**Scenario 1: Data Discovery and Classification**

You have successfully migrated your company’s databases to Azure SQL Managed Instance, and the IT Director has asked to you need to ensure sensitive data in these newly migrated databases are classified and labelled according company regulations as specified in the chart below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Database** | **Table** | **Column** | **Information Type** | **Sensitivity Label** |
| TeamXX\_LocalMasterDataDB | Users | Username | Name | Confidential – GDPR |
| TeamXX\_TenantDataDB | Users | Username | Name | Confidential – GDPR |
| TeamXX\_TenantDataDB | UserTransactions | TranAmount | Financial | Highly Confidential |

**LAB INSTRUCTIONS**

1. Using the data discovery and classification service within the azure portal set the information type and the sensitivity label, for the columns in the table above.

**HINT**: Check this at the database level and not the server level.

**Scenario 2**: **Auditing**

Users accessing the UserTransactions table needs to be audited because it contains sensitive information. You have been asked to ensure all DML operations onTeamXX\_TenantDataDB are audited and saved in blob storage.

**HINT**:

You will need the following storage account for your audit logs.

'https://<Replace with Storage Account>.blob.core.windows.net/auditlogs'

For Storage Account please find C:\\_SQLHACK\_\LABS\03-Security\StorageAccount.txt

**LAB INSTRUCTIONS**:

1. Using SSMS, create an Audit for you Managed Instance and define blob storage as the target path.

USE master ;

GO

-- Create the server audit.

CREATE SERVER AUDIT [<your\_server\_audit\_name>]

TO URL (PATH ='<container\_url>' )

GO

1. Create a Database Audit specification that maps to the Audit created in Step 1.

-- Create the database audit specification.

CREATE DATABASE AUDIT SPECIFICATION [<your\_specificatio\_name>]

FOR SERVER AUDIT [<your\_audit\_name>]

ADD (SELECT, INSERT

ON database\_name BY dbo)

WITH (STATE = ON);

GO

1. Enable the Audit.

-- Enable the server audit.

ALTER SERVER AUDIT [<your\_audit\_name>]

WITH (STATE = ON);

GO

1. Check the audit logs are being saved in the Blob storage account.

**Scenario 3: Dynamic Data Masking**

One of the new developers in your team, Peter, has been tasked to make changes and would need to access the database in order to test the changes. Your company policy states **only** members of the Accounting team should have visibility of the data in the TranAmount column.

Perform the steps required to mask the data in the TranAmount column from Peter.

**LAB INSTRUCTIONS:**

1. Connect to your Managed Instance from Management Studio.
2. Run the below script to mask the transaction amount in the UserTransactions table using the default masking function

ALTER TABLE [UserTransactions]

ALTER COLUMN [TranAmount] [decimal](18, 2) MASKED WITH (FUNCTION = 'DEFAULT()')

1. Check the table to ensure you are still able to see **ALL** the data

SELECT \* FROM UserTransactions

GO

1. Create a user for the developer and grant the read only access on the UserTransactions table

CREATE USER Peter WITHOUT LOGIN

GRANT SELECT ON dbo.UserTransactions TO Peter;

1. Run the query below to ensure we have been able to prevent the developer from accessing customers’ privacy information

EXECUTE AS USER = 'Peter';

SELECT \* FROM dbo.UserTransactions;

REVERT;

**Scenario 4: Vulnerability Assessment**

Examine the risk security vulnerabilities and deviations from best practices for the databases you have migrated to SQL Managed Instance by utilizing the Vulnerability Assessments service. To complete this section you will need to assess some the highlighted vulnerabilities.

**LAB INSTRUCTIONS**:

**Part 1: Transparent Data Encryption**

1. Review the vulnerability assessment for TEAMXX\_LocalMasterDataDB.
2. Using SSMS run the following to enable TDE on TEAMXX\_LocalMasterDataDB

USE MASTER

GO

ALTER DATABASE [TEAMXX\_LocalMasterDataDB]

SET ENCRYPTION ON

GO

**Part 2: CLR assemblies**

1. Review the vulnerability assessment results. The application uses CLR which has been approved by security. Using the vulnerability assessment mark the CLR result as acceptable by clicking on “**Approve as Baseline”.**

