All About Db2 for z/OS Packages

Anna McKee anna.mckee@ibm.com
Mark Rader mrader@us.ibm.com

IBM Z Worldwide Systems Center

May 13th, 2026

Agenda

- Packages and plans
- Static vs dynamic SQL in packages
- BIND options
- REBIND phase-in & AUTOBIND phase-in
- Profile tables to control packages
- Db2 package-related monitoring

Packages and Plans

A *package*

- Contains control structures that Db2 uses when it runs SQL statements
  - Control structures are the bound and operational form of SQL statements
- Is produced during program preparation

Packages and Plans

A package is identified by five different attributes:

- Location
  - an identifier that designates the database management system in which the package is stored
- Collection
  - an identifier that denotes a logical grouping of packages
- Name
  - the NAME of the package is automatically generated the same as the DBRM during the BIND
- Version
  - one instance of a package created from a DBRM that was generated with the VERSION keyword in either the pre-compile or the compile step of the DB2 coprocessor
- Consistency token
  - used to check that the two pieces of code generated for a DB2 program can be used together at execution time (precompiled program + package)

Packages and Plans

A *plan*

- Relates an application process to a local instance of Db2
- Specifies processing options
- Contains a list of package names
- Produced during the BIND process

The Db2 catalog records information about plans, packages, and services in:

- SYSIBM.SYSPLAN
- SYSIBM.SYSPACKAGE
- SYSIBM.DSNSERVICE

Packages make programs more flexible and easier to maintain

Plan A

Package A1

Package A2

Package A1: TABLE3
SELECT * FROM ~~TABLE1~~
^

BIND REPLACE Package A1

Binding packages

The bind process establishes a relationship between an application program and its relational data and does the following:

- Validates object references in SQL statements
- Verifies authorization of bind process with program owner
- Selects access paths for Db2 to access program data
- Uses BIND/REBIND PACKAGE commands
- Contains only one database request module (DBRM)

Package benefits

Support for multiple copies of a program

Granularity in bind options

Access path persistence

Ability to support remote access with static SQL

Ability to fall back with access path regression

Ability to use routines

Binding plans

All Db2 programs require an application plan

- To allocate Db2 resources
- To support SQL requests made at run time

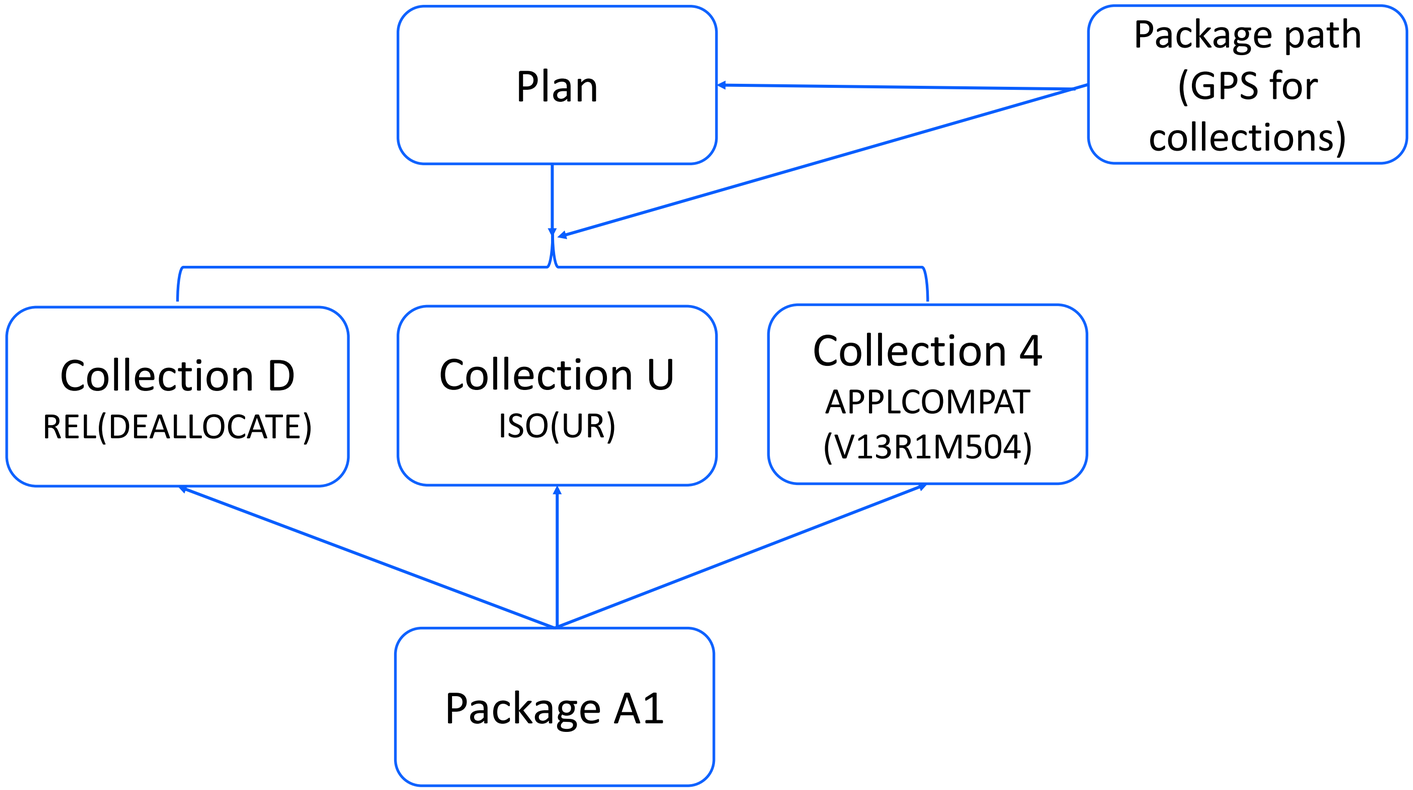
From a Db2 requester, you can run a plan by specifying it in the RUN subcommand

