



STRATEGIC BUSINESS SYSTEMS: 2009 WINNER

*Power Systems Innovation Award: Best Web Solution
from IBM and COMMON*



DB2 and PHP

Best practices on IBM i

Alan Seiden

PHP on IBM i consultant
Strategic Business Systems, Inc.

<http://alanseiden.com>

@alanseiden

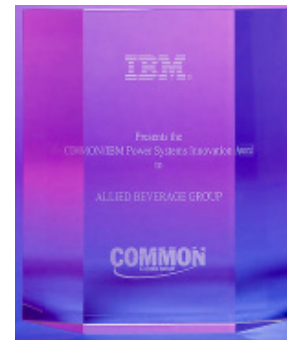


DB2 and PHP: Best Practices on
IBM i

About Alan

PHP on IBM i consultant

- Specialty: subsecond web performance on IBM i/iSeries
- Lead PHP developer for new toolkit from Zend/IBM
- Zend Framework contributor (DB2 for i enhancements)
- Developer, IBM/COMMON's "Best Web Solution," 2009



<http://alanseiden.com>

alan@alanseiden.com

• **twitter: @alanseiden**

• **201-327-9400**

Strategic Business Systems, Inc.



- **Based in Ramsey, Bergen County, New Jersey**
- **IBM Business Partner**
 - Power Systems hardware, software, development, consulting

Where to download these slides

From my site

<http://alanseiden.com/presentations>

On SlideShare

<http://slideshare.net/aseiden>

The latest version will be available on both sites

Why learn DB2 best practices?

- **As king of databases on IBM i, DB2 runs these:**
 - Most transaction processing systems
 - Stored procedures
 - Future: even more
 - New “XMLSERVICE” Toolkit available now, accessible with db2 stored procedures from PHP
- **DB2 knowledge will help you:**
 - Maximize speed
 - Reduce CPU usage
 - Maximize reliability
 - Avoid unexpected locking and other operational problems

DB2 drivers

DB2 drivers: past, future, present

- **Past: ODBC**

- PHP Extension based on Microsoft ODBC
- Was popular before `ibm_db2` became available

- **Future (maybe): IBM_PDO**

- PDO = PHP Data Objects (no IBM i support)

- **Present: `ibm_db2`**

- PHP extension provided by IBM
- Ships with Zend Server
- A db2-specific wrapper, with IBM i niceties, around standard SQL Call-Level Interface (CLI) API calls

ibm_db2 documentation

- **Manual page**
 - http://php.net/ibm_db2
- **Source code and additional documentation at the “PECL” PHP extension repository**
 - http://pecl.php.net/package/ibm_db2
 - Read the “C” source sometime—it’s educational
- **We will examine ibm_db2 in detail today**

Prerequisites

Zend Server for IBM i

- **Current PHP stack for IBM i**
 - Best of Zend Core and Platform in one licensed program
- **Two license levels**
 - Zend Server for IBM i Community Edition (CE)
 - Available at no charge per IBM partnership
 - Includes “Optimizer+” that speeds up code
 - One year silver (email) support
 - Zend Server for IBM i, a.k.a. Professional Edition (PE)
 - Subscription-based license
 - High value extra features, higher Support SLAs
- **Details of differences:**
 - <http://mikepavlak.blogspot.com/2010/08/i-want-to-do-php-on-ibm-i-so-what-do-i.html>

Installation of Zend Server

- **<http://www.zend.com/products/server/downloads>**
 - Click on the “IBM i” tab
 - Get the latest version of Zend Server for the newest ibm_db2
- **Zend Server prerequisites**
 - IBM i v5r4 or higher and:
 - http://files.zend.com/help/Zend-Server-IBMi/i5_installing_zend_server.htm
 - IBM’s FastCGI PTF (free)
- **Try Zend Studio’s IDE (no charge, courtesy of IBM)**
 - “Zend Studio for Eclipse, IBM i Edition”
 - <http://zend.com/en/products/studio/downloads>

Cheat sheet: upgrade Zend Core to Server

<http://alanseiden.com/2010/04/21/differences-between-zend-core-and-zend-server-on-ibm-i/>

