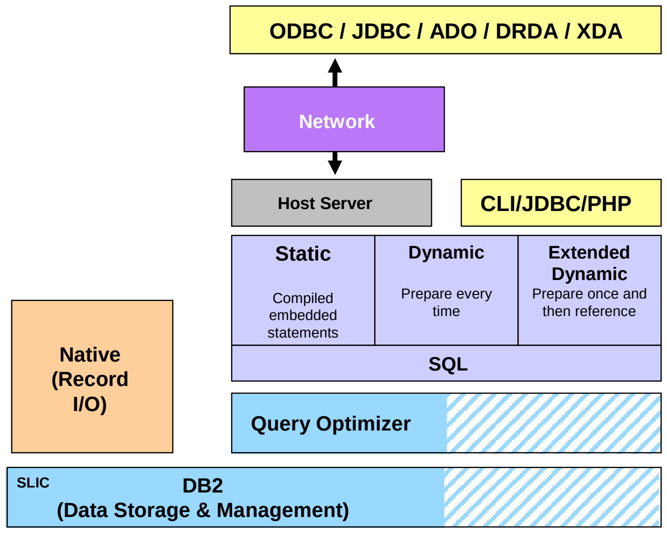
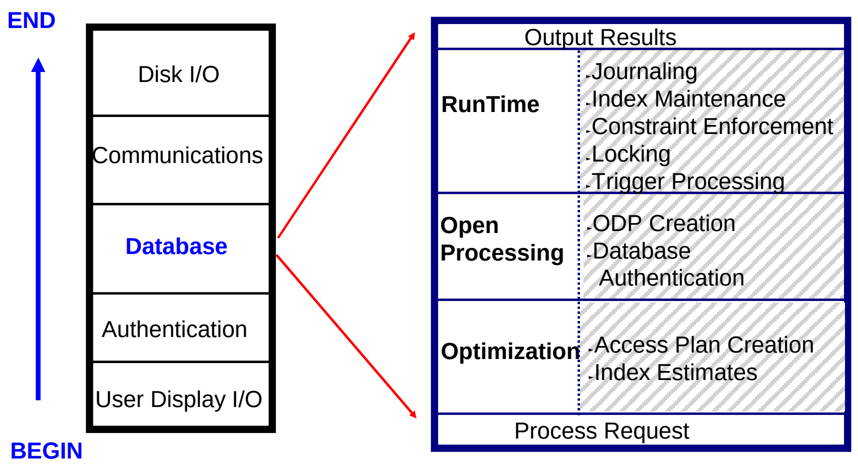
The ABCs of Coding High Performance SQL Apps
DB2 for IBM i

Presented by Jarek Miszczyk
IBM Rochester, ISV Enablement

SQL Interfaces

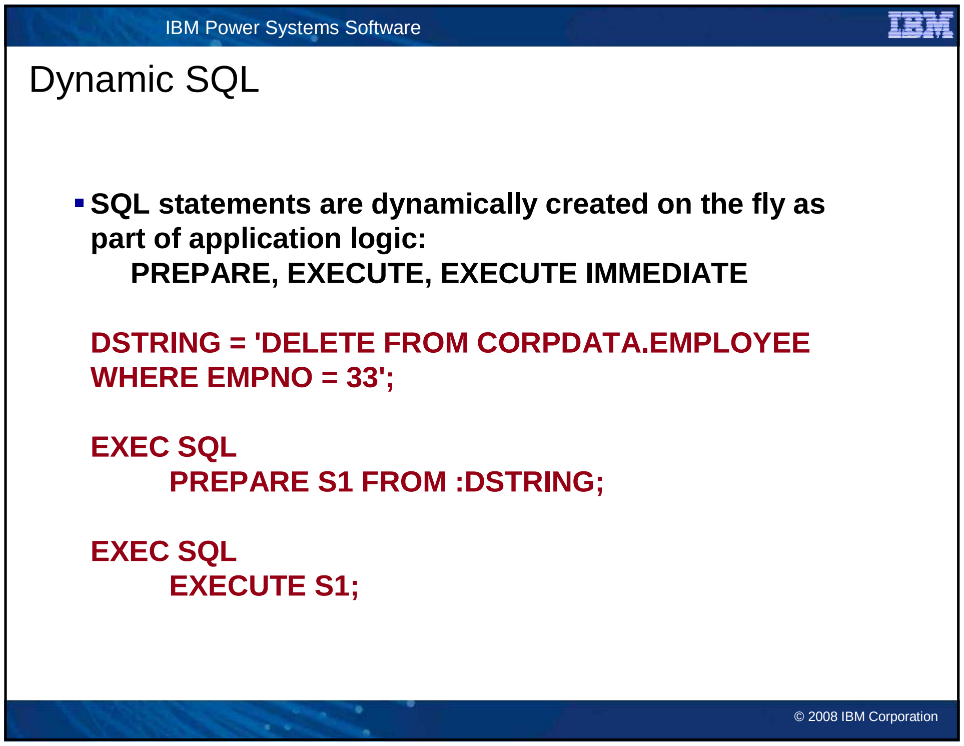


Measuring & Monitoring DB2 Performance



Static SQL

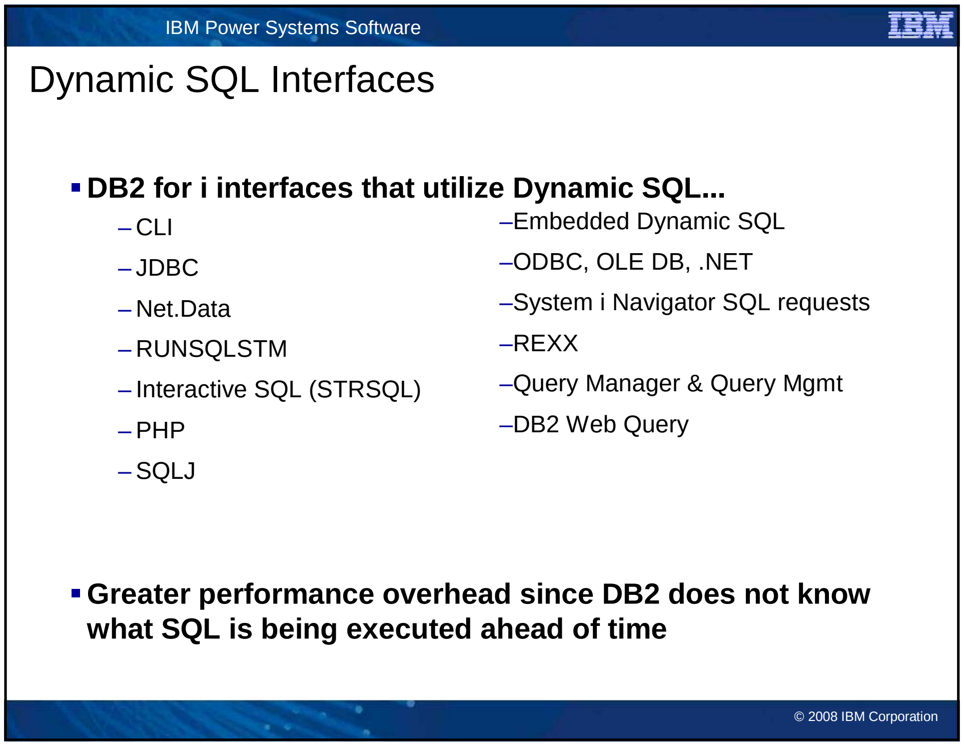
- **Non-dynamic SQL statements embedded in application programs**
- **Languages Supported:**
  - RPG
  - COBOL
  - C, C++
  - SQL Procedural Language
    - SQL embedded in C
  - PL/I
- **Most efficient SQL interface on IBM i**



Dynamic SQL

- **SQL statements are dynamically created on the fly as part of application logic:**
  PREPARE, EXECUTE, EXECUTE IMMEDIATE