**HINT**: You will need to click on the CLR result to accept the approve as baseline option.

**Part 3: Password**

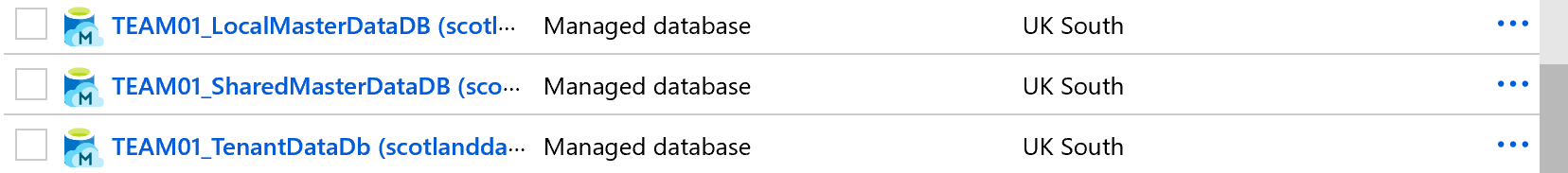
1. Review the vulnerability assessment results for “Login password should not be easily guessed”. Use the remediation script to change the password to a more secure password of your choice.
2. Re-run the vulnerability assessment ensuring the “Login password should not be easily guessed” has been removed.
3. Update the transaction reporting application with the new password.

**TASK 1: DISCOVER**

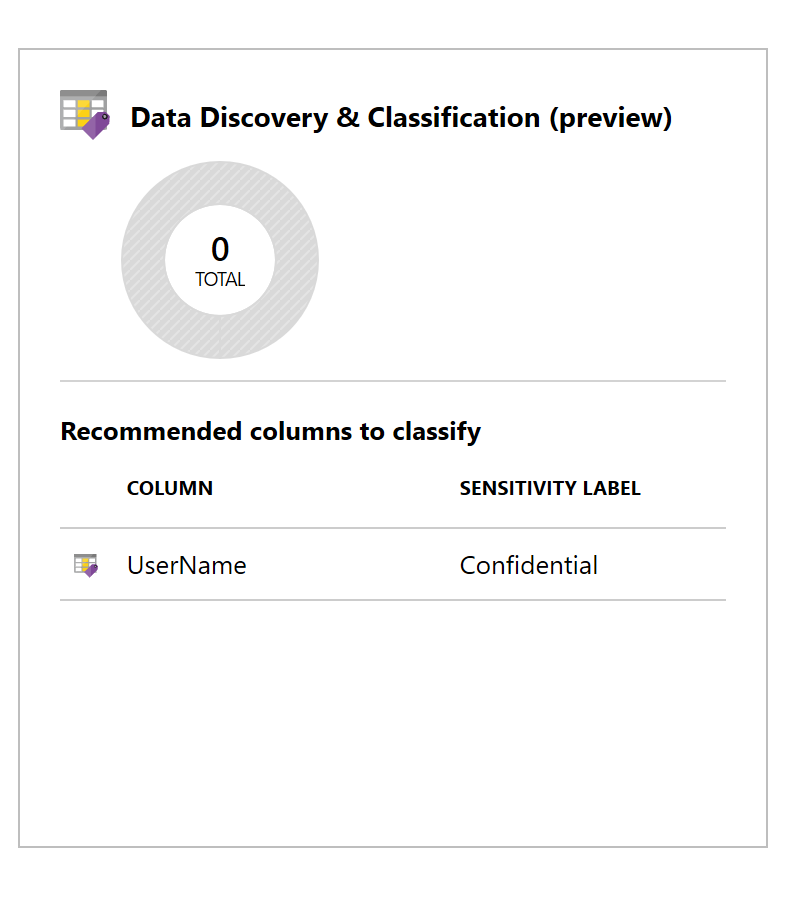
Data discovery & classification (currently in preview) provides advanced capabilities built into Azure SQL Database for **discovering**, **classifying**, **labeling** & **protecting** the sensitive data in your databases. It comes with a built-in set of sensitivity labels and a built-in set of information types and discovery logic.

In this lab, we will be exploring how to implement discovery, classification and labelling data in our managed instance database we migrated in the previous section.

