IBM Software Group

WebSphere/DB2 Pooling and Caching

Maryela Weihrauch
IBM Silicon Valley Lab., Jan 2007
weihrau@us.ibm.com

DB2 Information Management Software

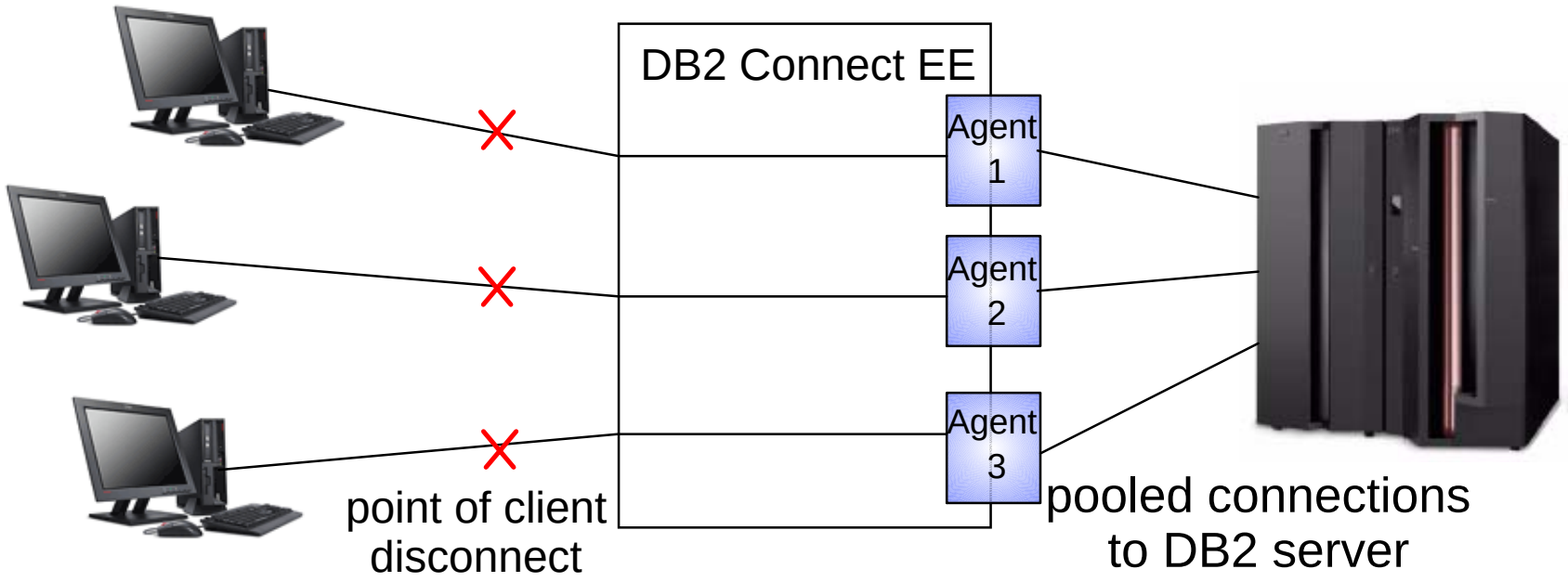
e business on demand software

Agenda

- **DB2 Connection Pooling/ Connection Concentrator**
- **WebSphere Connection Pooling**
- **Usage Scenarios**
- **DB2 Statement Caching**
- **WebSphere preparedStatement Object Caching**
- **Usage Scenarios**
- **Exploitation of Option KEEPDYNAMIC YES**
- **Summary**

DB2 Connect EE Connection Pooling

- **DB2 Connect EE V6 introduced Connection Pooling**
- **Ability to reuse server agents after client disconnected (physical connection between DB2 Connect and DB2 server)**
- **Avoid repeated processing to create and terminate DB2 threads**
- **NUM_POOLAGENTS <= MAXAGENTS**

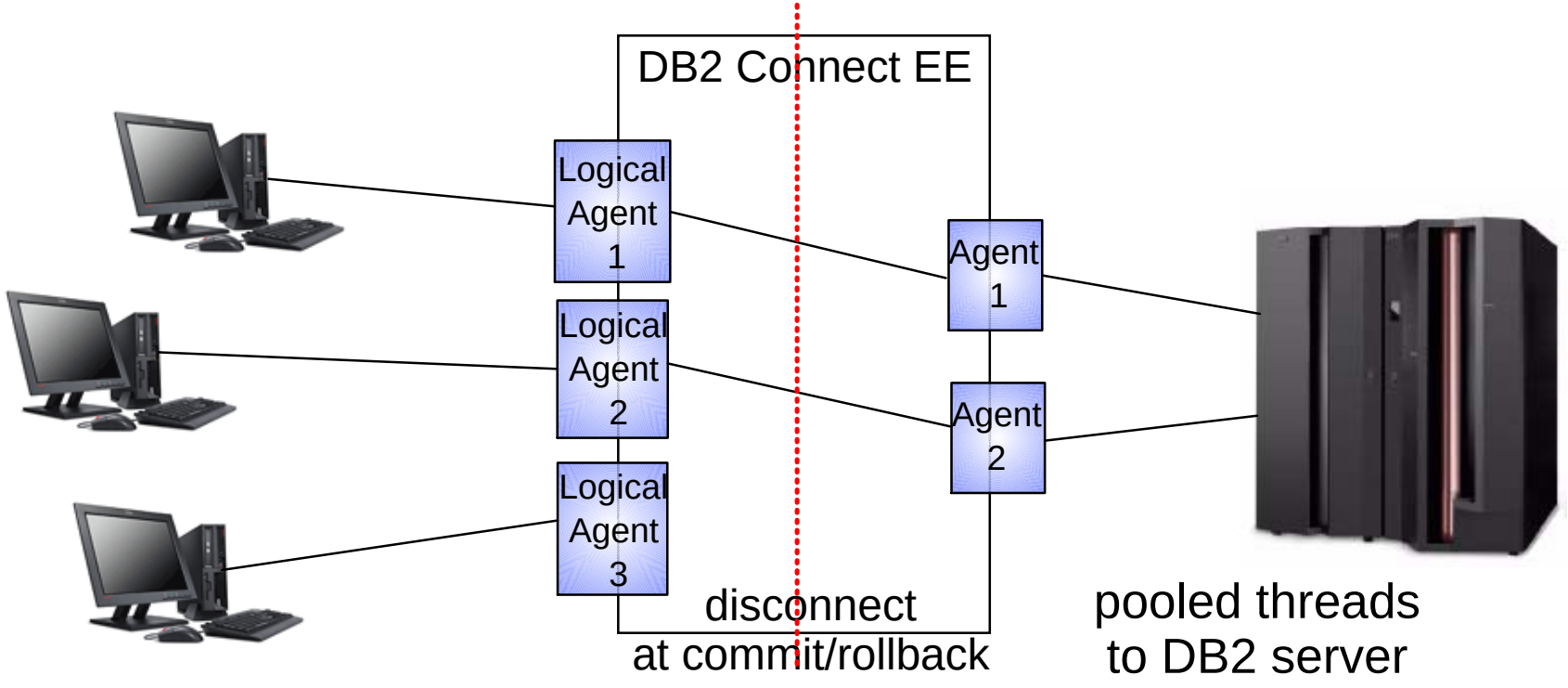


DB2 Connect EE Connection Pooling . . .

- MAXAGENTS - max. # threads to DB2 server
- NUM_POOLAGENTS - max. # of threads kept in the pool, default is 2% of MAXAGENTS
- Drawback of Connection Pooling when connected to DB2 Data Sharing group
  - DB2 Connect processes information about availability of data sharing members at creation of a "new" connection (thread)
  - Pooled connections could remain with data sharing member that has problems
  - DB2 Connect would not use Sysplex information to determine best connection for reuse
  - At member outage, all connections to this member would receive connection failure

DB2 Connect EE Connection Concentrator

- **DB2 Connect EE V7 introduced Connection Concentrator (also called Transaction Pooling)**
- **Ability to reuse server agents at application commit or rollback**
- **MAX_CONNECTIONS > MAXAGENTS to enable concentrator**



DB2 Connect EE Connection Concentrator . . .

