솔루션 상호운용성 인증 지원
[PHP 데이터베이스의 문제점 및 해결방안]

한국소프트웨어진흥원

공개SW기술지원센터

<Revision 정보>

일자	VERSION	변경내역	작성자
2007. 5. 16	0.1	초기 작성	김상운

	솔루션 상호운용성 인증지원	
	구분 : 기업 대 기관	단계: 완료
	작성자: 김상운	작성일: 2007년 5 월 16 일
	검토자: 허종윤	작성일: 2007년 5 월 16 일
	승인자: 박훈성	작성일: 2007년 5 월 16 일

1. 인증지원 기관 정보

구분	항목	내용	비고
기업/기관 정보	지역	서울시 송파구 가락동	
	기업/기관 명칭	한굴소프트웨어진흥원	
	부서	공개SW사업단	
	직책	수석연구원	
	담당자 이름	조유진	
	전화번호 / 팩스번호	02-2141-5066	
	E-Mail	yjcho@software.or.kr	

2. 인증 지원 사항

구분	항목	내용	비고
노트북 및 PC 호환성 검토	지원요청사항	PHP와 데이터베이스간의 2가지 문제점을 진단하고 개발적인 관점에서 해결방안 요청	
	지원상세내역	● PHP 데이터베이스의 2가지 문제점(MySQL을 직접 사용 및 자동증가기능을 사용하지 않음) 해결방안 제시	
	확인 사항	실제 소스코드를 사례로 하여 해결방안제시	

3.PHP 데이터베이스의 5가지 문제점과 해결방법

1) 문제점 1: MySQL을 직접 사용

첫 번째 문제는 mysql_ 함수를 사용하는 오래된 PHP 코드가 데이터베이스에 직접 액세스 하는 것이다. 글1) 은 데이터베이스에 직접 액세스 하는 방법을 보여주고 있다.

```
<?php
function get_user_id( $name )
{
  $db = mysql_connect( 'localhost', 'root', 'password' );
  mysql_select_db( 'users' );

  $res = mysql_query( "SELECT id FROM users WHERE login='".$name."'" );
  while( $row = mysql_fetch_array( $res ) ) { $id = $row[0]; }

  return $id;
}

var_dump( get_user_id( 'jack' ) );
?>
```

글 1) mysql 직접접근 소스

위의 글 1)을 보면 mysql_connect 함수를 사용하여 데이터베이스에 액세스 하고 있다. $name 매개변수를 쿼리에 추가하기 위해 스트링 연결(concatenation)을 사용하는 쿼리를 사용하는 구조이다. 이러한 적접 접근은 문제를 야기 할 수 있으므로 이를 대신할 2가지 좋은 방법으로는 PEAR DB 모듈과 PHP Data Objects (PDO) 클래스를 사용하는 것이다. 두 개 모두 선택된 데이터베이스에서 추상화를 제공한다. 따라서, 코드는 MySQL, PostgreSQL 또는 다른 데이터베이스에 맞춰 조정하지 않고도 실행될 수 있다.
PEAR DB 모듈과 PDO의 추상화 레이어를 사용할 때의 또 다른 이점은 SQL 문에 ? 연산자를 사용할 수 있다는 점이다. 이렇게 하면 SQL을 관리하기가 더 쉬워지고, SQL 인젝션 공격에서 애플리케이션을 보호할 수 있다.
PEAR DB를 사용하는 대안 코드는 다음과 같다.

```
<?php
require_once("DB.php");

function get_user_id( $name )
{
  $dsn = 'mysql://root:password@localhost/users';
  $db =& DB::Connect( $dsn, array() );
  if (PEAR::isError($db)) { die($db->getMessage()); }

  $res = $db->query( 'SELECT id FROM users WHERE login=?',
  array( $name ) );
  $id = null;
  while( $res->fetchInto( $row ) ) { $id = $row[0]; }

  return $id;
}

var_dump( get_user_id( 'jack' ) );
?>
```

글 2) PEAR DB를 사용하는 대안 코드

MySQL의 모든 직접적인 언급은 $dsn의 데이터베이스 연결 스트링을 제외하고는 없어진다. 게다가, 우리는 ? 연산자를 통해서 SQL에서 $name operator. Then, the data for the query is sent in through array 변수를 사용한다. 이 쿼리에 대한 데이터는 query() 메소드의 끝에서 array를 통해서 보내진다.

2) 문제점 2: 자동 증가 기능을 사용하지 않는 것

대부분의 데이터베이스와 마찬가지로, MySQL도 레코드 기반으로 자동 증가하는 고유 식별자를 만드는 기능이 있다. 그럼에도 불구하고, 우리는 여전히, 최대 id를 찾기 위해 SELECT 문을 처음 실행하고, 하나를 그 id, then adds one to that id와 새로운 레코드에 추가하는 코드를 본다. 글 3)은 잘못된 스키마 샘플이다.