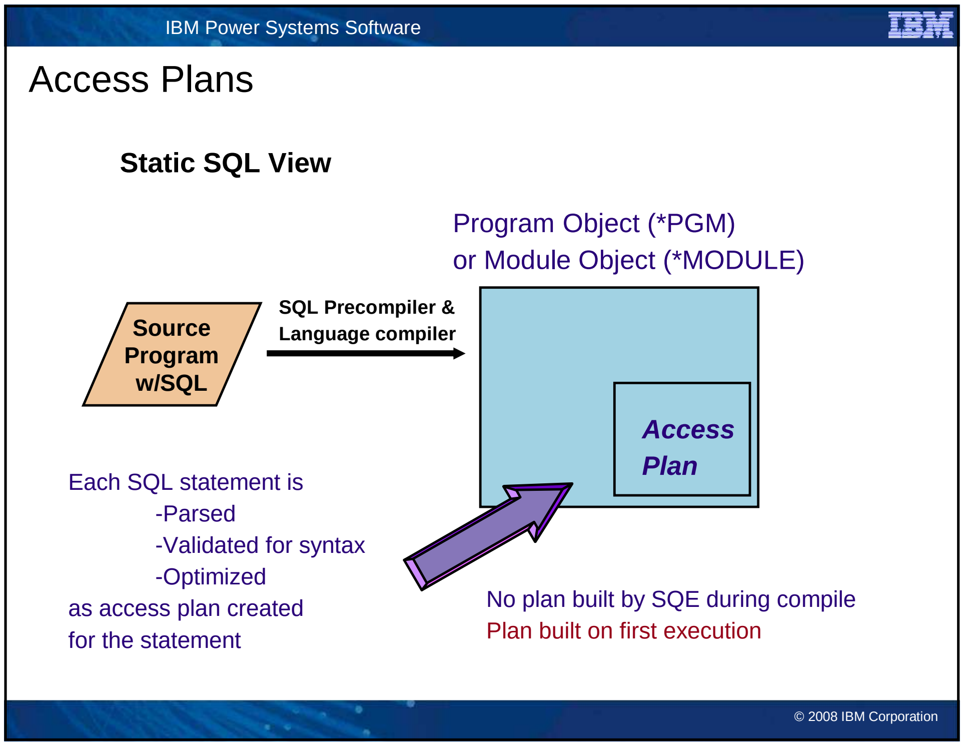
```
DSTRING = 'DELETE FROM CORPDATA.EMPLOYEE
WHERE EMPNO = 33';

EXEC SQL
      PREPARE S1 FROM :DSTRING;

EXEC SQL
      EXECUTE S1;
```



Dynamic SQL Interfaces

- **DB2 for i interfaces that utilize Dynamic SQL...**
  – CLI
  – JDBC
  – Net.Data
  – RUNSQLSTM
  – Interactive SQL (STRSQL)
  – PHP
  – SQLJ
  –Embedded Dynamic SQL
  –ODBC, OLE DB, .NET
  –System i Navigator SQL requests
  –REXX
  –Query Manager & Query Mgmt
  –DB2 Web Query

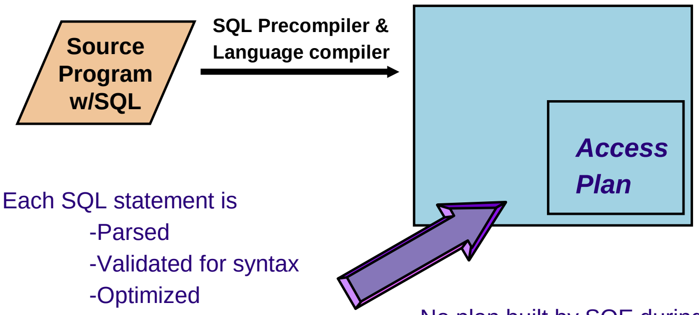
- **Greater performance overhead since DB2 does not know what SQL is being executed ahead of time**



Access Plans

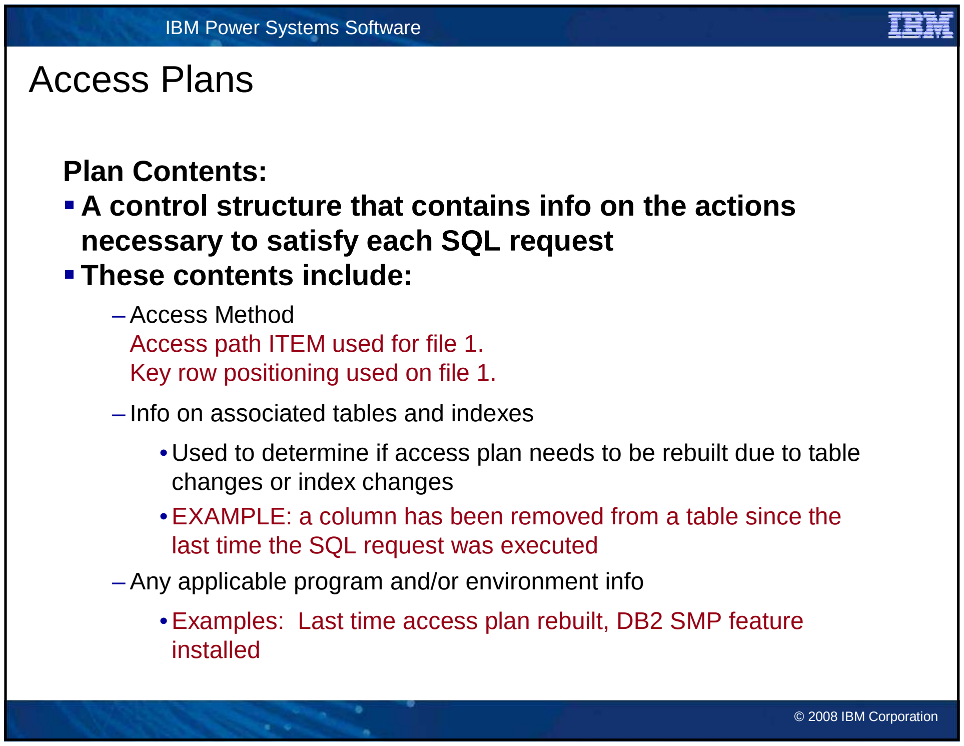
Static SQL View

Program Object (*PGM)
or Module Object (*MODULE)



Each SQL statement is
-Parsed
-Validated for syntax
-Optimized
as access plan created
for the statement

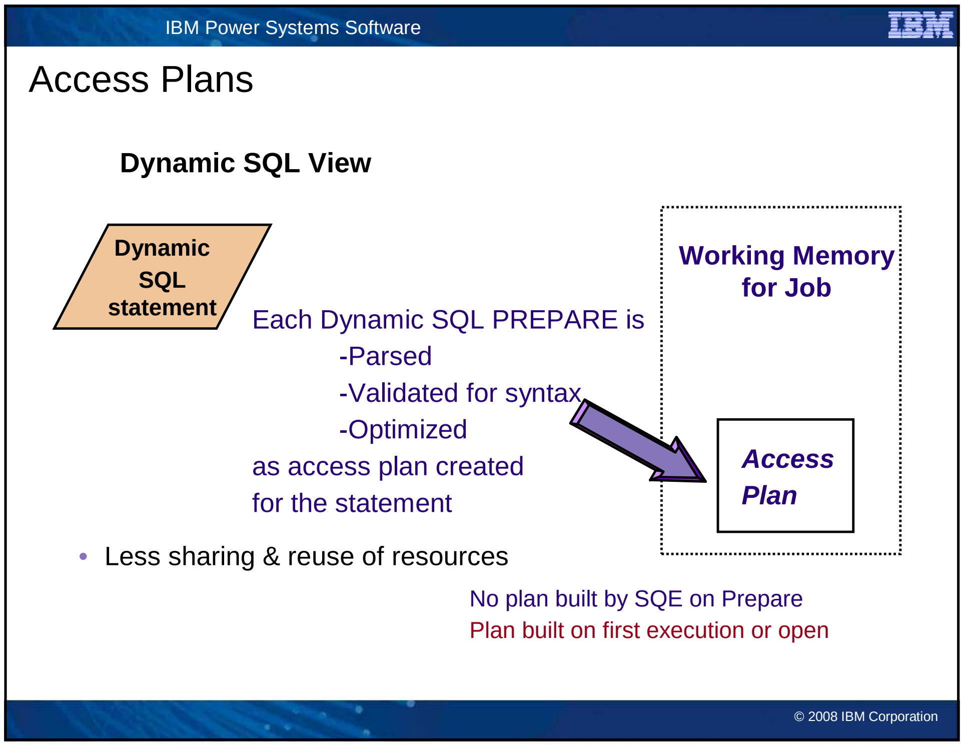
No plan built by SQE during compile
Plan built on first execution