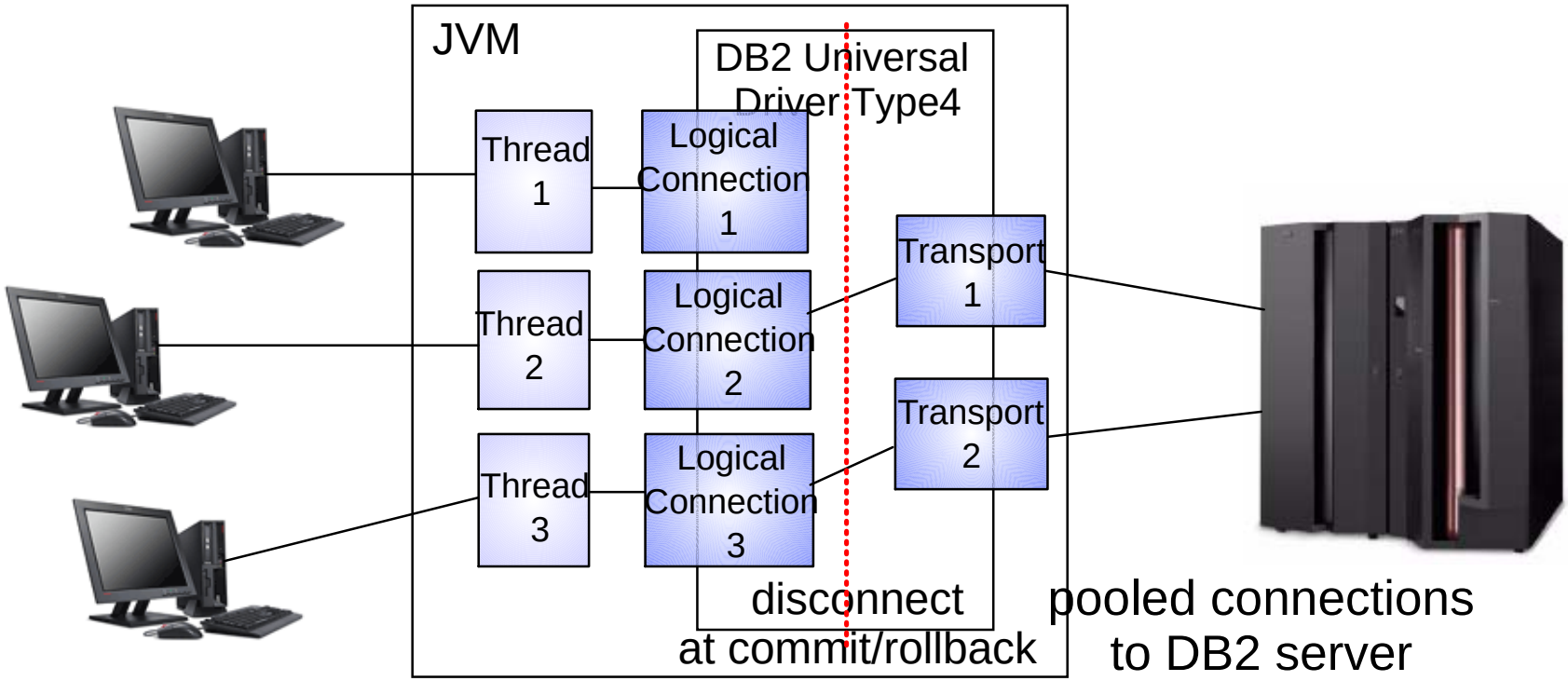
- MAXAGENTS - max. # threads to DB2 server
- MAX_CONNECTIONS - max. # of concurrent client connections to DB2 Connect (default is equal to MAXAGENTS - concentrator disabled)
- Conditions for connection reusability at commit:
  - No open WITH HOLD cursor
  - No declared global temporary tables must exist (used declared global temp tables must be explicitely or implicitely droped)
  - No reference to packages bound with KEEPDYNAMIC YES
- Rollback always leaves a connection in a reusable state

DB2 Connect EE Connection Concentrator . . .

- Parallel Sysplex Considerations:
  - DB2 Connect gets status information of each member of the data sharing group which is used to balance connections to each member and to route transactions
  - At outage of one member, only the transactions active in this particular member receive a connection failure (V7) or resource unavailable (V8)
  - All other clients will remain connected to DB2 Connect or continue transactions at other members of the group
- For best performance, there should be enough connections in the connection pool to contain all concurrently active clients

DB2 Universal Driver Connection Concentrator

- **IBM DB2 Driver for JDBC and SQLJ provides connection pool and connection concentrator functionality similar to DB2 Connect (DB2 V8 FP10 and later)**
- **Ability to reuse server agents at application commit or rollback**



DB2 Universal Driver Connection Concentrator . . .

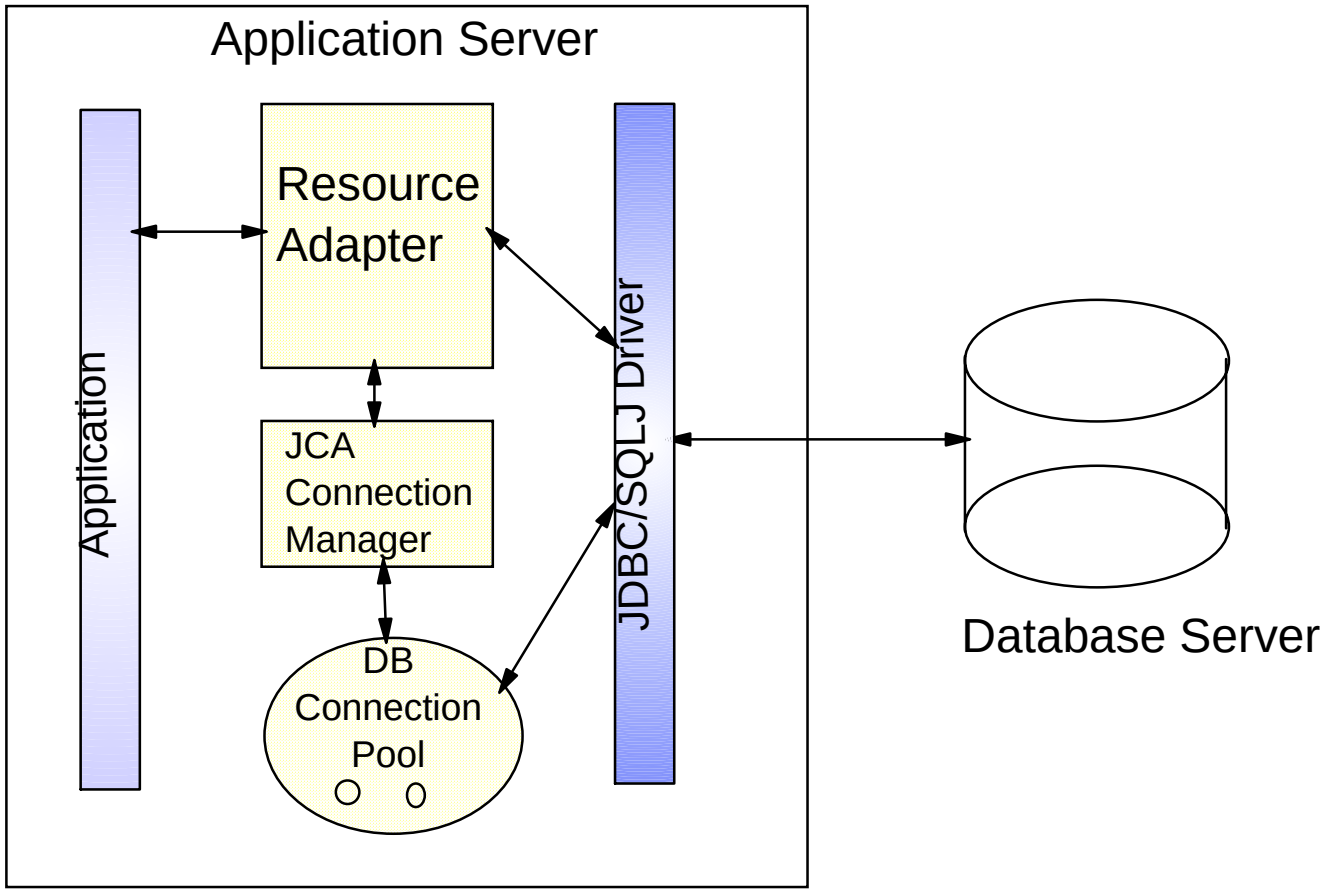
- New Global properties defined in Global Properties File:
  - db2.jcc.maxTransportObjects
    equivalent to MAXAGENTS - max # of connections to DB2 server across all datasources (default value is -1, meaning no limit)
  - db2.jcc.minTransportObjects
    equivalent to NUM_POOLAGENTS - # of connections kept in the pool across all datasources - # of transport objects will grow as requested but always stay (default value is 0)
  - db2.jcc.maxTransportObjectIdleTime
    time in sec., a connection stays idle in the pool before it is closed, until minTransportObject is reached
    (default value is 60 sec)
  - db2.jcc.maxTransportObjectWaitTime
    if maxTransportObjects is reached - time in sec., an application waits to get a connection before throwing a SQLException
    (default value is 5 sec)

