IBM

@server

Session: 401918
Agenda Key: 56CB

@server iSeries

Performance Tune iSeries Access ODBC

Brent Nelson - bmnelson@us.ibm.com
iSeries Access Development

ON DEMAND BUSINESS™

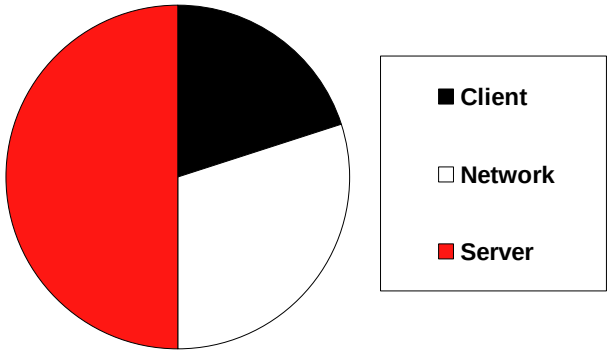
IBM

@server

Typical Performance Problems

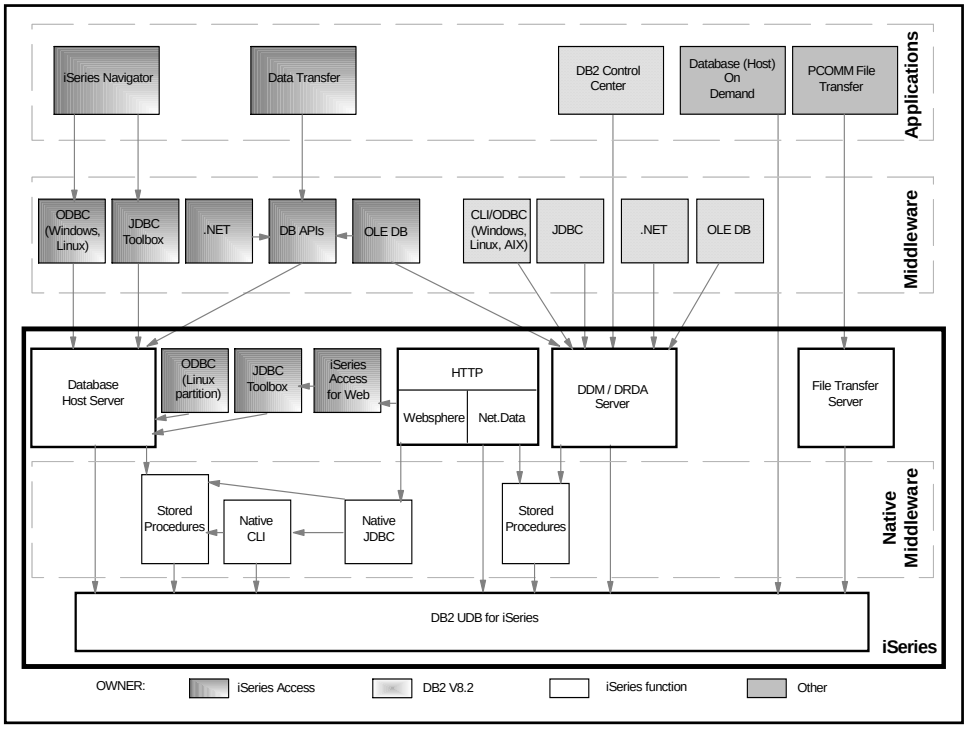
- Fetching data
- Long-running SQL queries
- Network issues
- Inserting data
- Lots of connections

Typical Performance Problems



Agenda

- Performance Considerations
  - Application Design
  - Network
  - Database Design
- Examples
  - 3-Tier Application
  - Off-the-shelf Applications
- Appendices



IBM

eserver

Performance Considerations

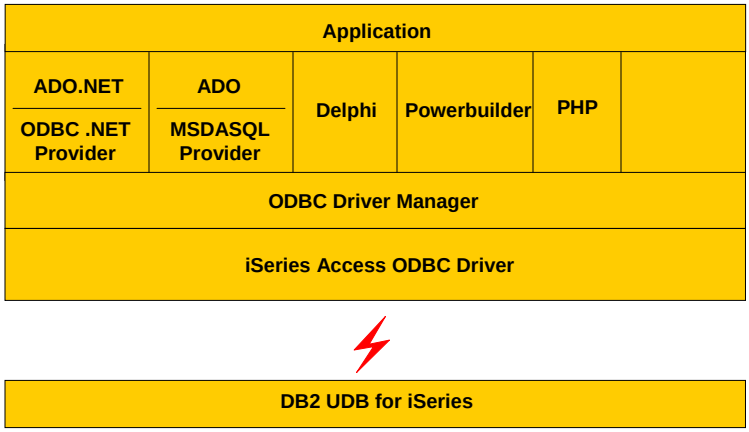
- **Application design**
  - Choice of programming interface
  - Tips on calling ODBC APIs
  - Block insert
  - Isolation level and concurrency
- Network
- Database design

