Implementing PHP/MySQL Striped Views

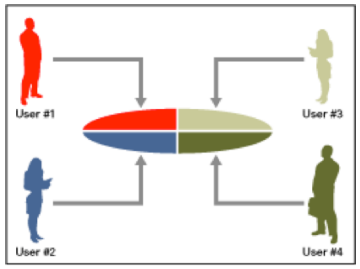
Michael McLaughlin
BYU - Idaho

Outline

- What is a Striped View?
- How do you Implement a Striped View?
- How do you write the PHP?

What is a Striped View?

- Is a stored query that depends on a runtime session variable
- A striped view lets users see only a subset of rows from a table
- A striped view is readable and writable



What is a Striped View?

Security through an ACL

- An Access Control List (ACL) is a table inside the database or LDAP server that maintains
  - A list of users and passwords
  - A set of permissions for group, language, and financial set of books access

What is a Striped View?

Dependency on Session Variables

- You can't build a view by directly referencing a session variable, like this

```
CREATE VIEW hobbit_v
  SELECT * FROM hobbit WHERE hobbit_name = @sv_login;
END;
/
```

- This syntax is disallowed, as explained in MySQL Reference 13.1.20 and raises the following error

```
ERROR 1351 (HY000): VIEW's SELECT contains a variable
                    or parameter
```

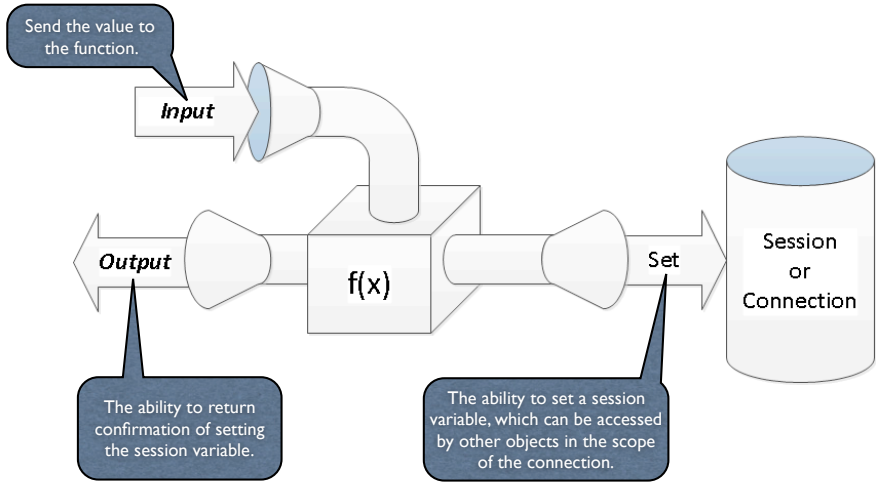
How do you Implement a Striped View?

The Steps

- You can write a striped view by comparing a column value to a function return value
- Functions can set and read MySQL session variables
- A setter function reads the ACL table, and sets session variables that stripe views
- A getter function reads the session variable and returns its value into a SQL context

How do you Implement a Striped View?

Case Study #1: Create a Sample Table



How do you Implement a Striped View?

Case Study #1: Create a Sample Table

```
-- Conditionally drop the hobbit table.
DROP TABLE IF EXISTS hobbit;

-- Create the hobbit table.
CREATE TABLE hobbit
( hobbit_id    int unsigned
, hobbit_name  varchar(20));

-- Seed the hobbit table with row two values.
INSERT INTO hobbit VALUES
( 1,'Bilbo'),( 1,'Frodo');
```

How do you Implement a Striped View?

Case Study #1: Create a Set Login Function

```
CREATE FUNCTION set_login_name
(pv_login_name VARCHAR(20)) RETURNS INT UNSIGNED
BEGIN
  /* Declare a local variable to verify completion of the task. */
  DECLARE  lv_success_flag  INT UNSIGNED  DEFAULT FALSE;

  /* Check whether the input value is something other than a null. */
  IF pv_login_name IS NOT NULL THEN

    /* Set the session variable and enable the success flag. */
    SET @sv_login_name := pv_login_name;
    SET lv_success_flag := TRUE;

  END IF;

  /* Return the success flag. */
  RETURN lv_success_flag;
END;
$$
```

How do you Implement a Striped View?