DB2 Universal Driver Connection Concentrator . . .

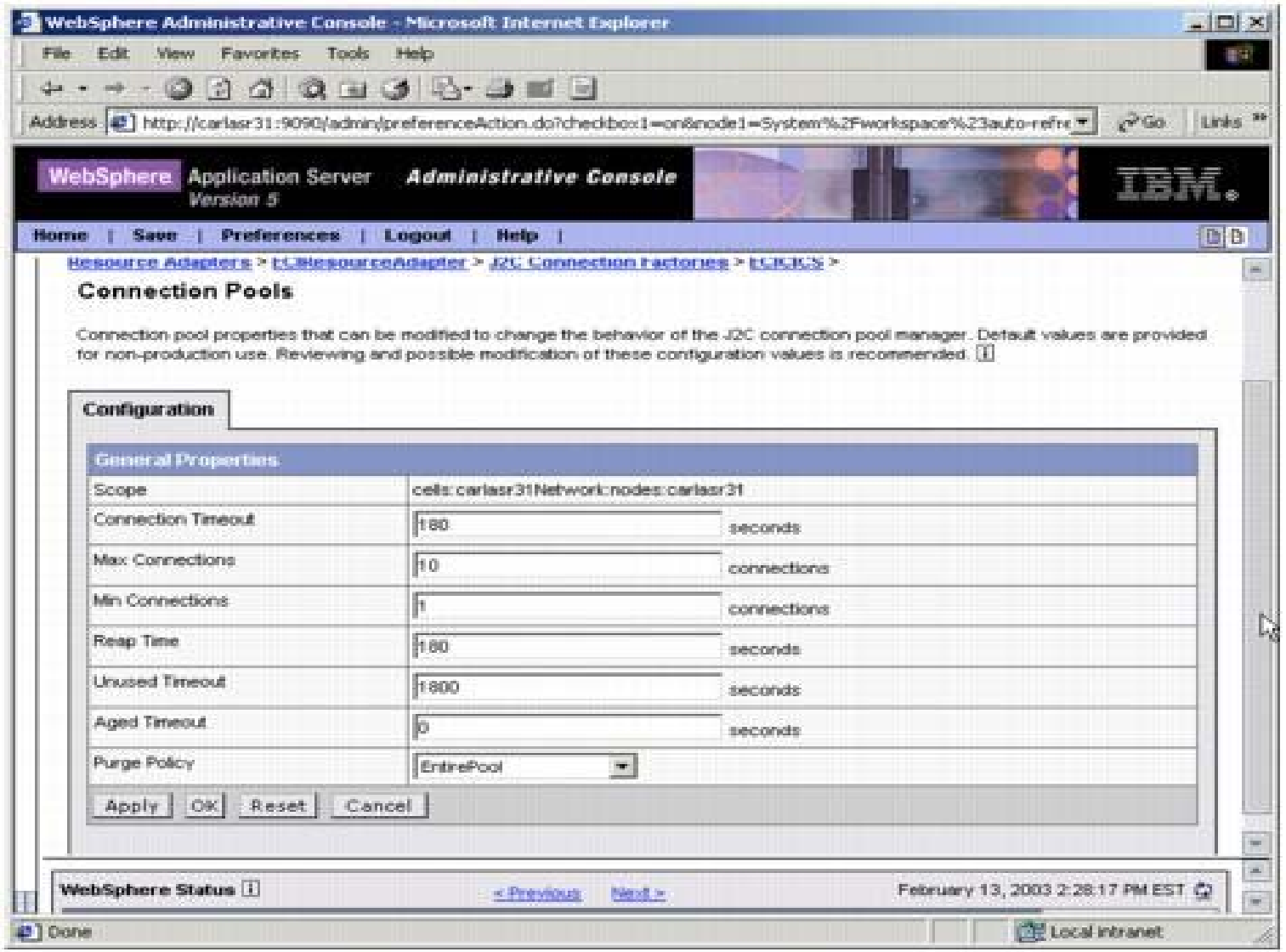
- New DataSource Properties
  - maxTransportObjects
    max # of connections to DB2 server from this DataSource. Can not be bigger than db2.jcc.maxTransportObjects
    (default value is -1, meaning no limit)
  - enableConnectionConcentrator
    enables connection concentrator functionality. Not allowed for DB2 LUW
    (default value is "false" - disabled)
  - enableSysplexWLB
    enables Sysplex Workload Balancing functionality. Not allowed for DB2 LUW
    (default value is "false" - disabled)
- Currently not supported for JDBC 1.2 DriverManager.getConnection()

WebSphere V5 Connection Pooling

- WebSphere pooles Connection Java objects, handed out by the JDBC/SQLJ driver



WebSphere Connetion Pooling . . .