Explanation of Scale

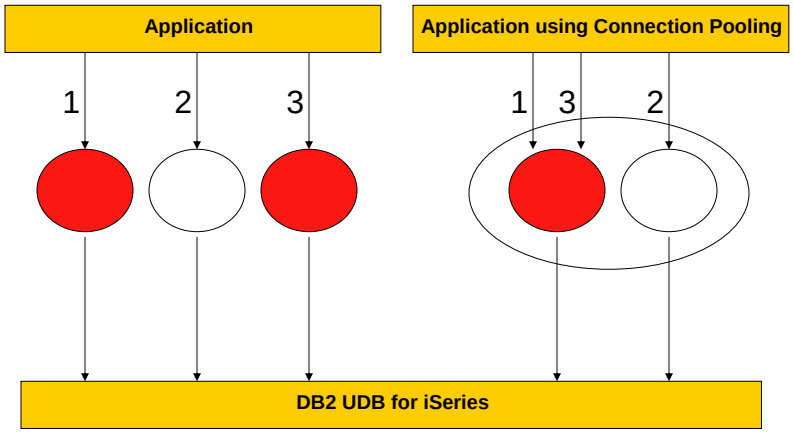
Programming Required?	Potential Benefit
Y	3
Y = Yes N = No	1 = Low 2 = Medium 3 = High

Y 2

Choosing a Programming Interface



Connection Pooling



Running SQL statements

- Statement pooling
- Use parameter markers (?'s)
- "Prepare once, Execute many" method

Insert Data - Insert with Constants

```
/*****************************************/
/* repeat for each row to be inserted */
/*****************************************/
strcpy(stmt,"insert into IBMLIB.TAB1 values('ax',123,'c')");

rc = SQLExecDirect(hStmt,stmt,SQL_NTS);
:
```


Insert Data - Prepare Once, Execute Many

```
strcpy(stmt,"insert into IBMLIB.TAB1 values (?,?,?)");
rc = SQLPrepare(hStmt,stmt,SQL_NTS);

/* Specify the bindings for each parameter */
rc = SQLBindParameter(hStmt, .... );

for (i=0;i<ROW_COUNT;i++) {
  /* Set variables for corresponding parameter markers */
  strcpy(col1,value);
  rc = SQLExecute(hStmt);
 }
```

Insert Data – Block Insert

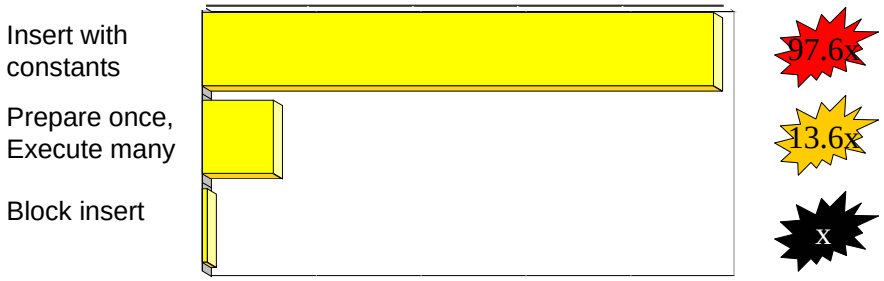
```
strcpy(stmt,"insert into IBMLIB.TAB1 values (?,?,?)");
rc = SQLPrepare(hStmt,stmt,SQL_NTS);

rc = SQLSetStmtAttr(hStmt,SQL_ATTR_PARAMSET_SIZE,
                    (PTR)ROW_COUNT,SQL_IS_INTEGER);

/* Specify the bindings for each parameter */
rc = SQLBindParameter(hStmt, .... );

for (i=0;i<ROW_COUNT;i++) {
  /* Set variables for corresponding parameter markers */
  strcpy(col1[i],value);
 }
rc = SQLExecute(hStmt);
```


Insert Data - Block Insert Notes

- Best alternative because:
  - Parsed only once
  - Avoids full open/close of target table
  - 1 send/receive for N rows
  - Path optimized from client->database
- Drawbacks
  - May not be practical for all applications
  - AS/400 only feature if "? rows" clause is used

Block Insert Performance

For 500 36-byte rows with three columns



Deleting data

- Consider:

CALL QSYS.QCMDEXC('CLRPFM FILE(MYLIB/MYFILE)',0000000025.00000)