- You cannot run a package directly
- You must include the package in a plan and then run the plan

Packages and plans

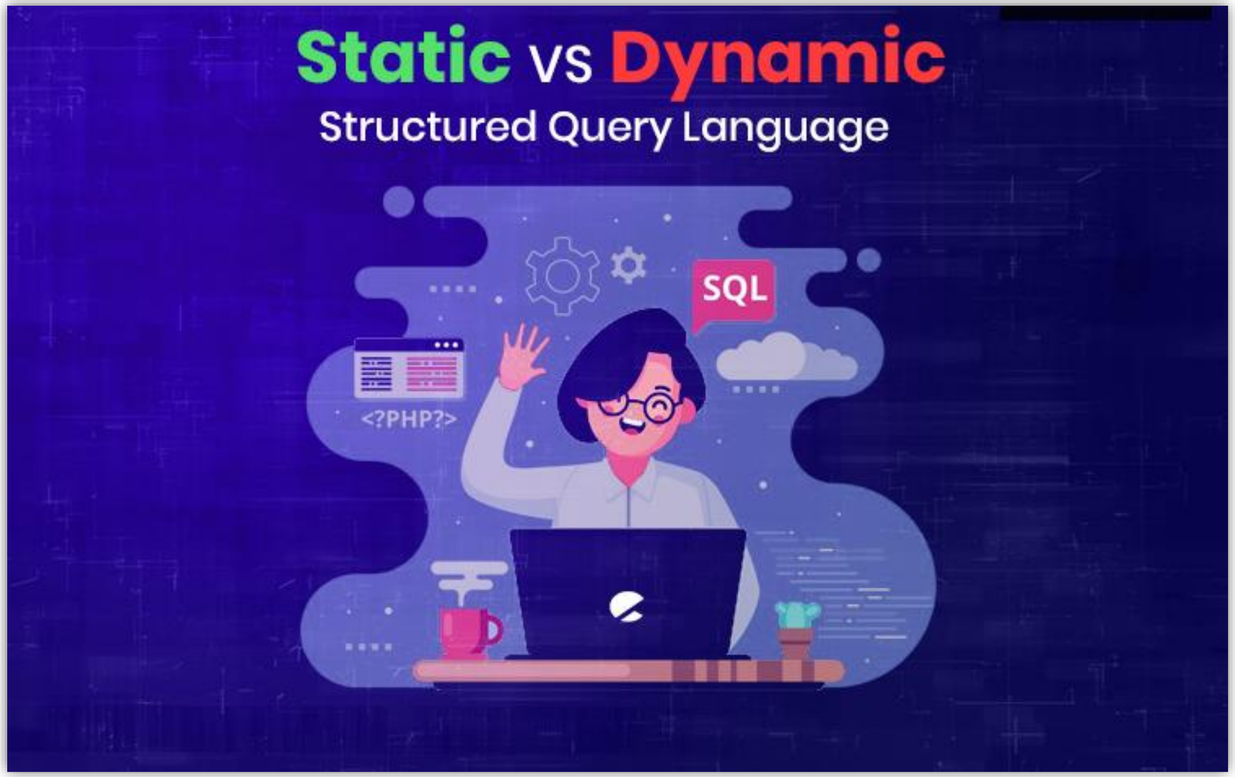
Plans can have many package lists (collections)