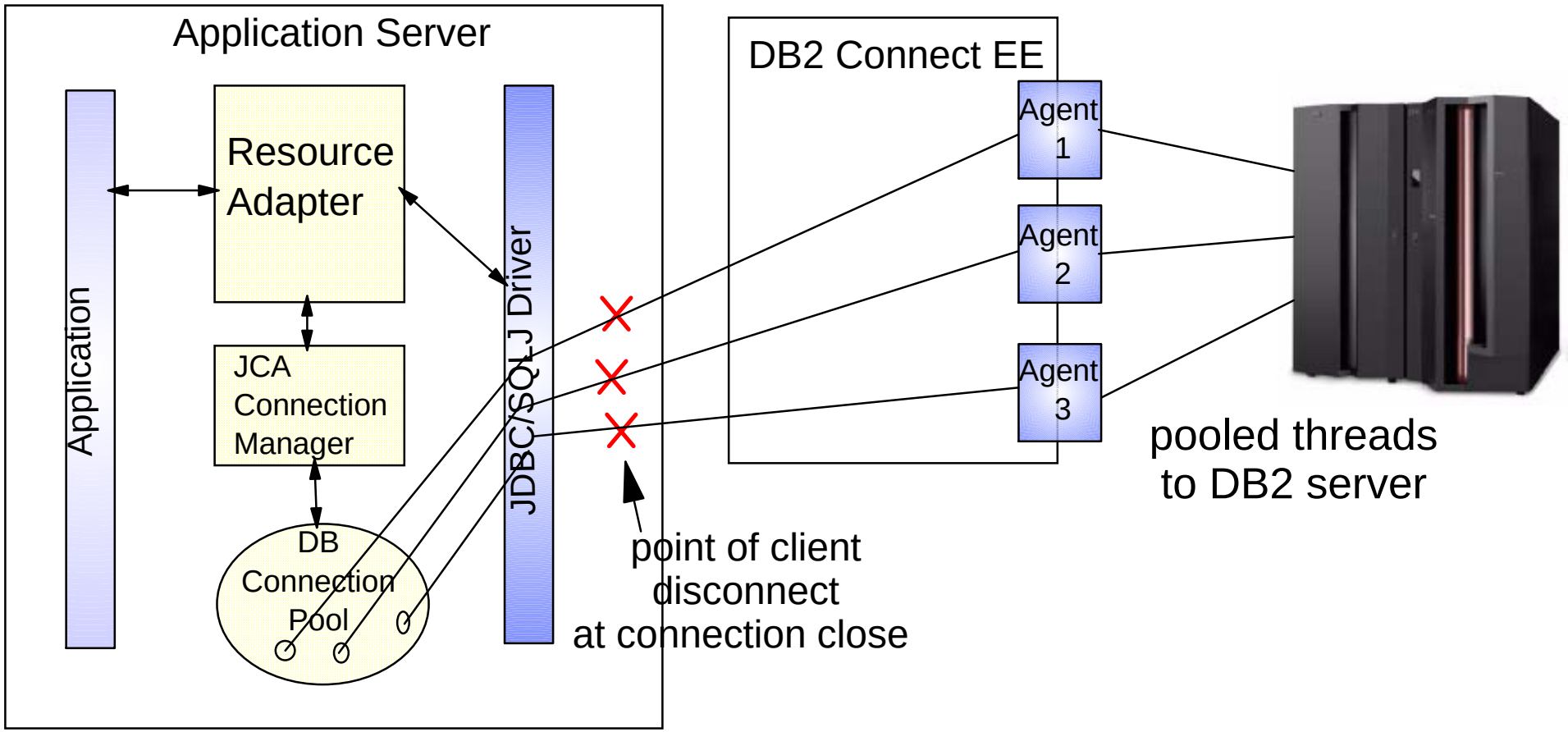


WebSphere Connetion Pooling

- Connection object pool currently maintained by WebSphere
  - saves creating/destroying Connection objects which is relatively expensive
  - part of the Connection object creation is creating a physical connection to DB2
- Connection object holds references to other Java objects like preparedStatement objects that would be destroyed with the Connection object

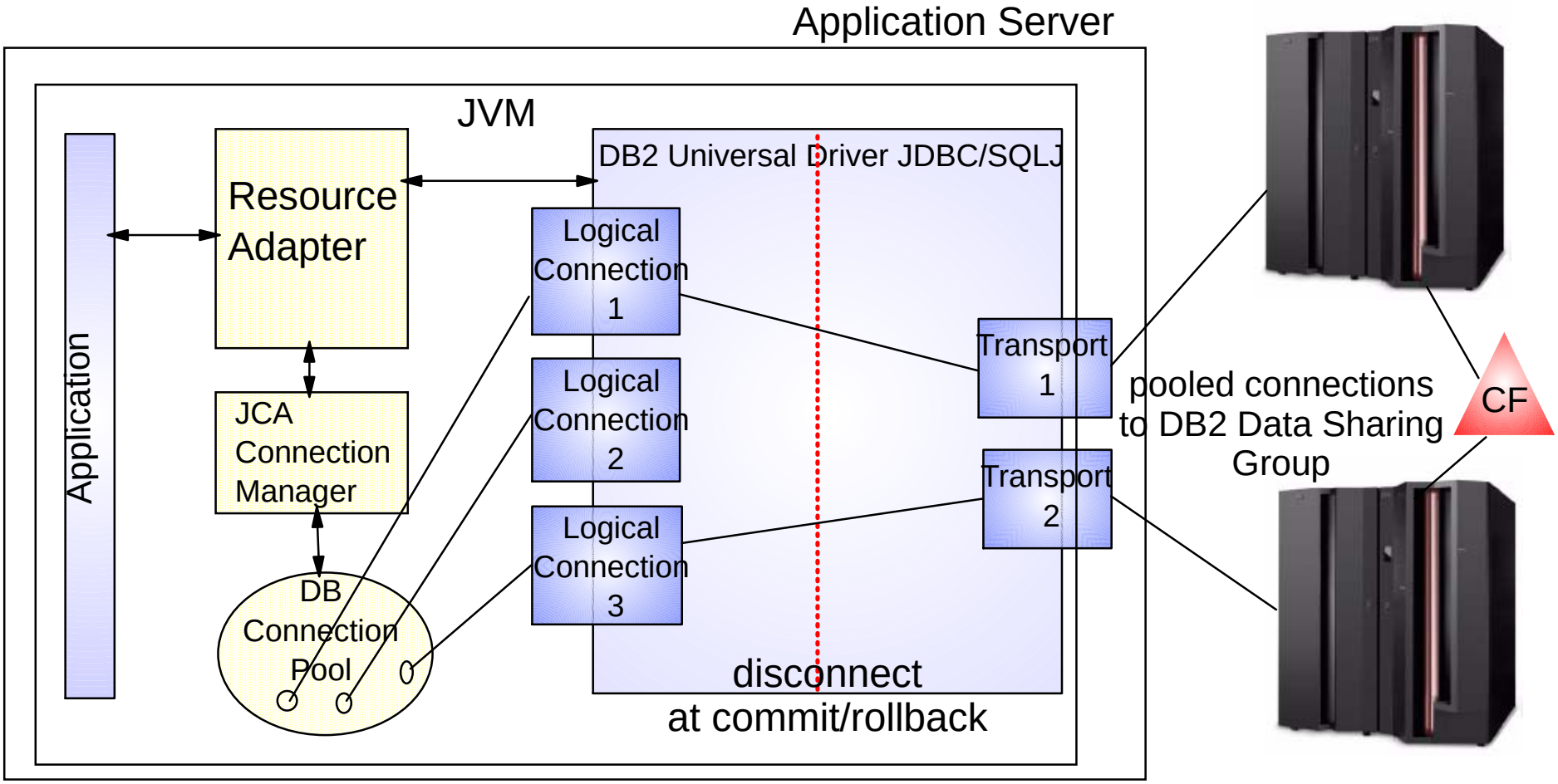
Which Connection Pool Should I use?

- If **NOT** connecting to DB2 Data Sharing Group, the connection pool closest to the application should be used.



Which Connection Pool Should I use?