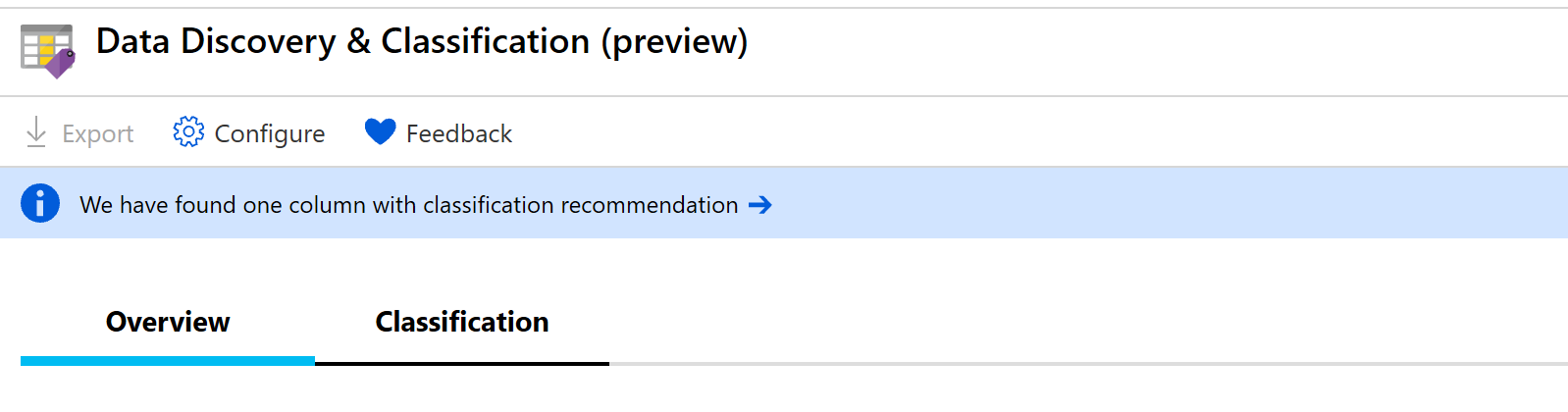
1. Go to the Azure portal
2. Navigate to the Resource group created for the Hack “SCOTLANDHACK”
3. Search for the Managed Instance Databases you have migrated in the previous exercise.



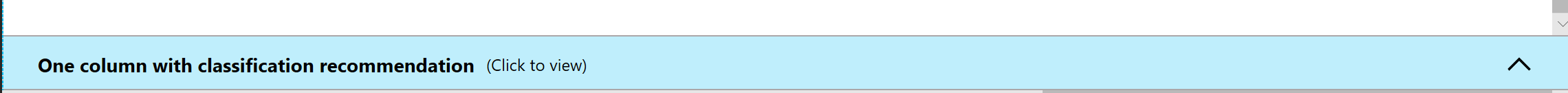
1. Select the TeamXX\_LocalMasterDataDB database by clicking on it.
2. Navigate to **Advanced Data Security** under the Security heading in your SQL Database pane. Click to enable advanced data security, and then click on the **Data discovery & classification (preview)** card as shown below:



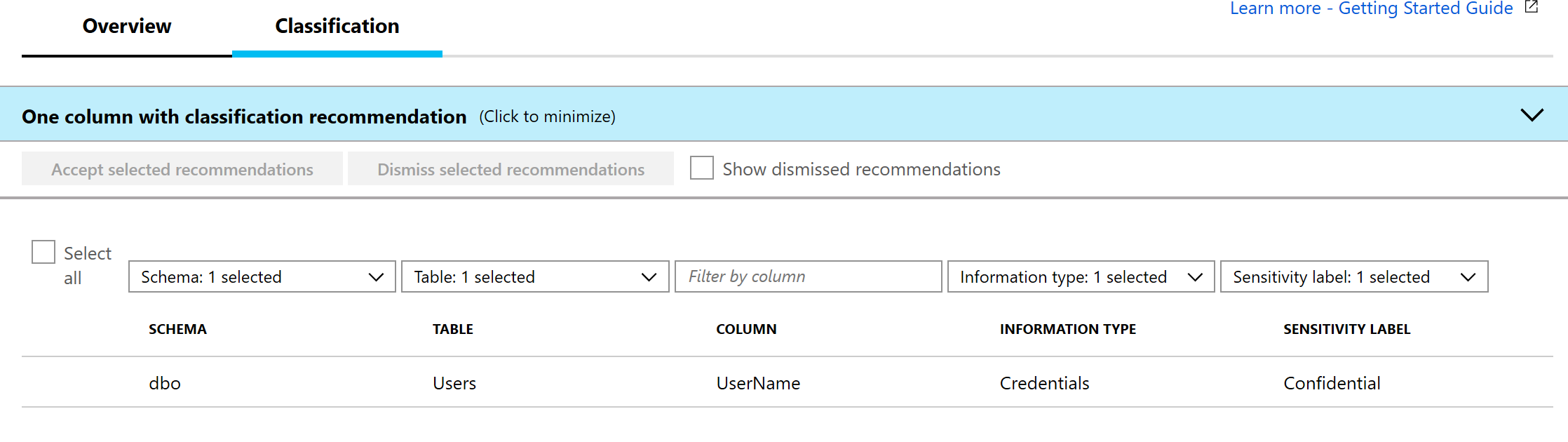
1. The **Overview** tab includes a summary of the current classification state of the database, including a detailed list of all classified columns, which you can also filter to view only specific schema parts, information types and labels.
2. To begin classifying your data, click on the **Classification tab** at the top of the window.