	Zend Core	Zend Server
Installation folder	/usr/local/zend/core	/usr/local/zendsvr
PHP.INI	/usr/local/zend/core/etc	/usr/local/zendsvr/etc
Web server root(s)	/www/zendcore, /usr/local/zend/apache2	/www/zendsvr
Document root	/www/zendcore/htdocs	/www/zendsvr/htdocs
Zend Framework	/usr/local /Zend/ZendFramework	/usr/local/zendsvr/share /ZendFramework
PHP binaries folder * * where php and php-cli reside	/usr/local/zend/core/bin	/usr/local/zendsvr/bin
PHP Log files	/usr/local/zend/core/logs	/usr/local/zendsvr/ var/log
Web user profile (assign authority to it)	NOBODY	QTMHHTTP
Default HTTP Port	:89	:10088
Admin Interface URL	http://yourIBMi:89 /ZendCore/	http://yourIBMi:10088 /ZendServer/

Get current with PTFs

- **Latest DB2 for IBM i group PTF level for your release**
 - 5.4: WRKPTFGRP SF99504 (was 32 as of Oct. 27, 2011)
 - 6.1: WRKPTFGRP SF99601 (was 24 as of Feb. 9, 2012)
 - 7.1: WRKPTFGRP SF99701 (was 13 as of March 1, 2012)
- For the latest levels:
<http://www-947.ibm.com/systems/support/i/fixes/index.html>
and click **Group PTFs**

Get the latest release that you can

- **DB2 improvements in 6.1 and 7.1:**
 - Lifted old restrictions on combination of SQL + DDS
 - Can optimize queries no matter how you write them
 - “MERGE” keyword lets you insert/update in one statement
- **Lets you do more in SQL and DB2**
 - Less code to maintain
 - Better performance when you let database do the work
- **Learn what's new in SQL and DB2 on IBM i**
 - <http://ibm.com/developerworks/ibmi>

Best practices

Prepare queries

“Prepared” = safe and fast

- **Prepared queries help in several ways**
 - Eliminate errors due to un-escaped single quotes
 - Protect your data from SQL Injection attacks
 - Speed up repeated queries
- **They are also known as prepared statements**
- **Here’s an example of the mischief they prevent**

Apostrophes confuse query parsers

```
// mysite.com?name=whatever
$name = $_GET['name'];
$sql = "select custno from custfile
      where name = '$name' and status = 'ACTIVE' ";
```

- Do you see any potential problems?

- What if the name is “O’Shea” ? Error!

```
$sql = "select custno from custfile
      where name = 'O'Shea' and status = 'ACTIVE' ";
```

- Single quotes confuse query parser when they serve two purposes
 - Used as apostrophe in data
 - Delimiter of string literals in the SQL syntax

Malicious users can try “SQL Injection”

```
// mysite.com?name=whatever
$name = $_GET['name'];
$sql = "select custno from custfile
      where name = '$name' and status = 'ACTIVE' ";
```

- **What if the name is the weird-looking “x' OR 1=1--”**
(That is, a user typed: mysite.com?name=x' OR 1=1--)

```
$sql = "select custno from custfile
      where name = 'x' OR 1=1--' and status = 'ACTIVE' ";
```

- **Every record in the table will be selected**
 - OR 1=1 will always be true
 - -- turns subsequent ‘where’ criteria into a comment (ignored!)

Safeguard data with prepared queries

```
// mysite.com?name=whatever
$name = $_GET['name'];
$sql = "select custno from custfile
      where name = ? and status = 'ACTIVE' ";
```

- **Represent parameters with question marks (?) instead of literal values**
- **It's fine to retain hard-coded values in the query**
 - Such as 'ACTIVE' in the example above
- **Supply parameters in an array**
 - `$params = array("O'Shea");`
- **Full example on next slide**

db2_prepare() with db2_execute()

```
$name = $_GET['name'];
$conn = db2_connect('', '', '');
$sql = "select custno from custfile
      where name = ? and status = 'ACTIVE' ";
$params = array($name); // can be "O'Shea" for all we care
$stmt = db2_prepare($conn, $sql);
if ($stmt) { // prepared OK
    $result = db2_execute($stmt, $params);
    if ($result) { // ran query OK with parameters
        while ($row = db2_fetch_assoc($stmt)) {
            echo "$row['custno']\n";
        }
    }
}
}
```