- To exploit DB2 Data Sharing workload balancing and transparent failover, both, application server connection pool AND connection concentrator/ connection pool should be used



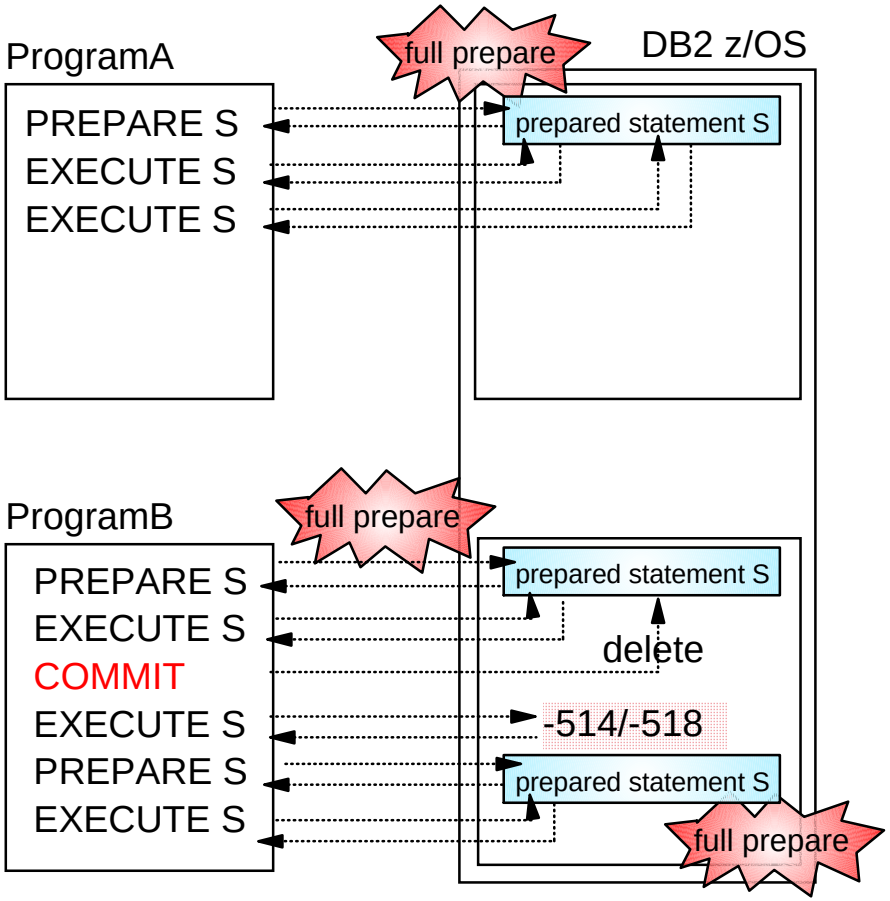
WAS/DB2 Active Thread - Tuning Considerations

- WAS connections in connection pool that keep an active thread in DB2 are target of the "idle thread timeout"
  - Set WAS "connection unused time" to a smaller value than DB2 "idle thread timeout" to avoid stale connection conditions
  - Consider setting min connections to 0 (zero) and connection unused time to appr. 10 - 15 min to free up unused resources in DB2 in a controled way and to reduce the exposure of long living threads

DB2 Statement Caching - Overview

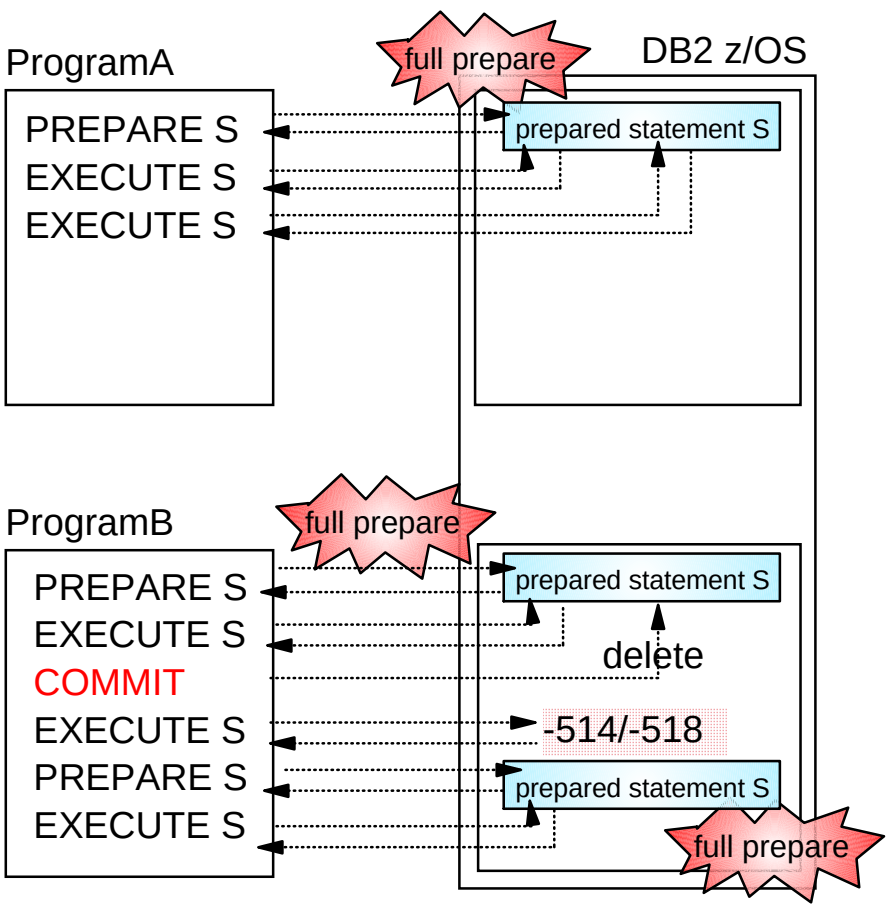
- **Introduced in DB2 z/OS V5**
- **Allows applications to reuse and share prepared SQL statements in DB2**
- **Conditions for reuse of SQL statement from dynamic statement cache**
  - SQL is dynamically prepared SELECT, UPDATE, DELETE or INSERT
  - The statement text is identical - character for character (literals problematic)
  - The authorization ID is the same
  - ...
- **ZPARM value CACHEDYN = YES turns on global cache**
  - Statement text and executable of the prepared statements are kept in the EDM pool for reuse across all threads
  - REOPT(VARS) disables use of cache for that plan/packages
- **BIND option KEEPDYNAMIC(YES) enables local cache**
  - prepared statements are kept in thread storage across COMMIT
  - local cache is thread specific
- **ZPARM value MAXKEEPD limits no. of SQL statements across all threads**

