

Welcome to our webinar!



- This webinar starts in a moment - please stay tuned
- This webinar will be recorded
- You will get the slides, recording and SQL snippets
- During the webinar, you may ask questions using the Q&A button - you may ask questions anonymously
- You can hop on and off on all the sessions

Day 1	14:00 - 15:00 Automate tasks using SYS and SYSADMIN schemas 15:15 - 16:00 The DV REST API and how to use it 16:15 - 17:00 Security
Day 2	15:00 - 16:00 Synchronization Methods 16:15 - 17:15 Working with Connectors
Day 3	15:00 - 16:00 Self-service Troubleshooting 16:15 - 17:15 Job Management Outlook

DATA VIRTUALITY MASTERCLASS

Topic: Security

What to expect from this session?

- In this track we will show you how to use DV's security features in order to implement a coherent security strategy. The following topics will be covered:
- role-based permissions
- column masking
- hashing
- row-level security
- network security
- own certificates

General remarks about DV security

- Permission system applies to Pipes Pro and LDW
- Pipes Pro: only when working on the data model, but not for replicated data
- Network
 - firewall settings restrict what ports can be accessed from where, we set this up when we host for you
- Encryption topic
 - Data in transit: use SSL (checkbox in DV studio, use encrypted connections to data sources)
 - Data at rest: databases may be encrypted as well
- Audit logging
 - tracks what users are doing, don't use admin for multiple users

DV's role based permission system - introduction

DV's role based permission system - introduction

- Mechanism to control access rights
- Allows differentiated access to data model/data sources
- Consists of users and roles
- Applies to all incoming connection types (ODBC/JDBC/REST)
- Entities that can be controlled:
 - schemas / data sources
 - tables / views
 - columns / rows
 - columns can be masked
- When LDAP integration is active:
 - roles (= AD groups) and users will be handled by LDAP -> only role permissions can be modified

Managing users and permissions

Managing users

- DV Studio: Service / Manage users
- Stored Procedures for automation:

- ```
EXEC "SYSADMIN.addUser"(name => 'matthias', pwd => '***', role_name => 'connect-dv-role');
```

connect-dv-role is the standard role. Additional roles can be added later using the  
SYSADMIN.addUserRole procedure

- ```
EXEC "SYSADMIN.changeUserPwd"("name" => 'matthias', "pwd" => 'string_pwd');
```
- ```
EXEC "SYSADMIN.deleteUser"("name" => 'matthias');
```
- These procedures can be used for automation, for example batch import a set of users



# Managing roles and permissions

- DV Studio: Service / Manage users / Roles Tab
- Stored Procedures:
  - `CALL "SYSADMIN.addRole"("name" => 'newrole', "allowCreateTempTables" => boolean_allowCreateTempTables);;`
  - `CALL "SYSADMIN.deleteRole"("name" => 'string_name');;`
- Assign roles:
  - Right click on any entity and select "set permissions"
  - `CALL "SYSADMIN.addUserRole"("user_name" => 'matthias', "role_name" => 'admin-role');;`
  - `CALL "SYSADMIN.setPermissions"(role_name => 'restricted_role', resourceName => 'mysql.salesorderdetail', permissions => 'R', condition => null, isConstraint => FALSE, mask => null, maskOrder => null)`
- You can always track DV Studio actions in the queries tab



# **Demo - Permission management in DV Studio**

**Auditing user's actions**





# Auditing user's actions

- DV Studio: Show queries
  - This is why it makes sense to have dedicated users instead of admin only
- Helpful system tables
  - SYSLOG.QueryLogs
  - SYSLOG.ViewDefinitionHistory
  - SYSLOG.ProcDefinitionHistory
- Watch out!
  - These tables will be cleaned by our maintenance jobs
  - Create a batch replication to secure this information



