



IBM System i™

Session: 401918  
Agenda Key: 42MK

## Performance Tune iSeries Access ODBC

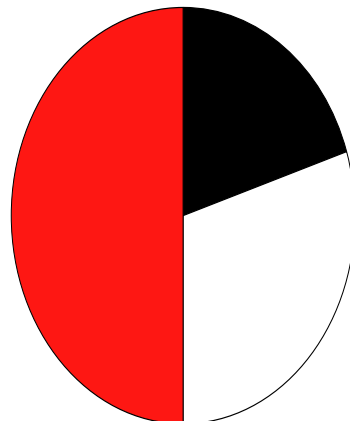
Brent Nelson - [bmnelson@us.ibm.com](mailto:bmnelson@us.ibm.com)  
iSeries Access Development

*i want stress-free IT.*  
*i want control.*  
*i want an **i**.*

© Copyright IBM Corporation, 2006. All Rights Reserved.  
This publication may refer to products that are not currently available in your country. IBM makes no commitment to make available any products referred to herein.

IBM System i

## Typical Performance Problems



■ Client

□ Network

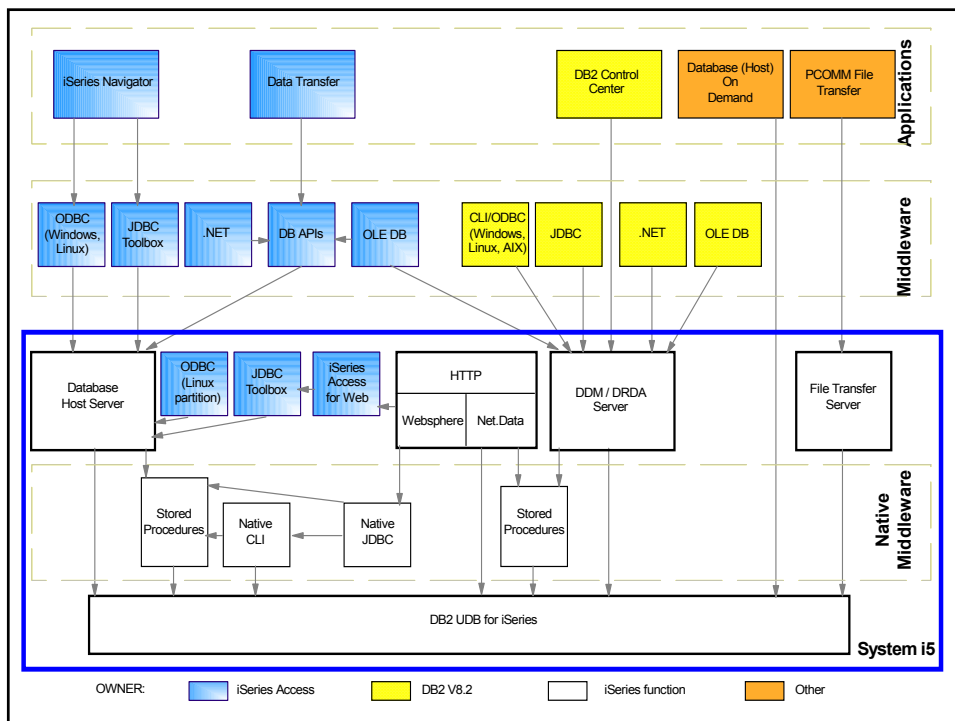
■ Server

*i want an **i**.*

© 2006 IBM Corporation

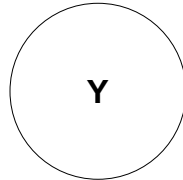
# Typical Performance Problems

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



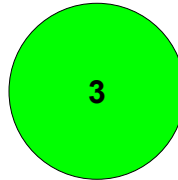
## Explanation of Scale

Programming  
Required?