DB2 Statement Caching - Global Caching

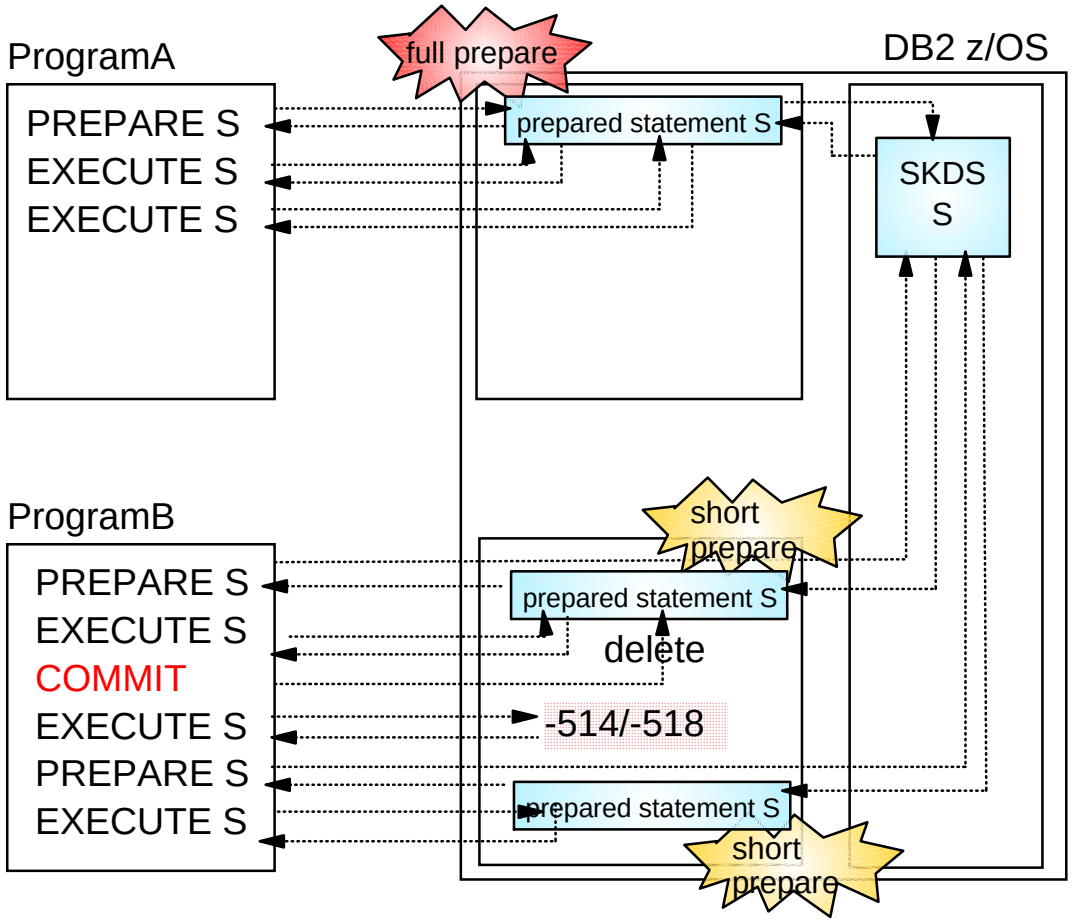


No Caching

DB2 Statement Caching - Global Caching



No Caching

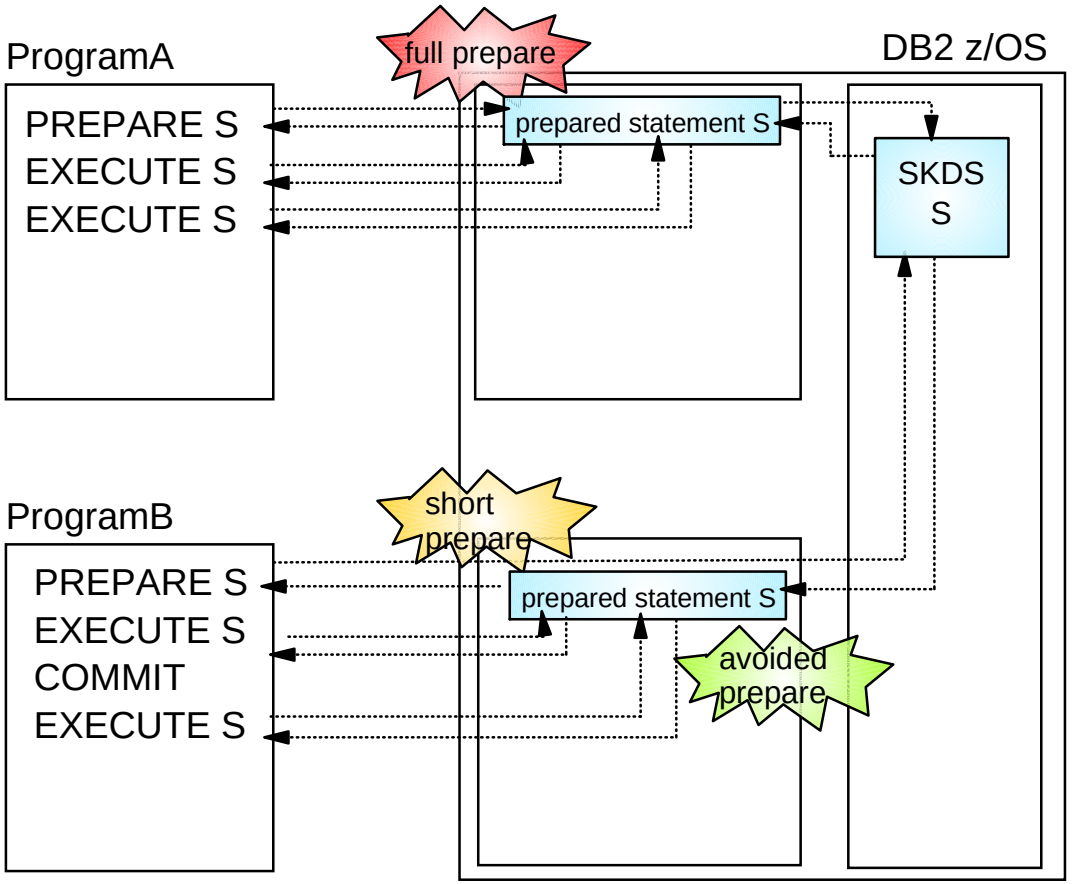


Global Caching Only

DB2 Statement Caching - Global Caching . . .

- **Significant cost to fully prepare a dynamic SQL statement**
- **Global dynamic statement cache**
  - statement text and executable (SKDS) is cached in EDM pool
    V7 by default in data space
    V8 above the bar
  - Only first prepare is full prepare, otherwise short prepare, which is a copy from global cache into thread storage
  - No prepared statement is kept in thread storage across commit
- **Should be turned on if dynamic SQL is executed in the DB2 system**
- **Best trade-off between storage and CPU consumption for applications executing dynamic SQL**

DB2 Statement Caching - Global and Local Caching



DB2 Statement Caching - Global and Local Caching . . .

- **Only first prepare is full prepare, otherwise short prepares**
- **Prepared statements kept in thread storage across commit (avoided prepares)**
  - same prepared sql statement can be stored in several threads
  - MAXKEEPD limits the stored executable only, the statement text is always stored in thread storage
  - application logic needs to reflect the bind option
- **Should only be used selectively for application with a limited number of SQL statements that are executed very frequently**
- **Should NOT be used for DB2 systems that are constraint in DBM1 storage**

WebSphere preparedStatement Object Cache

- WebSphere manages a cache of previously created preparedStatement objects on a connection
- When a new prepared statement is requested on a connection, the cached preparedStatement object is returned if available
- Creating a new preparedStatement object is costly in Java besides the cost to prepare the SQL to DB2

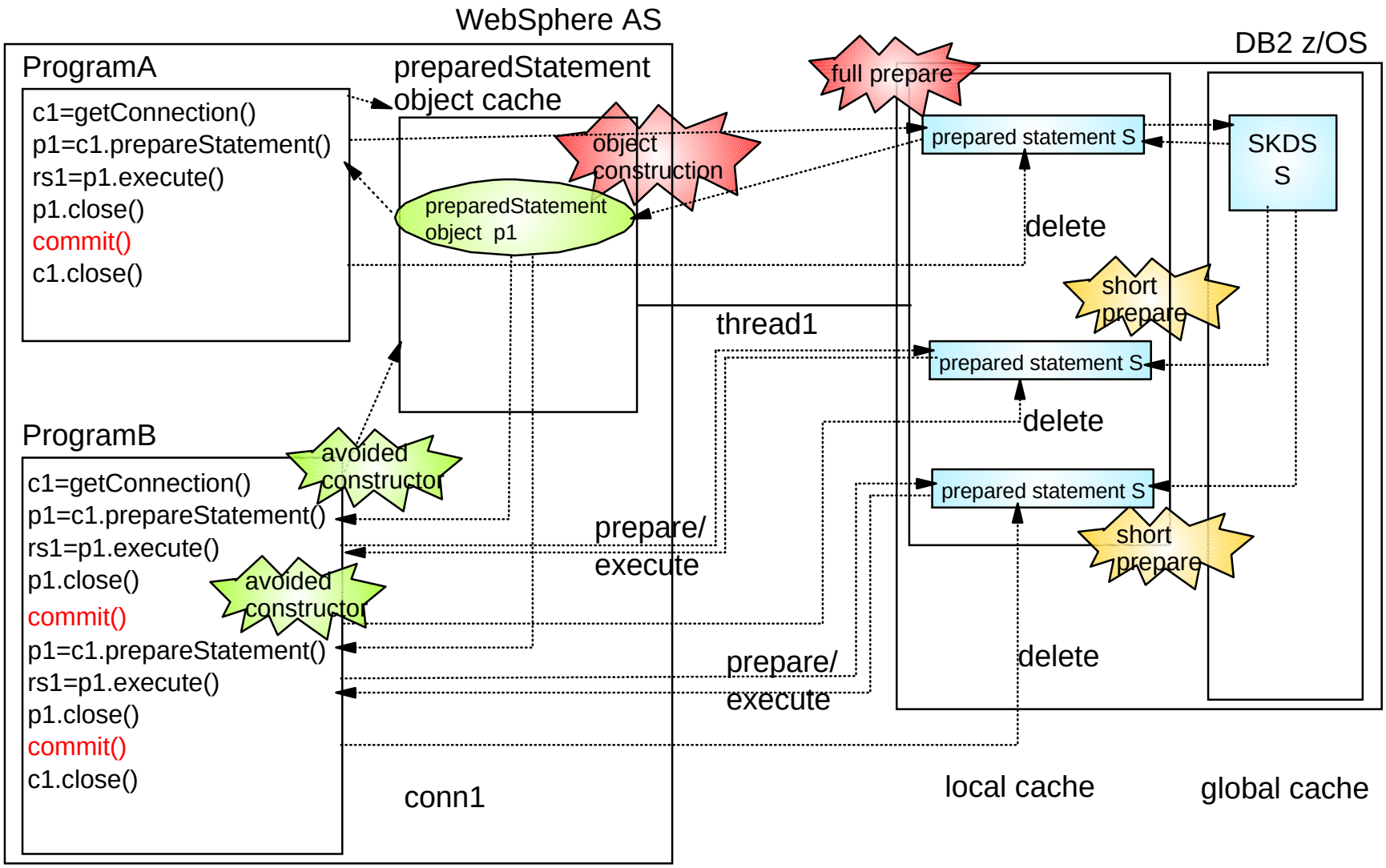
WebSphere preparedStatement Object Cache . . .