- Instead of:

DELETE FROM MYLIB.MYFILE

Fetching Data - Blocking

- Options that factor into blocking factor
  - Forward-only cursor
    - Block fetch of 1 row - (BLOCKFETCH keyword)
    - Block size – (BLOCKSIZE keyword)
    - Rowset size
  - Scrollable cursor
    - Rowset size

Fetching Data - Forward-only cursor examples

- Table retrieving from has a 32K row size
- **Example 1:**
  - Application is fetching one row at a time
  - Block fetch of 1 row is being used with a Block Size of 32K
  - Results: 1 Row at a time is fetched
- **Example 2:**
  - Application is fetching one row at a time
  - Block fetch of 1 row is being used with a Block Size of 512K
  - Results: ~16 Rows are fetched at a time
- **Example 3:**
  - Application is fetching with a rowset size of 50 rows
  - Results: 50 Rows are fetched at a time

Fetching Data - Scrollable cursor examples

- Table retrieving from has a 32K row size
- Example 1:
  - Application is fetching one row at a time
  - Results: 1 Row is fetched at a time
- Example 2:
  - Application is fetching with a rowset size of 50 rows
  - Results: 50 Rows are fetched at a time

Fetching Data – LOB threshold

- MAXFIELDLEN keyword
- Default is 32 (KB)
- Lower setting usually better

Fetching Data – Other Notes

- SQLBindCol vs. SQLGetData

Isolation Level

- By default, ODBC runs with autocommit ON
  - Equivalent to *NONE on pre-V5R3 iSeries
- Use lowest level of transaction isolation for least overhead

	Isolation Level	Overhead
(*NONE)	Commit immediate	less
(*CHG)	Read uncommitted	
(*CS)	Read committed	
(*ALL)	Repeatable read	
(*RR)	Serializable	more

Application Design Summary

- Use parameter markers
- SQLPrepare once, SQLExecute many times
- Use blocking effectively

Performance Considerations

- Application design
- **Network**
  - Reduce Trips to Server
  - Reduce Data Between Server
- Database design

Network

- About 1/3 of all ODBC performance problems
- ODBC sends much larger blocks of data then most applications

Reduce Trips to Server

- Stored procedures
- Triggers
- Connection pooling
- IP Address lookup in iSeries Navigator connection properties
- Port lookup in iSeries Navigator connection properties
- Block inserts
- Block fetches
- Lazy close
- Pre-fetch

Reduce Data Between Server

- Data compression
- LOB threshold
- Avoid “select *” SQL statements

N 3

Data Compression

- “Enable Data Compression” DSN setting ON by default
- Highly recommended for variable length fields
  - e.g. VARCHAR, VARGRAPHIC
- Improved compression algorithm on V5R1+ servers

CWBCOPWR

- Options to concentrate on:
  - Communication buffer size (Option /SC)
  - TCP/IP buffer size (Options /WSS and /WSR)
  - TCP/IP nagling (Option /NGL)

- Found in \Program Files\IBM\Client Access directory
- See CWBCOPWR.HTM for help

Network Summary

- Reduce Trips to Server
- Reduce Data Between Server

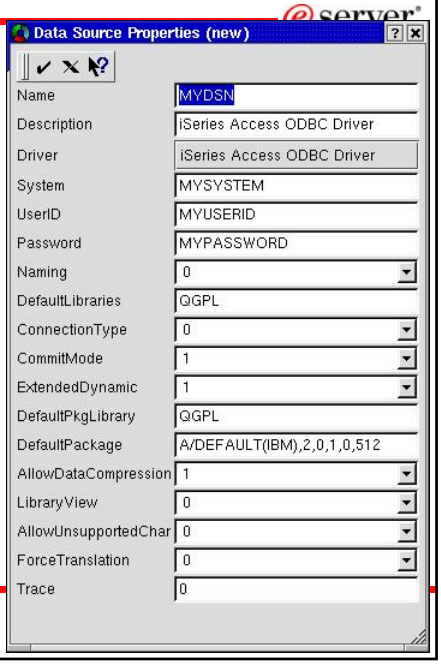
Windows ODBC DSN Setup GUI

- Performance tab
- Advanced performance options
- Package tab

- ODBC DSN keywords
  - http://publib.boulder.ibm.com/iseries/v5r2/ic2924/index.htm?info/rzaik/rzaik678.htm