Ordinary db2_exec() causes re-calc of plan

- **Ex. of non-prepared SQL repeated with different params**

```
$values = array('acme', 'shoprite', 'stop n shop');
foreach ($values as $value) {
    $sql = "select custno from custfile
           where name = '$value' and status = 'ACTIVE' ";
    // query gets re-optimized in each iteration
    $stmt = db2_exec($conn, $sql);
    if ($stmt) { // do something with $stmt }
}
```

- **The query plan will re-optimize on each db2_exec() because a new SQL string was supplied each time**
- **OK for one-off queries but not when repeated**

Prepared statement allows re-use of plan

- **Ex. of prepared SQL, execution with different params**

```
// prepare the query ONCE
$sql = "select custno from custfile
       where name = ? and status = 'ACTIVE' ";
$stmt = db2_prepare($conn, $sql);
// now execute with values only
$values = array('acme', 'shoprite', 'stop n shop');
foreach ($values as $value) {
    $result = db2_execute($stmt, array($value));
    if $result { // do something with $stmt }
}
}
```

- **The query plan is calculated ONCE and reused with each db2_execute(), saving time and CPU**

Prepared statements/queries are best

- **Replace `db2_exec()` with `db2_prepare()` and `db2_execute()` when you can**
- **Benefits**
 - Your queries will run as intended, with fewer surprises
 - Protection from a common form of hacking (SQL injection)
 - Performance will improve for repeated queries

db2_connect()

Connect with db2_connect()

- **db2_connect() creates a database connection**

- Accepts four parameters that you should master
 - Three main parameters, each a string
 - One array of additional options (not required)

```
resource db2_connect (string $database,  
                    string $username,  
                    string $password  
                    [, array $options ])
```

- **db2_pconnect() is similar**

- pconnect creates persistent connections (more on that later)

db2_connect() string parameters

- **\$database**

- Leave blank (' ') for default local database
- Use a db name from WRKRDBDIRE for a choice of databases
 - Database name can be for an LPAR, iASP, or another machine

- **\$username**

- Leave blank (' ') for default Apache user (QTMHHTTP)
- Use any valid user profile to associate queries with that user
 - Uppercase the user profile to be safe

- **\$password**

- Leave blank (' ') if \$database and \$username were blank
- Otherwise, provide password corresponding to \$username

db2_connect() basic examples

- **Empty params: default values will be used**

```
$conn = db2_connect( '', '', '' );
```

- Connects to default local database with web server user QTMHHTTP (it was NOBODY under Zend Core)

- **Override defaults with three basic parameters**

```
$conn = db2_connect( 'MYDB', 'MYUSER', 'MYPASS' );
```

- Connects to MYDB database with user MYUSER

\$options, parameter #4

- **An optional array to fine-tune the connection**
- **Below are choices that are most relevant for IBM i**
- **Details on next few slides**
 - **i5_lib**
 - Set a single default library
 - **i5_libl**
 - Set a library list (be sure to set i5_naming on)
 - **i5_naming**
 - Choose “system” or SQL naming
 - **i5_commit**
 - Commitment control options
 - **autocommit**
 - DB2_AUTOCOMMIT_ON (default) or _OFF

- **Specify one library as default**
 - `'i5_lib'=>'MYLIB'`
- **Any unqualified files/tables will be assumed to use this library**

i5_libl and other library list solutions

- **When using library lists, always set i5_naming to ON**
 - `'i5_naming' => DB2_I5_NAMING_ON`
 - More about `i5_naming` on next slide
- **i5_libl accepts a space-delimited library list**
 - `'i5_libl'=>'MYLIB YOURLIB ANYLIB'`
- **Two other ways to specify a library list**
 - Run a program via stored procedure that sets up your library list
 - Specify a user profile in `db2_connect()` whose job description has the library list you want

i5_naming

- **DB2_I5_NAMING_ON**

- A constant equal to 1
- Turns on “system naming” mode
- Table files are qualified using the slash (/) delimiter
- Unqualified files are resolved using the library list for the job

