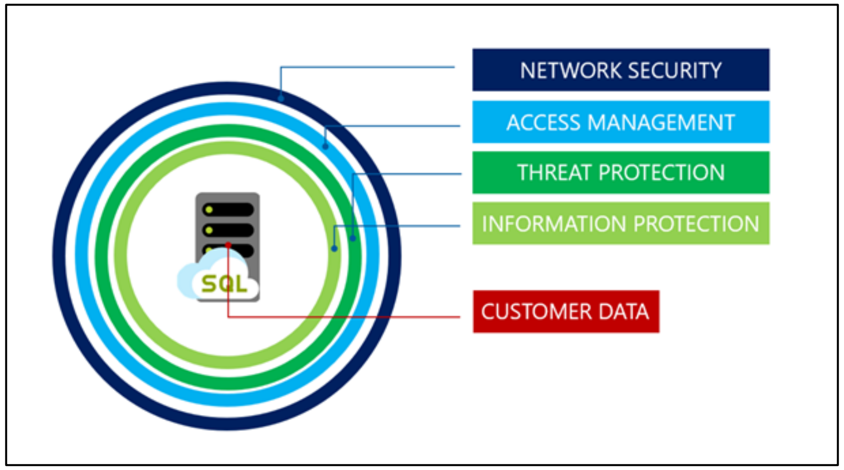
SQL Moderation Hack
Secure Your Data with Azure SQL DB Labs Step-by-step

V4.0

Table of Contents

1. Introduction

This hands-on lab will introduces you to the layered security model available when running databases in Azure. The activities within this hands-on lab will progress from the outer security layers that protect the perimeter of Azure SQL through to the inner layers that protect the information contained within the data.



Because SQL Managed Instance always runs in a private network the Network Security layer has already been implemented at the vNet level. Equally we have already defined and implemented Azure AD and SQL Server logins, roles and permissions so the Access Management tier has also been pre-built.

So this lab will focus on the Threat Protection, Information Protection and Customer Data layers of the security model and how these are implement in Azure SQL Managed Instance through:

- Review and configuring auditing within Azure SQL Managed Instance
- Using Data Discovery & Classification
- Azure Defender for SQL
    - Vulnerability Assessment
    - Advanced Threat Protection
- Information protection & encryption
    - Dynamic Data Masking
    - Always Encrypted

2. Azure SQL Database & Team VM Login Details

All the labs run against the TEAMXX_TenantDataDb that you migrated earlier using either SQL Server Management Studio or the Azure Portal.

Your Win10 VM (vm-TEAMXX) login credentials are also a member of SQL Server sysadmin role.

Username	**localhost\DemoUser**
Password	**Demo@pass1234567**

The Azure Portal credentials are those that your proctor will supply.

3. LAB 1: Auditing for Azure SQL Managed Instance

Auditing

For Azure SQL Managed Instance auditing is enabled at the server level and tracks events at both the server and database level (depending on your audit configuration).

The events are then written to a centralized log stored outside of the Managed Instance environment.

The log can be stored in either:

- A file in Azure Storage Account
- Log Analytics Workspace (a special centralized log storage location for logs from all Azure services)
- Azure Event Hub (an Azure native message queue where streaming messages can be consumed in real-time)
- Or any combination of the 3

More details on auditing in SQL Managed Instance can be found here:

SQL Managed Instance auditing - Azure SQL Managed Instance | Microsoft Docs

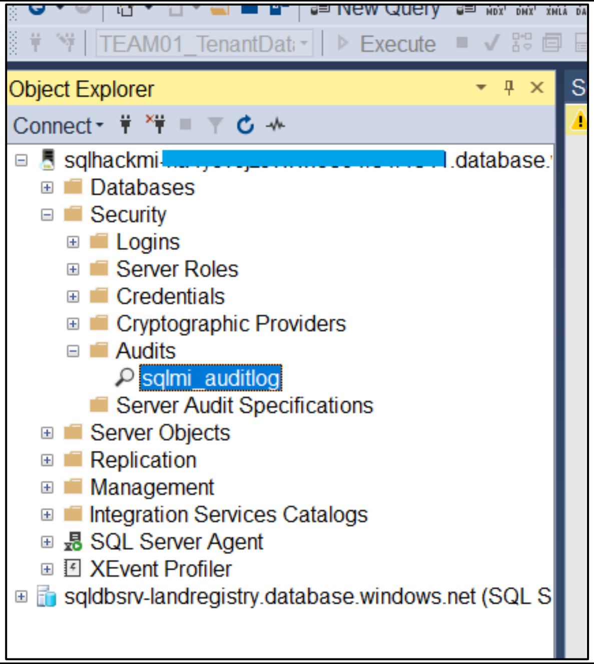
Confirm Auditing is enabled at the Logical SQL Server Level