Y = Yes  
N = No

Potential  
Benefit



1 = Low  
2 = Medium  
3 = High

## Agenda

- Process for Analyzing Performance Problems
- Performance Considerations
  - Application Design
  - Network
  - Database Design
- Examples
  - 3-Tier Application
  - Off-the-shelf Applications
- Appendices

## Process for Analyzing Performance Problems

1. Understand what the application is doing
2. Narrow down the problem
3. Understand options to fix the problem
4. Fix the problem

## Understand What the Application is Doing

- Lots of data retrieved?
- Lots of data inserted?
- Lots of connections?
- Complex queries?
- LOB fields?
- Problem related to scaling application?
- Did something change recently?
- ...

## Narrow Down the Problem

- Review database design
- Time parts of code

## Understand Options to Fix Problem

- Research options
- Understand the benefits to different alternatives

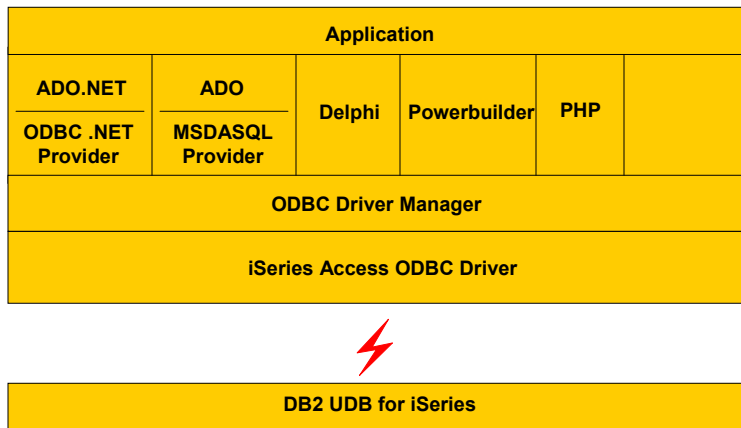
## Fix the Problem

- Make changes
- Compare performance before / after

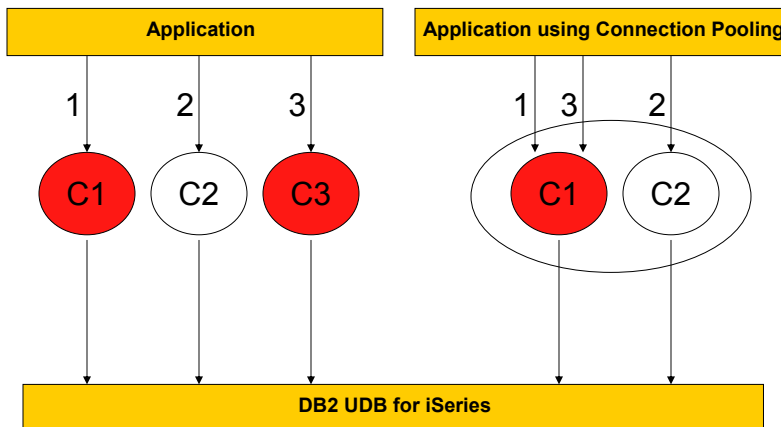
## Performance Considerations

- Application design
  - Choice of programming interface
  - Connection pooling
  - Inserting data
  - Fetching data
  - Isolation level and concurrency
- Network
- Database design

# Choosing a Programming Interface



# Connection Pooling



## Insert Data (Example Scenario)

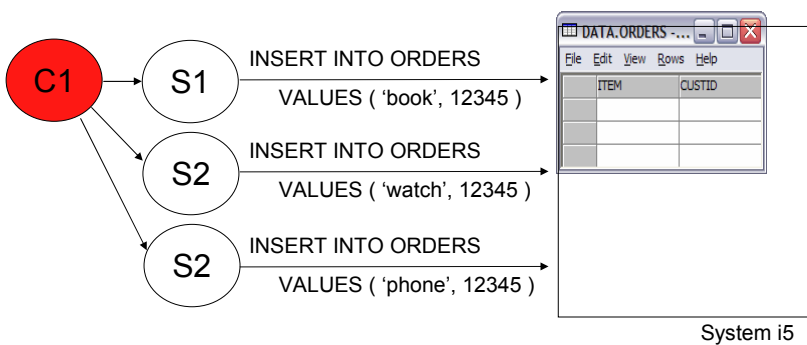
Customer has submitted an order for 3 items on our website



