Tran Cost after zIIP and/or zAAP redirect
COBOL Stored Procedure	**1X (BASE)**	0.80x (Some zIIP)
C Stored Procedure	1.02x	0.82x (Some zIIP)
SQLJ Stored Procedure	2.01x	1.11x (zAAP+ some zIIP)
JDBC Stored Procedure	2.97x	1.84x (zAAP+ some zIIP)
Native SQL Stored Procedure	**1.09x**	**0.59x (Significant zIIP)**

Performance Reporting – External Stored Procedure



External Stored Procedure Performance Summary - Plan-Level

- DB2 Accounting class 1 and 2 needed (3 is recommended)

AVERAGE	APPL(CL.1)	DB2 (CL.2)
ELAPSED TIME	0.003212	0.002575
NONNESTED	**0.000714**	**0.000694**
STORED PROC	**0.002498**	**0.001881**
UDF	0.000000	0.000000
TRIGGER	0.000000	0.000000
CP CPU TIME	0.000715	0.000654
AGENT	0.000715	0.000654
NONNESTED	**0.000149**	**0.000129**
STORED PRC	**0.000567**	**0.000525**
UDF	0.000000	0.000000
TRIGGER	0.000000	0.000000
PAR.TASKS	0.000000	0.000000

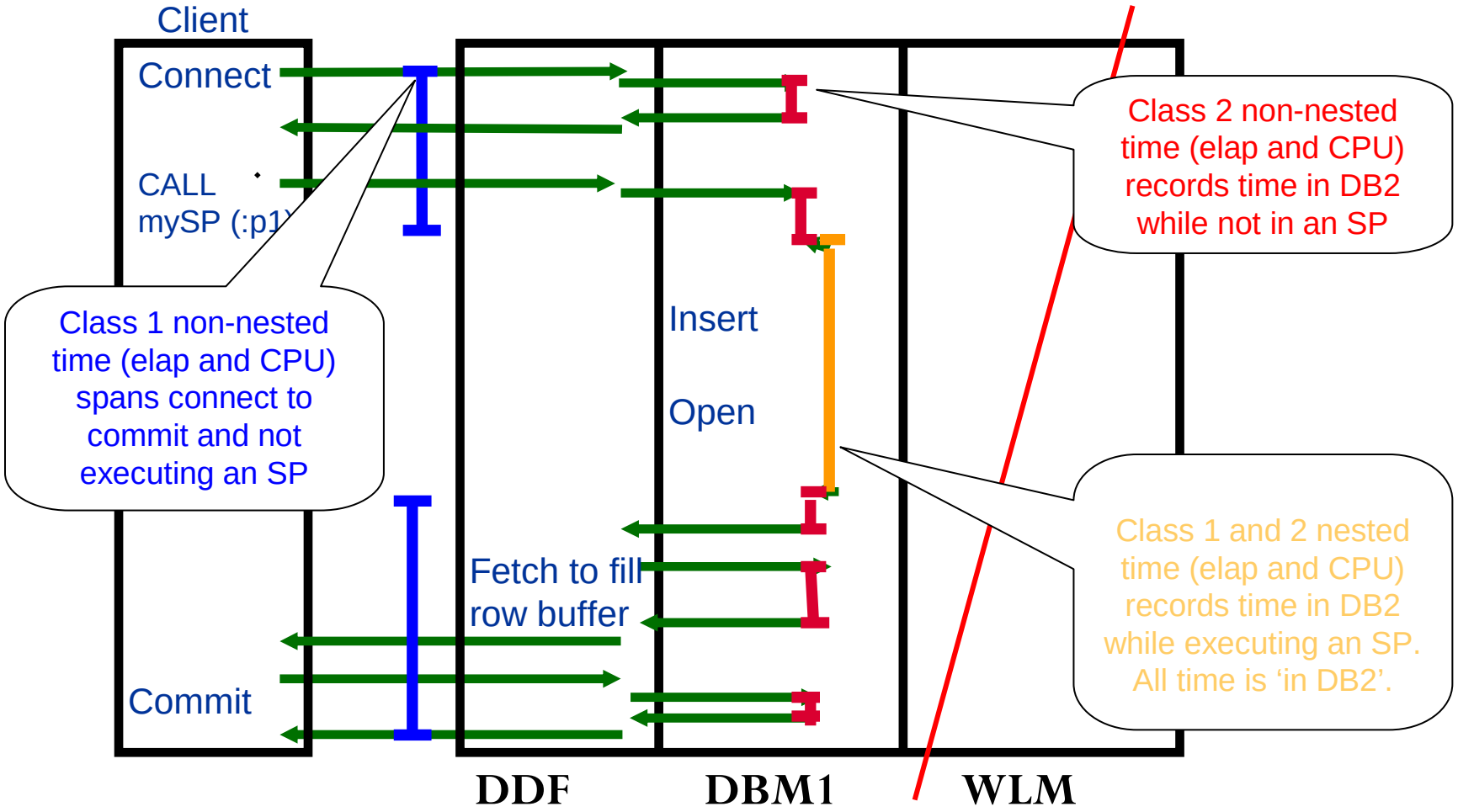
Class 1 non-nested time (ET & CPU)