- Collections can have many packages
- A package can be bound into more than one collection
  - E.g. BIND ADD or BIND COPY
  - Each collection may have different bind properties



Static vs dynamic SQL in packages

- Static SQL has already been prepared for execution with the BIND process before being executed by a program
- Dynamic SQL is prepared for execution when Db2 first sees it coming in from an application program



Dynamic SQL and parameter markers

- Prepared dynamic SQL statements can be reused if they are cached in the dynamic statement cache (CACHEDYN=YES)
  - Save cost of a full prepare
- What if you repeatedly use one statement where the predicate value changes?
  - Having a literal for that predicate means subsequent SQL will not match the cache as the literal changes

Dynamic SQL and parameter markers

- Instead of using the literal value in your dynamic SQL, you can use a parameter marker – ‘?’ – in place of the literal
  - On each new value, issue the same statement and pass a new value for the parameter
  - Now the statement can match what is in the cache, saving the cost of a full prepare

No parameter markers in your program?

- Must you undergo a full prepare each time your predicate changes?
- No, you can use concentrate statement with literals
  - CSWL for short
  - Not as efficient as parameter markers but better than a full prepare
- With CSWL, if Db2 does not find a match in the statement cache, it will replace the literal with ‘&’ and try again
  - This new version of the dynamic statement added to the cache
  - Now subsequent reuse of the dynamic statement can benefit from the cache and avoid a full prepare
- How do you enable CSWL?

CONCENTRATESTMT bind option

- The CONCENTRATESTMT option specifies whether to enforce statement concentration at the package level
  - CONCENTRATESTMT(NO) specifies that statement concentration is not enabled for the package
  - CONCENTRATESTMT(YES) specifies that statement concentration is enabled for the package
    - This option is a package-level alternative to specifying the ***statement*** attribute CONCENTRATE STATEMENTS WITH LITERALS in PREPARE statements

KEEPDYNAMIC bind option

- The KEEPDYNAMIC option determines whether Db2 keeps dynamic SQL statements after the point of commit or rollback
  - KEEPDYNAMIC (NO | YES)
- KEEPDYNAMIC(YES) keeps prepared statement in thread storage
  - Performance of dynamic SQL can approach that of equivalent static SQL
  - DBATs not pooled and remains active past commit (there is a limit!)
  - Subsystem parameter MAXKEEPD specifies maximum of system-wide storage to use for KEEPDYNAMIC(YES)

APPLCOMPAT bind option

APPLCOMPAT (application compatibility) is a bind option that

- Specifies the behavior for SQL statements in the package
- Enables the use of SQL functionality related to a particular Db2 version or function level
- Provides a shield that protects a program from SQL incompatibility
  - You can migrate your Db2 subsystem or member without worrying about whether your application behavior will change
- The APPLCOMPAT subsystem parameter specifies the default value to use when the APPLCOMPAT bind option is not specified in a BIND command

Isolation levels with package binds

ISOLATION levels determine the degree to which the specified data is locked or *isolated*

- Various *isolation levels* offer less or more concurrency at the cost of more or less protection from other application processes
  - More concurrency = less protection
  - Less concurrency = more protection

ISOLATION levels with package binds

There are 4 main ISOLATION level options for packages

- ISO(CS) is probably most common
  - Cursor stability – releases lock, if read lock was taken, when moving off row/page or at commit if data was changed
- ISO(UR) can be advantageous for queries, as no read locks requested
  - Uncommitted read – risk of reading changes that are later rolled back
- ISO(RR) and ISO(RS) reduce concurrency
  - Repeatable read – requires many read locks
  - Read stability – requires fewer read locks than RR