Access Plans

Plan Contents:

- **A control structure that contains info on the actions necessary to satisfy each SQL request**
- **These contents include:**
  - Access Method
    Access path ITEM used for file 1.
    Key row positioning used on file 1.
  - Info on associated tables and indexes
    - Used to determine if access plan needs to be rebuilt due to table changes or index changes
    - EXAMPLE: a column has been removed from a table since the last time the SQL request was executed
  - Any applicable program and/or environment info
    - Examples: Last time access plan rebuilt, DB2 SMP feature installed



Access Plans

Dynamic SQL View

Dynamic SQL statement

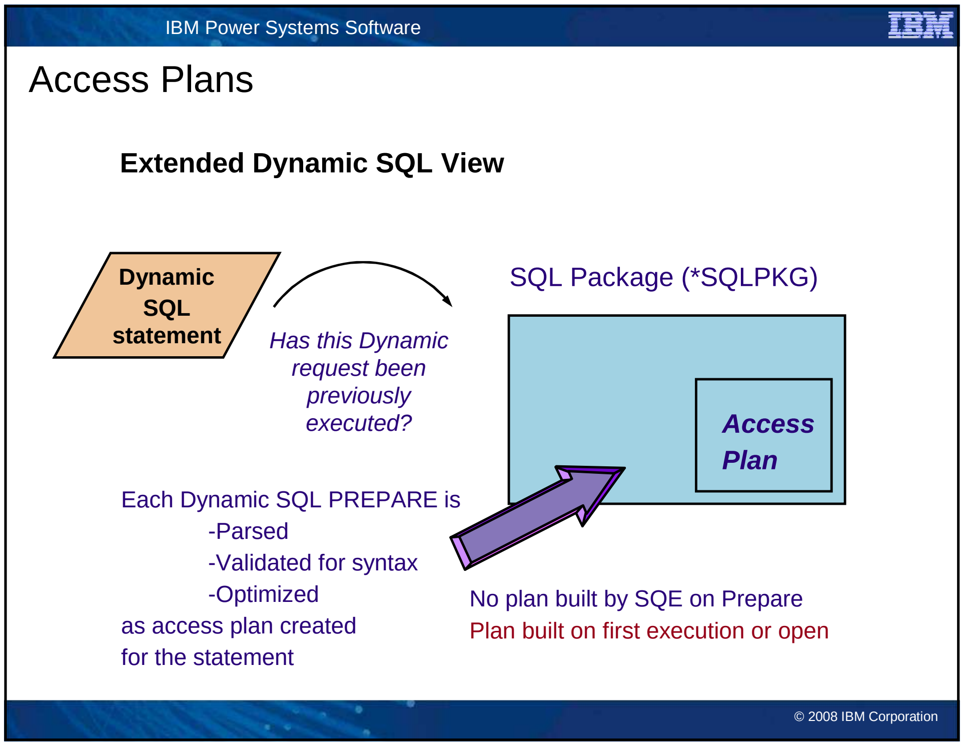
Each Dynamic SQL PREPARE is
-Parsed
-Validated for syntax
-Optimized
as access plan created for the statement

Working Memory for Job

Access Plan

- Less sharing & reuse of resources

No plan built by SQE on Prepare
Plan built on first execution or open



Access Plans

Extended Dynamic SQL View

Dynamic SQL statement

Has this Dynamic request been previously executed?

SQL Package (*SQLPKG)

Access Plan

Each Dynamic SQL PREPARE is
-Parsed
-Validated for syntax
-Optimized
as access plan created for the statement

No plan built by SQE on Prepare
Plan built on first execution or open

OPENing the Access Plan

- **Validate the Access Plan**
- **IF NOT Valid, THEN Reoptimize & update plan (late binding)**
  - Some of the more common reasons:
    - Different version of table object referenced (A1)
    - Significant change in Table row count (A4)
    - Index added (A5) or Index removed (A6)
    - Change in memory pool size (AB)
      - CQE optimizer only rebuilds plan when there has been a 2X change in memory pool size and runtime estimate greater than 2 seconds
      - SQE optimizer only rebuilds plan with a 2X change in memory pool size
  - All reasons document in DB2 Database Performance & Query Optimization book (or System Message IDs: CPI4323 & CPI4321)

- **Implement Access Plan: CREATE ODP (Open Data Path)**

Additional Access Plan Rebuild Reasons

- **Changes in the values of host variables and parameter markers**
  - Monitor reason code (A4 – 0002) for this type of plan rebuild, joblog rebuild messages may not be generated
  - Optimizer determines if new value changes "selectivity" enough to warrant a rebuild as part of plan validation...
    - When value used in selection against chosen index and selectivity is 10% different than value used with current access plan.
      - Selectivity change needs to be greater when Optimization time exceeds prior run time
      - CQE rebuild rules for selectivity rebuilds are similar
  - If program/package history shows current access plan used frequently in the past, then **new access plan** being built for data skew will be built as a **temporary access plan**

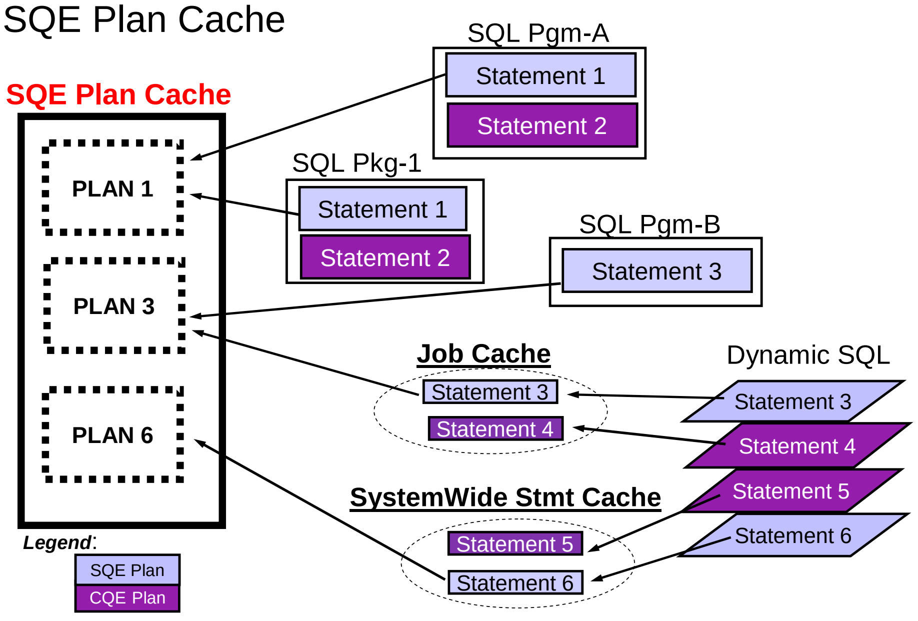
```
SELECT * FROM customers
WHERE state=:HV1
HV1 = 'NY'
```

```
SELECT * FROM customers
WHERE state=:HV1
HV1 = 'IA'
```

Access Plan Rebuild Considerations

- **Access plan updates are not always done in place**
  - If new space alllocated for rebuilt access plan, then size of program & package objects will grow over time - without any changes to the objects
  - Recreating program object is only way to reclaim "dead" access plan space
    - IBM utility now available: CALL QSYS/QSQCMPGM PARM('MYLIB' 'EMBPGM1')
    - DB2 has background compression algorithms for extended dynamic SQL packages
- **Static embedded SQL interfaces can have temporary access plan builds**
  - If DB2 unable to secure the necessary locks to update the program object, then a temporary access plan is built instead of waiting for the locks
  - If SQL programs have a heavy concurrent usage, may want to do more careful planning for Database Group PTF updates or IBM i upgrades
    - New IBM i releases causes all access plans to be rebuilt