Linux ODBC DSN GUI

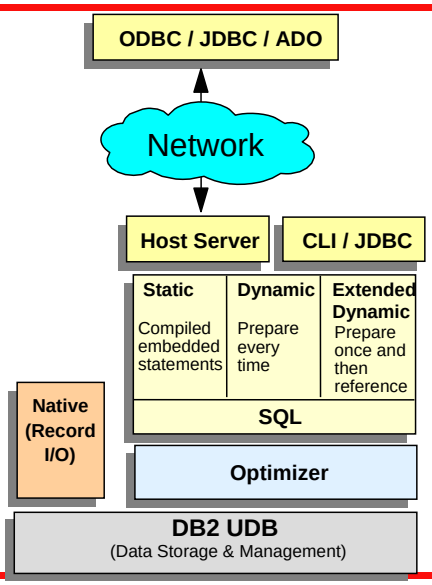
- Other options added to the .odbc.ini file or programmatically specified via SQLDriverConnect API



Performance Considerations

- Application design
- Network
- Database design
  - Indexes
  - Extended dynamic support
  - Stored procedures
  - Trigger programs

SQL Interfaces



SQL Statement Tuning

- Avoid SELECT *
- SQL clauses:
  - OPTIMIZE FOR N ROWS
  - FOR FETCH ONLY / FOR UPDATE
  - FETCH FIRST N ROWS ONLY

Indexes

- Two methods for accessing data--keyed and sequential
  - Lack of understanding can seriously impact performance
- Aimed specifically at optimizing queries (SELECTs)
- Most important aspect--proper indices for queries

Indexes

- Index is required for following cases:
  - ORDER BY
  - GROUP BY
  - JOIN of two tables
- Optimizer will create index if an appropriate one doesn't exist

Indexes

- Create index over tables when queries return less than 20% of table
  - Create index over columns used in WHERE clause
  - Create index over columns used to join tables
  - Create index on grouping columns

- White paper: "Indexing Strategies for DB2 UDB for iSeries"
  - http://www-919.ibm.com/servers/eserver/iseries/developer/bi/documents/strategy/strategy.pdf

Access Plans and ODPs

- Minimize access plan builds
  - Reduces CPU use on server
- Reuse (Open Data Path) ODPs
  - Prepare once/run many
  - Statement pooling
  - Connection pooling
  - Use parameter markers
  - Cache package locally

Extended Dynamic Access (Packages)

1. Application runs:

SELECT * FROM MYTABLE WHERE COL1=?

2. Application re-runs:

SELECT * FROM MYTABLE WHERE COL1=?

Package

Extended Dynamic Access (Packages)

- Caches SQL statements on server or local
- Allows reuse of statements (across sessions)
  - Prepare once, execute many
- Statements can be shared among many users
- Can reuse the ODP if local package caching used

SQL Statements in Packages

- The following SQL statements are put into extended dynamic packages:
  - Statements that contain parameter markers
  - INSERT with subselect
  - Positioned UPDATE or DELETE
  - SELECT FOR UPDATE
  - DECLARE PROCEDURE

iSeries Navigator Tools

- SQL Performance Monitor
  - DSN setting for “Enable Database Monitor” located on Diagnostic tab
  - File stored in QUSRSYS/QODBxxx where xxx is the job number
- Visual Explain
  - Query Access Plan Diagram
  - Index Advisor

Stored Procedures

- Powerful tool--accepts input parameters, returns output parameters
- SQL procedure
- Stored procedure as external procedure
  - Does not need to contain SQL
  - Can be any C, RPG, CL, COBOL, Java program
- Can return multiple result sets
- Can hide details of application from user

Stored Procedures

- Stored Procedures can be utilized to provide...
  - Static SQL performance behavior to dynamic ODBC & JDBC client requests
  - Access to tuning knobs/precompiler options such as ALWCPYDTA, etc

Stored Procedures Example 1

Example:
Logic to execute 2 statements w/o stored procedure:

```
/* Application runs INSERT statement - if the     */
/* statement is successful, it runs an UPDATE   */
/* statement.                                                */
strcpy(stmt,"INSERT INTO IBMLIB.TAB1 VALUES(?,?,?)");
rc=SQLPrepare(hstmt1,stmt,SQL_NTS);
strcpy(stmt2,"UPDATE IBMLIB.TAB2 SET COL1 = ...");
rc=SQLPrepare(hstmt2,stmt2,SQL_NTS);
rc=SQLBindParameter(hstmt1,... );
:
rc = SQLExecute(hstmt1);
if (rc==SQL_SUCCESS)
 {
   rc=SQLExecute(hstmt2);
   if (rc==SQL_SUCCESS)
   :
```