CURRENTDATA bind option

- The *CURRENTDATA option* of an application specifies whether data currency is required for read-only and ambiguous cursors when the ISOLATION(CS) option is used
- This option enables a trade-off between the improved ability of multiple applications to access the same data concurrently and the risk that non-current data might be returned to the application
- CURRENTDATA(NO) should be used in most circumstances
  - Allows lock avoidance to take place for read-only and ambiguous cursors
  - Allows block fetch of data from remote locations
- CURRENTDATA(YES) requires Db2 to acquire additional page or row locks to maintain currency for read-only and ambiguous cursors

RELEASE bind option

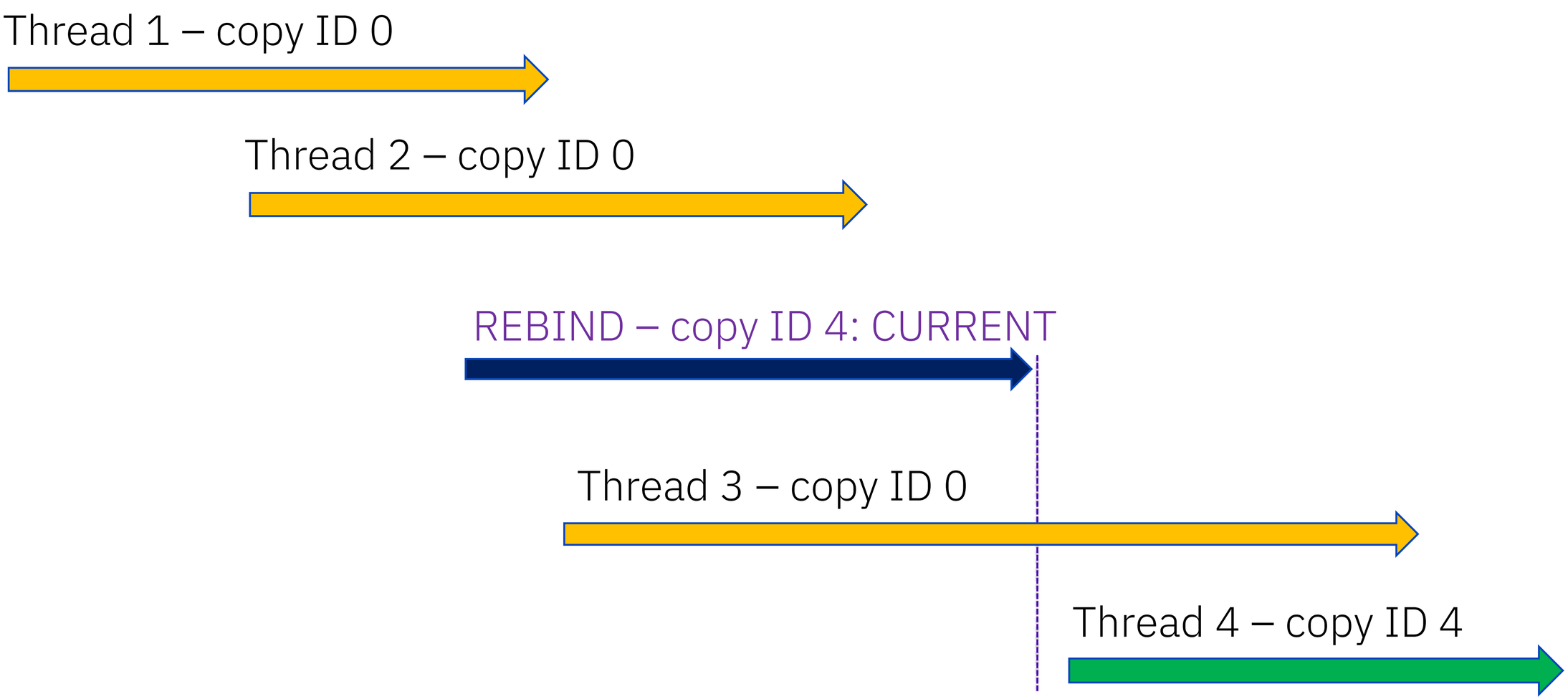
- The RELEASE option determines when to release resources that a program uses, either at each commit point or when the program terminates.
  - RELEASE (COMMIT | DEALLOCATE)
- RELEASE (DEALLOCATE) can reduce CPU for some longer running processes, such as batch jobs or CICS protected entry threads…
  - Save cost of releasing and reacquiring certain resources, such as parent locks
  - RELEASE(DEALLOCATE) is pre-requisite for high performance DBATs

Rebind phase-in

What if you need to REBIND a package, but that package is constantly in use?