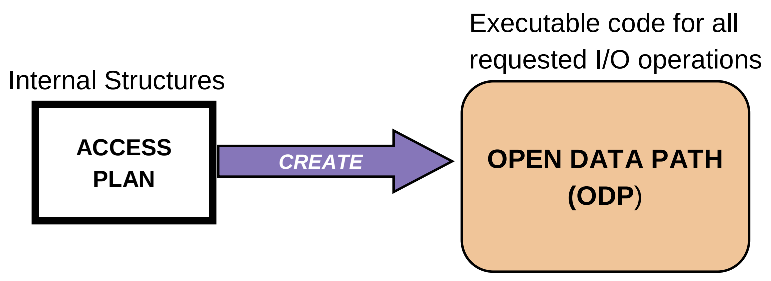
SQE Plan Cache



SQE Plan Cache

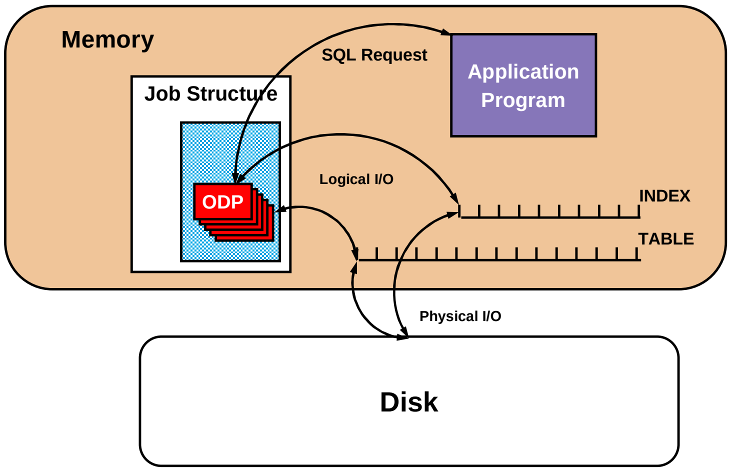
- **Self-managed cache for all plans produced by SQE Optimizer**
  - Allows more reuse of existing plans regardless of interface for identical SQL statements
    - Room for about 6000-10000 SQL statements
    - Plans are stored in a compressed mode
    - Up to 3 plans can be stored per SQL statement
  - Access is optimized to minimize contention on plan entries across system
  - Cache is automatically maintained to keep most active queries available for reuse
  - Foundation for a self-learning query optimizer to interrogate the plans to make wiser costing decisions
- **SQE Access Plans actually divided between Plan Cache & Containing Object (Program, Package, etc)**
  - Plan Cache stores the optimized portion (e.g., the index scan recipe) of the access plan
  - The access plan components needed for validating an SQL request (such as the SQL statement text and object information) is left in the original access plan location along with a virtual link to the plan in the Plan Cache
  - Plan cache entry also contains information on automatic stats collection & refresh
- **Plan Cache is cleared at IPL**

Access Plan to ODP



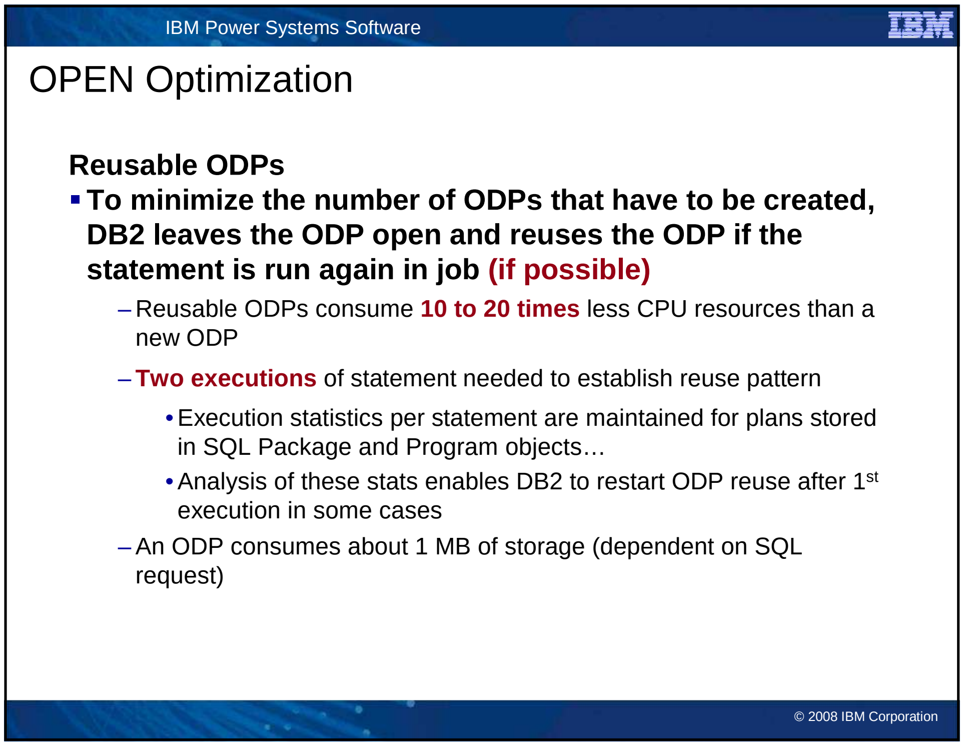
- **Create process is EXPENSIVE**
  - Longer execution time the first time an SQL statement is executed
- **Emphasizes the need of REUSABLE ODPs**

ODP's "In Action"



OPEN Optimization

- **OPENs can occur on:**
  - OPEN Statement
  - SELECT Into Statement
  - INSERT statement with a VALUES clause
  - INSERT statement with a SELECT (2 OPENs)
  - Searched UPDATE's
  - Searched DELETE's
  - Some SET statements
  - VALUES INTO statement
  - Certain subqueries may require one Open per subselect
- **The request and environment determine if the OPEN requires an ODP Creation ("Full" Open)**



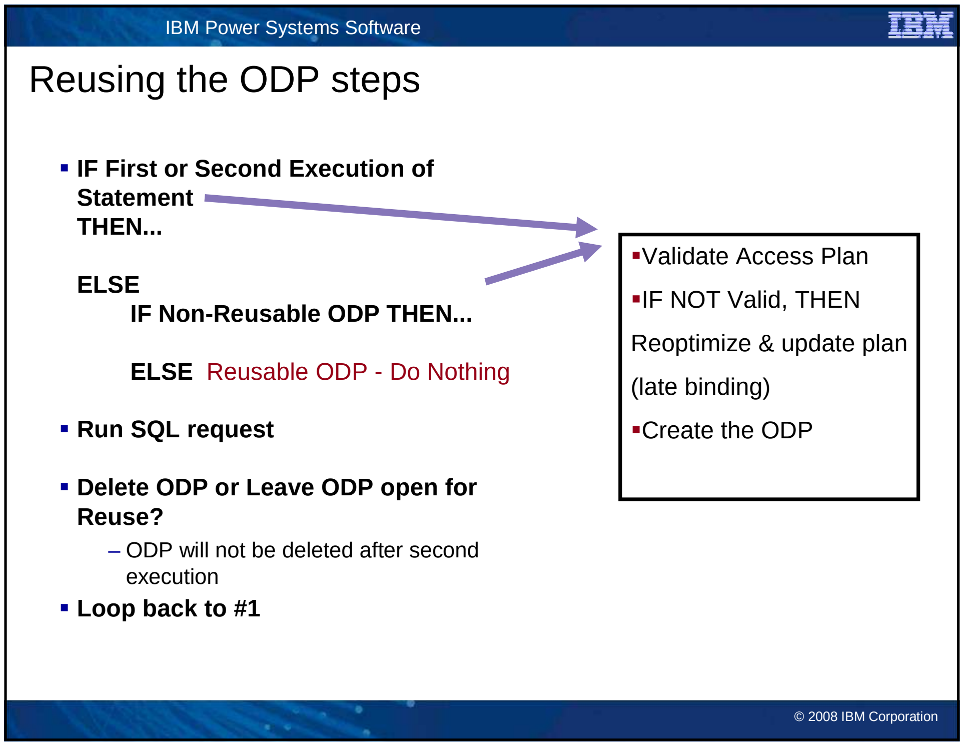
IBM Power Systems Software

OPEN Optimization

Reusable ODPs

- **To minimize the number of ODPs that have to be created, DB2 leaves the ODP open and reuses the ODP if the statement is run again in job (if possible)**
  - Reusable ODPs consume **10 to 20 times** less CPU resources than a new ODP
  - **Two executions** of statement needed to establish reuse pattern
    - Execution statistics per statement are maintained for plans stored in SQL Package and Program objects…
    - Analysis of these stats enables DB2 to restart ODP reuse after 1st execution in some cases
  - An ODP consumes about 1 MB of storage (dependent on SQL request)

© 2008 IBM Corporation



IBM Power Systems Software

Reusing the ODP steps

- **IF First or Second Execution of Statement THEN...**

  ELSE

  IF Non-Reusable ODP THEN...