1. The classification engine scans your database for columns containing potentially sensitive data and provides a list of **recommended column classifications**. To view and apply classification recommendations click on the classification recommendation as shown below:



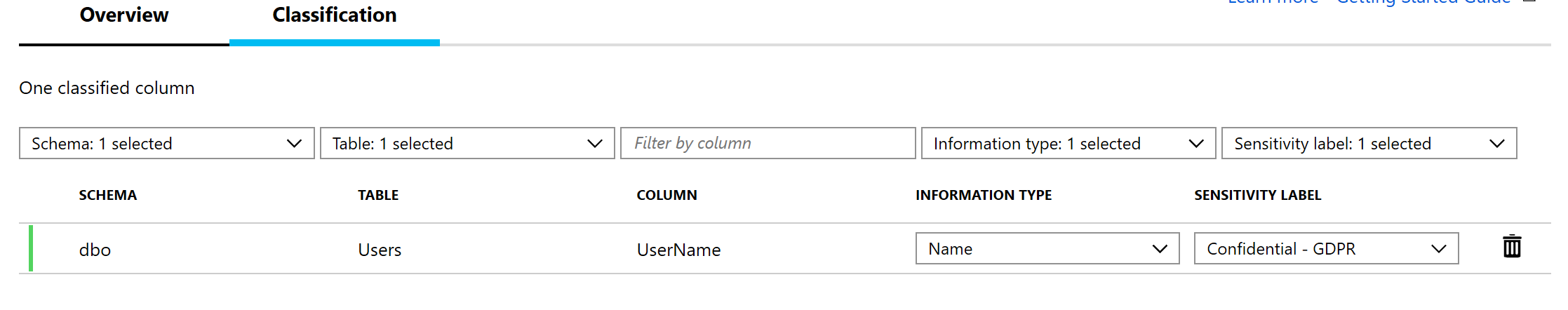
1. To view the list of recommended column classifications, click on the recommendation panel at the bottom of the window:



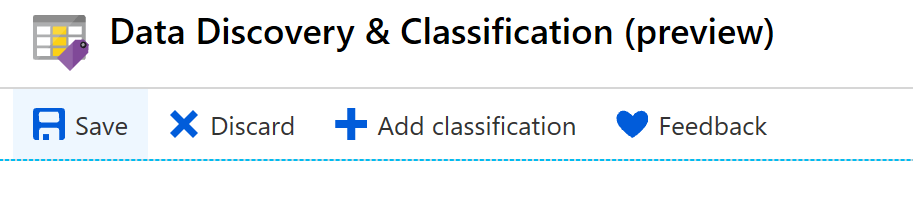
1. Review the list of recommendations.

Accept the recommendation and modify the information type and sensitivity. To accept the recommendation for the username column in the Users table, check the checkbox in the left column of the Users row and select **Accept selected recommendations.**

1. Once the recommendation has been accepted it will show as a classified column. Change the **Information type** to **Name** and the **Sensitivity** to **Confidential – GDPR** as shown below:



1. Click save button to commit your changes.



1. Perform steps 3-12 for the following additional classifications:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Database** | **Table** | **Column** | **Information Type** | **Sensitivity Label** |
| TeamXX\_LocalMasterDataDB | Users | Email | Contact Info | Confidential |
| TeamXX\_TenantDataDB | Users | Username | Name | Confidential – GDPR |
| TeamXX\_TenantDataDB | UserTransactions | TranAmount | Financial | Highly Confidential |

**Database Modernisation Workshop**: **Auditing**

**TASK 2: AUDIT**

Auditing in Managed Instance is done at the server level and stores .XEL files in Azure Blob storage. In this task, we will be auditing access to the sensitive data in [TEAMXX\_TenantDatabaseDB] and [TEAMXX\_LocalMasterDataDB] databases which we classified and label in the previous task.

1. Create an audit for your Managed Instance and define the target

-- Create the server audit.

-- Change the path to a path to Blob Storage Path for auditing

CREATE SERVER AUDIT [TEAMXX\_ScotlandDataAuditLogs]

TO URL (PATH ='https://scotlandauditlogs.blob.core.windows.net/auditlogs')

GO

1. Create a database audit specification that maps to the audit by running the following query in Management Studio

USE [TEAMXX\_TenantDataDb]

GO

-- Create the database audit specification.

CREATE DATABASE AUDIT SPECIFICATION TEAMXX\_Audit\_Data --change xx to team number

FOR SERVER AUDIT [TEAMXX\_ScotlandDataAuditLogs] --change to server audit name created in step 1

ADD (INSERT, UPDATE, DELETE, SELECT

ON SCHEMA::DBO BY dbo )

WITH (STATE = ON);

GO

1. Enable the Audit

-- Enable the server audit.

ALTER SERVER AUDIT [TEAMXX\_ScotlandDataAuditLogs]

WITH (STATE = ON);

GO

1. Check the audit logs are being saved in the Blob storage account.

**Data Modernisation Workshop: Dynamic Data Masking**

**TASK 3: PROTECT**

Dynamic data masking helps prevent unauthorized access to sensitive data by enabling customers to designate how much of the sensitive data to reveal with minimal impact on the application layer.