We need to record this in the ORDERS table on the System i5

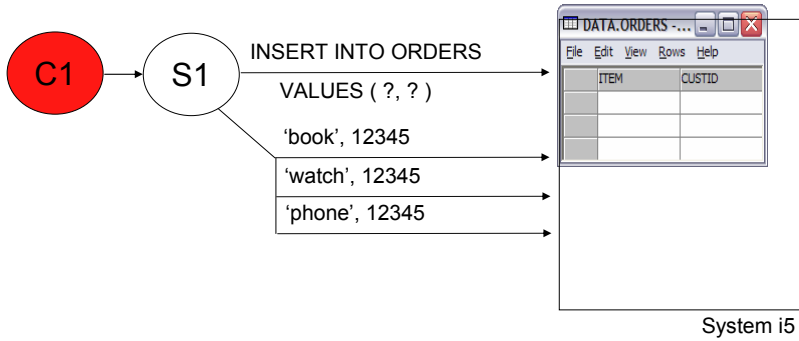
ITEM	CUSTID
book	12345
watch	12345
phone	12345

## Insert Data - Insert with Constants

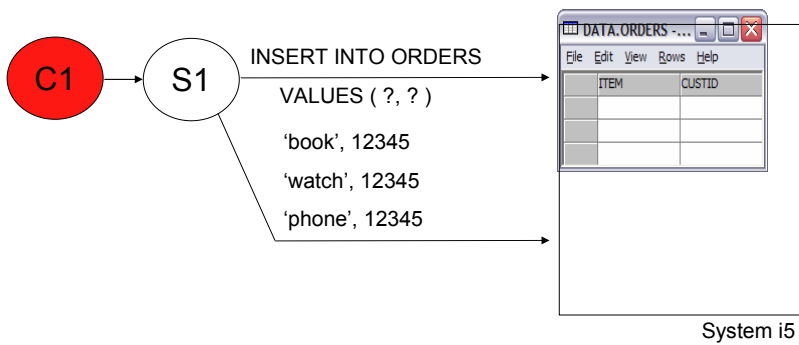




# Insert Data - Prepare Once, Execute Many



# Insert Data - Block Insert

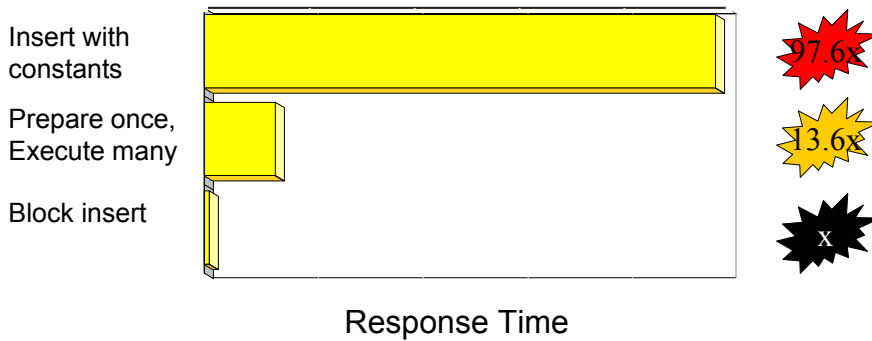


## Insert Data - Block Insert Notes

- 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 use "? rows" SQL clause

## Block Insert Performance

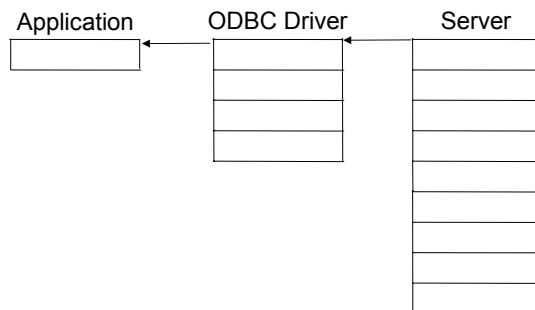
For 500 36-byte rows with three columns



## Deleting Data

- Consider:  
CALL QSYS.QCMDXEC('CLRPFM FILE(MYLIB/MYFILE)',0000000025.00000)
- Instead of:  
DELETE FROM MYLIB.MYFILE
- NOTE: This is not necessary on V5R3 (and later) systems.