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
  id MEDIUMINT,
  login TEXT,
  password TEXT
);

INSERT INTO users VALUES ( 1, 'jack', 'pass' );
INSERT INTO users VALUES ( 2, 'joan', 'pass' );
INSERT INTO users VALUES ( 1, 'jane', 'pass' );
```

글3) 잘못된 스키마 양식

여기에서 CREATE 문 다음에 나오는 INSERT 문과 같이 보면 id 필드는 간단히 정수로 지정되는데 value 값이 적용되야 하는데도 불구하고 사용자가 원하는 어떤 값을 추가할 수 있다. 글 4)는 사용자들을 이러한 유형의 스키마에 추가하는 PHP 코드 모습이다.

```
<?php
require_once("DB.php");

function add_user( $name, $pass )
{
  $rows = array();
  $dsn = 'mysql://root:password@localhost/bad_badid';
  $db =& DB::Connect( $dsn, array() );
  if (PEAR::isError($db)) { die($db->getMessage()); }
  $res = $db->query( "SELECT max(id) FROM users" );
  $id = null;
  while( $res->fetchInto( $row ) ) { $id = $row[0]; }
  $id += 1;
  $sth = $db->prepare( "INSERT INTO users VALUES(?,?,?)" );
  $db->execute( $sth, array( $id, $name, $pass ) );
  return $id;
}
$id = add_user( 'jerry', 'pass' );
var_dump( $id );
?>
```

글 4) 사용자 추가에 대한 잘못된 스키마 양식

add_user.php의 코드는 먼저 id의 최대 값을 찾기 위해 쿼리를 수행한다. 그런 다음 파일은 id 값에 하나를 더하여 INSERT 문을 실행한다. 이 코드는 부하가 큰 서버에서 경쟁 조건에서 실패할 수 있다. 게다가 이는 효율성도 떨어진다.
대안으로는 MySQL의 자동 증가 기능을 사용하여 각 삽입에 대한 고유 ID를 자동으로 만드는 것이 적합한 방법일 수 있으며 그 방법으로는 아래의 글 5)를 보자

```
DROP TABLE IF EXISTS users;
CREATE TABLE users (
  id MEDIUMINT NOT NULL AUTO_INCREMENT,
  login TEXT NOT NULL,
  password TEXT NOT NULL,
  PRIMARY KEY( id )
);
INSERT INTO users VALUES ( null, 'jack', 'pass' );
INSERT INTO users VALUES ( null, 'joan', 'pass' );
INSERT INTO users VALUES ( null, 'jane', 'pass' );
```

글 5) 수정된 id 값 사용 소스

위에서 보면 NOT NULL 플래그를 추가하여 이 필드가 무효가 되어서는 안된다는 것을 나타냈다. 또한 AUTO_INCREMENT 플래그를 추가하여 이 필드가 자동 증가한다는 것을 나타냈고, PRIMARY KEY 플래그를 통해서 어떤 필드가 id인지를 나타냈다. 이러한 변화는 처리 속도의 개선에도 영향을 미친다. 글 6)은 위 수정된 내용을 바탕으로 업데이트 된 PHP 코드로서 사용자를 테이블에 삽입한다.

```
<?php
require_once("DB.php");
function add_user( $name, $pass )
{
  $dsn = 'mysql://root:password@localhost/good_genid';
  $db =& DB::Connect( $dsn, array() );
  if (PEAR::isError($db)) { die($db->getMessage()); }
  $sth = $db->prepare( "INSERT INTO users VALUES(null,?,?)" );
  $db->execute( $sth, array( $name, $pass ) );
  $res = $db->query( "SELECT last_insert_id()" );
  $id = null;
  while( $res->fetchInto( $row ) ) { $id = $row[0]; }
  return $id;
}
$id = add_user( 'jerry', 'pass' );
var_dump( $id );
?>
```

글 6) 수정된 사용자 추가 PHP 소스

최대 id 값을 얻는 대신, 지금 INSERT 문을 사용하여 데이터를 삽입하고, SELECT 문을 사용하여 최근 삽입된 레코드의 id 를 가져온다. 이 코드는 원래 버전과 관련 스키마 보다 훨씬 더 단순하고 보다 효율적이다.

MySQL의 자동 증가 기능을 사용하는 것을 대신할 방법은 PEAR DB 시스템에서 nextId() 메소드를 사용하는 것이다. MySQL의 경우, 이것은 새로운 시퀀스 테이블을 만들고, 정교한 잠금 메커니즘을 사용하여 이것을 관리한다. 이 방식을 사용하면 다른 데이터베이스 시스템들에서도 작동한다는 이점이 있다.

어떤 방법을 사용하든, 증가하는 고유 ID를 관리하는 시스템을 사용해야 하고, 처음 쿼리했던 시스템에 의존하지 말아야 하며, 값을 스스로 증가시키고, 레코드를 추가하도록 한다. 후자의 방식은 높은 용량의 사이트에서 경쟁 조건이 될수 있다.

4.참고문헌

- Five common PHP database problems :
http://www.ibm.com/developerworks/library/os-php-dbmistake/index.html
http://www.ibm.com/developerworks/opensource/top-projects/php.html