Class 2 non-nested time (ET & CPU)

Class 1 nested time (ET & CPU)

Class 2 nested time (ET & CPU)

Performance Reporting – Native SQL Stored Procedure



Native SQL Stored Procedure Performance Summary - Plan-Level

- DB2 Accounting class 1 and 2 needed (3 is recommended)

AVERAGE	APPL(CL.1)	DB2 (CL.2)
ELAPSED TIME	0.004834	0.002789
NONNESTED	**0.002819**	**0.000774**
STORED PROC	0.002015	0.002015
UDF	0.000000	0.000000
TRIGGER	0.000000	0.000000
CP CPU TIME	0.000963	0.000909
AGENT	0.000963	0.000909
NONNESTED	**0.000198**	**0.000143**
STORED PRC	0.000765	0.000765
UDF	0.000000	0.000000
TRIGGER	0.000000	0.000000
PAR.TASKS	0.000000	0.000000

CL1 and CL2 will always be equal!

CL1 and CL2 will always be equal!

Stored Procedure Detail Reporting - Package level Reporting

- **Accounting class 7 and/or 8 needed**
- SYSSTAT package contains time for CALL statement, result set processing, SET special registers, and VALUES statements for LOB handling

SYSSTAT	VALUE	SYSSTAT	TIMES
TYPE	PACKAGE	ELAP-CL7 TIME-AVG	0.000387
		CP CPU TIME	0.000072
LOCATION	DSND91B	AGENT	0.000072
COLLECTION ID	NULLID	PAR.TASKS	0.000000
PROGRAM NAME	SYSSTAT	SE CPU TIME	0.000000

NSQLNEW	VALUE	NSQLNEW	TIMES
TYPE	PACKAGE	ELAP-CL7 TIME-AVG	0.004751
		CP CPU TIME	0.001667
LOCATION	DSND91B	AGENT	0.001667
COLLECTION ID	USRT001	PAR.TASKS	0.000000
PROGRAM NAME	NSQLNEW	SE CPU TIME	0.000000

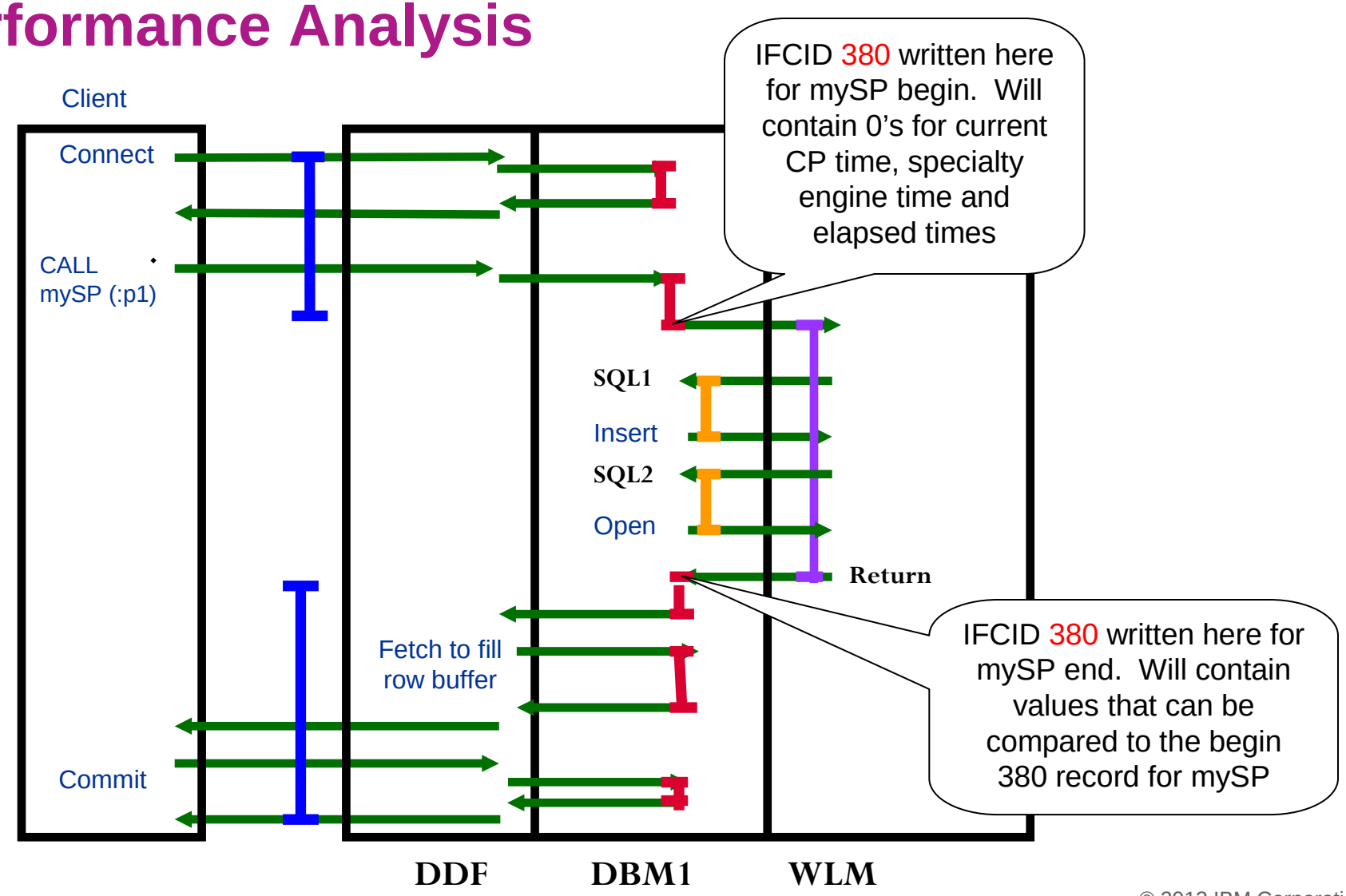
Issues with Plan and Package Level Stored Procedure Analysis