## Fetching Data - Blocking

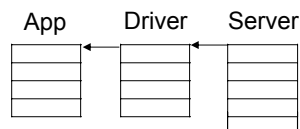
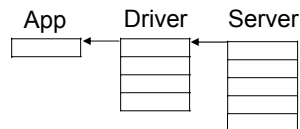
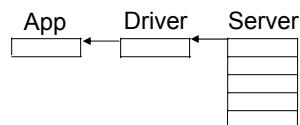


## Fetching Data – Settings that Affect Blocking

- Cursor Type
  - SQLSetStmtAttr API with SQL\_ATTR\_CURSOR\_TYPE option
  - Forward-only versus Scrollable
- Rowset size
  - SQLSetStmtAttr API with SQL\_ATTR\_ROW\_ARRAY\_SIZE option
- Block fetch of 1 row option (BLOCKFETCH keyword)
  - Block size option – (BLOCKSIZE keyword)
- Cursor Concurrency
  - SQLSetStmtAttr API with SQL\_ATTR\_CONCURRENCY option

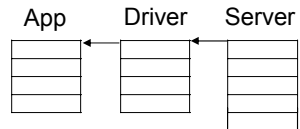
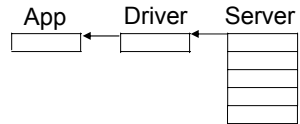
## Fetching Data - Forward-only Cursor Examples

- Table retrieving from has a 32K row size
- Example 1:
  - Application is fetching one row at a time
  - Block fetch of 1 row option 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 option with a Block Size of 128K
  - Results: ~4 rows are fetched at a time
- Example 3:
  - Application is fetching with rowset size of 4 rows
  - Results: 4 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 rowset size of 4 rows
  - Results: 4 Rows are fetched at a time



## Fetching Data – SQLBindCol vs SQLGetData



- Example:
  - SQLBindCol usage:
    - 3 SQLBindCol calls
    - up to 5 SQLFetch calls
  - SQLGetData usage:
    - 5 SQLFetch calls
    - 15 SQLGetData calls

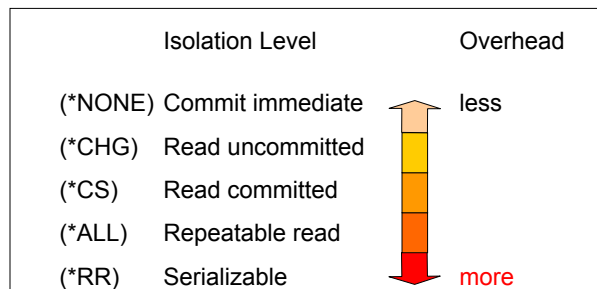
COL1	COL2	COL3

## Large Objects (LOBs)

- MAXFIELDLEN keyword
  - Default is 32 (KB)
  - Lower setting usually better
- Inserting data
  - Use SQLParamData / SQLPutData
- Fetching data
  - Use SQLGetData

## Isolation Level

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



## Application Design Summary

- Use connection pooling
- 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 than most applications

## Reduce Trips to Server

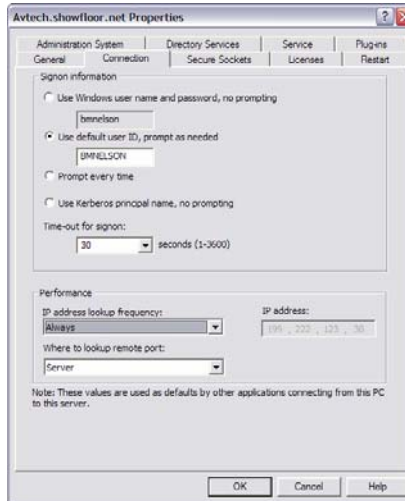


- Stored procedures
- Triggers
- Connection pooling
- Block inserts
- Block fetches
- Lazy close
- Pre-fetch