Stored Procedures Example 2

Example:
Logic to execute the same statements using stored procedure:

```
/* Application invokes stored procedure passing all      */
/* the parameters necessary for both the INSERT and */
/* UPDATE statement                                      */
strcpy(stmt,"CALL IBMLIB.PROC1 (?,?,?)");
rc = SQLPrepare(hstmt1,stmt,SQL_NTS);
rc = SQLBindParameter(hstmt1,... );
:
rc = SQLExecute(hstmt1);
if (rc==SQL_SUCCESS) {
  :
```


Trigger Programs

- Based on target file/table
- Invoked for each INSERT/UPDATE/DELETE against the table as defined by the triggers
- Advantage over stored procedures
  - Better performance (marginal)
- Caveat: All operations, regardless of origin, fire the trigger program

Database Tools

- Graphical
  - Centerfield Technology DB Essentials
    - http://www.centerfieldtechnology.com
  - iSeries Navigator
    - http://www-1.ibm.com/servers/eserver/iseries/access/
  - Visual Explain
    - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/index.htm?info/rzajq/visexpl.htm
- Text-based
  - DBMON (Database monitor)
  - Debug joblogs
  - SST (iSeries Communication Trace)
  - ODBC trace (SQL.LOG)

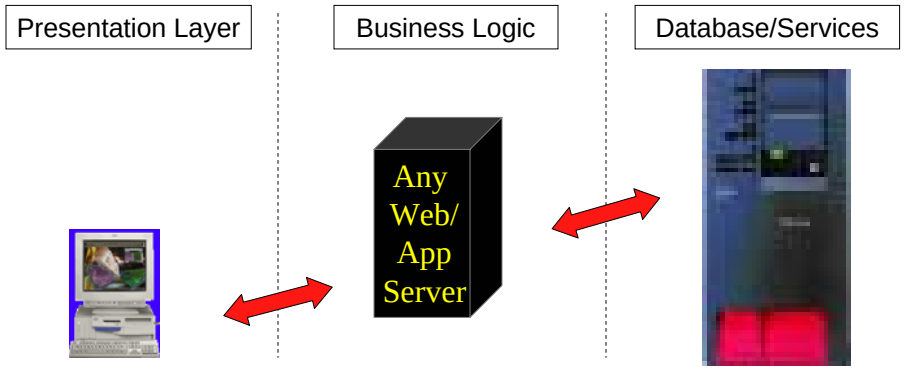
Database Design Summary

- Understand your application and the queries it runs
- Create indices where needed
- Use extended dynamic access
- Use stored procedures where possible

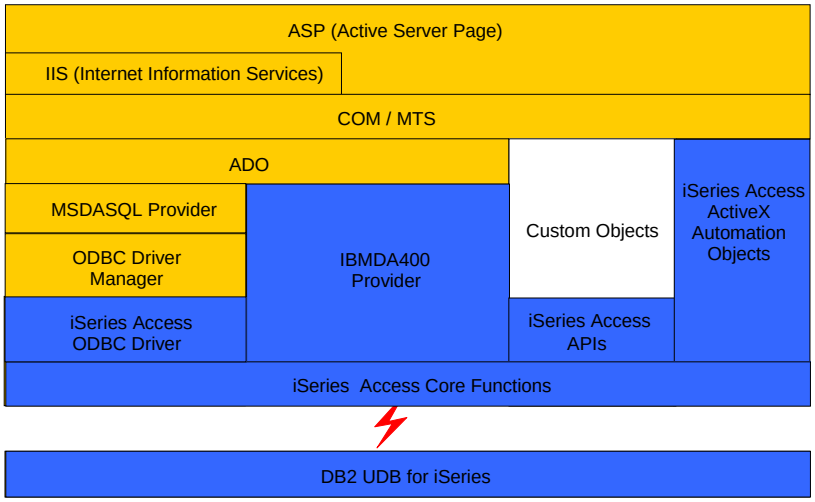
Examples

- 3-Tier application
- Stand-alone application
- Solutions for typical performance problems

Example scenarios (3-Tier application)



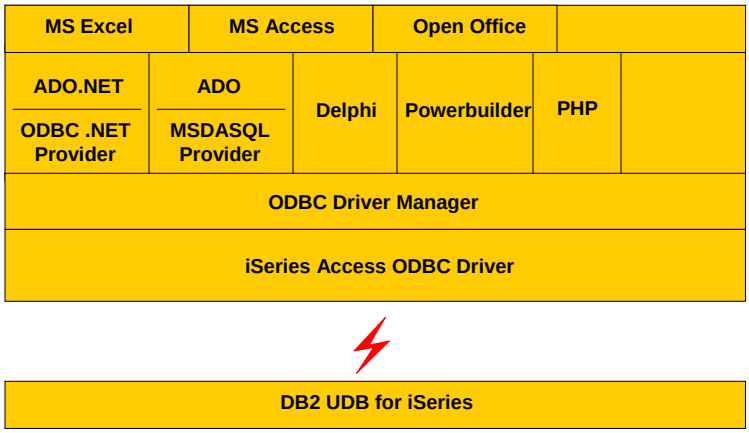
iSeries Access in Windows Environment



Helpful settings with 3-Tier applications

- Connection pooling
- Stored procedures
- Block fetches

Example scenarios (Stand-alone application)