In this task we will mask the transaction amount from Peter who is a developer.

1. Connect to your Managed Instance from Management Studio.
2. Run the below script to mask the transaction amount in the UserTransactions table using the default masking function

ALTER TABLE [UserTransactions]

ALTER COLUMN [TranAmount] [decimal](18, 2) MASKED WITH (FUNCTION = 'DEFAULT()')

1. Check the table to ensure you are still able to see **ALL** the data

SELECT \* FROM UserTransactions

GO

1. Create a user for the developer and grant the read only access on the UserTransactions table

CREATE USER Peter WITHOUT LOGIN

GRANT SELECT ON dbo.UserTransactions TO Peter;

1. Run the query below to ensure we have been able to prevent the developer from accessing customers’ privacy information

EXECUTE AS USER = 'Peter';

SELECT \* FROM dbo.UserTransactions;

REVERT;

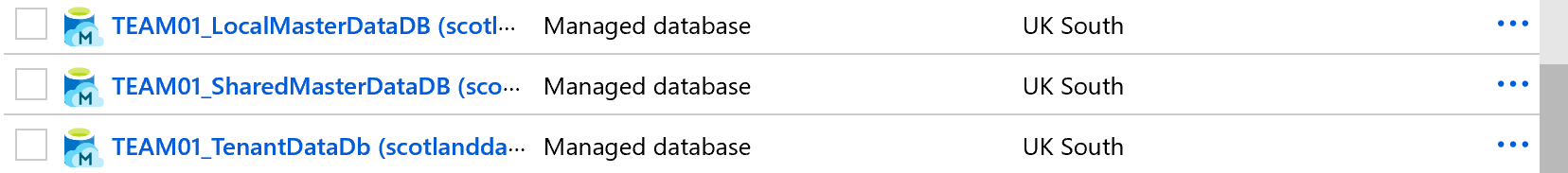
**Data Modernisation Workshop: Vulnerability Assessment**

**Task 4: DETECT**

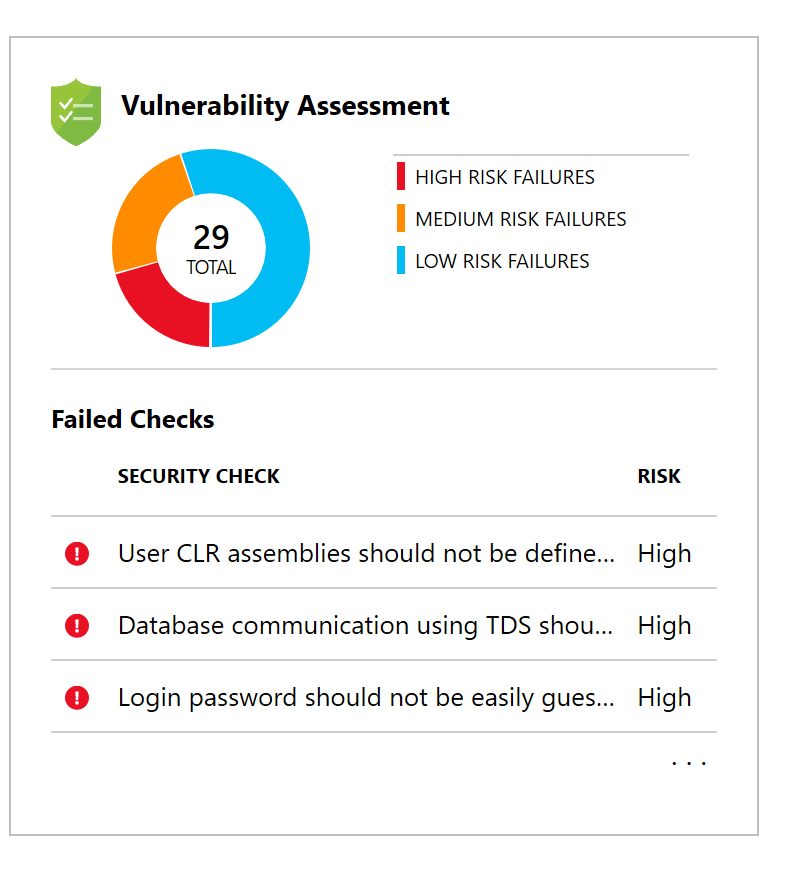
SQL Vulnerability Assessment (VA) is a service that provides visibility into your security state, and includes actionable steps to resolve security issues, and enhance your database security.

In this exercise we will be examining some of the risk security vulnerabilities and deviations from best practices.