## Reduce Trips to Server

- iSeries Navigator connection properties
  - IP Address lookup
  - Port lookup



## Reduce Data Between Server

- Data compression
- LOB threshold
- Avoid "SELECT \*" SQL statements

## Data Compression

- “Enable Data Compression” DSN setting ON by default
- 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

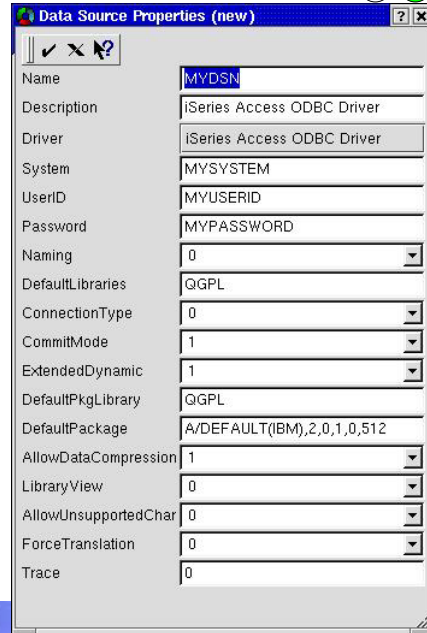
ODBC DSN keywords:

<http://publib.boulder.ibm.com/infocenter/series/v5r4/index.jsp?topic=/rzaik/connectkeywords.htm>



## Linux ODBC DSN GUI

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

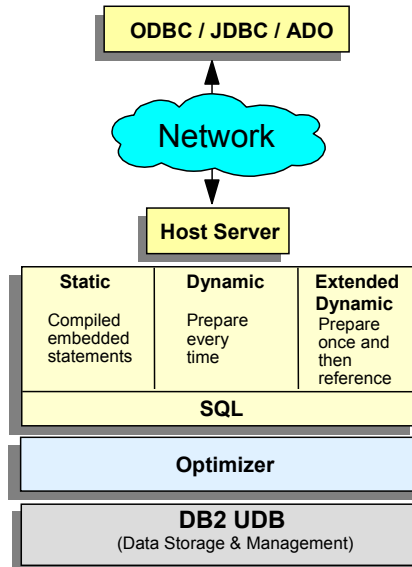


Property	Value
Name	MYDSN
Description	iSeries Access ODBC Driver
Driver	iSeries Access ODBC Driver
System	MYSYSTEM
UserID	MYUSERID
Password	MYPASSWORD
Naming	0
DefaultLibraries	QGPL
ConnectionType	0
CommitMode	1
ExtendedDynamic	1
DefaultPkgLibrary	QGPL
DefaultPackage	A/DEFAULT(IBM),2,0,1,0,512
AllowDataCompression	1
LibraryView	0
AllowUnsupportedChar	0
ForceTranslation	0
Trace	0

## Performance Considerations

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

## SQL Interfaces



## SQL Statement Tuning

(Y) 2

- 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-03.ibm.com/servers/enable/site/education/abstracts/indxng\\_abs.html](http://www-03.ibm.com/servers/enable/site/education/abstracts/indxng_abs.html)