- Db2 12 FL 505 introduced REBIND phase-in
  - REBIND a new copy of the package
  - Once the REBIND completes, new thread requests will use the new copy of the package
  - Old package copies get removed as the last thread that used them deallocates
- Rebind phase-in support extended to native SQL routines and trigger packages in V13R1M508

REBIND PHASE-IN



AUTOBIND can cause disruption

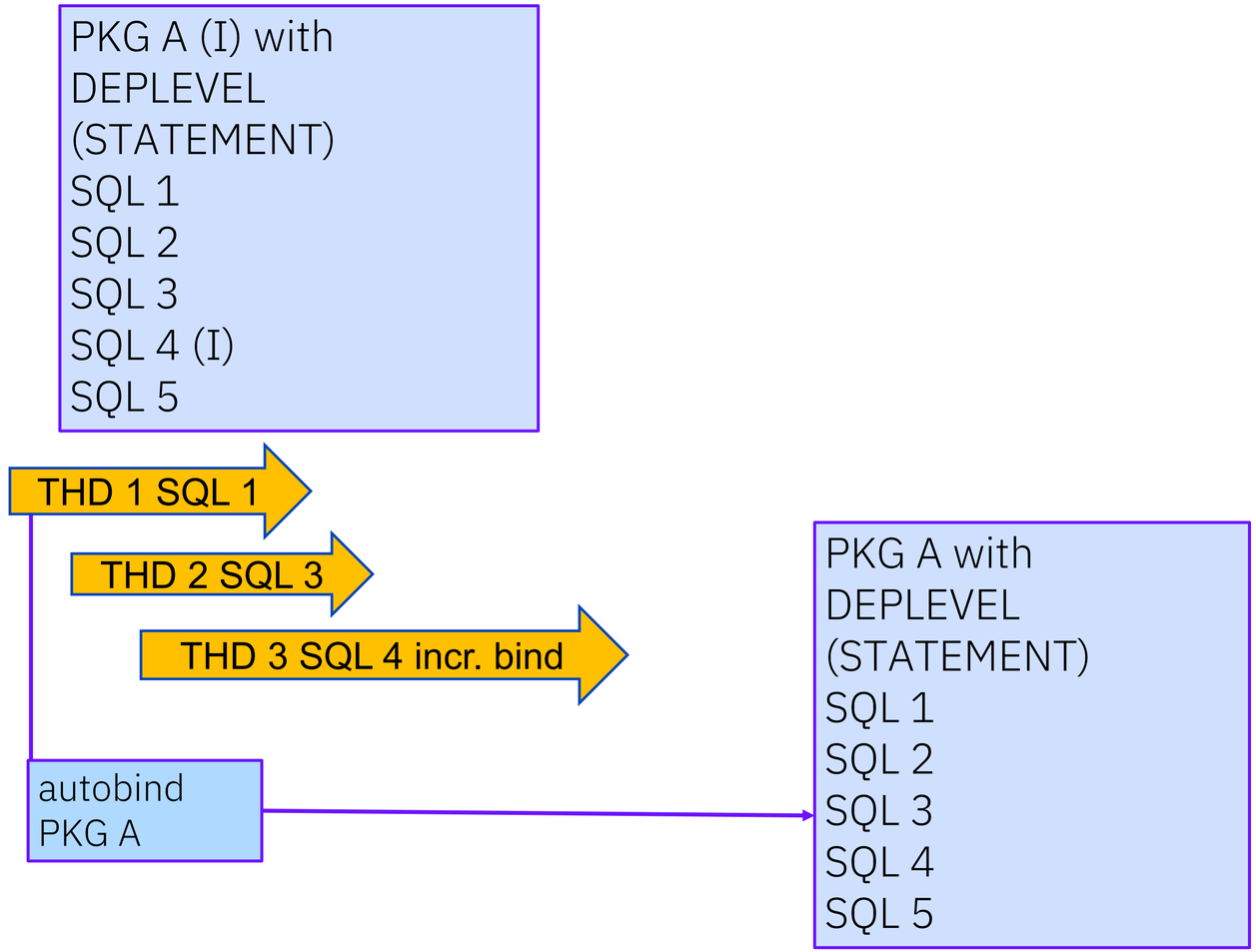
- If an ALTER or DROP invalidates one or more packages, those packages will get auto bound when next referenced
  - This can cause disruptions, especially during busy periods
  - Disruption can persist if there is an autobind failure
    - No threads can use the package until an explicit rebind
- Db2 13 FL 504 introduces autobind phase-in to reduce the disruption
  - First, targeted packages will continue to execute even if invalid
  - Second, autobind failure will not prevent package execution of those packages
    - No requirement for immediate explicit rebind