Case Study #1: Create a Get Login Function

```
CREATE FUNCTION get_login_name() RETURNS VARCHAR(20)
BEGIN
  /* Return the success flag. */
  RETURN @sv_login_name;
END;
$$
```

How do you Implement a Striped View?

Case Study #1: Create a Striped View

```
-- Conditionally drop the view.
DROP VIEW IF EXISTS hobbit_v;

-- Create the function-enabled view.
CREATE VIEW hobbit_v AS
  SELECT *
  FROM   hobbit
  WHERE  hobbit_name = get_login_name();
```

How do you Implement a Striped View?

Case Study #1: Query the Table

```
-- Query table.
SELECT * FROM hobbit;
```

```
+-----------+-------------+
| hobbit_id | hobbit_name |
+-----------+-------------+
|         1 | Bilbo       |
|         1 | Frodo       |
+-----------+-------------+
2 rows in set (0.00 sec)
```

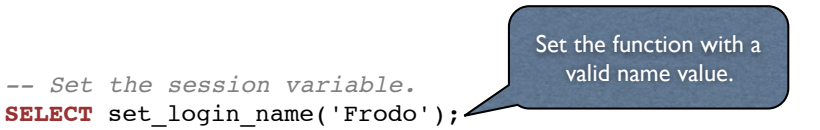
```
-- Query view.
SELECT * FROM hobbit_v;
```

```
Empty set (0.00 sec)
```

A query without setting the session variable returns a `NULL` value; and doesn't raise an exception because the session varaible isn't set.

How do you Implement a Striped View?

Case Study #1: Set & Get the Striped Value

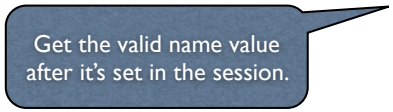



```
-- Set the session variable.
SELECT set_login_name('Frodo');

+--------------------------+
| set_login_name('Frodo') |
+--------------------------+
|                        1 |
+--------------------------+
1 row in set (0.00 sec)
```

```
-- Get the session name.
SELECT get_login_name();

+------------------+
| get_login_name() |
+------------------+
| Frodo            |
+------------------+
1 row in set (0.00 sec)
```



How do you Implement a Striped View?

Case Study #1: Query the View

```
-- Query table.
SELECT * FROM hobbit;
```

```
+-----------+-------------+
| hobbit_id | hobbit_name |
+-----------+-------------+
|         1 | Bilbo       |
|         1 | Frodo       |
+-----------+-------------+
2 rows in set (0.00 sec)
```

```
-- Query striped view.
SELECT * FROM hobbit_v;
```

```
+-----------+-------------+
| hobbit_id | hobbit_name |
+-----------+-------------+
|         1 | Frodo       |
+-----------+-------------+
1 row in set (0.00 sec)
```

A query only returns one row when the session variable has been set, which

How do you Implement a Striped View?

Case Study #2: Create a Striped ACL View

```
-- Conditionally drop the view.
DROP TABLE IF EXISTS application_user;

-- Create the function-enabled view.
CREATE TABLE application_user
( user_id  int(10) unsigned PRIMARY KEY AUTO_INCREMENT
, user_name  varchar(20) NOT NULL
, user_role  varchar(20) NOT NULL
, user_group_id  int(10) unsigned NOT NULL
, user_type  int(10) unsigned NOT NULL
, first_name  varchar(20)
, middle_name  varchar(20)
, last_name  varchar(20)
, created_by  int(10) unsigned NOT NULL
, creation_date  datetime NOT NULL
, last_updated_by  int(10) unsigned NOT NULL
, last_update_date  datetime NOT NULL
, CONSTRAINT natural_key UNIQUE (user_name)
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8;
```

How do you Implement a Striped View?