- Multiple Stored Procedures called in a transaction are summed at the plan level. By definition this affects the analysis of nested SPs.
- Package level analysis can be difficult if an Stored Procedure execute different paths and SQL based on parameters. How do you differentiate between the invocations?
- Package level analysis does not apply to Stored Procedures that do not execute SQL

Enhanced Instrumentation for Stored Procedure Performance Analysis

- **PM53243 (DB2 10) New IFCIDs 380 and 381 are created for Stored Procedure and User-Defined Function detail respectively. These records:**
  - Identify the stored procedure or UDF beginning or ending
  - Include the current CP, specialty engine, and elapsed time details for nested activity
- **These record can be used to determine the CP, specialty engine, and elapsed time for a given Stored Procedure or UDF invocation**

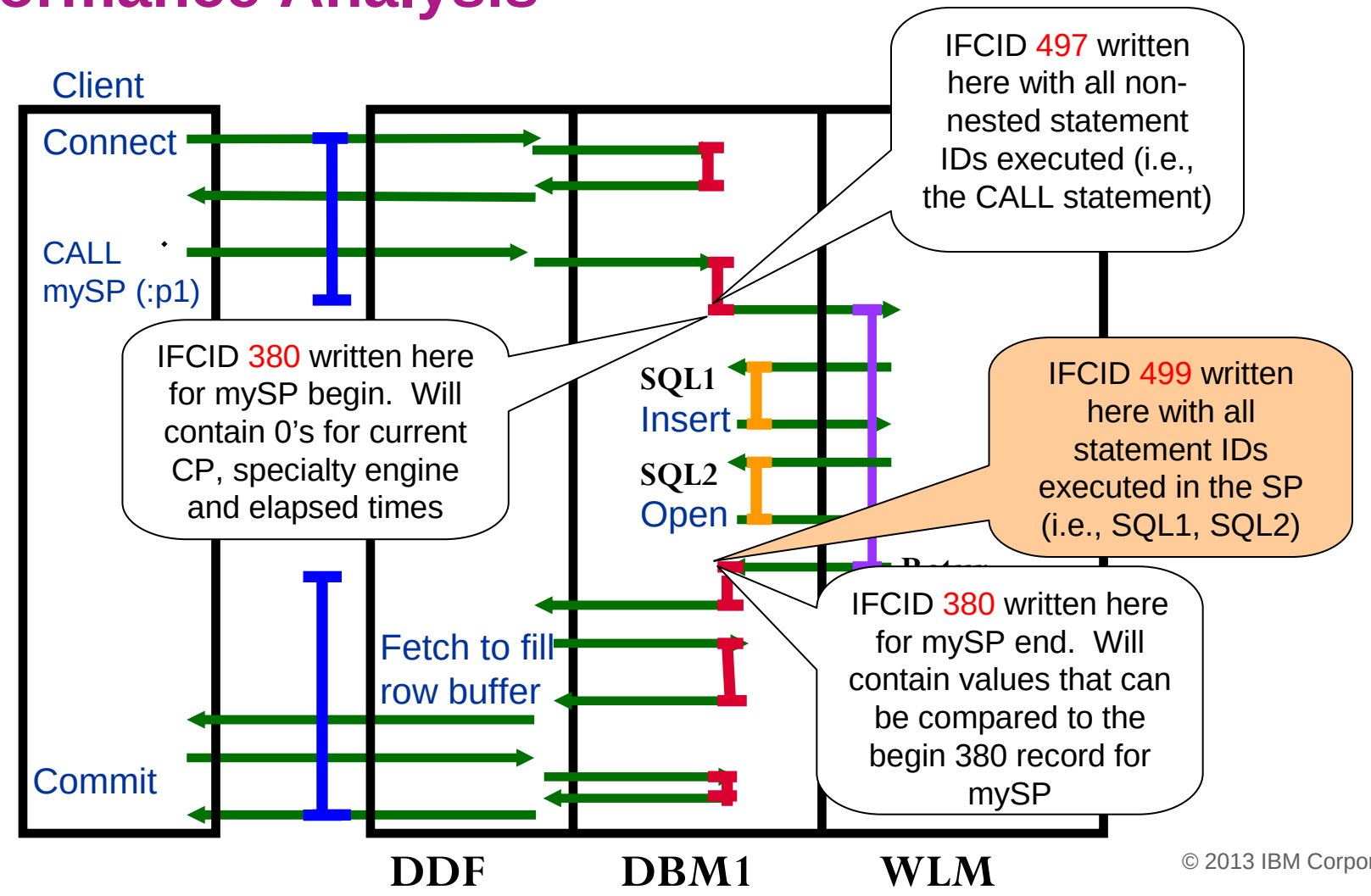
Enhanced Instrumentation for Stored Procedure Performance Analysis