AUTOBIND phase-in

- Package BIND/REBIND with DEPLEVEL(STATEMENT) option
  - Record statement-level dependencies in SYSPACKSTMTDEP
- At function level V13R1504, if ALTER invalidates package bound with DEPLEVEL(STATEMENT), next package request triggers autobind
  - Autobind in background (phase-in)
  - Package executes before autobind completes
    - Non-invalidated statements as usual
    - Invalidated statements with incremental bind

AUTOBIND phase-in

- Package A is invalid
  - Only statement 4 affected
- Next thread (THD 1) triggers autobind in background
  - THD 1 executes SQL 1
    - Completes normally
  - THD 2 executes SQL 3
  - THD 3 executes SQL 4
    - Incrementally bound
- New copy of PKG A phased-in



Profile tables with Data Server Driver packages

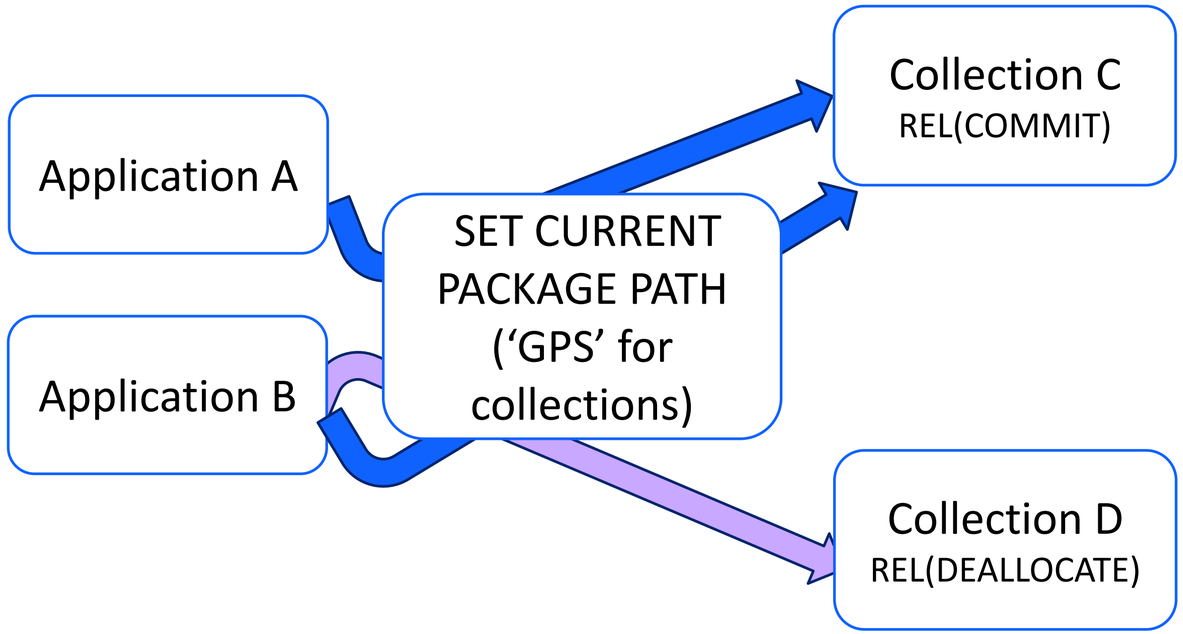
Profiles have been used since DB2 10 to manage DDF related behavior

- Starting in DB2 11, you can set special registers and global variables for DDF threads
  - And in Db2 13 for z/OS, that extends to local applications, too!
- Setting special registers can be valuable for directing IBM Data Server Driver packages to use particular collections without changing the properties at the driver
  - For example: SET CURRENT PACKAGE PATH
  - The result would override the driver’s property settings

Profile tables with Data Server Driver packages

Perhaps you have an application that could take advantage of high performance DBATS

- Set the CURRENT PACKAGE PATH to point to a collection with packages bound with RELEASE (DEALLOCATE)