  ELSE Reusable ODP - Do Nothing
- **Run SQL request**
- **Delete ODP or Leave ODP open for Reuse?**
  - ODP will not be deleted after second execution
- **Loop back to #1**

- Validate Access Plan
- IF NOT Valid, THEN Reoptimize & update plan (late binding)
- Create the ODP

© 2008 IBM Corporation

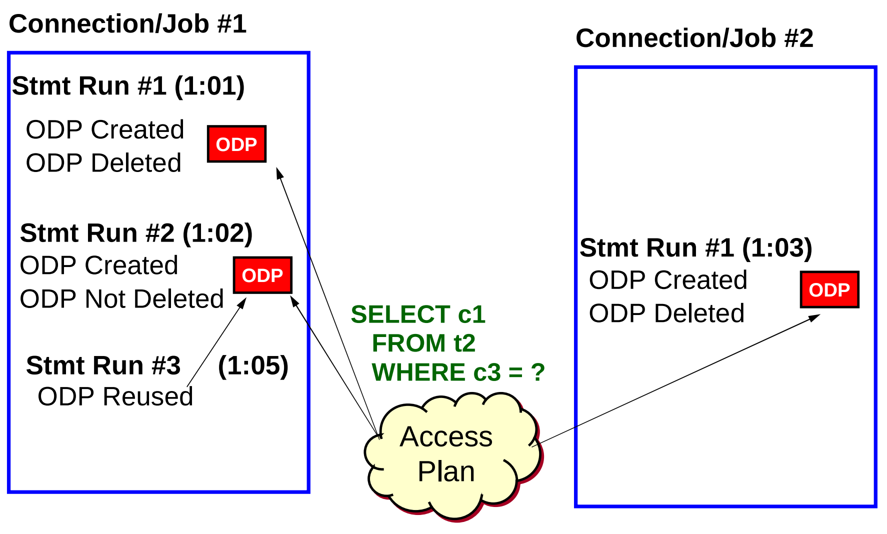
OPEN Optimization

Reusable ODP Example

```
INSERT INTO resultTable
   SELECT id, name
      FROM customers
         WHERE region = 'Central'
```

```
SQL7912   ODP created.
SQL7912   ODP created.          <------
...
SQL7913   ODP deleted.
SQL7913   ODP deleted.
SQL7985   CALL statement complete
SQL7912   ODP created.
SQL7912   ODP created.
...
SQL7914   ODP not deleted.
SQL7914   ODP not deleted.      <------
SQL7985   CALL statement complete
SQL7911   ODP reused.
SQL7911   ODP reused.           <----------
...
...
SQL7914   ODP not deleted.
SQL7914   ODP not deleted.
SQL7985   CALL statement complete
```

ODPs & Plans In Action



Miscellaneous considerations

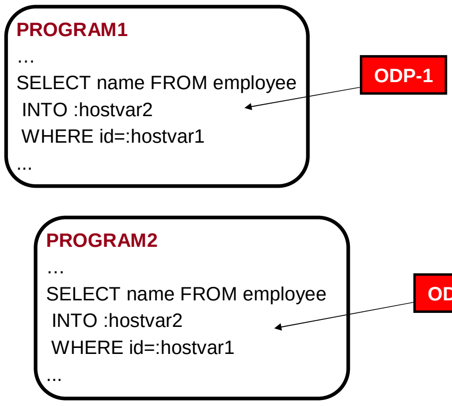
Reusable ODP Control - QSQPSCLS1 Data Area

- **Existence of data area allows the reuse behavior after first execution of SQL statement instead of the second execution**
  - DB2 checks for data area named QSQPSCLS1 in job's library list - existence only checked at the beginning of the job (first SQL ODP)
  - **USE CAREFULLY** since cursors that are not reused will consume extra storage
  - Data area contents, type, and length are not applicable

Reusable ODP Tips & Techniques

OPEN Optimization - Reuse Roadblocks

- **With static SQL, ODPs are NOT reused for the same SQL statement in different program objects**
  - Program objects include: Service Programs, SQL Procedures & Functions



PROGRAM1

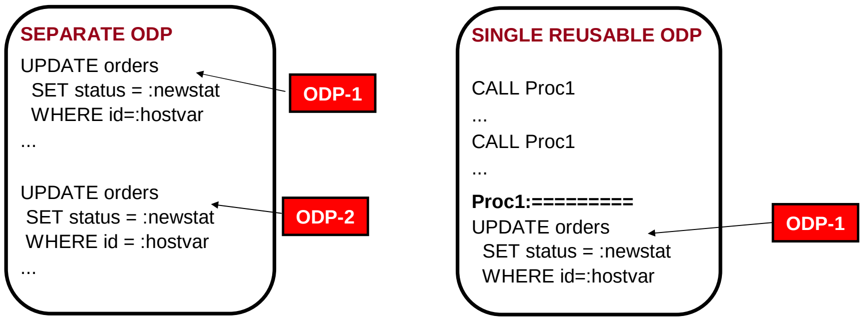
```
…
SELECT name FROM employee
 INTO :hostvar2
 WHERE id=:hostvar1
…
```

PROGRAM2

```
…
SELECT name FROM employee
 INTO :hostvar2
 WHERE id=:hostvar1
…
```

OPEN Optimization - Reuse Roadblocks

- **With static SQL, DB2 only reuses ODPs opened by the same statement**
  - If same statement will be executed multiple times, need to code logic so that statement is in a shared subroutine that can called



SEPARATE ODP

```
UPDATE orders
  SET status = :newstat
  WHERE id=:hostvar
…

UPDATE orders
 SET status = :newstat
 WHERE id = :hostvar
…
```

SINGLE REUSABLE ODP

```
CALL Proc1
…
CALL Proc1
…
Proc1:=========
UPDATE orders
  SET status = :newstat
  WHERE id=:hostvar
```

OPEN Optimization - Reuse Roadblocks

Location of DB2 objects may have changed:

- Unqualified table and the library list has changed since the ODP was opened with *SYS naming mode (RC: O)
  - If table location is not changing (library list just changing for other objects), then default collection can be used to enable reuse
  - Default collection exists for static, dynamic, and extended dynamic SQL
    - SET CURRENT SCHEMA to specify default schema for dynamic SQL
- Override Database File (OVRDBF) or Delete Override (DLTOVR) command issued for tables associated with an ODP that was previously opened (RC: J)
- SQL Path changed effecting resolution of UDF Calls (RC: J)
- Program being shared across Switchable Independent ASPs (IASP) where library name is the same in each IASP

OPEN Optimization - Reuse Roadblocks

- **SET SESSION AUTHORIZATION statement (RC: Q)**
- **System CL commands such as CLRPFM (RC: G)**
- **Commit or Rollback involving Declared Temporary Table that was created with “ON COMMIT DELETE ROWS” (RC: E)**
- **Commit or Rollback due to the abnormal termination of a database connection (RC: E)**
- **Temporary tables when multiple jobs are sharing the same program**

OPEN Optimization - Reuse Roadblocks

- **ODP requires temporary index**
  - Temporary index build does not always cause an ODP to be non-reusable, optimizer does try to reuse temporary index if possible
    - If SQL run multiple times and index is built on each execution, creating a permanent index could make ODP reusable
    - If host variable value used to build selection into temporary index (ie, sparse), then ODP is not reusable because temporary index selection can be different on every execution of the query
      - Optimizer will tend to avoid creating sparse indexes if the statement execution history shows it to be a "frequently executed" statement
  - Temporary indexes are not usable by other ODP's, unless they are SQE Autonomic Indexes

OPEN Optimization

UPDATE WHERE CURRENT OF Reuse

- **If an UPDATE WHERE CURRENT OF request contains a function or operator on the SET clause, then an open operation must be performed**
- **Can avoid this open by performing the function or operation in the host language**
  - **Code operation into host language...**

```
FETCH EMPT INTO :Salary;
Salary = Salary + 1000;
UPDATE EMPLOYEE
     SET Salary = :Salary
      WHERE CURRENT OF Empt;
```

  - **Instead of...**

```
FETCH EMPT INTO :Salary;
UPDATE Employee
     SET Salary = :Salary+1000
     WHERE CURRENT OF Empt;
```