- **DB2_I5_NAMING_OFF**

- A constant equal to 0 (default)
- Enables “SQL naming” mode
- Files are qualified using the period (.) delimiter
- Unqualified files are resolved using either the default library (i5_lib) or the current user profile
 - Message to watch for: “*MYFILE* in *USERPROFILE* type *FILE not found. SQLCODE=-204” where MYFILE is your table and USERPROFILE is the user profile, such as QTMHHTTP or whatever you used in db2_connect()

db2_connect() example with \$options

```
$database = 'MYDB';
$user     = 'MYUSER';
$password = 'MYPASS';

$options = array('i5_naming' => DB2_I5_NAMING_ON,
                'i5_lib1'   => 'MYLIB1 MYLIB2'
               );
$conn = db2_connect($database, $user, $password, $options);

if ($conn) {
    echo "Connection succeeded.";
} else {
    echo "Connection failed.";
}

// MYTABLE will be found, if in library MYLIB1 or MYLIB2
$stmt=db2_exec($conn,"SELECT * FROM MYTABLE");
```

i5_commit

- **i5_commit**

- Options for commitment control
- Gives you fine-grained control (and ability to turn off altogether)
- Only relevant when commitment control is enabled system-wide
 - `ibm_db2.i5_allow_commit = 1` in INI file
- Choices:
 - `DB2_I5_TXN_NO_COMMIT` – turns off commitment control for this connection
 - `DB2_I5_TXN_READ_UNCOMMITTED`
 - `DB2_I5_TXN_READ_COMMITTED`
 - `DB2_I5_TXN_REPEATABLE_READ`
 - `DB2_I5_TXN_SERIALIZABLE`

autocommit

- **DB2_AUTOCOMMIT_ON**

- A constant equal to 1 (default)
- Turns autocommit on
 - End of script causes commit
- Only relevant when commitment control is used
- Convenient: insert/update/delete will work without `db2_commit()`

- **DB2_AUTOCOMMIT_OFF**

- A constant equal to 0
- Turns autocommit off
- Only relevant when commitment control is used
- Provides flexibility to ensure data integrity in multi-step transactions by using `db2_commit()/db2_rollback()` around groups of insert/update/delete queries

Use commitment control for data integrity

- **Example: “all or none” for two INSERTS**

- with `ibm_db2.i5_allow_commit = 1`
- and `autocommit = DB2_AUTOCOMMIT_OFF`

```
$conn = db2_pconnect('', '', '', array('autocommit'=>DB2_AUTOCOMMIT_OFF));
$stmt=db2_prepare($conn,"INSERT INTO MYTABLE (IDNUM, NAME) VALUES(?, ?)");

$result1 = db2_execute($stmt, array(1, 'jane')); // should insert OK
$result2 = db2_execute($stmt, array('x', 'bob')); // not numeric!

// check if both INSERTs succeeded
if ($result1 && $result2) {
    // Success. Commit both inserts
    db2_commit($conn);
} else {
    // *** Error with one of the inserts; roll them both back ***
    db2_rollback($conn);
}
// Neither record will be in the table. We rolled back.
```

Commitment control tips

- **Set `ibm_db2.i5_allow_commit = 1`**
- **Choose autocommit on or off**
- **Turn on journaling for schemas/collections/libraries**
 - Automatic for schemas/collections created by SQL
 - Extra step for libraries created via CRTLIB
 - Start Journal Library (STRJRNLIB, v6.1+) makes a library a journaled object. Any objects eligible to be journaled that are added to the library can be automatically journaled
 - <http://www.redbooks.ibm.com/abstracts/tips0662.html>
 - <http://kb.zend.com/index.php?View=entry&EntryID=235>

Persistent = fast

Use db2_pconnect() to connect persistently

- `resource db2_pconnect (string $database , string $username , string $password [, array $options])`
- **Persistent is much faster than non-persistent**
 - db2_pconnect can reuse connections, reducing the time needed to connect (after the first time) to almost zero
- **How db2_pconnect() reuses connections**
 - Connections defined by *database*, *username*, and *password*
 - Tries to reuse an existing connection matching these 3 params
 - db2_close() on a persistent connection does nothing
 - db2_pclose() forces the conn to close

Rules for using persistence

- **Because connections are shared when defined with an identical database, user, and password, please:**
 - Avoid unpredictable results by also specifying the same \$options for these connections
 - Promote sharing of jobs by minimizing the number of user profiles that you connect with
 - Each user profile creates a new set of database jobs
 - Each set of jobs consumes system resources
- **If using commitment control:**
 - Set DB2_AUTOCOMMIT_ON or diligently use db2_commit()
 - Reason: When a script ends, the driver will automatically roll back uncommitted insert/update/deletes to avoid “unwanted” transaction sharing between shared persistent connections

