Virtual DDBMS PHP Direct-Engine-Call API Documentation Draft v0.0.0.3

Copyright (C) 2008-2009 Muhammad Muzzammil bin Mohd Salahudin & Chan Huah Yong
Grid Computing Lab - University Sains Malaysia

This document is licensed under the GNU Free Documentation License (GNU FDL) either version 1.3 of the license, or (at your option) any later version as published by the Free Software Foundation.

The DDBMSConnection Class

Class Synopsis

```
class DDBMSConnection {
        /* Properties */

        /* Methods */
        bool connected();
        void connect(string $username, string $database_type, int $port_no);
        string addResource(string $database_type, string $host_address, int $port_no, string $username, string
$password, string $host_name);
        void removeResource(string $resource_id)
        void execSQL(string $sql)
        void evalSQL(string $sql)
        array fetchNextRow()
}
```

Description

resource DDBMSConnection(string $unique_str)
Represents a connection between PHP and DDBMS.

Parameters

String $unique_str
A unique string to differentiate between multiple DDBMS objects.

Return Values

Returns the DDBMS OO object.

Example

```
$str = session_id(); // can be any string, so long as it's always unique (eg. time)
$ddbms = new DDBMSConnection($str);
```

Function connected

Description
bool connected(void)
checks if the newly created DDBMS connection is connected to the server or not.

Parameters
none

Return Values
Returns **TRUE** if the connection exists, **FALSE** otherwise

Example
```
$ddbms = new DDBMSConnection( session_id() );
if( $ddbms->connected() ){ echo "Connection exists"; }
else{ echo "Connection does not exists"; }
```

Method connect

Description
void connect(string $username, string $database_type, int $port_no)
Creates a connection to the DDBMS server.

Parameters
string $username
Represents the user who is going to connect to the DDBMS server.

string $database_type
Represents the database type. Currently only accepts "mysql".

int $port_no
Represents the database connection port number.

Return Values
The DDBMSConnection object gets connected to the DDBMS server. No return values sent.

Example
```
$ddbms = new DDBMSConnection( session_id() );
if( ! $ddbms->connected() ){
        $ddbms->connect( "username", "mysql", 3306 );
}
```

Method addResource

Description
string addResource(string $database_type, string $host_address, int $port_no, string $db_name, string $username, string $password, string $odbc_options, string $description)

Parameters
string $database_type
Represents the database type of the new database which is going to be added to DDBMS.

string $host_address
Represents the hostname / ip address where the database is located.

int $port_no
Represents the database connection port number.

string $username
Represents the username who is allowed to access this database

string $password
Represents the password of the user mentioned above.

string $odbc_options
Optional options for ODBC.

string $ description
Optional string to describe the newly created database.

Return Values
Returns the (unique) resource ID of the newly created resource. In addition to that, the new resource is added to the global catalog.

Example
// assume we have created a $ddbms connection
$resID = $ddbms->addResource("mysql", "10.207.207.123", 3306, "test_user", "password", "Sample Database");
// now $resID contains the resource ID of the newly added database.

Method removeResource

Description
void removeResource(string $resource_id)

Parameters
string $resource_id
Represents the resource ID of the resource to be removed. You can use the query "SHOW RESOURCES" to get the resource ID of the database, which can be used here when removing it.

Return Values
The resource is removed from the global catalog.

Example
```
$resID = 'xY5U'; // assume we know the resource ID
$ddbms->removeResource( $resID );
```

Method evalSQL

Description
void evalSQL(string $sql)

Method execSQL

Description
void execSQL(string $sql)

Parameters
string $sql
The SQL query which we want to execute.

Return Values
No values are returned from this method. However, the DDBMS object has executed the query, and if the query is expecting a result (eg. SELECT query, SHOW query, etc), then the function fetchNextRow() is used to retrieve the result.

Example
```
// create database
$ddbms->execSQL( "CREATE DATABASE TestDB" );

// create table in TestDB using dot-notation
$ddbms->execSQL( "CREATE TABLE TestDB.User (id INT NOT NULL UNIQUE PRIMARY KEY, name TINYTEXT, address TEXT)" );

// create table in TestDB without using dot-notation
$ddbms->execSQL( "USE TestDB" );
$ddbms->execSQL( "CREATE TABLE User (id INT NOT NULL UNIQUE PRIMARY KEY, name TINYTEXT, address TEXT)" );

// insert records into table
$ddbms->execSQL( "INSERT INTO TestDB.User VALUES (1, 'Name 1', 'Address 1')" );
$ddbms->execSQL( "INSERT INTO TestDB.User VALUES (2, 'Name 2', 'Address 3')" );

// the examples below currently are not supported
$ddbms->execSQL( "UPDATE TestDB.User SET address='Address 2' WHERE id=2" );
$ddbms->execSQL( "SELECT * FROM TestDB.User" );
```

Method fetchNextRow

Description
array fetchNextRow(void)

Parameters
None

Return Values
Returns a row of the result set. Use loops to retrieve all record sets. Please note that the first row of the record set is always the field names. The actual data will start from second row onwards.

Examples
```
$ddbms->execSQL( "SHOW RESOURCES" );
while( $row = $ddbms->fetchNextRow() ){
        // do anything with the records here
}
```

void ddbmsConnectionDestroySessionObject(String unique_str)