Enhanced Instrumentation for Stored Procedure Performance Analysis

- Additionally PM53243 (DB2 10) added IFCID 497, 498, 499 for SQL drill down analysis. These records contain the dynamic or static statement IDs for non-nested, UDF, and SP work respectively.
- The statement IDs can be correlated to IFCID 316 dynamic statement or IFCID 401 static statement cache data.

Enhanced Instrumentation for Stored Procedure Performance Analysis

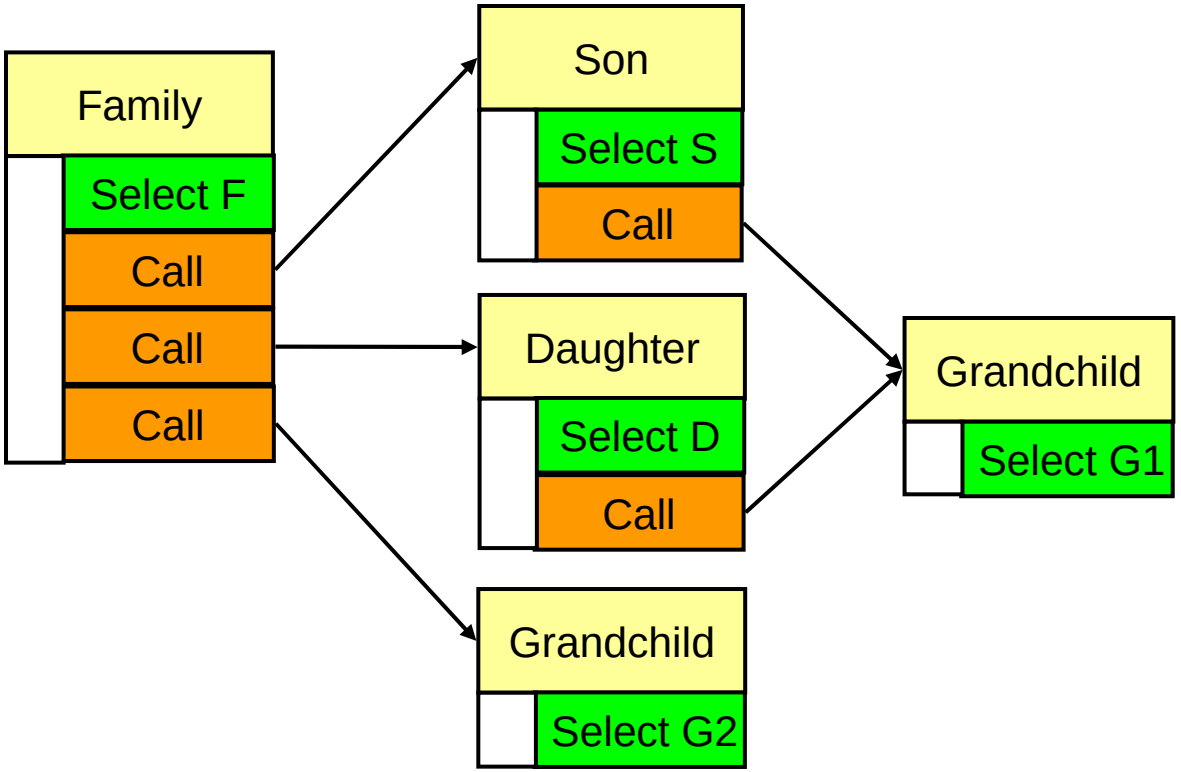


Monitoring Stored Procedures using new trace records

- The new DB2 instrumentation records for Stored Procedures are ingested by the OMEGAMON Collector, aggregated on a system level and returned to the (Optim Performance Monitor) Repository Server.
- The OMEGAMON Collector processing includes the sequencing logic and the calculation of elapsed times for the different accounting class times written in the IFI records as timestamps, considering nesting as well.
- In parallel the IFCID 316/401 data for the Statement Caches is collected and a correlation to the executed stored procedure statements via IFCID 499 is made.
- Full RECTRACE support for all new IFCIDs is provided