More about db2_connect and db2_pconnect

- **Manual pages**

- <http://www.php.net/manual/en/function.db2-connect.php>
- <http://www.php.net/manual/en/function.db2-pconnect.php>
- <http://www.php.net/manual/en/features.persistent-connections.php>

Global settings

A choice of config files for ibm_db2

- **php.ini**

- Located in `/usr/local/zendsvr/etc/`
- A large file containing hundreds of settings
- Add or modify settings under the section `[ibm_db2]`

- **ibm_db2.ini**

- Located in `/usr/local/zendsvr/etc/conf.d/`
- A small file containing only `ibm_db2` settings
- No need for a `[ibm_db2]` section
- Initial contents:
`extension=ibm_db2.so`

Useful runtime configurations

Setting	Default value	What it does
<code>ibm_db2.i5_all_pconnect</code>	"0"	Force persistent connections
<code>ibm_db2.i5_ignore_userid</code>	"0"	Ignore user/pwds specified in <code>db2_connect()</code>
<code>ibm_db2.i5_allow_commit</code>	"0"	turn commitment control on/off for schemas (and journal-able libraries)

- **Full definitions online**

- <http://www.php.net/manual/en/ibm-db2.configuration.php>

- **Details coming up**

ibm_db2.i5_all_pconnect

When `ibm_db2.i5_all_pconnect = 1`:

- **All `db2_connect()` calls are run as `db2_pconnect()`**
 - All db2 connections become persistent
 - No source code changes are needed
- **Why was this setting invented?**
 - On IBM i, `db2_pconnect()` performs dramatically better with lower machine stress than `db2_connect()`

ibm_db2.i5_ignore_userid

When `ibm_db2.i5_ignore_userid = 1`:

- **All `db2_(p)connect()` calls are overridden to ignore db, user and password**
 - Becomes `db2_(p)connect("", "", "" [, $options])`
 - The default user profile (QTMHHTTP) will be used
 - No source code changes are needed
- **Why was this setting invented?**
 - To prevent the launching of db2 server jobs, keeping all activity inline within Apache (more on this later)

ibm_db2.i5_allow_commit

When `ibm_db2.i5_allow_commit = 1`:

- **Enables commitment control**
- **See earlier slides for details**

Can't override global settings with ini_set()

- **Many PHP INI settings can be overridden in scripts with the PHP function ini_set()**
 - Example:

```
ini_set('display_errors', 1);
```
- **ini_set() cannot override most ibm_db2 settings**
 - <http://php.net/manual/en/ibm-db2.configuration.php>
 - In the “changeable” column in the documentation, ibm_db2 settings are labeled as “PHP_INI_SYSTEM”
- **Instead, use \$options in db2_connect() to override where possible (e.g. 'i5_commit' option)**

Inline Mode vs. Server Mode

DB2 logic can run in two environments

1. Inline, in your PHP Apache job

For “Plain vanilla” connections only

No specific user profile or database

Enforced with the setting: `ibm_db2.i5_ignore_userid=1`

2. “Server mode,” in separate shared jobs

More flexible connections

How to make it happen:

Specify a user profile when you connect or

specify persistent connection when you connect

Inline Mode

DB2 queries run directly in your PHP job, “inline”

Advantage: no extra jobs need to be spawned

Disadvantages:

- Can't choose user profile
 - Jobs run under QTMHHTTP user profile
- No persistence or sharing with other PHP jobs
- Connect to only one RDB (database) per PHP job

```
ZENDSVR      QTMHHTTP    BATCHI      ACTIVE      PGM-php-cgi.bi
ZENDSVR      QTMHHTTP    BATCHI      ACTIVE      PGM-php-cgi.bi
ZENDSVR      QTMHHTTP    BATCHI      ACTIVE      PGM-php-cgi.bi
ZENDSVR      QTMHHTTP    BATCHI      ACTIVE      PGM-php-cgi.bi
ZENDSVR      QTMHHTTP    BATCHI      ACTIVE      PGM-php-cgi.bi
```