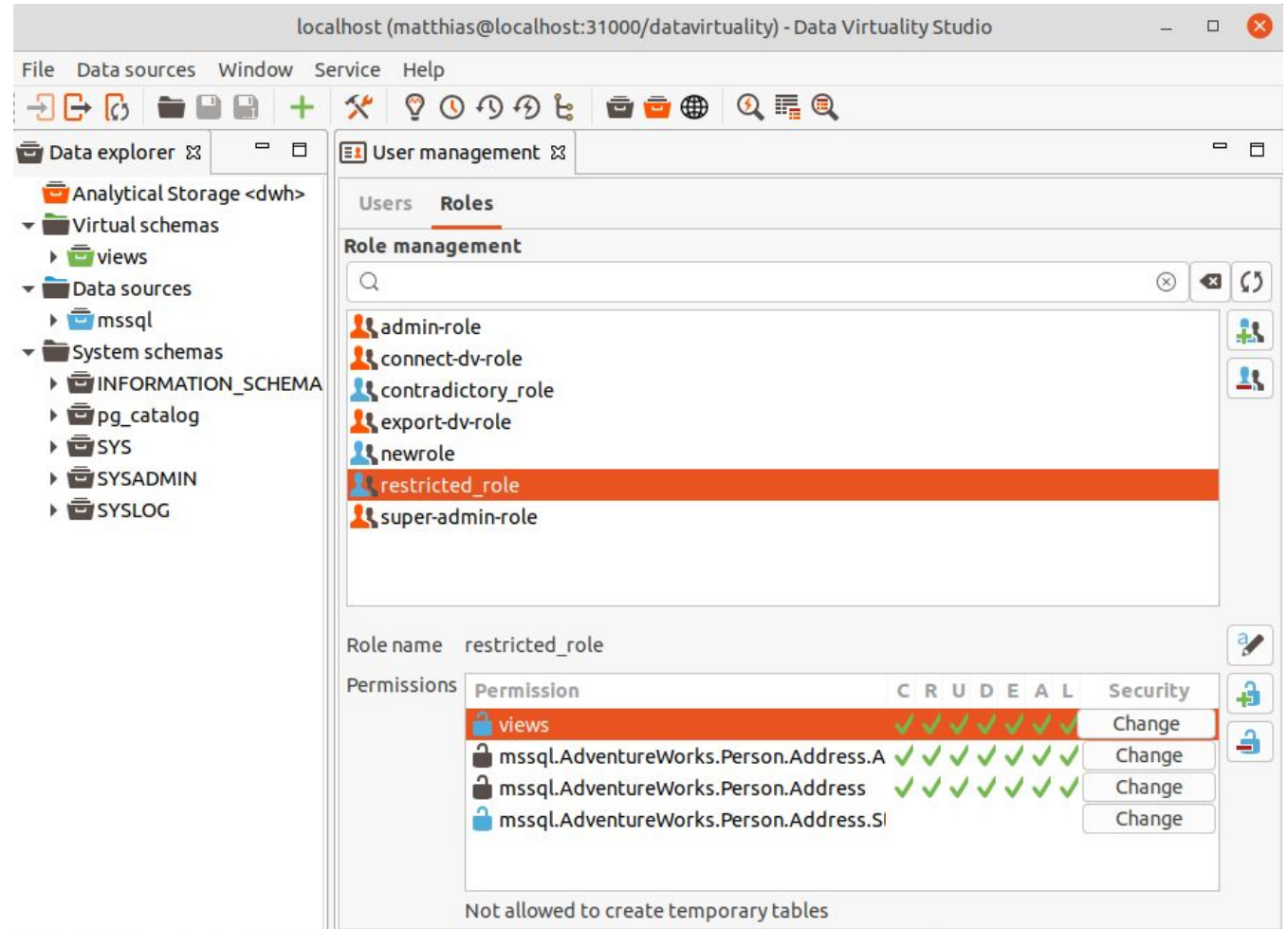
# **Demo - Auditing user's actions**

## Permission calculation and effects

- | Permission                                                                                                                                | C | R | U | D | E | A | L | Security                |
|-------------------------------------------------------------------------------------------------------------------------------------------|---|---|---|---|---|---|---|-------------------------|
|  views                                               | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | <button>Change</button> |
|  mssql.AdventureWorks.Person.Address.AddressLine1    | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | <button>Change</button> |
|  mssql.AdventureWorks.Person.Address                 | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | <button>Change</button> |
|  mssql.AdventureWorks.Person.Address.StateProvinceID | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | <button>Change</button> |

# Metadata visibility

- Starting at version 2.3, metadata will not be visible if you don't have the permissions
- Querying items will result in an error
- Visibility is ODBC / JDBC and REST API
- Important Storage schema permissions



The screenshot shows the 'User management' window in Data Virtuality Studio, specifically the 'Roles' tab. The 'Role management' section lists several roles, with 'restricted\_role' selected. Below this, the 'Permissions' table shows the permissions granted to 'restricted\_role'.

| Permission                             | C | R | U | D | E | A | L | Security |
|----------------------------------------|---|---|---|---|---|---|---|----------|
| views                                  | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | Change   |
| mssql.AdventureWorks.Person.Address.A  | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | Change   |
| mssql.AdventureWorks.Person.Address    | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | Change   |
| mssql.AdventureWorks.Person.Address.SI |   |   |   |   |   |   |   | Change   |