OPEN Optimization - Reuse Considerations

- **Reusable ODP's do have one shortcoming... once reuse mode has started access plan is NOT rebuilt when the environment changes**
  - –What happens to performance if Reusable ODP is now run against a table that started out empty and that table is now substantially bigger than the first execution? *******
  - –What if index added for tuning after 5th execution of statement in the job? *******
  - –What if selectively of host variable or parameter marker greatly different on 5th execution of statement?
  - –*******NOT an issue with SQE since V5R3 – SQE recognizes new indexes and table size changes while in ODP reuse mode (RC: A)

Dynamic & Extended Dynamic SQL

Dynamic SQL Tuning

- **With Dynamic interfaces, full opens are avoided by using a "PREPARE once, EXECUTE many" design point when an SQL statement is going to be executed more than once**

- **A PREPARE does NOT automatically create a new ODP on each execution**
  - –DB2 performs caching on PREPARE & OPEN within a job/connections
  - –DB2 caching is not perfect (and subject to change)
    - White space and different case (upper vs lower) will negatively impact the DB2 caching
    - DB2 caches reside in the System ASP in a Switchable IASP environment
  - –Good application design is ONLY way to guarantee ODP reuse

Dynamic SQL Tuning - Parameter Markers

- **Parameter Markers are one implementation method for "EXECUTE many"**
  - –Improves chance for reusable ODPs
  - –Ex: want to run the same SELECT statement several times using different values for customer state
    - 50 different statements/opens for each of the states OR...
    - *Single SQL statement that allows you to plug in the needed state value*
  - –DB2 does <u>attempt</u> to automate this behavior

Dynamic SQL Tuning- Parameter Markers

Parameter Marker Example

```
StmtString = 'DELETE FROM employee WHERE empno=?';
...
PREPARE s1 USING :StmtString;
...
 EXECUTE s1 USING :InputEmpNo;
...
```

Dynamic SQL Tuning - Parameter Markers

Automatic Parameter Marker Conversion

- DB2 automatically tries to convert literals into parameter markers to make statement look repetitive

SELECT name, address FROM customers
WHERE orderamount > 1000.00 AND state = 'NY'

CONVERTED TO:

SELECT name, address FROM customers
WHERE orderamount > ? AND state = ?

UPDATE customers SET status = 'A'
WHERE orderamount >= 10000

CONVERTED TO:

UPDATE customers SET status = ?
WHERE orderamount >= ?

Extended Dynamic & Packages

- **Package is searched to see if there is a statement with the same SQL and attributes**
  - Hash tables used to make statement searches faster
- **If a match is found, then a new statement entry name is allocated with a pointer to the existing statement information (access plan, etc)**
  - DB Monitor can be used to determine if "packaged" statement used at execution time:

```
SELECT qqc103, qqc21, qq1000 from ‹db monitor table›
WHERE qqrid=1000 AND qvc18='E'
```

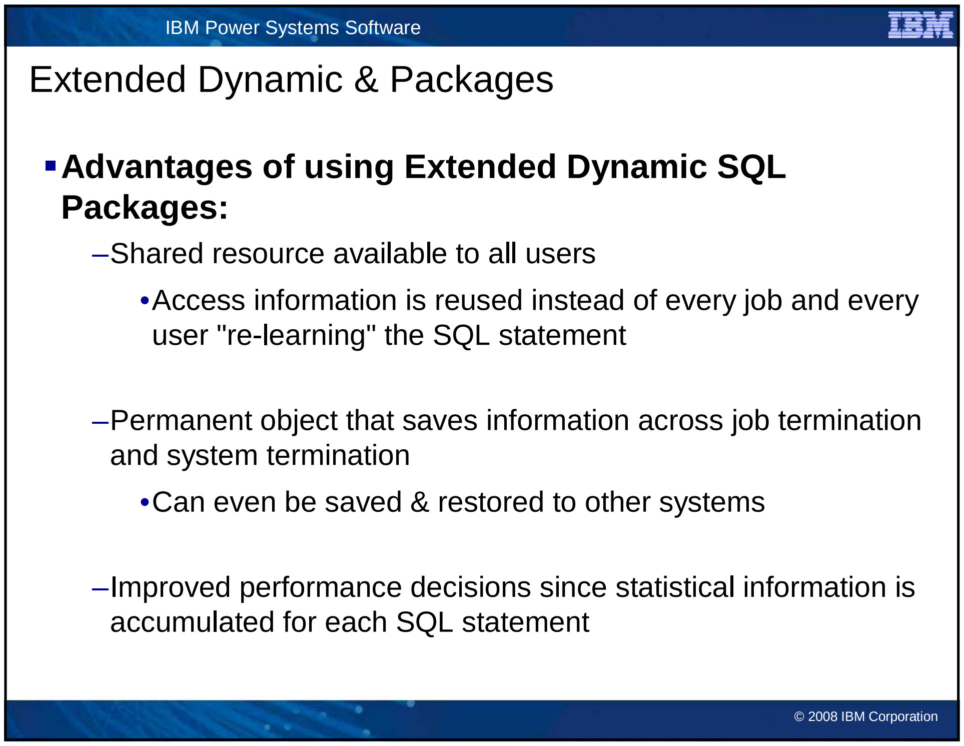
Extended Dynamic & Packages

Package Contents:

- Statement name
- Statement text
- Statement parse tree
- Access Plan

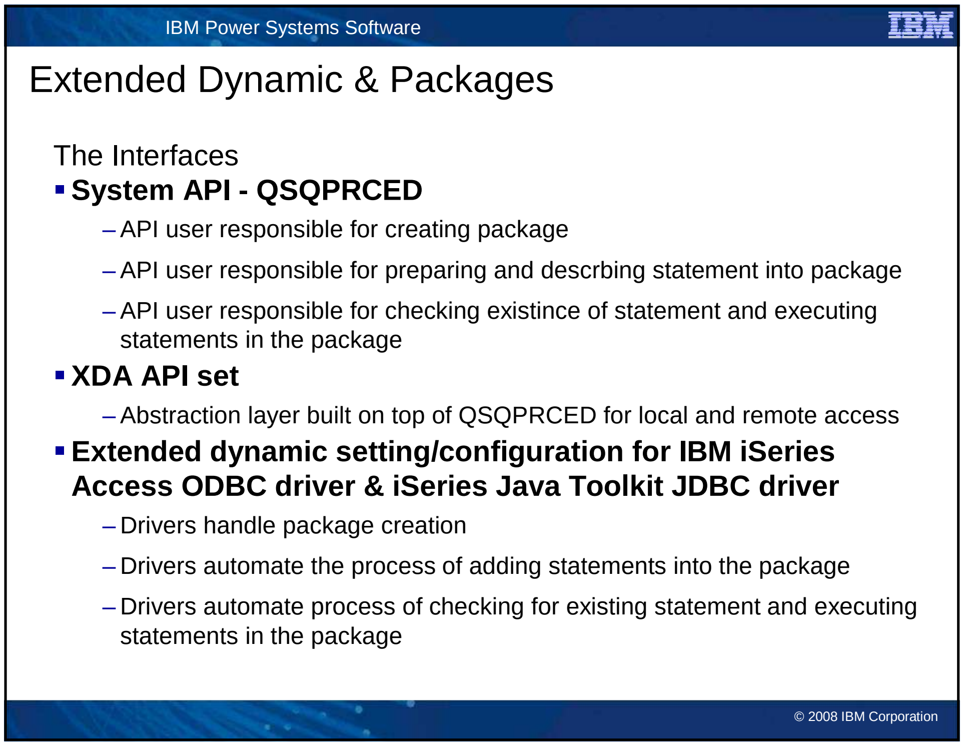
PRTSQLINF output

```
STATEMENT NAME:  QZ7A6B3E74C31D0000
Select IID, INAME, IPRICE, IDATA from TEST/ITEM where
IID in ( ?, ?, ?, ?)
  SQL4021  Access plan last saved on 12/16/96 at 20:21:45.
  SQL4020  Estimated query run time is 1 seconds.
  SQL4008  Access path ITEM used for file 1.
  SQL4011  Key row positioning used on file 1.
...
STATEMENT NAME:  QZ7A6B3E74DD6D8000
Select CLAST, CDCT, CCREDT, WTAX from  TEST/CSTMR,
TEST//WRHS where  CWID=? and CDID=?
   SQL4021  Access plan last saved on 12/16/96 at 20:21:43.
   SQL4020  Estimated query run time is 1 seconds.
   SQL4007  Query implementation for join position 1 file 2.
   SQL4008  Access path WRHS used for file 2.
   SQL4011  Key row positioning used on file 2.
   SQL4007  Query implementation for join position 2 file 1.
   SQL4006  All access paths considered for file 1.
   SQL4008  Access path CSTMR used for file 1.
   SQL4014  0 join field pair(s) are used for this join position.
   SQL4011  Key row positioning used on file 1.
```