Server Mode

DB2 queries run in separate prestart jobs

- **Advantages**

- Can choose user profile
- Can use persistent connections (fast!)
- Connect to multiple RDBs (databases) concurrently from PHP

- **Disadvantages**

- Potential delays to spawn jobs, if non-persistent
- Overhead if too many jobs are spawned
- Creates *SHRRD pseudo-locks that may require attention (more on this later)

```
Subsystem . . . . . : QSYSWRK
Opt  Job      User      Type      Status
---  ---      ---      ---      ---
  1  QSQSRVR  QUSER    PJ        ACTIVE
  2  QSQSRVR  QUSER    PJ        ACTIVE
```

Server Mode configuration

- **Prestart jobs named QSQSRVR run in QSYSWRK**
- **Pool of jobs is configurable**

```
CHGPJE SBSDB(QSYS/QSYSWRK) PGM(QSYS/QSQSRVR)  
STRJOBS(*YES) INLJOBS(XX) THRESHOLD(XX)  
ADLJOBS(XX) MAXUSE(XX OR *NOMAX)
```

- **More information about server mode**
 - <http://www.redbooks.ibm.com/abstracts/tips0658.html>
 - <http://www.mcpressonline.com/tips-techniques/database/techtip-grab-control-of-the-db2-qsqsrvr-jobs.html>
 - <http://www.mcpressonline.com/database/db2/finding-sql-server-mode-connecting-jobs.html>

Freeing resources

\$stmt = '' releases resources

- **End-of-script normally releases resources/memory**
- **For longer scripts, consider releasing these sooner**
 - Set statement to '' (empty string)
 - Releases memory, resources; closes cursor
 - Old way: db2_free_stmt(), now deprecated
 - Take care that you don't need the \$stmt anymore

```
$stmt=db2_exec($conn,"SELECT * FROM MYTABLE");  
while($row=db2_fetch_array($stmt)) {  
    echo "\n<br>";  
    var_dump($row);  
}  
// free statement resources  
$stmt = '';
```

db2_free_result() clears “pseudo locks”

- **In persistent mode, SELECT statements can cause *SHRRD (shared read) pseudo locks**
 - See them in QSQRVR jobs via the WRKOBJLCK command
 - Pseudo locks help retain performance-enhancing cursors
 - Normally, CLRPFM and other exclusive ops will clear the locks

- **If exclusive operations may occur while your script is active, use db2_free_result() to release cursors**

```
$stmt=db2_exec($conn,"SELECT * FROM MYTABLE");  
while($row=db2_fetch_array($stmt)) {  
    echo "\n<br>";  
    var_dump($row);  
}  
db2_free_result ($stmt); // allow exclusive ops
```

Windows/Linux dev clients

“How do I develop on non-i but deploy on i?”

- **People often ask how they can use their PC to develop and test**
- **Goal: To develop IBM i-based PHP code on a non-i machine (Linux, Windows) but connect to DB2 on IBM i and have it run identically, for eventual production use on i**
- **Requires a separate product from IBM**

IBM DB2 Connect

- **Purchase IBM's "DB2 Connect, Personal Edition"**
 - <http://www-01.ibm.com/software/data/db2/db2connect>
 - Not to be confused with `db2_connect()`
- **Your non-i computer will host a local "dummy" DB2 database that actually accesses DB2 on your IBM i**
 - Use the same `ibm_db2` functions that you normally do
 - Configuration tips:
<http://174.79.32.155/wiki/index.php/Tier2/DB2Connect>

Logical files

Tips for using logical files (LFs) in SQL

- **SQL and DB2 can use the indexes of DDS-based logical files**
- **Remember your “ORDER BY” and “WHERE” clauses**
 - Indexes from LFs will be used automatically as appropriate
 - Do not rely on the query optimizer to use any particular logical file, even if you specified one
- **Avoid confusion by specifying only physical files**

More tips for logical files

- **Avoid using select/omit in LFs if your system is not at v7.1 yet**
- **Before 7.1, select/omit criteria in LFs caused the query optimizer to revert to Classic Query Engine (CQE) instead of the SQL Query Engine (SQE)**
- **SQE is better because it utilizes improvements of recent years**
- **Solution: use a WHERE clause when possible instead of select/omit (or upgrade to 7.1)**