Case Study #2: Seed the Striped ACL Table

```
-- Seed the ACL table.
INSERT INTO application_user VALUES
 ( null, 'potterhj', 'System Admin', 1, 1
 ,'Harry', 'James', 'Potter', 1, NOW(), 1, NOW())
,( null, 'weasilyr', 'Guest', 0, 1
 ,'Ronald', null, 'Weasily', 1, NOW(), 1, NOW())
,( null, 'longbottomn', 'Guest', 0, 1
 ,'Neville', null, 'Longbottom', 1, NOW(), 1, NOW())
,( null, 'holmess', 'DBA', 2, 1
 ,'Sherlock', null, 'Holmes', 1, NOW(), 1, NOW())
,( null, 'watsonj', 'DBA', 2, 1
 ,'John', 'H', 'Watson', 1, NOW(), 1, NOW());
```

How do you Implement a Striped View?

Case Study #2: Set the Application Function

```
CREATE FUNCTION set_login(pv_login_name VARCHAR(20)) RETURNS INT UNSIGNED
BEGIN

  /* Declare a local variable to verify completion of the task. */
  DECLARE  lv_success_flag  INT UNSIGNED  DEFAULT FALSE;
  DECLARE  lv_login_id  INT UNSIGNED;
  DECLARE  lv_group_id  INT UNSIGNED;

  /* Declare a cursor to return an authorized user id. */
  DECLARE authorize_cursor CURSOR FOR
    SELECT   a.user_id
    ,        a.user_group_id
    FROM     application_user a
    WHERE    a.user_name = pv_login_name;

  /* Check whether the input value is something other than a null value. */
  IF pv_login_name IS NOT NULL THEN

    OPEN  authorize_cursor;
    FETCH authorize_cursor INTO lv_login_id, lv_group_id;
    CLOSE authorize_cursor;

    /* Set the success flag. */
    SET @sv_login_id := lv_login_id;
    SET @sv_group_id := lv_group_id;
    SET lv_success_flag := TRUE;

  END IF;

  /* Return the success flag. */
  RETURN lv_success_flag;
END;
$$;
```

This sets the login identifier, that links to an individual

This sets the group identifier, that links to an individual's group membership

How do you Implement a Striped View?

Case Study #2: A two element striped view

```
CREATE VIEW authorized_user AS
SELECT    au.user_id
,         au.user_name
,         au.user_role
,         CONCAT(au.last_name,", ",au.first_name,"
",IFNULL(au.middle_name,"")) AS full_name
FROM      application_user au CROSS JOIN
         (SELECT   get_login_id() AS login_id
          ,        get_group_id() AS group_id) fq
WHERE    (au.user_group_id = 0
AND       au.user_group_id = fq.group_id
AND       au.user_id = fq.login_id)
OR        fq.group_id = 1
OR       (fq.group_id > 1
AND       au.user_group_id = fq.group_id);
```

A subquery in the FROM clause is disallowed.

How do you Implement a Striped View?

Case Study #2: A two element striped view

```
CREATE VIEW function_query AS
SELECT   get_login_id() AS login_id
,        get_group_id() AS group_id;

CREATE VIEW authorized_user AS
SELECT   au.user_id
,        au.user_name
,        au.user_role
,        CONCAT(au.last_name,", ",au.first_name," "
         ,IFNULL(au.middle_name,"")) AS full_name
FROM     application_user au CROSS JOIN function_query fq
WHERE   (au.user_group_id = 0
AND      au.user_group_id = fq.group_id
AND      au.user_id = fq.login_id)
OR       fq.group_id = 1
OR      (fq.group_id > 1
AND      au.user_group_id = fq.group_id);
```

How do you Implement a Striped View?