Extended Dynamic & Packages

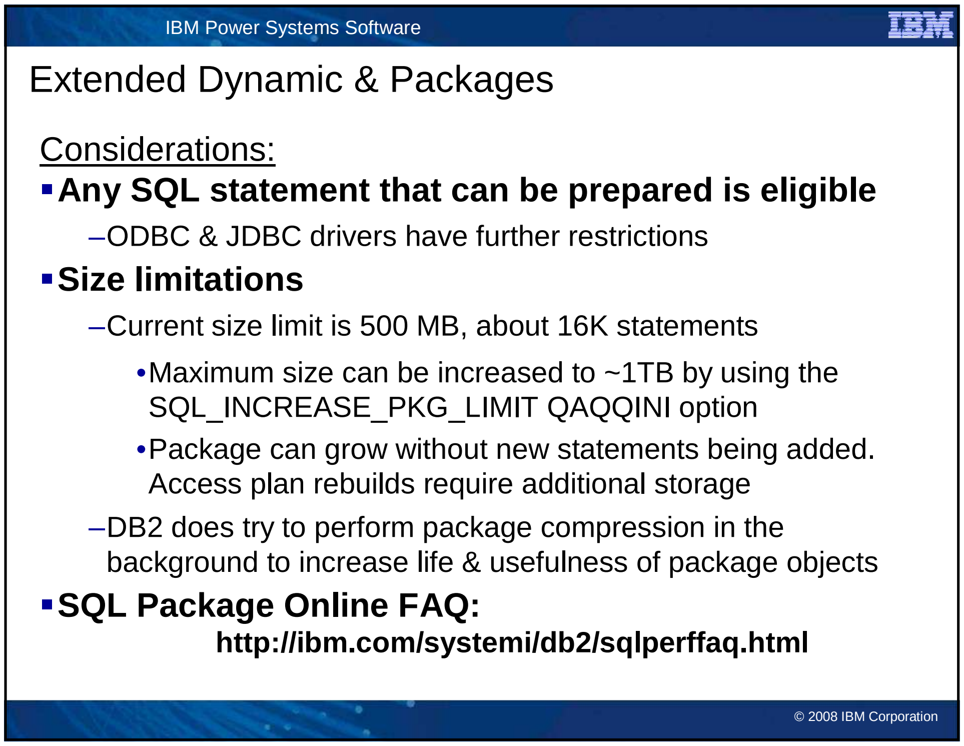
- **Advantages of using Extended Dynamic SQL Packages:**
  - –Shared resource available to all users
    - •Access information is reused instead of every job and every user "re-learning" the SQL statement
  - –Permanent object that saves information across job termination and system termination
    - •Can even be saved & restored to other systems
  - –Improved performance decisions since statistical information is accumulated for each SQL statement



Extended Dynamic & Packages

The Interfaces

- **System API - QSQPRCED**
  - –API user responsible for creating package
  - –API user responsible for preparing and descrbing statement into package
  - –API user responsible for checking existince of statement and executing statements in the package
- **XDA API set**
  - –Abstraction layer built on top of QSQPRCED for local and remote access
- **Extended dynamic setting/configuration for IBM iSeries Access ODBC driver & iSeries Java Toolkit JDBC driver**
  - –Drivers handle package creation
  - –Drivers automate the process of adding statements into the package
  - –Drivers automate process of checking for existing statement and executing statements in the package

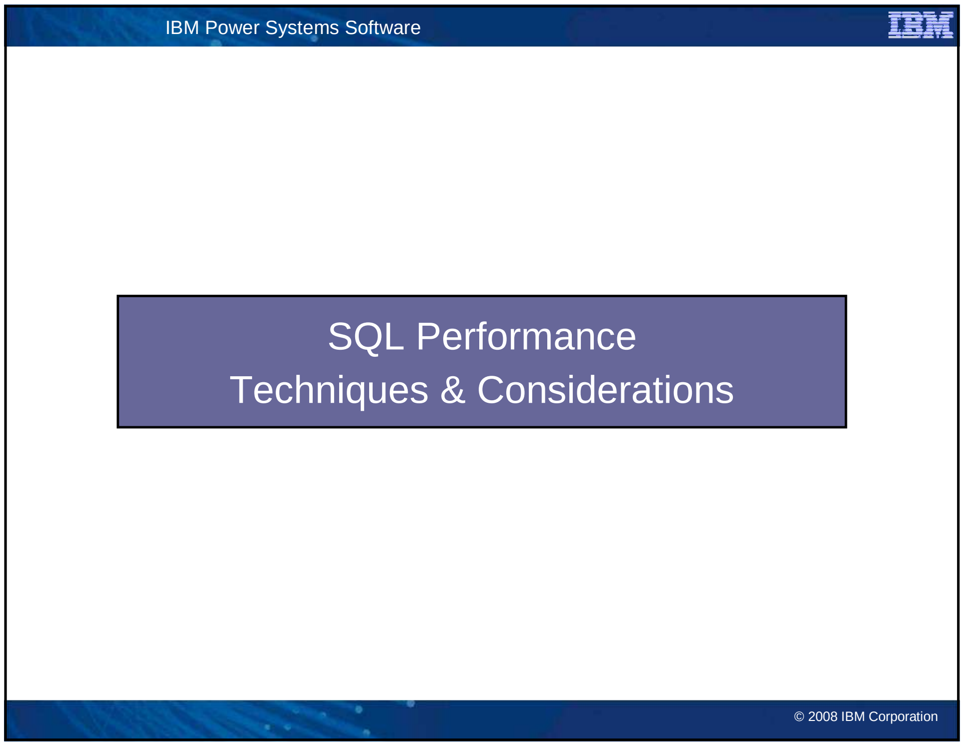


Extended Dynamic & Packages

Considerations:

- **Any SQL statement that can be prepared is eligible**
  - ODBC & JDBC drivers have further restrictions
- **Size limitations**
  - Current size limit is 500 MB, about 16K statements
    - Maximum size can be increased to ~1TB by using the SQL_INCREASE_PKG_LIMIT QAQQINI option
    - Package can grow without new statements being added. Access plan rebuilds require additional storage
  - DB2 does try to perform package compression in the background to increase life & usefulness of package objects
- **SQL Package Online FAQ:**

  http://ibm.com/systemi/db2/sqlperffaq.html



SQL Performance Techniques & Considerations

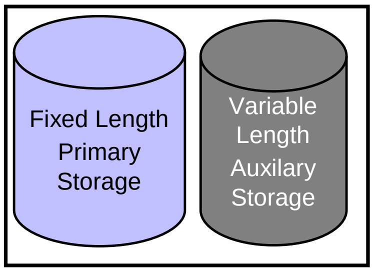
VARCHAR considerations

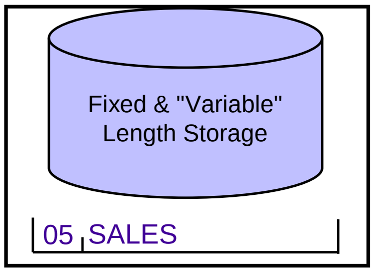
- **Variable length columns (VARCHAR/VARGRAPHIC)**
  - If primary goal is space saving, include ALLOCATE(0) with VARCHAR definition
  - If primary goal is performance, ALLOCATE value should be wide enough to accommodate 90-95% of the values that will be assigned to the varying length column
    - Minimizes number of times that DB2 has to touch data in overflow storage area
  - BLOB/CLOB columns stored in the same overflow container
- **VARCHAR columns more efficient on wildcard searches**
  - DB2 able to stop searching after the end of the string - with fixed length characters it must search to the end of string, even if all blanks