Choose a CCSID

CCSID is magical (when it works)

- **Coded Character Set Identifier (CCSID) helps convert data in a transparent manner**
 - EBCDIC to ASCII with many human languages
- **Lets DB2 operate in multiple language environments**
- **CCSIDs are listed here**
 - http://www-01.ibm.com/software/globalization/ccsid/ccsid_registered.html
- **I use CCSID 37, good for USA, Canada, Netherlands, Portugal, Brazil, Australia, New Zealand**

Default CCSID of 65535 can cause trouble

- **When a table/file has CCSID 65535, you get gibberish**
`SELECT CUST from CUSTFILE WHERE ID = 1`
- **Result:**
 - \$#%#%#(*#\$
- **A clue in a Zend Server startup message**
 - “The Zend Server Apache job CCSID is set to 65535. This setting might have unpredictable results when accessing data base data.”
- **QCCSID value is 65535 by default**
 - 65535 means to treat data as hex or “binary,” not to be converted

Solutions (old and new)

- **Old solution*: SQL Casting**

- CAST data to the proper EBCDIC CCSID, such as 37
- Requires that you provide the data type such as char(6)
- **SELECT CAST(CUST as char(6) CCSID 37) FROM CUSTFILE WHERE ID = 1**

* Casting still required when CCSID explicitly set at field level (such as in older JDE systems)

- **New, easy solution using Apache directives**

- Add to Apache ZENDSVR configuration file
 - /www/zendsvr/conf/httpd.conf

DefaultFsCCSID 37

CGIJobCCSID 37

To sum up

Summary of recommendations (slide 1 of 3)

- **Get current software**
 - Zend Server 5.6.0
 - Current DB2 group PTF
- **Use prepared queries/statements for security and performance**
- **Use persistent connections for best connection performance**

```
$conn = db2_pconnect('MYDB', 'MYUSER', 'MYPASS',  
                    array(...));
```

- Limit the number of user profiles that you connect with
- Be consistent about the options you use

Summary of recommendations (slide 2 of 3)

- **Understand how and when to use commitment control**
 - Defaults of commit=0 & autocommit=ON are fine to start
 - As you get experience, try commitment control
- **Clear resources with db2_free_stmt() and db2_free_result() as appropriate**
- **Avoid specifying logical files in SQL**
 - The query optimizer may ignore your wishes anyway
 - Enforce your wishes with ORDER BY and WHERE clauses

Summary of recommendations (slide 3 of 3)

- **To connect from Windows or Linux as if natively on the i, get DB2 Connect from IBM**
- **Add CCSID to your Apache config**
 - `DefaultFsCCSID 37`
 - `CGIJobCCSID 37`

DB2 and PHP Resources

- **IBM**

- IBM_DB2 manual and open source repository
 - http://php.net/ibm_db2, http://pecl.php.net/package/ibm_db2
- DeveloperWorks wiki
 - <http://ibm.com/developerworks/ibmi>
- Performance tips from IBM's Tony Cairns
 - <http://www.youngiprofessionals.com/wiki/FastCGI>

- **Zend and Alan**

- Zend Server for IBM i
 - <http://www.zend.com/en/products/server/zend-server-ibm-i>
- Forums for PHP on IBM i
 - <http://forums.zend.com/viewforum.php?f=67>
- Zend_Db2 component (Zend Framework)
 - <http://www.zend.com/en/resources/webinars/i5-os>
 - Scroll to Alan Seiden's webinar: "From Zero to ZF"

Keep up to date

- **These slides are online**
- **Send me feedback so I can continue to hone this information**
- **Try the Zend forums, too, so everyone can benefit from the answers:**
<http://forums.zend.com/viewforum.php?f=67>

Questions

Alan's upcoming appearances

MITEC, OCEAN....

Complete list here:

<http://alanseiden.com/presentations/upcoming>

In New York City? I host the NYC Zend Framework Meetup

<http://www.meetup.com/ZendFramework-NYCmetro/>

Contact | Get tips

Alan Seiden

201-327-9400

aseiden@sbsusa.com

<http://alanseiden.com>

Free tips newsletter:

<http://alanseiden.com/tips>

Strategic Business Systems, Inc,

17 S. Franklin Tpk

Ramsey, NJ, 07446

Twitter: @alanseiden