Stored Procedures analysis – sample scenario

- **Workload:**

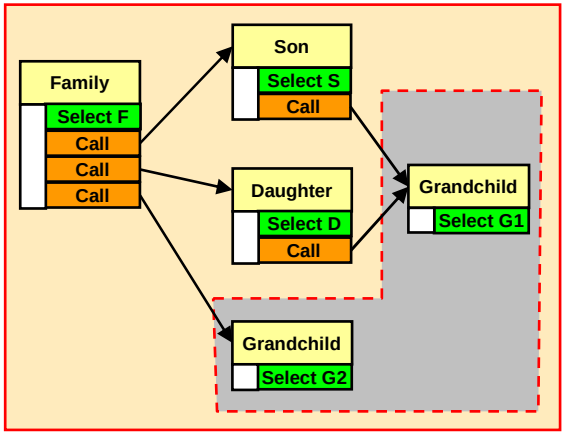


SQL Dashboard – aggregation by ROUTINEID

- Workload at SQL dashboard ("All statements" view) executed in the selected time period (time slider), valid for all subsequent views

Σ of Family

Σ of Grandchild



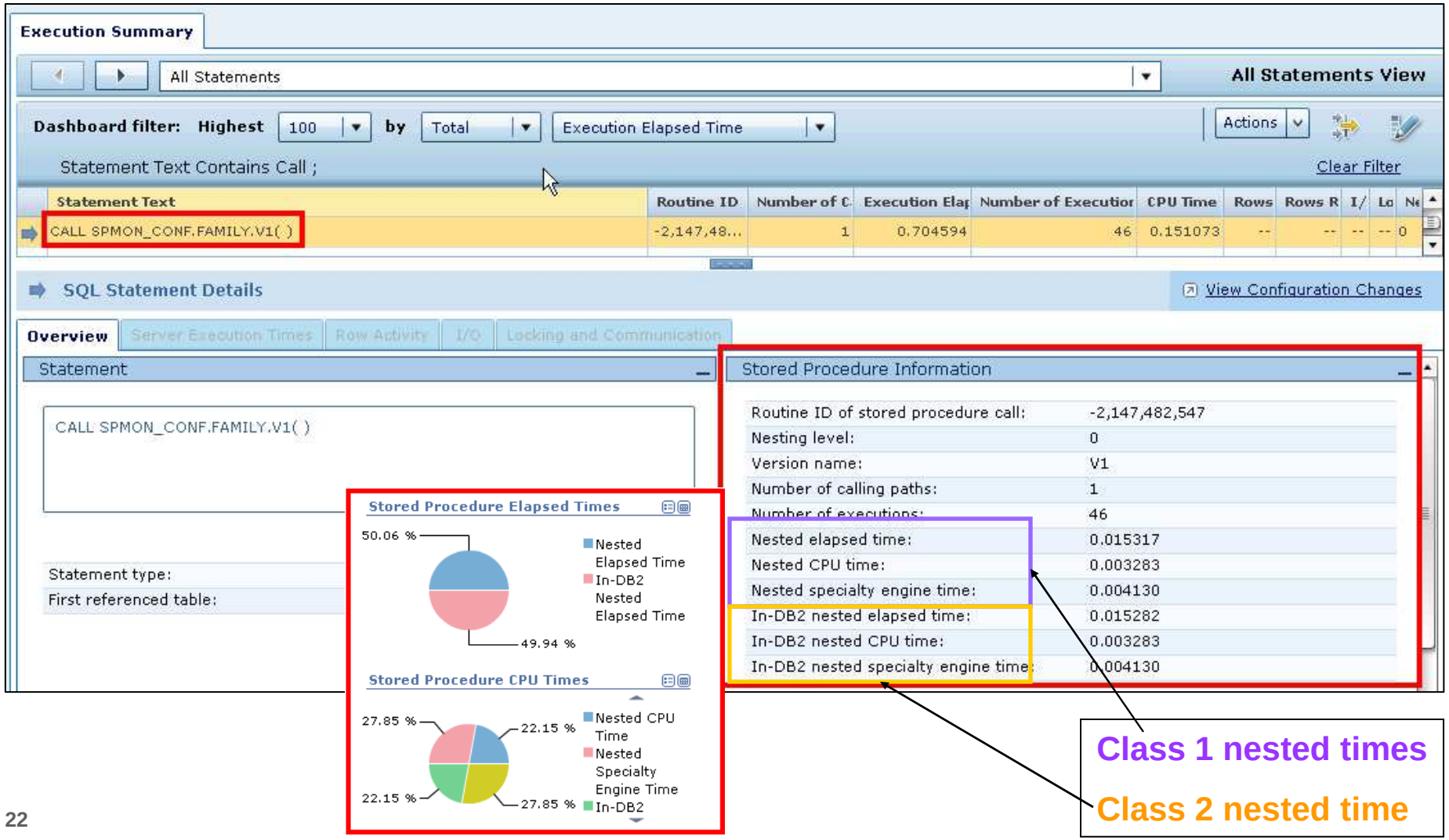
All Statements | All Statements View

Dashboard filter: Highest 20 by Total Execution Elapsed Time | Actions

Statement Text Contains call ; Clear Filter

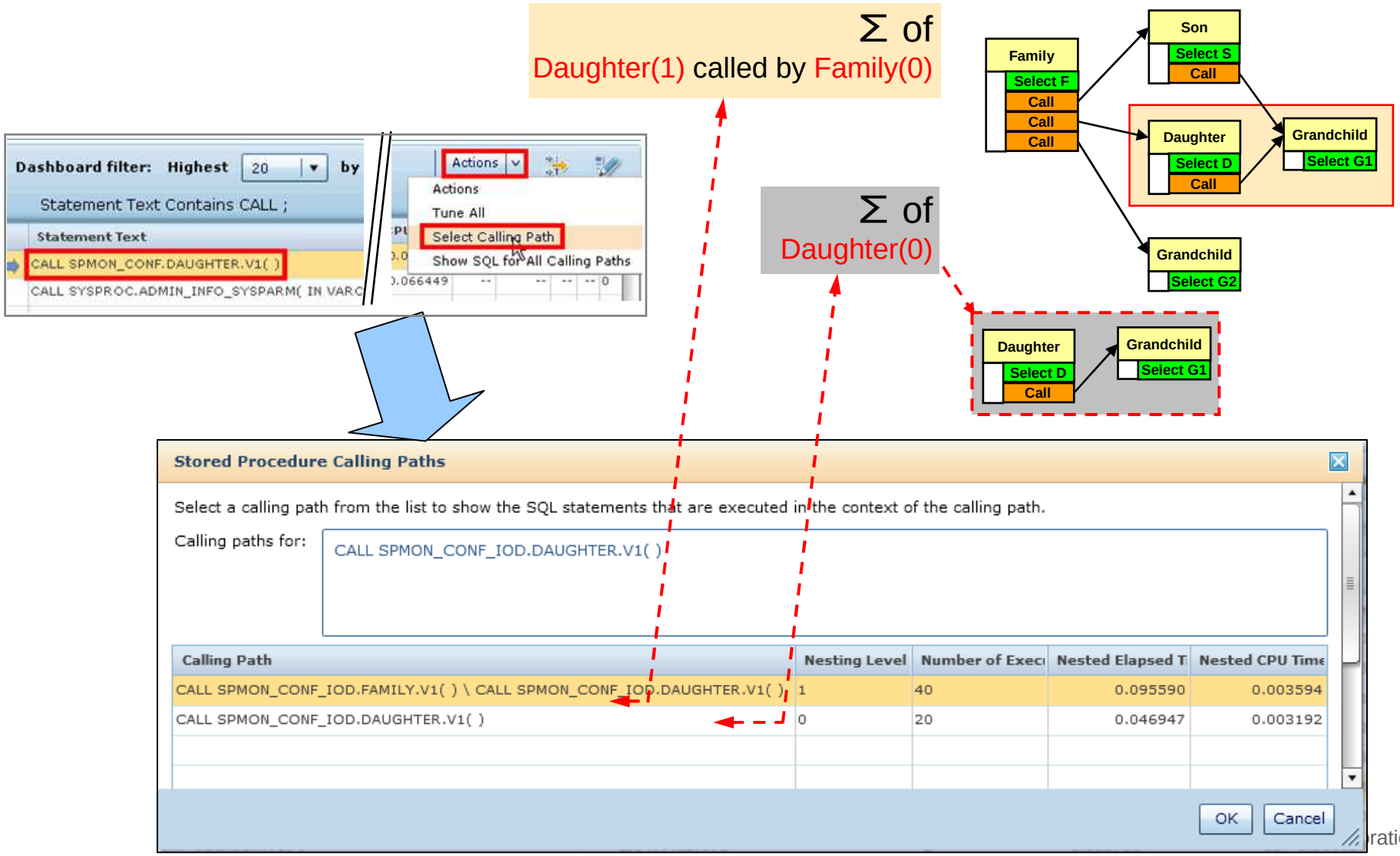
Statement Text	Routine ID	Number of Calling Paths	Execution Elapsed Time	Number of Execu	CPU Time
CALL SYSIBM.SQLPROCEDURECOLS(IN VARCHAR, IN VARCHAR, IN ...	-2,147,483,102	1	1.160051	168	0.148540
CALL SPMON_CONF_IOD.FAMILY.V1( )	-2,147,482,976	1	0.403588	40	0.018785
CALL SYSPROC.ADMIN_COMMAND_DB2(IN VARCHAR, IN INTEGER, I...	-2,147,483,148	2	0.372614	13	0.065811
CALL SYSPROC.ADMIN_INFO_SYSPARM(IN VARCHAR, OUT INTEGER,...	-2,147,483,134	1	0.360020	2	0.033512
CALL SYSIBM.SQLPROCEDURES(IN VARCHAR, IN VARCHAR, IN VAR...	-2,147,483,101	1	0.268017	84	0.051966
CALL SPMON_CONF_IOD.DAUGHTER.V1( )	-2,147,482,977	2	0.142537	60	0.006785
CALL SPMON_CONF_IOD.GRANDCHILD.V1( )	-2,147,482,979	6	0.108440	164	0.005870
CALL SPMON_CONF_IOD.SON.V1( )	-2,147,482,978	2	0.083759	52	0.009528
CALL OPM.DB2MON_LOC.V1(OUT VARCHAR)	-2,147,482,972	1	0.023140	1	0.004644

