UserIdentifiers

User Names & Authorization Identifiers in Derby

Overview

A user in Derby is represented by a case-sensitive value, called an authorization identifier. E.g. EVE, eve, eVe, eve@yahoo.com are all different users. A user name can be provided to or read from Derby in a number of forms, e.g. SQL identifiers, a String in a Java program etc. This wiki page summarizes the how a user name converted to an authorization identifier in various situations.

Note that rules for user names in Derby are independent on how that user name is defined to Derby or authenticated. Thus these rules apply if the database is using the BUILTIN authentication or LDAP authentication.

User Name Rules

<table>
<thead>
<tr>
<th>Context</th>
<th>Definition</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL identifiers</td>
<td>User identifier is a SQL identifier, section 5.4 SQL 2003. A regular identifier is upper-cased to represent an authorization identifier, e.g. eve, eVe and EVE all represent the authorization identifier EVE. A delimited identifier (with double quotes) does not have any case conversion to its authorization identifier, e.g. &quot;eVe&quot; represents the authorization identifier eVe. Note that the delimited identifier &quot;EVE&quot; represents the authorization identifier EVE which is the same user as the regular identifiers eve, EVE and EvE etc. The authorization identifier represents how a user name is stored/processed by the SQL engine in a SQL context, e.g. the system tables representing granted permissions. Note that derby.* properties are not in the SQL context.</td>
<td>grantee in REVOKE statement</td>
</tr>
<tr>
<td>current user expression</td>
<td>Returns the authorization identifier for the current user.</td>
<td>VALUES CURRENT USER</td>
</tr>
<tr>
<td>VALUES SESSION_USER</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VALUES (In user())</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SQL Routine</td>
<td>A system SQL routine taking a USERNAME parameter requires the caller pass in the authorization identifier, this is to align with the value being passed in from CURRENT USER or a value obtained from a system table representing a user.</td>
<td>SYSCS_UTIL.SYSCS_GET_USER_ACCESSS DERBY-3095</td>
</tr>
<tr>
<td>JDBC Connection request</td>
<td>Follows the rules of SQL identifiers including support for delimited identifiers.</td>
<td>User named passed into method call, e.g. DataSource, setUser, DriverManager, getConnection</td>
</tr>
<tr>
<td>User set as user property in DriverManager connection request</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User name on JDBC URL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JDBC DatabaseMeta Data: getUsername()</td>
<td>Returns the user name used to make the JDBC connection request which is a SQL identifier and not the authorization identifier.</td>
<td>conn.getMetaData().getUserName()</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>Derby BUILTIN authentication</td>
<td>Follows the rules of SQL identifiers including support for delimited identifiers. DERBY-3150</td>
<td>System property derby.user.username= password</td>
</tr>
<tr>
<td>CALL SYSCS_UTIL.SYSCS_SET_DATABASE_PROPERTY('derby.user.userid', 'password')</td>
<td>User names set in Java or database properties. Documented as following the rules of SQL identifiers including support for delimited identifiers. Thus derby.database.fullAccessUsers=eVe and derby.database.fullAccessUsers=EVE have identical meaning.</td>
<td>derby.database.fullAccessUsers=eVe</td>
</tr>
<tr>
<td>Connection authorization</td>
<td>derby.database.readOnlyAccessUsers=eVe</td>
<td>derby.database.readOnlyAccessUsers =eVe</td>
</tr>
</tbody>
</table>

**Example - Authorization identifier EVE**

All of these examples are valid for authorization identifier EVE

```sql
GRANT SELECT ON t TO eve
REVOKE SELECT ON t FROM EVE
GRANT SELECT ON t TO EVE
GRANT SELECT ON t TO 'EVE'
VALUES CURRENT_USER will return EVE
CALL SYSCS_UTIL.SYSCS_SET_USER_ACCESS('EVE', NULL)
VALUES SYSCS_UTIL.SYSCS_GET_USER_ACCESS('EVE')
jdbc:derby:db;user=eve
jdbc:derby:db;user=EVE
ds.getConnection("eve", password); // Java code DatabaseMetaData.getUserame() will return eve
ds.getConnection("EVE", password); // Java code DatabaseMetaData.getUserame() will return EVE
CALL SYSCS_UTIL.SYSCS_SET_DATABASE_PROPERTY('derby.user.EVE', 'password')
CALL SYSCS_UTIL.SYSCS_SET_DATABASE_PROPERTY('derby.user.eve', 'password')
derby.database.readOnlyAccesssUsers=eVe
```

**Example - Authorization identifier eve**

All of these examples are valid for authorization identifier eve

```sql
GRANT SELECT ON t TO "eve"
REVOKE SELECT ON t FROM "eve"
VALUES CURRENT_USER will return eve
CALL SYSCS_UTIL.SYSCS_SET_USER_ACCESS('eve', NULL)
VALUES SYSCS_UTIL.SYSCS_GET_USER_ACCESS('eve')
```
jdbc:derby:db;user="eve"

ds.getConnection("eve", password); // Java code DatabaseMetaData.getUserName() will return "eve"

CALL SYSCS_UTIL.SYSCS_SET_DATABASE_PROPERTY('derby.user."eve"', 'password')
derby.database.readOnlyAccessUsers="eve"

Note

Earlier versions of this document used normal user name to represent authorization identifier. Thanks to Rick's comments in DERBY-2109 I decided to re-write this in terms of authorization identifier.