## 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=?
```



```
SELECT * FROM MYTABLE  
WHERE COL1=?  
Access plan
```

Package

## Extended Dynamic Access (Packages)

- Caches SQL statements on server or local
- Allows reuse of statements (across sessions)
- Can be shared among many users



## 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

## QAQQINI file



- OPTIMIZATION\_GOAL
  - \*DEFAULT
    - Packages: \*ALLIO
    - No Packages: \*FIRSTIO
  - \*FIRSTIO
  - \*ALLIO
  
- QAQQINI query options:  
<http://publib.boulder.ibm.com/infocenter/iseres/v5r4/index.jsp?topic=/rzajq/qryopt.htm>

## 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 ALWCOPYDTA, etc

## Insert Data (Example Scenario)

Customer has submitted an order for 3 items on our website



We need to record this in the ORDERS table on the System i5

ITEM	CUSTID
book	12345
watch	12345
phone	12345

## Insert Data - Stored Procedure Example

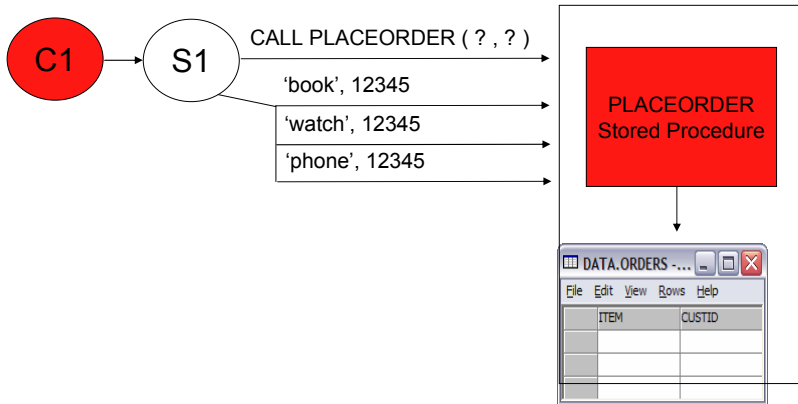
```
create procedure PLACEORDER( ITEM in varchar(20), CUSTID in int)
language sql
begin

declare sqlstmt char(2048);

set sqlstmt = 'insert into ORDERS values ( "' || ITEM || '", ' || CUSTID || ' )';
execute immediate sqlstmt;

end
```

## Insert Data - Stored Procedure



## Insert Data - Stored Procedure Example

```

create procedure PLACEORDER( ITEM in varchar(20), CUSTID in int)
language sql
begin

declare sqlstmt char(2048);

set sqlstmt = 'insert into ORDERS values ( "" || ITEM || "" , ' || CUSTID || ' )';
execute immediate sqlstmt;

set sqlstmt =
  'update ORDERLOG set TIME=CURRENT_TIMESTAMP
    where CUSTID = ' || CUSTID;
execute immediate sqlstmt;