Profile tables with Data Server Driver packages

SYSIBM.DSN_PROFILE_TABLE

PROFILEID	LOCATION	ROLE	AUTHID	PRDID	COLLID	PKGNAME	PROFILE_ENABLED
11	Null	Null	Null	Null	PKG_A1	Null	Y
12	Null	ROLE_DBA	USER*	Null	Null	Null	Y
13	TEST.SVL.IBM.COM	Null	Null	Null	Null	Null	Y
14	Null	Null	Null	JCC04030	Null	Null	Y

SYSIBM.DSN_PROFILE_ATTRIBUTES

PROFILE ID	KEYWORDS	ATTRIBUTE1	ATTRIBUTE2	ATTRIBUTE3	ATTRIBUTE_TIMESTAMP
11	SPECIAL_REGISTER	SET CURRENT PACKAGE PATH = CURRENT PACKAGE PATH, :collid1;	Null	Null	2021-12-19...

Monitor-reported metrics

Package metrics can provide deeper insights into application behavior and performance

- Omegamon and Intellimagic are just two examples of tools that can help you evaluate application performance
- Accounting (application) data produced by Db2 is input to such tools
  - SMFACCT(1,2,3,7,8) will send elapsed time, CPU time and suspend time to SMF
    - 1,2,3 at the plan level – 7 and 8 are the package equivalents
  - ACCOUNTING LONG reports can be filtered by package name

Monitor-reported metrics

Accounting long reports include detailed package metrics

PACKAGE NAME	VALUE	PACKAGE INFO	TIMES	PACKAGE INFO	AVERAGE TIME	AVG.EV	TIME/EVENT
TYPE	PACKAGE	ELAP-CL7 TIME-AVG	0.000381	LOCK/LATCH	0.000001	0.00	0.000420
		CP CPU TIME	0.000224	IRLM LOCK+LATCH	0.000001	0.00	0.000420
LOCATION	LOC_NAME	AGENT	0.000224	DB2 LATCH	0.000000	0.00	N/C
COLLECTION ID	COLL_ID	PAR.TASKS	0.000000	SYNCHRONOUS I/O	0.000000	0.00	N/C
PROGRAM NAME	PRG_NAME	SE CPU TIME	0.000000	OTHER READ I/O	0.000000	0.00	N/C
		SUSPENSION-CL8	0.000095	OTHER WRITE I/O	0.000000	0.00	N/C
ACTIVITY TYPE	NONNESTED	AGENT	0.000095	SERV.TASK SWITCH	0.000093	0.12	0.000766
ACTIVITY NAME	'BLANK'	PAR.TASKS	0.000000	ARCH.LOG(QUIESCE)	0.000000	0.00	N/C
SCHEMA NAME	'BLANK'	NOT ACCOUNTED	0.000062	ARCHIVE LOG READ	0.000000	0.00	N/C
INCOMPATIBILITY	NO	AVG.DB2 ENTRY/EXIT	10.22	DRAIN LOCK	0.000000	0.00	N/C
SUCC AUTH CHECK	0	DB2 ENTRY/EXIT	6544	CLAIM RELEASE	0.000000	0.00	N/C
OCCURRENCES	640			PAGE LATCH	0.000000	0.00	N/C
NBR OF ALLOCATIONS	0	CP CPU SU	21.12	NOTIFY MESSAGES	0.000001	0.00	0.000170
SQL STMT - AVERAGE	4.08	AGENT	21.12	GLOBAL CONTENTION	0.000000	0.00	N/C
SQL STMT - TOTAL	2614	PAR.TASKS	0.00	TCP/IP LOB XML	0.000000	0.00	N/C
NBR RLUP THREADS	640	SE CPU SU	0.00	ACCELERATOR	0.000000	0.00	N/C
				PQ SYNCHRONIZATION	0.000000	0.00	N/C
				FAST INSERT PIPE	0.000000	0.00	N/C
				TOTAL CL8 SUSPENS.	0.000095	0.13	0.000740

Monitor-reported metrics

Intellimagic reports include charts with package information and accounting data

Questions?

Summary

- Packages are control structures for running SQL in Db2 and are collected and listed in plans
- There are many ways to aid in managing your packages
  - Bind options
  - REBIND/AUTOBIND phase-in
  - Profile tables
  - Package monitoring

Thank you!

Anna McKee
Anna.mckee@ibm.com

Mark Rader
mrader@us.ibm.com