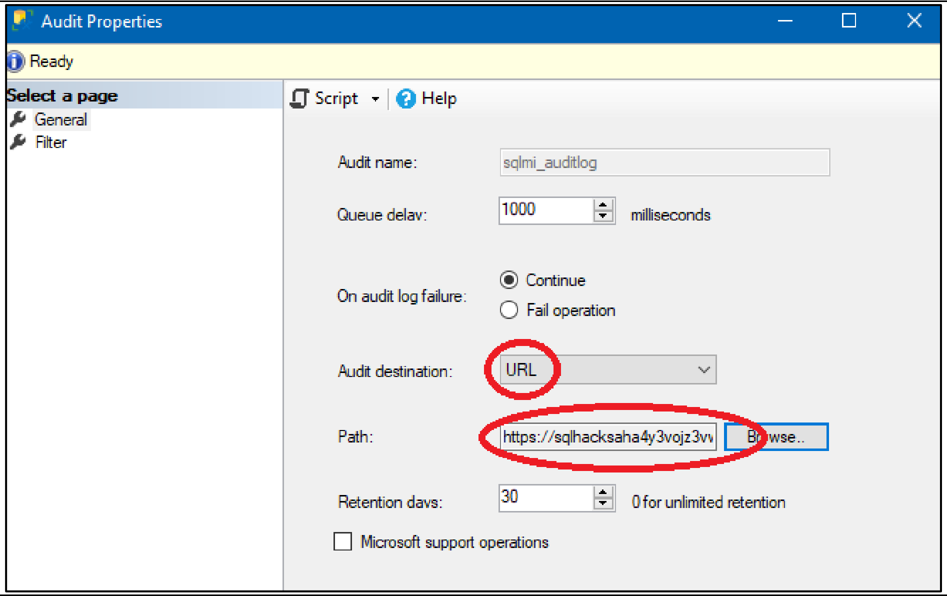
Because auditing is switched on at the server level we have already done this within the lab environment and configured it to write to a log file held in an Azure Storage account.

For more information on setting up auditing on SQL Managed Instance see this documentation:

SQL Managed Instance auditing - Azure SQL Managed Instance | Microsoft Docs

But for now let's confirm that auditing is enabled.

Narrative	Screenshot/Code	Notes
1. On your team Win10 VM open SQL Server Management Studio, connect to the shared SQL Managed Instance and expand the Security\Audits folder	 sqlhackmi-[redacted].database. Databases Security Logins Server Roles Credentials Cryptographic Providers Audits sqlmi_auditlog Server Audit Specifications Server Objects Replication Management Integration Services Catalogs SQL Server Agent XEvent Profiler sqldbsrv-landregistry.database.windows.net (SQL S	

2. Right-click on the **sqlmi_auditlog** and look at it's properties. Notice that the Audit Destination is set to URL and the Path points to our shared Storage Account.		

Audit name: sqlmi_auditlog

Queue delay: 1000 milliseconds

On audit log failure: [✓ Continue] Fail operation

Audit destination: URL

Path: https://sqlhacksaha4y3vojz3vv

Retention days: 30 0 for unlimited retention

☐ Microsoft support operations

Create an Audit Specification for tracking queries against specific tables

Although a physical Audit Log has been created and enabled, to actually capture events we need to create Audit Specifications.

Audit Specifications define what actions and operations are audited and at what level. It is quite normal to have separate Audit Specifications for each database as well as at the server level depending on what activities you want to track. All these specifications will write to the same Audit Log.