Container managed persistence	☑ Use this Data Source in container managed persistence (CMP)	Enable if this data source will be used for container managed persistence of EJBs. This will cause a corresponding CMP connection factory which corresponds to this datasource to be created for the relational resource adapter.
Description	Webbank Datasource	An optional description for the resource.
Category		An optional category string which can be used to classify or group the resource.
Statement Cache Size	10 statements	Number of free prepared statements per connection. This is different from the old datasource which is defined as number of free prepared statements per data source.
Datasource Helper Classname	com.ibm.websphere.rsadapter.DB2L	The datastore helper that is used to perform specific database functions.
Component-managed Authentication Alias	(none)	References authentication data for component-managed signon to the resource.
Container-managed Authentication Alias	(none)	References authentication data for container-managed signon to the resource.
Mapping-Configuration Alias	(none)	Select a suitable JAAS login configuration from the security-JAAS configuration panel to map the user identity and credentials to a resource principal and credentials that is required to open a connection to the back-end server.

Apply OK Reset Cancel

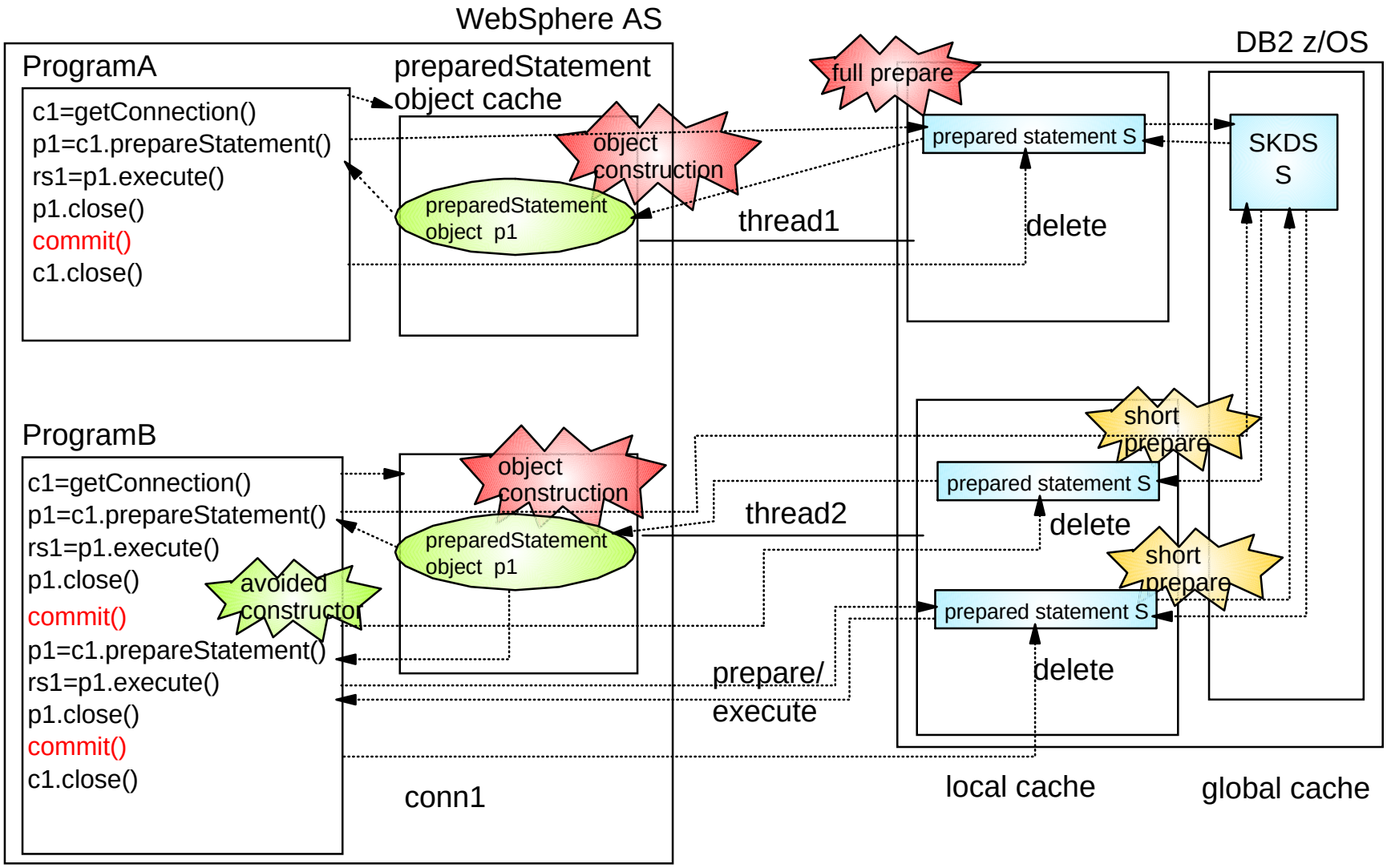
WebSphere/DB2 - How the Caches Play Together...

- PreparedStatement object caching in WebSphere AND global dynamic statement caching in DB2 is recommended for JDBC application.