Helpful settings with stand-alone applications

- Packages
- Block fetch of 1 row

Solutions for typical performance problems

- Fetching data
- Long-running SQL queries
- Network issues
- Inserting data
- Lots of connections

Summary

- Application Design:
  - Use parameter markers
  - Prepare once, execute many
- Network:
  - Use data compression
  - Use blocking
- Database Design:
  - Indexes
  - Use extended dynamic
  - Use stored procedures

iSeries Access for Windows – Sessions in Chicago

1. 26GH – MS Office with Client Access
2. 31GJ - Administration of iSeries Access for Windows: Advanced Tips
3. 31GH - MS Office and Client Access Integration Session 1: Setup and Overview
4. 32GH – MS Office and Client Access Integration Session 2: Word and Excel
5. 33GH – MS Office and Client Access Integration Session 3: Access-Web-Sending Data
6. 36CA - iSeries Access for Windows: What's New in V5R3
7. 41CB - iSeries Access Data Transfer: Tips and Techniques
8. 41LC - LAB: MS Office with CA/400
9. 42CB - iSeries Access for Windows: Security and Communications Tips
10. 44CA - iSeries Access for Windows in a .NET World
11. 45LA - OPEN LAB: iSeries Access for Windows with the Experts
12. 52CB - Everything you wanted to know about PC5250 emulation
13. 56CB - Performance Tune iSeries Access ODBC Driver

Session Title: Performance Tune iSeries Access ODBC

Session ID: 401918

Agenda Key: 56CB

Speaker: Brent Nelson

Appendices

- A – Reference Information
- B – Extended Dynamic Packages
- C – Compression
- D – Running CL Commands Through ODBC
- E – Exit Programs
- F – Identifying ODBC Job
- G – 3-Tier Application Picturesictures

A.1 - Additional Client-side Information

- ODBC
  - http://publib.boulder.ibm.com/infocenter/iseries/v5r3/ic2924/index.htm?info/rzaik/rzaikappodbc.htm
- OLE DB
  - See OLE DB Tech Ref installed with iSeries Access
- .NET
  - See .NET Tech Ref installed with V5R3 iSeries Access
- JDBC
  - http://publib.boulder.ibm.com/iseries/v5r3/ic2924/index.htm?info/rzahh/jdbc.htm

A.2 - Additional Information

- DB2 UDB for iSeries home page
  - http://ibm.com/servers/eserver/iseries/db2/
- Newsgroups
  - comp.sys.ibm.as400.misc
  - comp.databases.ibm-db2
- Education Resources - Classroom & Online
  - http://ibm.com/servers/eserver/iseries/service/igs/db2performance.html
- DB2 UDB for iSeries Publications
  - Online Manuals: http://ibm.com/servers/eserver/iseries/db2/books.htm
  - Indexing Strategies for DB2 UDB for iSeries: http://www.iseries.ibm.com/developer/bi/documents/strategy/strategy.pdf
  - DB2 UDB for AS/400 Redbooks (http://ibm.com/redbooks)
    - DB2 UDB for AS/400 Object Relational Support (SG24-5409)
    - DB2/400 Advanced Database Functions (SG24-4249-02)
- SQL/400 Developer's Guide by Paul Conte & Mike Cravitz
  - 29th Street Press, ISBN 1-882419-70-7
  - http://as400network.com/str/books/Uniquebook2.cfm?NextBook=183

A.3 - Performance Service Tips

- Before calling SupportLine with a query performance problem...
  - Run query in DEBUG mode and check JOBLOG
    - Index recommendations
    - Understand query implementation
  - Check resources and Work Management
    - QQRYDEGREE or CHGQRYA
    - Memory and MAX ACTIVE settings
    - What else is running?
    - Does QQQOPTIONS data area exist?
  - Check file stats
    - Size of objects, number of rows
    - Number of indexes
  - Understand your data
  - Save JOBLOGs and system settings