Narrative	Screenshot/Code	Notes
1. On your team Win10 VM open SQL Server Management Studio, connect to the shared SQL Managed Instance and open a new query window to your **TEAMXX_TenantDataDb**		

2. Run this query to create an Audit Spec that will monitor all SELECT queries run against all tables in the SalesLT schema:	`--RUN AGAINST YOUR TEAM'S [TenantDataDb]:`
	`USE [TEAMXX_TenantDataDb];`
	`CREATE DATABASE AUDIT SPECIFICATION audit_sensitve_data`
	`FOR SERVER AUDIT [sqlmi_auditlog]`
	`ADD (SELECT ON Schema::SalesLT BY public)`
	`WITH (STATE = ON)`
		For more information on Audit Specification see this documentation: Create Server Audit & Server Audit Specification - SQL Server | Microsoft Docs

We'll return to the Audit Logs later to see what it has captured for us.

4. LAB 2: Data Discovery & Classification

Data Discovery & Classification

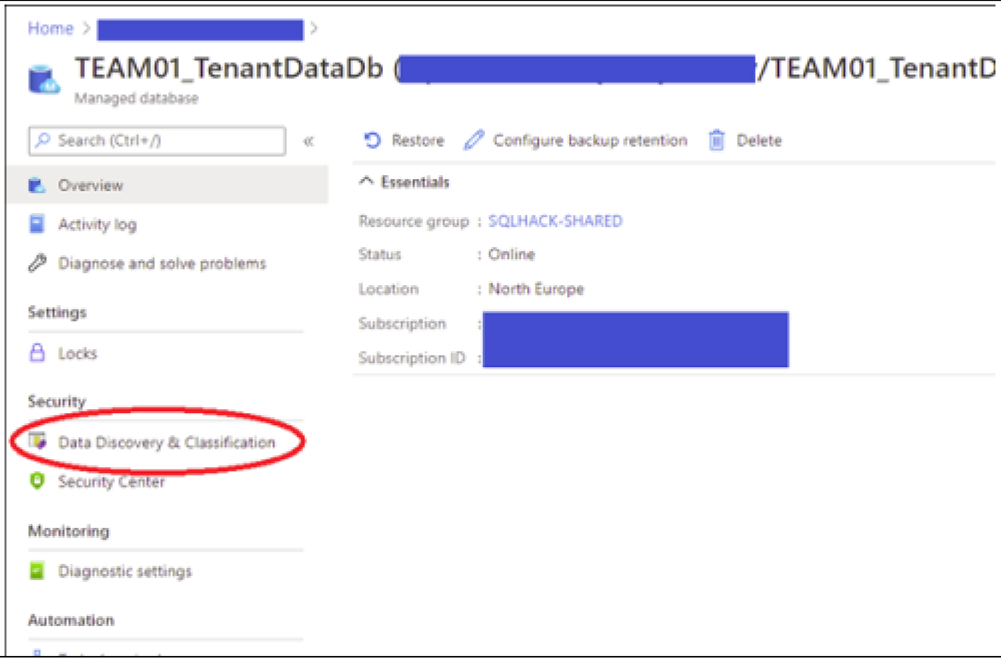
Data Discovery & Classification is a built-in capability for discovering, classifying, labelling and protecting sensitive data in databases. It can be used to support many use cases including financial, healthcare, personally identifiable (PII) data and help meet data privacy standards and regulatory compliance.

