SQL Anywhere PHP 関数と MySQL PHP 関数の違い

2009/02/26

このホワイトペーパーは、SQL Anywhere 11を対象に書かれました。
しかし、この内容は過去及び将来のリリースでも適用できる場合があります。

目次

はじめに

このホワイトペーパーでは、PHP 開発者が MySQL から SQL Anywhere に移行する際に発見すると思われるいくつかの相違点について紹介します。

SQL Anywhere 関数と MySQL 関数

このセクションで説明される関数は類似していますが、わずかに異なります。各関数の目的を説明し、その相違点を挙げていきます。

SQL Anywhere の場合、すべての PHP 関数が sasql クラスに存在しますが、同等の MySQL 関数は、mysql クラスと mysqli (改良版拡張モジュール) クラスの両方に存在します。

SASQL_DATA_SEEK() / MYSQL_DATA_SEEK()

これらの関数は、結果リソース内で内部カーソルの位置を前進させます。カーソルの新しい位置は、指定された整数に関連付けられたローを指します。たとえば、0 を指定するとカーソルが結果セットの最初のローに移動し、5 を指定すると 6 番目のローに移動します。負の数は、結果セットの最後を基準とするローを表します。たとえば、-1 を指定するとカーソルが結果セットの最後のローに移動し、-2 を指定すると最後から 2 番目のローに移動します。

相違点：

mysql_data_seek() では、負のロー番号を検索する機能がサポートされません。したがって、結果セットの最後を基準にしてローを参照することができません。

SASQL_FETCH_OBJECT() / MYSQL_FETCH_OBJECT()

これらの関数は、結果セットからローをオブジェクトとして返します。オブジェクトは、各プロパティ名が結果セットのカラム名のいずれかと一致する結果セットのフェッチされたローです。結果セットにローがそれ以上ない場合、FALSE を返します。

相違点：

mysql_fetch_object() には、指定したタイプのオブジェクトをインスタンス化する機能がありますが、sasql_fetch_object() は、stdClass オブジェクトを返すだけです。

SASQL_ESCAPE_STRING() / MYSQL_REAL_ESCAPE_STRING()

これらの関数は、文字列内の特殊文字をエスケープし、エスケープされた文字列を返します。

相違点：

mysql_real_escape_string():

以下の文字の先頭に円記号を付けます。¥x00,¥n,¥r,¥,',"および¥x1a

sasql_escape_string():

以下の特殊文字をエスケープします。:¥r,¥n,',",;,¥,および NULL 文字.

MySQL と SQL Anywhere は、特殊文字のエスケープ方法が異なります。特に、一重引用符 (') と 二重引用符 (") の処理方法が異なります。

MySQL は、すべての特殊文字の先頭に円記号を付けます。しかし、SQL Anywhere は、一重引用符をもう 1 つ追加することで一重引用符をエスケープし、文字を 16 進表現と置き換えることで二重引用符をエスケープします。

文字列リテラルでエスケープ・シーケンスを使用する方法の例を以下に示します。

- 一重引用符は、文字列リテラルの先頭と末尾の印として使用されます。そのため、文字列内の一重引用符は、'John''s database' のように一重引用符をもう 1 つ追加することによってエスケープする必要があります。
- 16 進のエスケープ・シーケンスは、あらゆる文字またはバイナリ値に使用できます。16 進のエスケープ・シーケンスは、円記号とその後に x と 2 桁の 16 進数がある文字列です。16 進数値は、CHAR と NCHAR の両方の文字列リテラルで CHAR 文字セットの文字として解釈されます。たとえば、コード・ページ 1252 の例 '123¥x80' は、数字 1、2、3 に続けてユーロ通貨記号を表現しています。
- 改行文字を表現するには、'First line:¥nSecond line:' のように円記号と n (¥n) を使用します。
- 円記号はエスケープ・シーケンスの開始を示すときに使用するため、文字列内の円記号は、'c:¥¥temp' のように円記号をもう 1 つ追加することによってエスケープする必要があります。

SASQL_QUERY() / MYSQL_QUERY()

これらの関数は、指定された接続で SQL 文を準備して実行します。

相違点：

SQL Anywhere インタフェースは、MySQL 改良版拡張モジュール・インタフェースに従います。そのため、sasql_query() と mysqli_query() は同じです。しかし、sasql_query() と mysql_query() は同じ引数をとりますが、順序が逆です。

SASQL_FETCH_FIELD() / MYSQL_FETCH_FIELD()

これらの関数は、特定のカラム (フィールド) に関する情報を含むオブジェクトを返します。

相違点：

SQL Anywhere と MySQL によって返されるプロパティが異なります。それらのプロパティについて、以下の表にまとめます。

SQL Anywhere のプロパティ

プロパティ名	説明
id	フィールド番号を格納します。
name	フィールド名を格納します。
numeric	フィールドが数値かどうかを示します。
length	フィールドのネイティブ・ストレージ・サイズを返します。
type	フィールド・タイプを返します。
native_type	フィールドのネイティブ・タイプを返します。DT_FIXCHAR、DT_DECIMAL、または DT_DATE などの値です。
precision	フィールドの数値精度を返します。このプロパティは、native_type が DT_DECIMAL であるフィールドに対してのみ設定します。
scale	フィールドの数値スケールを返します。このプロパティは、native_type が DT_DECIMAL であるフィールドに対してのみ設定します。