Not allowed to create temporary tables





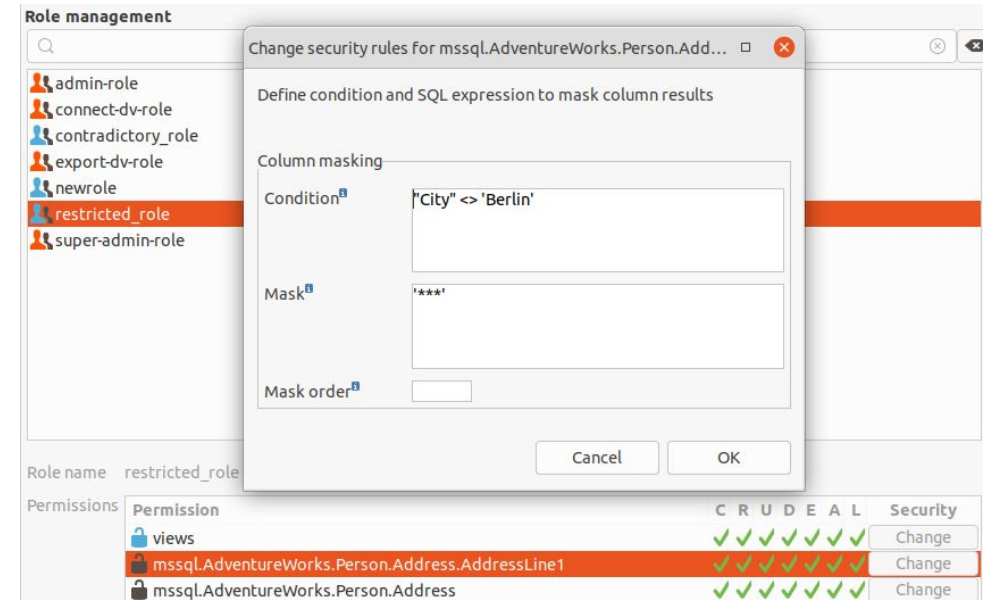
# **Demo - Metadata visibility**

# Column masking and metadata visibility

# Column Masking

- V2.3+, only available on column level permissions
- Works like a WHERE clause
- In the example:
  - user will see AddressLine1 only when the column City = 'Berlin'
  - all other AddressLine1 will be masked
  - don't forget the quotes for the mask
- Example procedure call:

```
EXEC "SYSADMIN.setPermissions"(role_name => 'restricted_role',
resourceName => 'mssql.AdventureWorks.Person.Address.AddressLine1',
permissions => 'CRUDEAL', condition => '"City" <> ''Berlin''',
isConstraint => FALSE, mask => ''***'', maskOrder => null)
```







# **Demo - Column Masking and Row Level Security**

**Anonymization**

# Hashing

- Can be used to anonymize data (semi securely - can always be brute forced)
- md5() available
  - possibly pushed down
- Use salt to anonymize even more:
  - `SELECT to_chars(SHA2_512('Katharinenstrasse' || 'SecretSalt'), 'HEX')`
  - it is hard to keep a salt secret (where would you store it) - it protects only if the person has no DV access/permissions

## Cryptographic Hash Functions

These functions compute the hash of the given value.

| Function      | Definition                                 | Data Type Constraint                          |
|---------------|--------------------------------------------|-----------------------------------------------|
| MD5_BINARY(x) | Return the MD5 hash of the value           | x in { string, varbinary }, returns varbinary |
| SHA1(x)       | Return the SHA-1 hash of the value         | x in { string, varbinary }, returns varbinary |
| SHA2_256(x)   | Return the SHA-2 256 bit hash of the value | x in { string, varbinary }, returns varbinary |
| SHA2_512(x)   | Return the SHA-2 512 bit hash of the value | x in {string, varbinary}, returns varbinary   |

To convert the output varbinary value to hex representation, you can use the `to_chars(VALUE, 'HEX')` function.

# Demo - Hashing



**Own certificates**

# Own Certificates

- Areas to protect:
  - ODBC, JDBC, DV Web, REST API
- Example: securing ODBC with your own certificate which has been imported to a java key store, config in dvserver/standalone/configuration/dvserver-standalone.xml

```
<subsystem xmlns="urn:jboss:domain:teiid:1.0">
...
<transport name="odbc-ssl" socket-binding="teiid-odbc-ssl" protocol="pg">
 <authenticationsecurity-domain="teiid-security"/>
 <sslmode="enabled" ssl-protocol="SSLv3">
 <keystorename="%pathToKeystoreFile%" password="%keystorePassword%"/>
 </ssl>
</transport>
...
</subsystem>
```

# Securing DV server with https

- For hosted clients
  - we will do this for you
- For on premise installation, here's an overview
  - we provide the detailed documentation on demand
  - create the certificate
  - adapt dvserver-standalone.xml
  - secure landing page in dvserver/standalone/deployments/ROOT.war/WEB-INF/web.xml
  - secure REST API in dvserver/standalone/deployments/rest-<YourVersion>.war

## Summary

- DV provides many options to implement a coherent security strategy
- Use permissions to restrict what users can see and do
- Use encryption and firewall rules to prevent unwanted access
- Use the audit log to track user's actions





**Any feedback / questions?**



# Where to get help

Reference Guide section about security functions

[https://datavirtuality.com/docs/#doc=Reference%20Guide&version=2.3&page=Security\\_Functions.html](https://datavirtuality.com/docs/#doc=Reference%20Guide&version=2.3&page=Security_Functions.html)

Reference Guide section about system functions

[https://datavirtuality.com/docs/#doc=Reference%20Guide&version=2.3&page=System\\_Functions.html](https://datavirtuality.com/docs/#doc=Reference%20Guide&version=2.3&page=System_Functions.html)

Help Center

<https://support.datavirtuality.com/hc>

Community

<https://support.datavirtuality.com/hc/en-us/community/topics>

# Thank you!

Please feel free to contact us at:  
[info@datavirtuality.com](mailto:info@datavirtuality.com)

or

visit us at:  
[datavirtuality.com](https://datavirtuality.com)