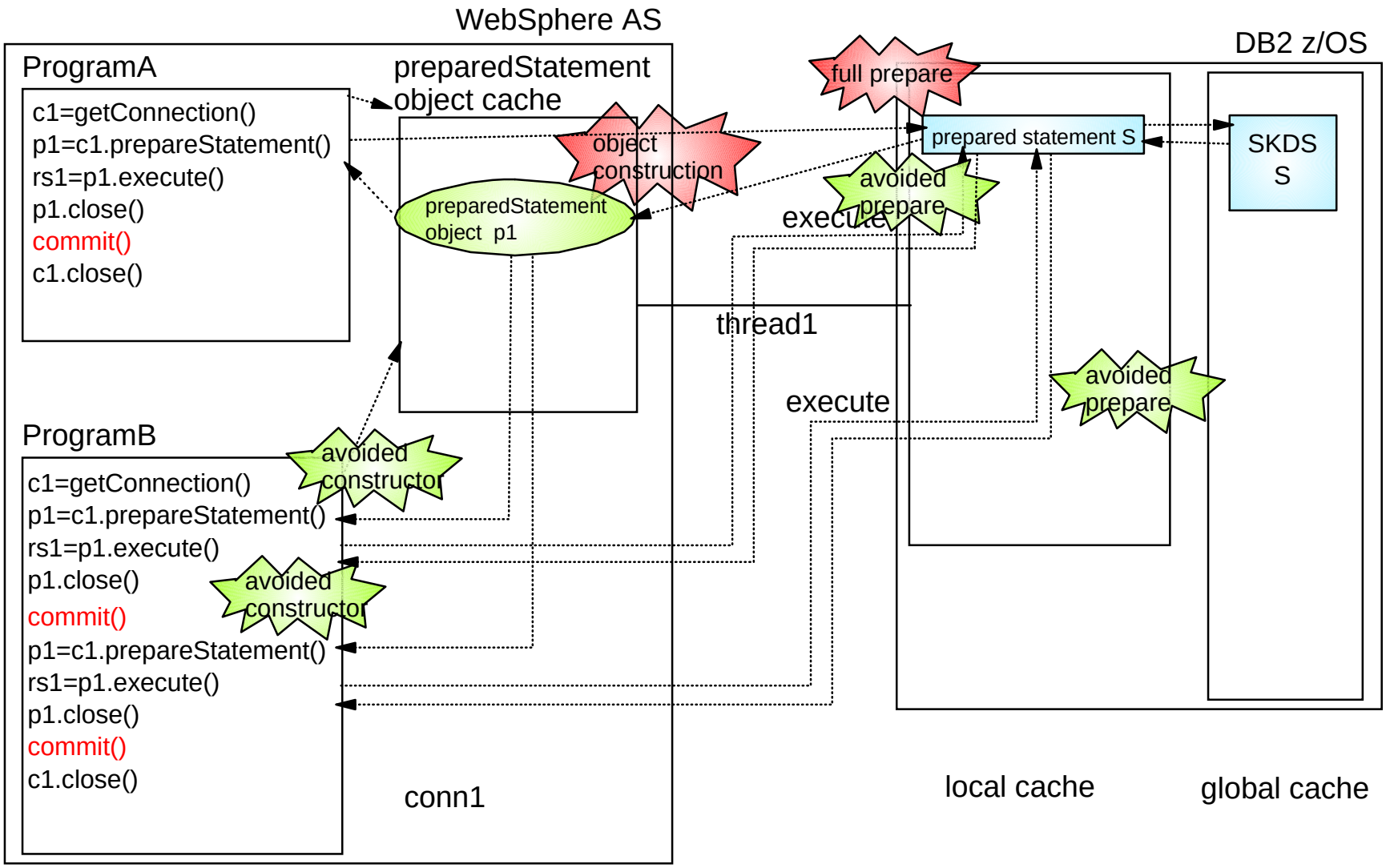


WebSphere/DB2 - How the Caches Play Together...

- PreparedStatement objects are cached per connection - multiple objects for the same statement

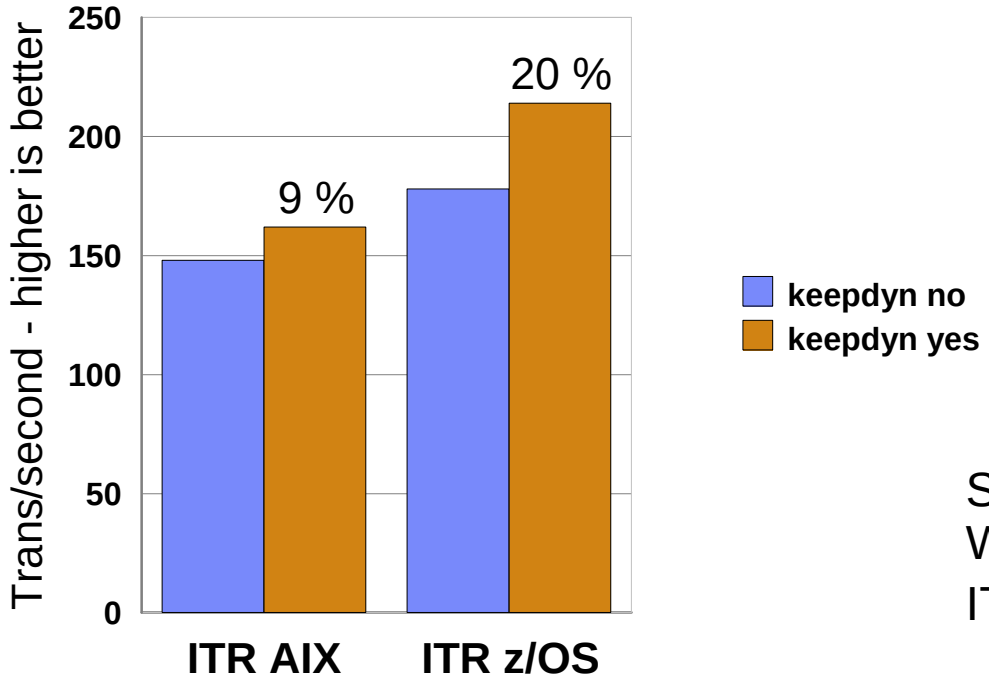


WebSphere/DB2 - Using KEEPDYNAMIC YES



WebSphere/DB2 - Using KEEPDYNAMIC YES

- Bind JCC packages with
  - -keepdynamic=1 (YES) and
  - -collection = collname
- Define datasource property (WebSphere customProperty)
  - -keepDynamic=1 and
  - -JdbcCollection=collname



SpecJApp2002
WebSphere/AIX V502- DB2 z/OS V8
ITR = tops/cpu%*100

keepDynamic YES - Usage Considerations

- Pro
  - avoids prepare of statements after commit that were prepared earlier
  - saves CPU on application server (WebSphere) as well as DB2
- Contra
  - statements are saved on thread level
    consumes a significant amount of storage
    not recommended for storage constraint systems
  - type 2 distributed threads can not turn inactive
    DB2 resources for distr. threads are not freed at commit
    disables transaction workload balancing in Data Sharing Env.
- Recommended for applications with a limited amount of SQL statements that are executed very often
- Not recommended for applications with a large number of SQL statements that are executed infrequently
- DB2 V8 maint: PQ86787/UQ87049,PQ87126//UQ88971,PQ87129/UQ87633

Summary

- Connection Pooling
  - if no DB2 data sharing, use connection pool closest to application (e.g. WebSphere connection pool)
  - if DB2 data sharing advantages are exploited, use connection pool of application server AND DB2 Connect / JCC type 4 connection concentrator
- Dynamic Statement Cache
  - always use WebSphere preparedStatement object cache and global dynamic statement cache in DB2
  - consider using local dynamic statement cache for application that execute a limited amount of sql very frequently and DB2 is not storage constraint.
  - SQLJ (static SQL) does not use dynamic statement cache with equal or better performance than dynamic SQL exploiting local and global cache.

Some References . . .

- **WebSphere Connection Pooling white paper**

  http://www-306.ibm.com/software/webservers/appserv/whitepapers/connection_pool.pdf

- **IBM WebSphere Application Server V5.1 System Management and Configuration WebSphere Handbook Series**

  http://www.redbooks.ibm.com/abstracts/sg246195.html?Open

- **DB2 for z/OS: Ready for Java SG24-6435**

  http://www.redbooks.ibm.com/abstracts/sg246435.html?Open

- **DB2 for z/OS and WebSphere: The Perfect Couple SG24-6319**

  http://www.redbooks.ibm.com/abstracts/sg246319.html?Open