VARCHAR considerations

```
CREATE TABLE dept
(
  id      CHAR(4),
  name VARCHAR(40),
  bldg_num INTEGER
 )
```

```
CREATE TABLE dept
(
  id      CHAR(4),
  name VARCHAR(40)
            ALLOCATE(40),
  bldg_num INTEGER
 )
```


Fixed Length Primary Storage
Variable Length Auxilary Storage



SQL Table considerations

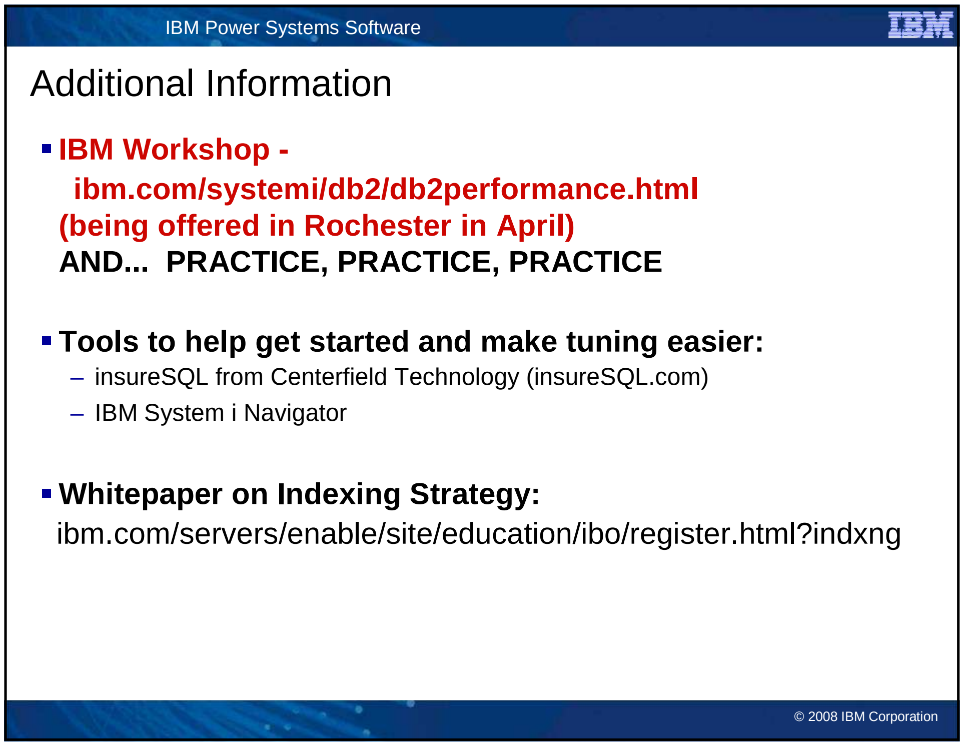
- **SQL-created tables are faster on reads and slower on writes that DDS-created tables**
- **Tables with high number of concurrent inserts may also benefit from Concurrent Insert feature ("Holey Inserts")**
  - Activated by doing a CALL QDBENCWT '1' & then IPLing system
  - Default starting with V5R3, unless the release is slip-installed
- **If you have tables that receive a high-velocity of inserts in concurrent enviroments, then it may be beneficial to pre-allocate storage for the table**
  - CHGPF FILE(lib/table1) SIZE(**125000** 1000 3) ALLOCATE(*YES)
  - After CHGPF, a CLRPFM or RGZPFM command must be executed to "activate" the allocation

Stored Procedures

- **Procedures most effective from a performance perspective when multiple operations performed on a single procedure call**
- **SQL Procedure Language (PSM) considerations**
  - Generated C code with embedded SQL will not be as efficient as user-written code, **big improvements with V5R4**
  - No support for blocked fetches & inserts
  - Local variable suggestions
    - Declare local variables as not null
    - Use integer instead of decimal precision with 0
    - Minimize the usage of character & date variables
    - Use the same data type, length and scale for numeric variables that are used together in assignments
  - Minimize the number of nested calls to other SQL procedures
  - Consider moving handlers for a specific condition/statement within a nested compound statement

```
BEGIN
  DECLARE CONTINUE HANDLER
             FOR SQLSTATE ' 23504'...
  ...
  DELETE FROM master WHERE id=1;
  ...
```

```
BEGIN
 ...
  BEGIN
      DECLARE CONTINUE HANDLER FOR
                          SQLSTATE ' 23504'...
      DELETE FROM master WHERE id=1;
   END
...
```

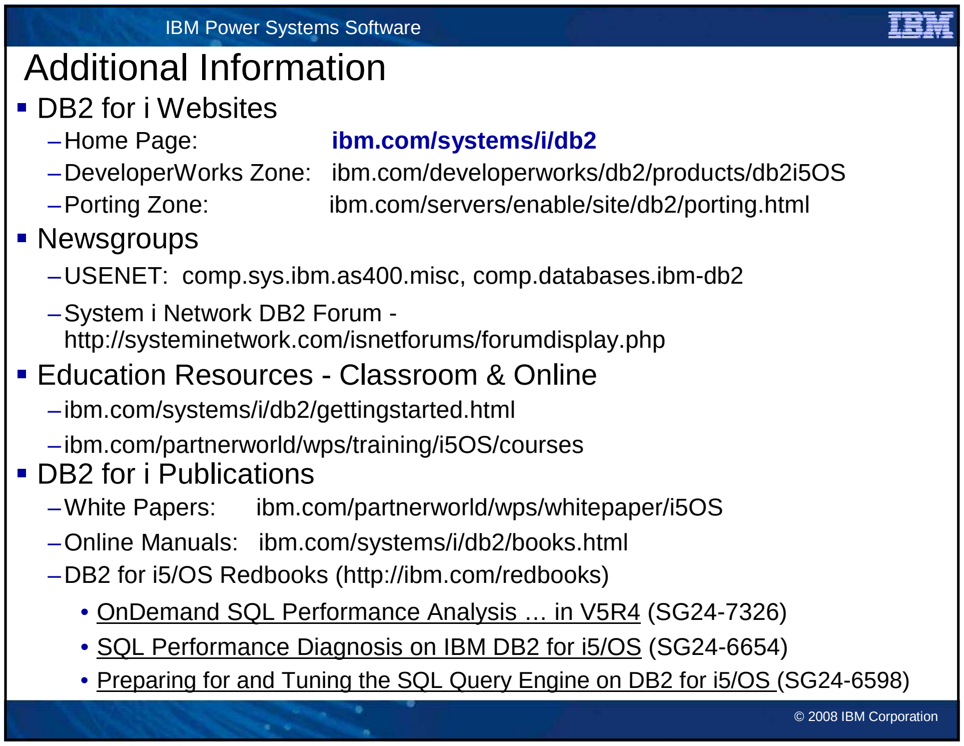


Additional Information

- **IBM Workshop -**
  ibm.com/systemi/db2/db2performance.html
  (being offered in Rochester in April)
  AND... PRACTICE, PRACTICE, PRACTICE

- **Tools to help get started and make tuning easier:**
  – insureSQL from Centerfield Technology (insureSQL.com)
  – IBM System i Navigator

- **Whitepaper on Indexing Strategy:**
  ibm.com/servers/enable/site/education/ibo/register.html?indxng



Additional Information

- DB2 for i Websites
  – Home Page: **ibm.com/systems/i/db2**
  – DeveloperWorks Zone: ibm.com/developerworks/db2/products/db2i5OS
  – Porting Zone: ibm.com/servers/enable/site/db2/porting.html
- Newsgroups
  – USENET: comp.sys.ibm.as400.misc, comp.databases.ibm-db2
  – System i Network DB2 Forum -
  http://systeminetwork.com/isnetforums/forumdisplay.php
- Education Resources - Classroom & Online
  – ibm.com/systems/i/db2/gettingstarted.html
  – ibm.com/partnerworld/wps/training/i5OS/courses
- DB2 for i Publications
  – White Papers: ibm.com/partnerworld/wps/whitepaper/i5OS
  – Online Manuals: ibm.com/systems/i/db2/books.html
  – DB2 for i5/OS Redbooks (http://ibm.com/redbooks)
    - OnDemand SQL Performance Analysis … in V5R4 (SG24-7326)
    - SQL Performance Diagnosis on IBM DB2 for i5/OS (SG24-6654)
    - Preparing for and Tuning the SQL Query Engine on DB2 for i5/OS (SG24-6598)