MySQL のプロパティ

プロパティ名	説明
name	カラム名

table	カラムが属するテーブル名
def	カラムのデフォルト値
max_length	カラムの最大長
not_null	カラムに NULL を設定できない場合は 1
primary_key	カラムがプライマリ・キーの場合は 1
unique_key	カラムがユニーク・キーの場合は 1
multiple_key	カラムがユニーク・キーでない場合は 1
numeric	カラムが数値の場合は 1
blob	カラムが BLOB の場合は 1
type	カラム型
unsigned	カラムが符号なしの場合は 1
zerofill	カラムが 0 で埋められる場合は 1

SASQL_SET_OPTION() / MYSQLI_OPTIONS()

これらの関数は、指定された接続の指定されたオプションの値を設定します。

相違点：

SQL Anywhere と MySQL で使用できるオプションが異なります。使用できるオプションについて、以下の表にまとめます。

SQL Anywhere のオプション

オプション名	説明	デフォルト
auto_commit	このオプションを ON に設定すると、各文の実行後にデータベース・サーバがコミットします。	ON
row_counts	このオプションを FALSE に設定すると、sasql_num_rows 関数が、影響を受けるロー数の推定値を返します。正確なロー数を取得したい場合、このオプションを TRUE に設定します。	FALSE
verbose_errors	このオプションを TRUE に設定すると、PHP ドライバが冗長エラーを返します。このオプションを FALSE に設定した場合、詳細なエラー情報を取得するには、sasql_error または sasql_errorcode 関数を呼び出す必要があります。	TRUE

MySQL のオプション

オプション名	説明
MYSQLI_OPT_CONNECT_TIMEOUT	秒単位の接続タイムアウト
MYSQLI_OPT_LOCAL_INFILE	LOAD LOCAL INFILE の使用を有効化／無効化します。
MYSQLI_INIT_COMMAND	MySQL サーバへの接続後に実行するコマンド
MYSQLI_READ_DEFAULT_FILE	my.cnf の代わりに指定されたオプション・ファイルからオプションを読み込みます。
MYSQLI_READ_DEFAULT_GROUP	my.cnf または MYSQL_READ_DEFAULT_FILE で指定されたファイルから指定されたグループのオプションを読み込みます。

SQL Anywhere 専用の関数

以下の関数は、SQL Anywhere PHP クラスのみに存在します。

SASQL_MESSAGE()

この関数は、サーバ・コンソールにメッセージを書き込みます。

SASQL_RESULT_ALL()

この関数は、すべての結果をフェッチし、オプションのフォーマット文字列を使用して HTML 出力テーブルを生成します。

SASQL_STMT_BIND_PARAM_EX()

この関数は、PHP 変数を文にバインドするときにより高い細分性を許容するという点で、sasql_stmt_bind_param() と異なります。

MySQL 専用の関数

以下の表には、現在のところ SQL Anywhere PHP extension 2.0.4 には同等の関数がない関数をまとめます。

関数名	説明
mysql_client_encoding	現在の接続の文字セット名を返します。
mysql_db_name	データベース名を返します。
mysql_fetch_lengths	結果における各出力の長さを取得します。
mysql_field_flags	結果における指定されたフィールドに関連付けられたフラグを取得します。
mysql_field_len	指定されたフィールドの長さを返します。
mysql_field_name	結果における指定されたフィールドの名前を取得します。
mysql_field_seek	指定されたフィールド・オフセットに結果ポインタを設定します。
mysql_field_table	指定されたフィールドが属するテーブルの名前を取得します。
mysql_field_type	結果における指定されたフィールド・タイプを取得します。
mysql_get_host_info	MySQL のホスト情報を取得します。
mysql_get_proto_info	MySQL のプロトコル情報を取得します。

mysql_info	直前のクエリに関する情報を取得します。
mysql_list_processes	MySQL のプロセスをリストします。
mysql_ping	サーバ接続を ping するかまたは接続がない場合は再接続します。
mysql_result	結果データを取得します。
mysql_set_charset	クライアントの文字セットを設定します。
mysql_stat	現在のシステム状況を取得します。
mysql_thread_id	現在のスレッドの ID を返します。
mysql_unbuffered_query	結果ローをフェッチおよびバッファリングせずに、SQL クエリを MySQL に送信します。

SQL Anywhere で同様の結果を得るには

mysql_field_len() mysql_field_name() mysql_field_type()

MySQL は、これらの専用関数を提供することにより、指定されたフィールドの長さ、名前、および型の取得を単純化します。

SQL Anywhere でこれらのプロパティを取得するには、sasql_sql_fetch_field() 関数を使用します。返されるオブジェクトには、フィールドの長さ、名前、および型が含まれています。

例：

```
<?php
   ... $field = sasql_fetch_field($result);  //object with field information $field =
   $field_info->length;        //get length of the field $field = $field_info->name;
   //get name of the field $field = $field_info->type;          //get the type of the field ...
?>
```

まとめ

SQL Anywhere と MySQL には、動作がごくわずかしか違わない類似した PHP 関数があります。

一般に、MySQL から SQL Anywhere への移行は難しいものではありません。このホワイトペーパーで説明されているような違いがわずかにあることを覚えておけば、より簡単に移行プロセスを実行することができます。

法的注意