end

```

## Trigger Programs

- Based on target file/table
- Can be invoked for each INSERT/UPDATE/DELETE against the table
- Slight performance advantage over stored procedures
- Be careful: All operations, regardless of origin, fire the trigger program

## 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
- Links:
  - <http://www-1.ibm.com/servers/eserver/series/access/>
  - <http://publib.boulder.ibm.com/infocenter/series/v5r4/index.jsp?topic=/rzajq/visexpl.htm>

## Other Database Tools

- Graphical
  - Centerfield Technology DB Essentials
    - <http://www.centerfieldtechnology.com>
- Text-based
  - Debug joblogs
  - SST (iSeries Communication Trace)
  - ODBC trace (SQL.LOG)

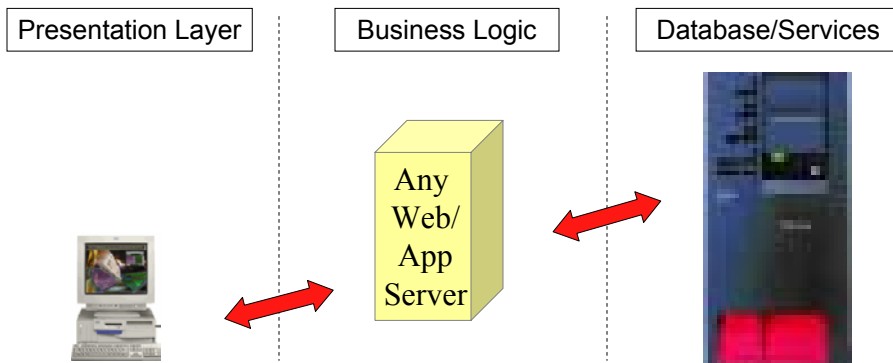
## Database Design Summary

- Create indices where needed
- Use stored procedures where possible

## Examples

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

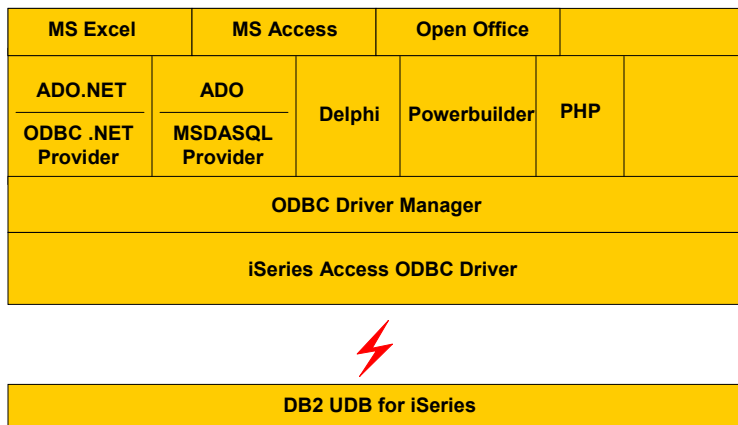
## Example Scenarios (3-Tier Application)



## Helpful Settings with 3-Tier Applications

- Goals
- Settings
  - Connection pooling
  - Stored procedures
  - Block fetches

## Example scenarios (Stand-alone application)





## Helpful settings with stand-alone applications

- Goals
- Settings
  - Block fetch of 1 row
  - Compression
  - LOB threshold
  - Packages

## Summary

- Process
  - Understand application
  - Narrow problem
  - Understand options
  - Fix problem
- Application Design:
  - Use parameter markers
  - Prepare once, execute many
- Network:
  - Adjust data compression as needed
  - Use blocking
- Database Design:
  - Indexes
  - Use stored procedures

**iSeries Access for Windows sessions:**

- 32MC - iSeries Access for Windows: What's New in V5R4
- 33MI - iSeries Access Data Transfer: Tips & Techniques
- 42MK - Performance Tune iSeries Access ODBC
- 43MC - PC5250 Emulation: Everything You Need To Know
- 46ME - iSeries Access in the .NET World

**iSeries Access for Linux session:**

- 53ML – Creating the iSeries Linux Desktop

Stop in EXPO for a demo...

*Functional enhancements can be submitted via the FITS system. The url is:*

<http://www.ibm.com/eserver/series/access/>

And click on link "[Request for Design Change](#)"

**Session Title:** Performance Tune iSeries Access ODBC

**Session ID:** 401918

**Agenda Key:** 42MK

**Speaker:** Brent Nelson

## Appendices

- A – Reference Information
- B – Insert data example code
- C – Compression
- D – Extended Dynamic Packages

## A.1 - Additional Client-side Information

- ODBC
  - <http://publib.boulder.ibm.com/infocenter/series/v5r4/index.jsp?topic=/rzaik/rzaikappodbc.htm>
- OLE DB
  - See OLE DB Tech Ref installed with iSeries Access
- .NET
  - See .NET Tech Ref installed with V5R4 iSeries Access
- JDBC
  - <http://publib.boulder.ibm.com/infocenter/series/v5r4/index.jsp?topic=/rzahh/page1.htm>

## A.2 - Additional Information

- DB2 UDB for iSeries home page
  - <http://ibm.com/servers/eserver/series/db2/>
- Newsgroups
  - comp.sys.ibm.as400.misc
  - comp.databases.ibm-db2
- Education Resources - Classroom & Online
  - <http://ibm.com/servers/eserver/series/service/igs/db2performance.html>
- DB2 UDB for iSeries Publications
  - Online Manuals: <http://www-03.ibm.com/servers/eserver/series/db2/books.html>
  - Indexing Strategies for DB2 UDB for iSeries: [http://www-03.ibm.com/servers/enable/site/education/abstracts/indxng\\_abs.html](http://www-03.ibm.com/servers/enable/site/education/abstracts/indxng_abs.html)
  - 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
    - QQRVDEGREE or CHGQRYA
    - Memory and MAX ACTIVE settings
    - What else is running?
    - Does QQOPTIONS 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 – iSeries/i5 Performance Analysis Workshop:
    - [http://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=course\\_description&courseCode=AS024](http://www-304.ibm.com/jct03001c/services/learning/ites.wss/us/en?pageType=course_description&courseCode=AS024)
  - DB2 Symmetric Multiprocessing and DB2 OLAP Utilities:
    - <http://www-128.ibm.com/developerworks/db2/library/techarticle/0301milligan/0301milligan.html>

## B.1 - Insert Data - Insert with Constants