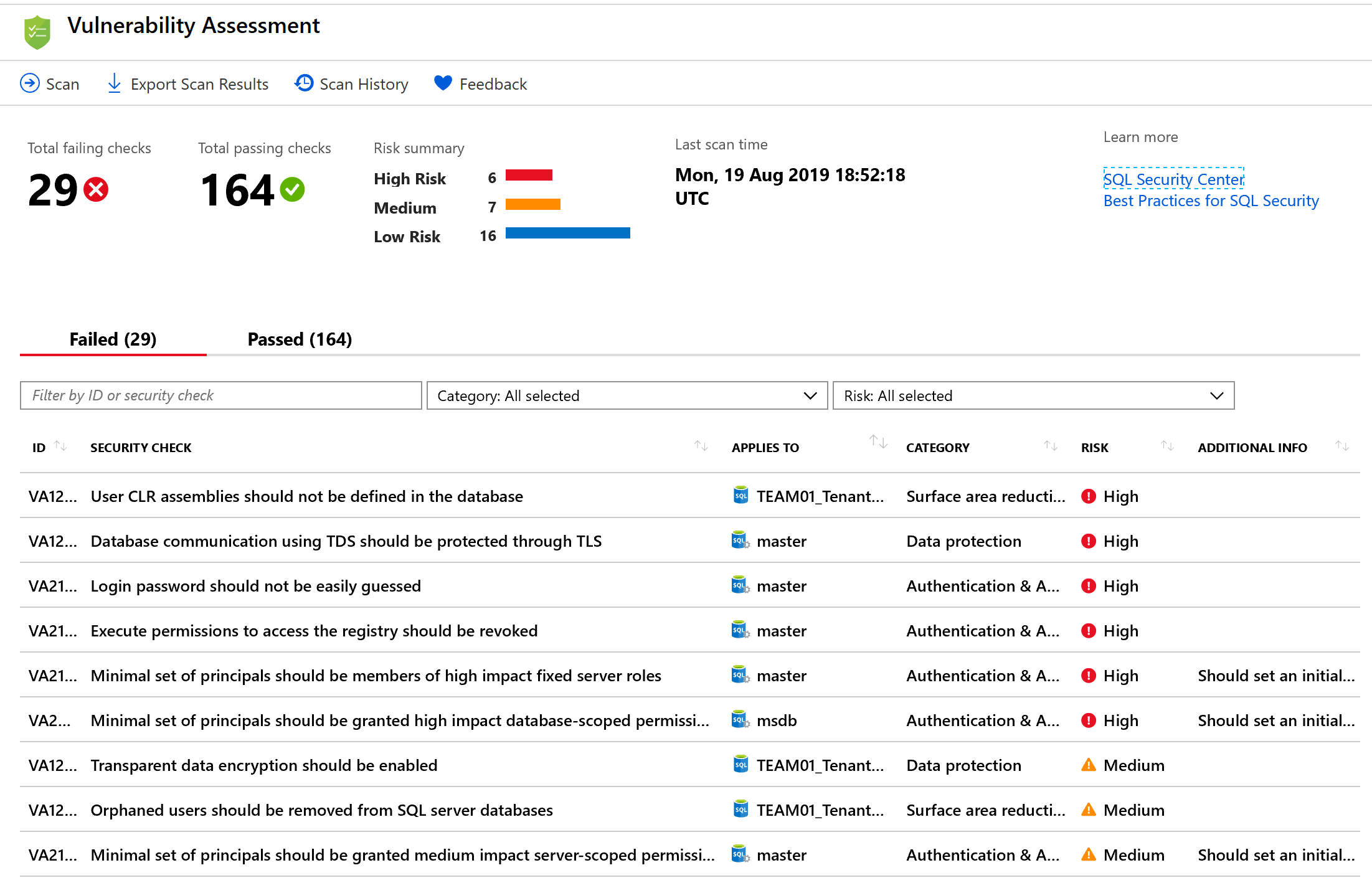
1. Go to the Azure portal
2. Navigate to the Resource group created for the Hack “SCOTLANDHACK”
3. Search for the Managed Instance Database you have migrated in the previous exercise.