Case Study #1: Superuser Query

Setting the view with a superuser name gives you visibility to all rows in the table, which effectively unstripes the view.

```
-- Set the login.
SELECT set_login('potterhj');

-- Query table.
SELECT * FROM authorized_user;
```

```
+---------+-------------+--------------+----------------------+
| user_id | user_name   | user_role    | full_name            |
+---------+-------------+--------------+----------------------+
|       1 | potterhj    | System Admin | Potter, Harry James  |
|       2 | weasilyr    | Guest        | Weasily, Ronald      |
|       3 | longbottomn | Guest        | Longbottom, Neville  |
|       4 | holmess     | DBA          | Sherlock, Holmes     |
|       5 | watsonj     | DBA          | John, Watson H       |
+---------+-------------+--------------+----------------------+
5 rows in set (0.00 sec)
```

How do you Implement a Striped View?

Case Study #1: Group Privileged User

Setting the view with a group name gives you visibility to all rows in the table that belong to the same group.

```
-- Set the login.
SELECT set_login('holmess');

-- Query table.
SELECT * FROM authorized_user;
```

```
+---------+-----------+-----------+-------------------+
| user_id | user_name | user_role | full_name         |
+---------+-----------+-----------+-------------------+
|       4 | holmess   | DBA       | Holmes, Sherlock  |
|       5 | watsonj   | DBA       | Watson, John H    |
+---------+-----------+-----------+-------------------+
2 rows in set (0.00 sec)
```

How do you Implement a Striped View?

Case Study #1: Unprivileged User Query

Setting the view with an unprivileged user name gives you visibility to only the row that represents the user.

```
-- Set the login.
SELECT set_login('weasilyr');

-- Query table.
SELECT * FROM authorized_user;
```

```
+---------+-----------+-----------+-------------------+
| user_id | user_name | user_role | full_name         |
+---------+-----------+-----------+-------------------+
|       2 | weasilyr  | Guest     | Weasily, Ronald   |
+---------+-----------+-----------+-------------------+
1 row in set (0.00 sec)
```

How do you write the PHP?

Process for Setting

- Within the scope of the connection
  - Call the `set_login()` function in a query
  - Managing the return value is possible but not required since an unset variable returns a null record set
  - Call the striped view

How do you write the PHP?

Case Study #1: Query the Table

```
// Declare a dynamic function call.
$query = "SELECT set_login(?)";

// Attempt preparing statement.
if (!$stmt = $mysqli->prepare($query)) {

  // Print failure to resolve query message.
  print $mysqli->error."<br />";
  print "Failed to resolve query ...<br />";
}
else {

  // Bind variable to SQL statement and execute it.
  $stmt->bind_param("s", $input);
  $stmt->execute();
  $stmt->close();
}
```

After opening the connection, you call the local function to set a session variable.

How do you write the PHP?

Case Study #1: Query the Striped View

```
// Declare a static query.
$query = "SELECT au.user_id, au.user_name, au.user_role, au.full_name FROM authorized_user au" ;

// Loop through a result set until completed.
do {
  // Attempt query and exit with failure before processing.
  if (!$stmt = $mysqli->query($query)) {
    // Print failure to resolve query message.
    print $mysqli->error."<br />";
    print "Failed to resolve query ...<br />";
  }
  else {
    // Print the opening HTML table tag.
    print '<table><tr><th class="ID">ID</th><th class="Label">User Name</th>'
    .     '<th class="Label">User Role</th><th class="Label">Full Name</th></tr>';

    // Fetch a row for processing.
    while( $row = $stmt->fetch_row() ) {
      // Print the opening HTML row tag.
      print "<tr>";
      // Loop through the row's columns.
      for ($i = 0;$i < $mysqli->field_count;$i++) {
        // Handle column one differently.
        if ($i == 0)
          print '<td class="ID">'.$row[$i]."</td>";
        else
          print '<td class="Label">'.$row[$i]."</td>";
        }
        // Print the closing HTML row tag.
        print "</tr>";
      }
    }
  } while( $mysqli->next_result());

  // Print the closing HTML table tag.
  print "</table>";

  // Release connection resource.
  $mysqli->close(); }
```

After opening the connection, you call the local function to set a session variable.