Showing Stored Procedure Details



Showing calling paths of Stored Procedures (1/2)

- Select Calling Path for Daughter



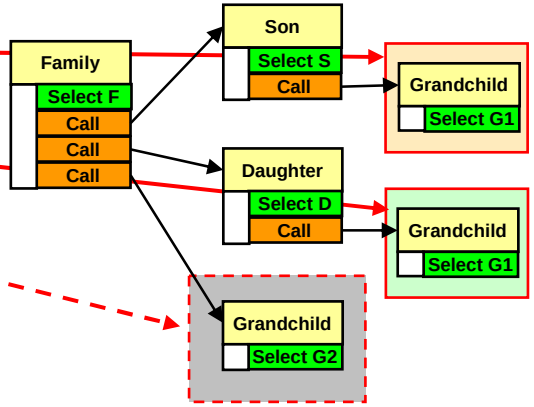
Showing calling paths of Stored Procedures (2/2)

- Select Calling Path for Grandchild

Σ of Grandchild(2) called by Family(0)+Son(1)

Σ of Granchild(2) called by Family (0)+Daughter(1)

Σ of Grandchild(1) called by Family (0)



Stored Procedure Calling Paths

Select a calling path from the list to show the SQL statements that are executed in the context of the calling path.

Calling paths for: CALL SPMON_CONF_IOD.GRANDCHILD.V1( )

Calling Path	Nesting Level	Number of Execut	Nested Elapsed T	Nested CPU Ti
CALL SPMON_CONF_IOD.FAMILY.V1( ) \ CALL SPMON_CONF_IOD.GRANDCHILD.V1( )	1	40	0.059185	0.001066
CALL SPMON_CONF_IOD.FAMILY.V1( ) \ CALL SPMON_CONF_IOD.SON.V1( ) \ CALL SPMON_CONF_IOD.GRAND...	2	40	0.024043	0.002116
CALL SPMON_CONF_IOD.GRANDCHILD.V1( )	0	12	0.020522	0.000740
CALL SPMON_CONF_IOD.DAUGHTER.V1( ) \ CALL SPMON_CONF_IOD.GRANDCHILD.V1( )	1	20	0.001797	0.000783
CALL SPMON_CONF_IOD.FAMILY.V1( ) \ CALL SPMON_CONF_IOD.DAUGHTER.V1( ) \ CALL SPMON_CONF_IOD....	2	40	0.001689	0.000641
CALL SPMON_CONF_IOD.SON.V1( ) \ CALL SPMON_CONF_IOD.GRANDCHILD.V1( )	1	12	0.001203	0.000524

OK Cancel

Show SQL executed by a Stored Procedure (1/2)

- Action: Show SQL for **This** Calling Path

Select of Family (0)
shows

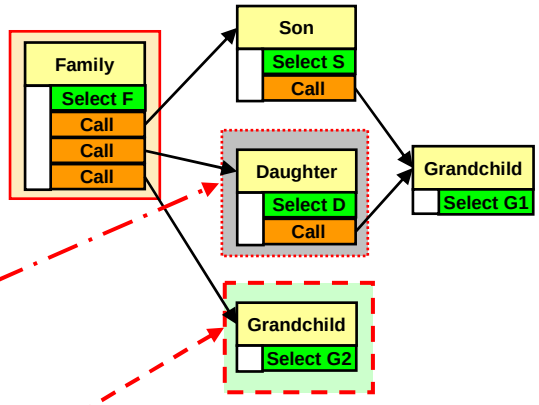
SELECT F *
Σ of Call Son(1) called by Family(0)
Σ of Call Daughter(1) called by Family(0)
Σ of Call Grandchild(1) called by Family(0)