```
strcpy(stmt, "insert into ORDERS values('book', 12345)");  
rc = SQLExecDirect(hStmt, stmt, SQL_NTS);
```

```
strcpy(stmt, "insert into ORDERS values('watch', 12345)");  
rc = SQLExecDirect(hStmt, stmt, SQL_NTS);
```

```
strcpy(stmt, "insert into ORDERS values('phone', 12345)");  
rc = SQLExecDirect(hStmt, stmt, SQL_NTS);
```

## B.2 - Insert Data - Prepare Once, Execute Many

```
strcpy(stmt, "insert into ORDERS values (?,?)");  
rc = SQLPrepare(hStmt, stmt, SQL_NTS);  
  
rc = SQLBindParameter(hStmt, 1, ...., szItem, ... );  
rc = SQLBindParameter(hStmt, 2, ...., &custID, ... );  
  
strcpy(szItem, "book");  
custID = 123  
rc = SQLExecute(hStmt);  
  
strcpy(szItem, "watch");  
rc = SQLExecute(hStmt);  
  
strcpy(szItem, "phone");  
rc = SQLExecute(hStmt);
```

## B.3 - Insert Data – Block Insert

```
strcpy(stmt, "insert into ORDERS values (?,?)");  
rc = SQLPrepare(hStmt, stmt, SQL_NTS);  
  
rc = SQLSetStmtAttr(hStmt, SQL_ATTR_PARAMSET_SIZE, (PTR)3, ... );  
  
rc = SQLBindParameter(hStmt, 1, ...., szItemArray[0], ... );  
rc = SQLBindParameter(hStmt, 2, ...., &custID, ... );  
  
strcpy(szItemArray[0], "book");  
strcpy(szItemArray[1], "watch");  
strcpy(szItemArray[2], "phone");  
custID = 12345;  
  
rc = SQLExecute(hStmt);
```

## B.4 - Insert Data - Stored Procedure

```
strcpy(stmt, "CALL PLACEORDER (?,?)");  
rc = SQLPrepare(hstmt1, stmt, SQL_NTS);  
  
rc = SQLBindParameter(hStmt, 1, ...., szItem, ... );  
rc = SQLBindParameter(hStmt, 2, ...., &custID, ... );  
  
strcpy(szItem, "book");  
custID = 123  
rc = SQLExecute(hStmt);  
  
strcpy(szItem, "watch");  
rc = SQLExecute(hStmt);  
  
strcpy(szItem, "phone");  
rc = SQLExecute(hStmt);
```

## 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 Package contents

- Extended dynamic
  - Use iSeries Navigator to view the contents of the package
  - Alternatively, you can use the PRTSQLINF command on the System i5 to dump the package containing your statements
  - PRTSQLINF produces spoolfile showing syntax and optimization information

## D.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 lstrnam=?
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.
```



## D.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

## D.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

## 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

© IBM Corporation 1994-2006. All rights reserved.

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

Trademarks of International Business Machines Corporation in the United States, other countries, or both can be found on the World Wide Web at <http://www.ibm.com/legal/copytrade.shtml>.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds 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.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

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

The 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.

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 may be engineering prototypes. Changes may be incorporated in production models.