A.4 - Tech Tip: Improve Query Performance

- DB2 UDB for iSeries has a phenomenal query optimizer built into it.
  - without the DB2 Symmetric Multiprocessing (SMP) feature your SQL database tasks and index builds are running single-threaded?
  - the DEFAULT system tuning setup could be significantly hindering ODBC performance?
  - the database utility called DB2 OLAP can provide sub-second response times to complex queries?
- For more information about query optimization, check out these resources:
  - S6140 - DB2 UDB for iSeries SQL and Query Performance Tuning and Monitoring Workshop:
    - http://www-3.ibm.com/servlet/com.ibm.ls.lsow.servlets.CourseDescriptionServlet?coursecode=S6140
  - DB2 Symmetric Multiprocessing and DB2 OLAP Utilities:
    - http://isource.ibm.com/cgi-bin/goto?on=c4268db2/products

B.1 Package contents

- Extended dynamic
  - Use PRTSQLINF command on the iSeries to dump the package containing your statements
  - PRTSQLINF produces spoolfile showing syntax and optimization information
  - Use PRTSQLINF to ensure all SELECT statements have parameter markers

B.2 - Package contents sample

Extended Dynamic
Sample PRTSQLINF output:

```
5722SS1 V5R2M0 030905     Print SQL information         SQL package QGPL/ODBCXXXFBA
Object name.............QGPL/ODBCXXXFBA
Object type.............*SQLPKG
  CRTSQL***
     PGM(QGPL/ODBCXXXFBA)
     SRCFILE(         /         )
     SRCMBR(         )
     COMMIT(*NONE)
     OPTION(*SQL *PERIOD)
     TGTRLS(*PRV)
     ALWCPYDTA(*OPTIMIZE)
     CLOSQLCSR(*ENDPGM)
     STATEMENT TEXT CCSID(37)
STATEMENT NAME:  QZ84DC1FE6AC488000
select * from qiws.qcustcdt where lstnam=?
  SQL4021  Access plan last saved on 09/16/02 at 12:47:36.
  SQL4020  Estimated query run time is 1 seconds.
  SQL4027  Access plan was saved with DB2 UDB Symmetric Multiprocessing installed on the system.
  SQL4010  Table scan access for table 1.
```


B.3 - Unusable packages

- Package can become unusable if package attributes do not match application
  - Different CCSID, Date & time format attributes, decimal delimiter, default collection, etc
  - With ODBC packages, a default collection for unqualified names can be specified - if package already exists and the client application has a different default collection, then package cannot be used
  - If package unusable, new requests are executed as "pure" Dynamic SQL

B.4 - Package names

- First time an SQL statement is prepared, the package is created (if it doesn't exist yet)
- Can specify a name and location for package on the data source or let the system do that work
  - Default ODBC SQL package name is created by taken the first 7 characters of the application name and appending 3 letters that are encoding of the package configuration attributes
    - Default package name for Lotus Approach would be: APPROACFBA
    - New setup GUI allows setting of package name for a specific application
  - Default library determined by data source configuration

C.1 - Compression

- Can be activated at the connection level or statement level
- Connection level settings
  - COMPRESSION=1 in SQLDriverConnect connection string OR...
  - SQLSetConnectAttr(hdbc, 2106, 1) OR...
  - “Enable Data Compression” option on ODBC DSN setup GUI
- Statement level settings
  - SQLSetStmtAttr(hstmt, 2106, 1)

D.1 - Executing CL Commands

- Execute system CL commands via the QCMDEXC stored procedure
- CALL QSYS/QCMDEXC ('CHGQRYA QRYTIMLMT (0)', 0000000021.00000)
  - 21 is the character length of the CL command string

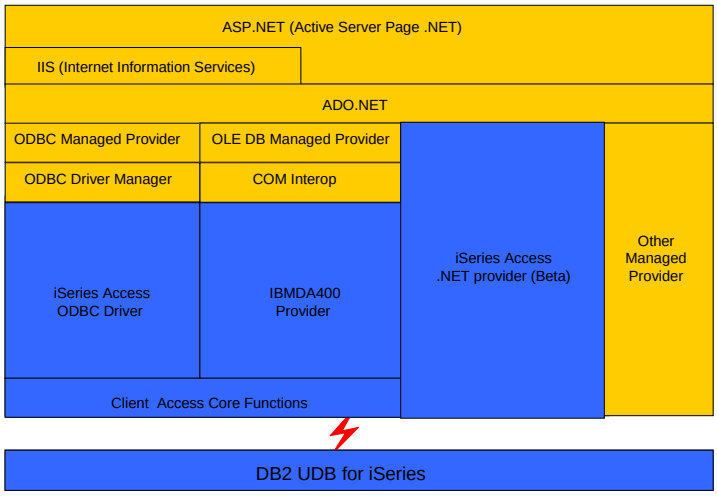
E.1 - Exit Programs For Job Initialization