Select of Daughter(1)
shows

SELECT D
Σ of Call Grandchild(2) called by Daughter(1)

Select of Grandchild(1)
shows

SELECT G

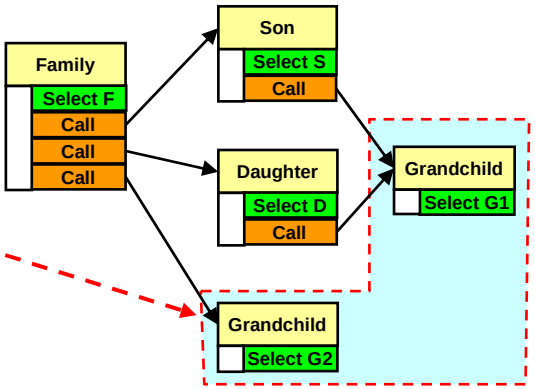


* see next slide

- Action: Show SQL for **All** Calling Paths

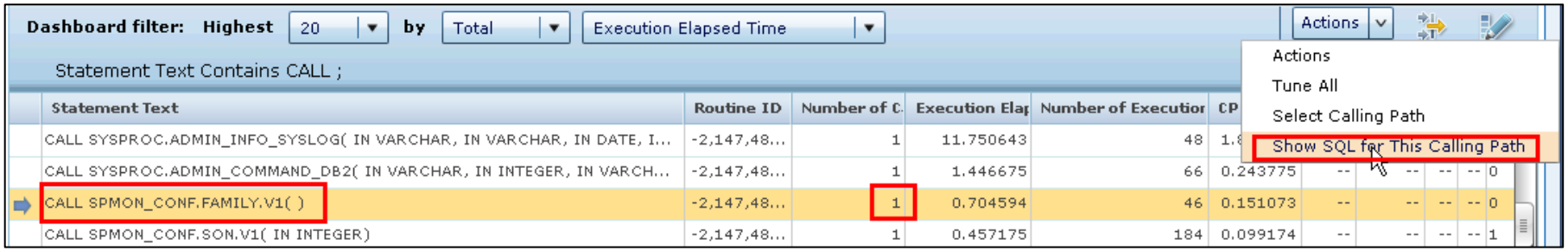
Select Grandchild()
shows Σ

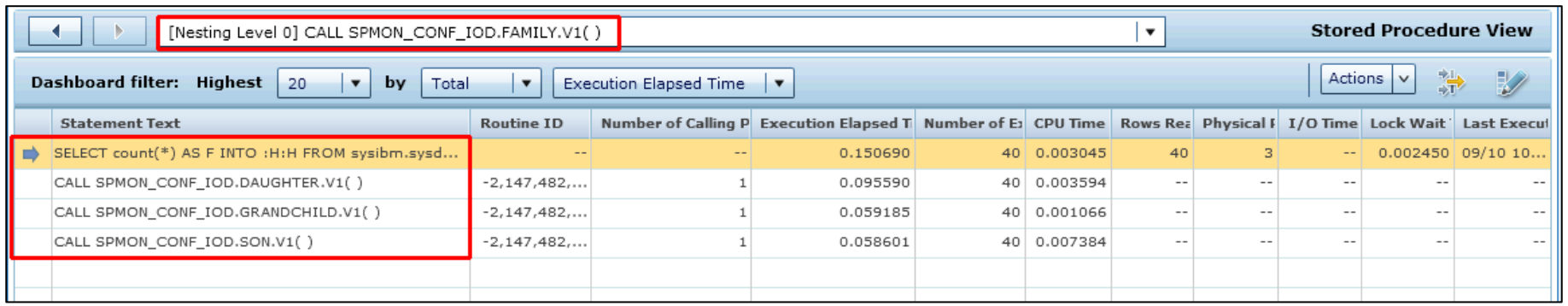
SELECT G1
SELECT G2



Show SQL executed by a Stored Procedure (2/2)

- Show SQL for **This** Calling Path for Family(0)





More Information

- **Websites**
  - DB2 for z/OS home page (http://www-01.ibm.com/software/data/db2/zos/family/)
  - DB2 Tools for z/OS home page (http://www-01.ibm.com/software/data/db2imstools/products/db2-zos-tools.html)
  - Tivoli OMEGAMON XE for DB2 PE on z/OS home page (http://www-03.ibm.com/software/products/us/en/tivoliomegamonxefordb2performanceexpertonzos/)
  - DB2 for z/OS: Information Roadmap (http://www-01.ibm.com/support/docview.wss?rs=64&uid=swg27017080)
- **Other resources**
  - Online demo: Stored procedure monitoring and analysis (http://www.youtube.com/watch?v=fdfEh4Urcjc)
  - eBook: Optimizing database performance through an integrated solution for DB2 (http://public.dhe.ibm.com/common/ssi/ecm/en/imm14094usen/IMM14094USEN.PDF)

Thank you !