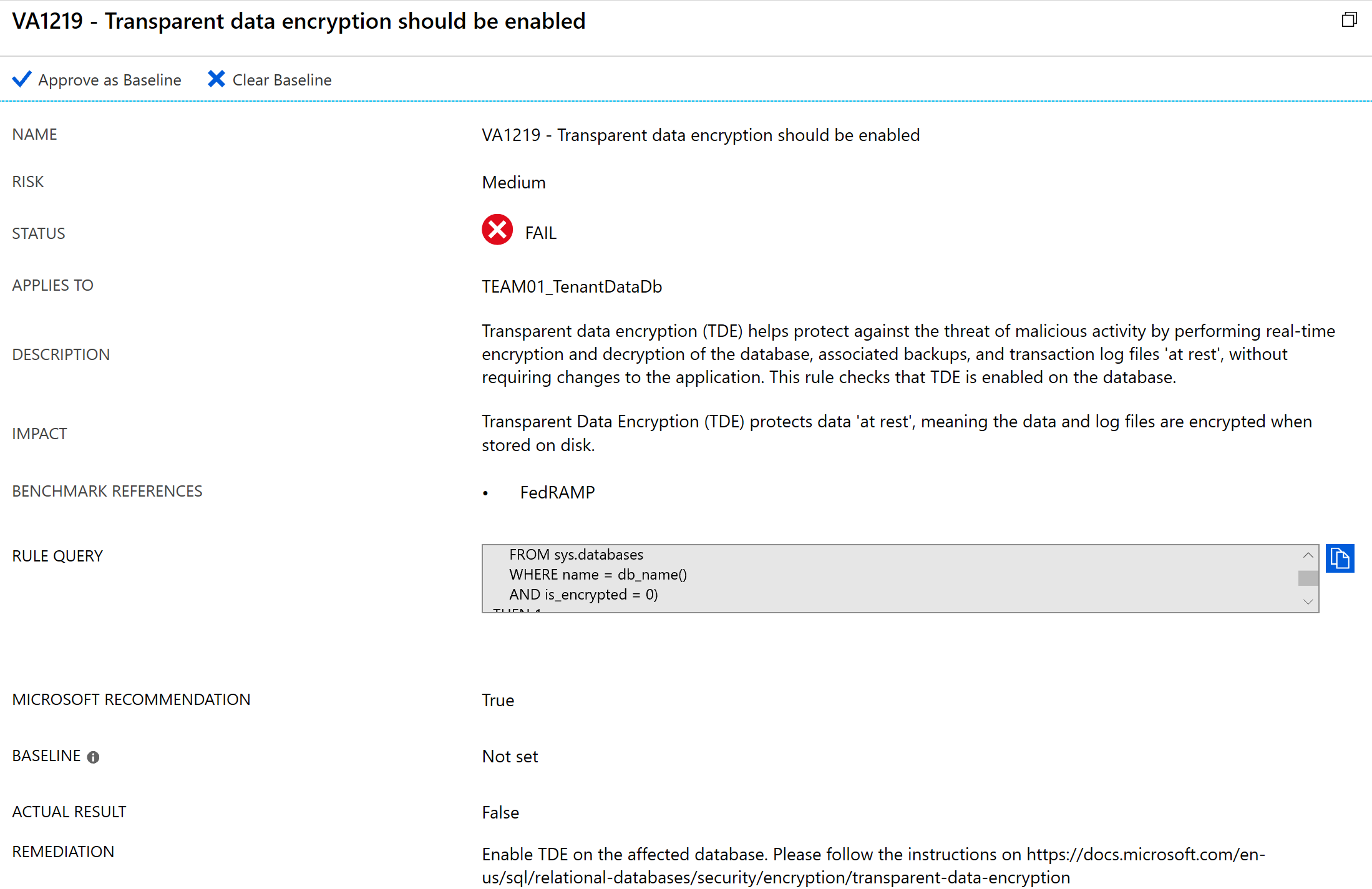
1. Select the TeamXX\_TenantDataDB database.
2. Navigate to **Advanced Data Security** under the Security heading in your SQL Database pane. Then click on the **Vulnerability assessment** card as shown below:



1. View the Vulnerability Assessment report. It would look similar to the below.



1. Click Description and impact for “**Transparent data encryption should be enabled**”



1. To enable TDE, connect to your Managed Instance via SQL Server Management Studio and execute the following query:

USE MASTER

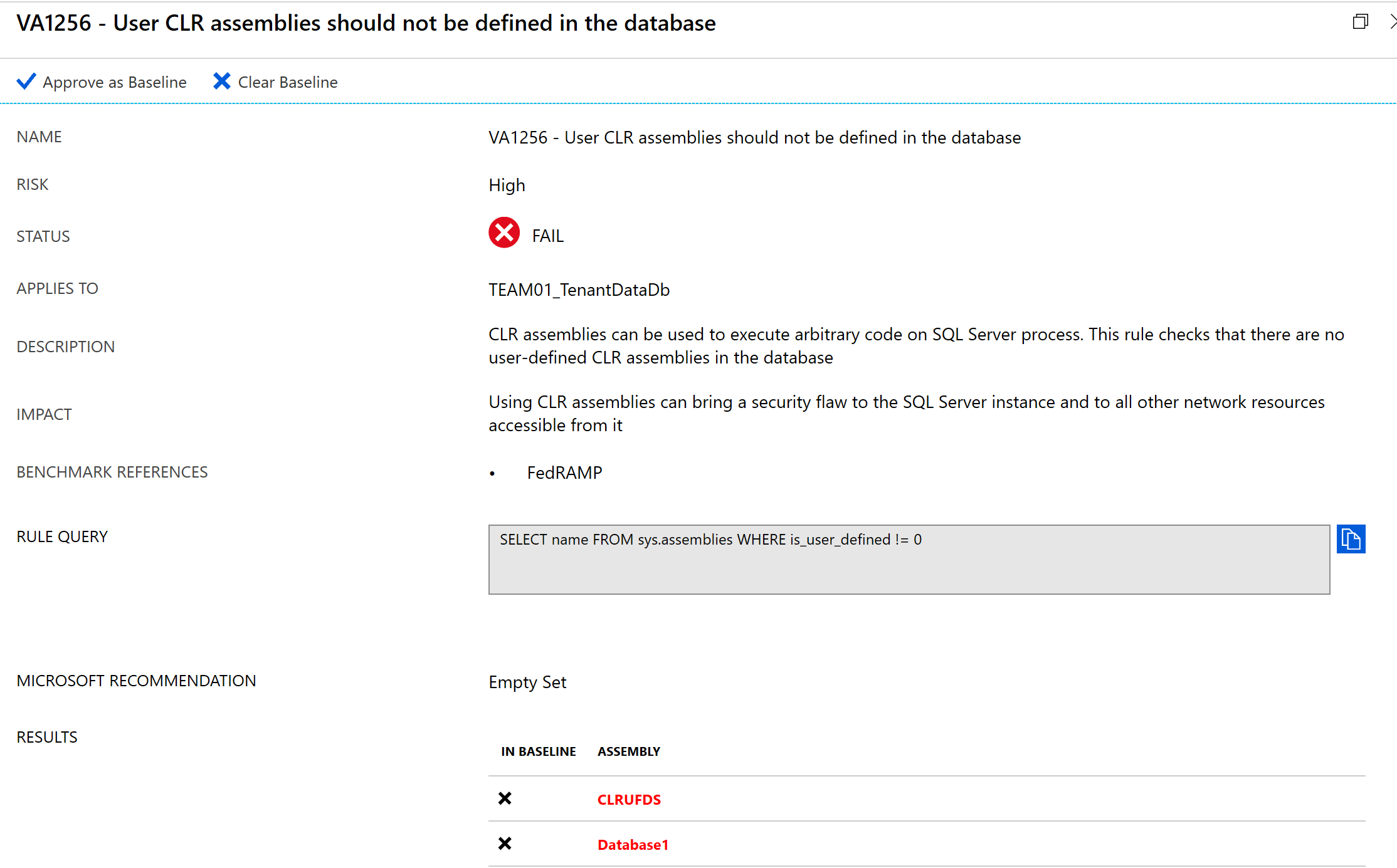
GO

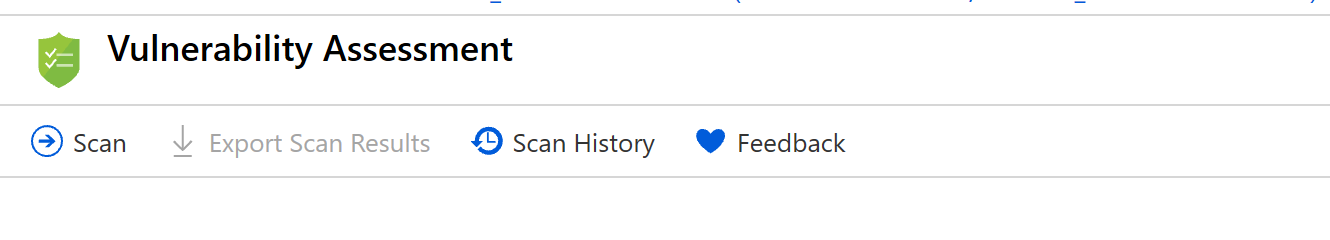
ALTER DATABASE [TEAMXX\_LocalMasterDataDB]

SET ENCRYPTION ON

GO

1. Enable TDE for the other databases TEAMXX\_SharedMasterDataDB and TEAMXX\_TenantDataDB in your managed instance.
2. Navigate back to the Vulnerability Assessment report in Azure Portal.
3. ToClick on security check - “**User CLR assemblies should not be defined in the database**” and review description and remediation.
4. Review the assessment results, in this case we will mark specific results as being an acceptable Baseline in our environment by clicking on “**Approve as Baseline**”.



1. Run the scan on the TEAMXX\_LocalMasterDataDB
2. The VA now reports *only* the security issues that deviate from your approved baseline state and the security issues which haven’t been addressed.