- When a JDBC or ODBC job (QZDASOINIT) is initiated in subsystem (QSERVER), the registered exit program is called automatically
- Add program to exit point for ODBCINIT via WRKREGINF CL command at QIBM_QZDA_INIT
- Example: Create an exit program with CL that conditionally activates debug mode (STRDBG UPDPROD(*YES)) so that you can see the optimizer debug messages for an ODBC or JDBC request

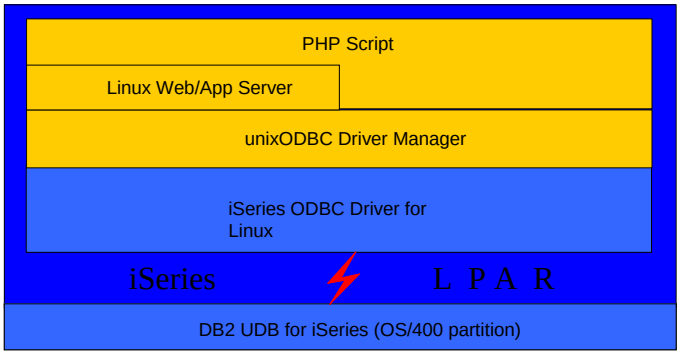
F.1 - Identifying your ODBC Server Job

- How do you determine which ODBC server job (QZDASOINIT) is the one that you want to analyze?
  - WRKOBJLCK OBJ(userid) OBJTYPE(*usrprf)
  - Returns the QZDASOINIT job servicing your ODBC request
- In V5R1, job information can be obtained by calling SQLGetConnectAttr API with option 2110.

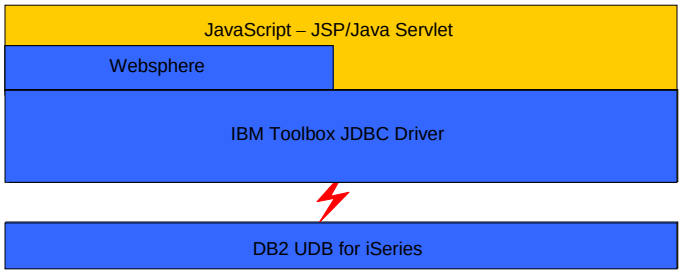
G.1 - iSeries Access in .NET Environment



G.2 - iSeries ODBC in Linux Environment



G.3 - Toolbox JDBC in Java Environment



Sample Code Disclaimer

This material contains IBM copyrighted sample programming source code for your consideration. This sample code has not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function. IBM provides no program services for this material. This material is provided "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THE ABOVE EXCLUSIONS MAY NOT APPLY TO YOU. IN NO EVENT WILL IBM BE LIABLE TO ANY PARTY FOR ANY DIRECT, INDIRECT, SPECIAL OR OTHER CONSEQUENTIAL DAMAGES FOR ANY USE OF THIS MATERIAL INCLUDING, WITHOUT LIMITATION, ANY LOST PROFITS, BUSINESS INTERRUPTION, LOSS OF PROGRAMS OR OTHER DATA ON YOUR INFORMATION HANDLING SYSTEM OR OTHERWISE, EVEN IF EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Trademarks and Disclaimers

References in this document to IBM products or services do not imply that IBM intends to make them available in every country.

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

AS/400	e-business on demand	OS/400
AS/400e	IBM	i5/OS
eServer	IBM (logo)	
eserver	iSeries	

Rational is a trademark of International Business Machines Corporation and Rational Software Corporation in the United States, other countries, or both.
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
Intel, Intel Inside (logos), MMX and Pentium are trademarks of Intel Corporation in the United States, other countries, or both.
UNIX is a registered trademark of The Open Group in the United States and other countries.
SET and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.
Other company, product or service names may be trademarks or service marks of others.

Information is provided "AS IS" without warranty of any kind.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer.

Information concerning non-IBM products was obtained from a supplier of these products, published announcement material, or other publicly available sources and does not constitute an endorsement of such products by IBM. Sources for non-IBM list prices and performance numbers are taken from publicly available information, including vendor announcements and vendor worldwide homepages. IBM has not tested these products and cannot confirm the accuracy of performance, capability, or any other claims related to non-IBM products. Questions on the capability of non-IBM products should be addressed to the supplier of those products.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. Contact your local IBM office or IBM authorized reseller for the full text of the specific Statement of Direction.

Some information addresses anticipated future capabilities. Such information is not intended as a definitive statement of a commitment to specific levels of performance, function or delivery schedules with respect to any future products. Such commitments are only made in IBM product announcements. The information is presented here to communicate IBM's current investment and development activities as a good faith effort to help with our customers' future planning.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.