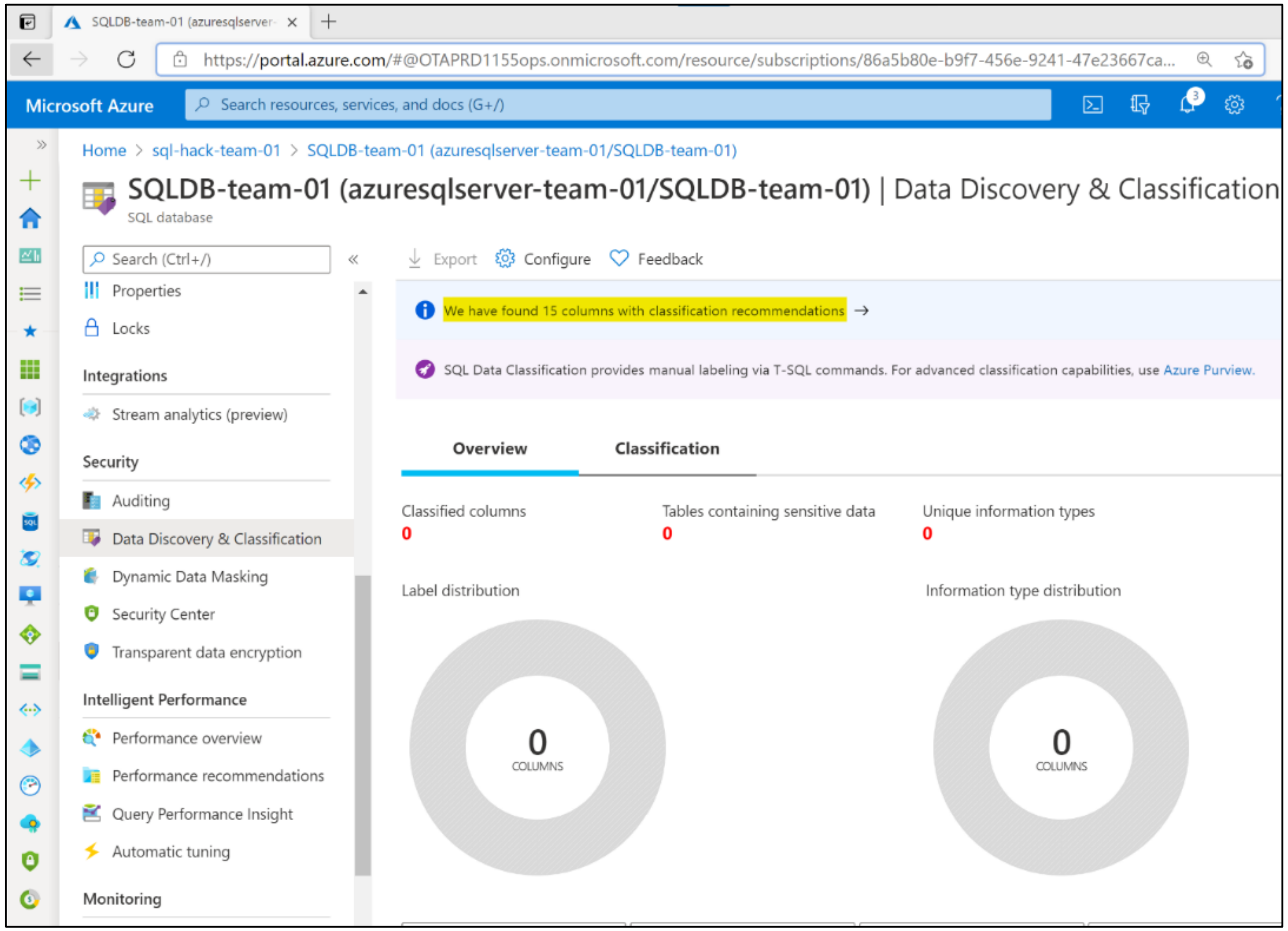
More information on Data Discovery & Classification can be found here:

https://docs.microsoft.com/en-us/azure/azure-sql/database/data-discovery-and-classification-overview

Viewing Data Classification Recommendations

Whenever a database is deployed or schema changes are made to an existing database, the Data Discovery & Classification engine automatically performs a scan to identify columns that may potentially contain sensitive data.

Narrative	Screenshot/Code	Notes
1. Within the Azure Portal navigate to the shared Azure SQL Managed Instance screen. Scroll down to the list of databases and click on your teams **TEAMXX_TenantDataDb** database.		
2. On the blade on the left, under the **Security** section click "**Data Discovery & Classification**"	 Home > [redacted] > TEAM01_TenantDataDb ([redacted]/TEAM01_TenantD Managed database. Resource group : SQLHACK-SHARED; Status : Online; Location : North Europe; Subscription : [redacted]; Subscription ID :	

The Data Discovery and Classification **Overview** shows that no data classifications have been made but based on the automatic classification scan there are a number of potential data classification recommendations as shown at the top of the report: 3. Click the blue information bar (highlighted in yellow) to view the data classification recommendations	

SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) | Data Discovery & Classification

SQL database

We have found 15 columns with classification recommendations →

SQL Data Classification provides manual labeling via T-SQL commands. For advanced classification capabilities, use Azure Purview.

Overview | Classification

Classified columns: 0

Tables containing sensitive data: 0

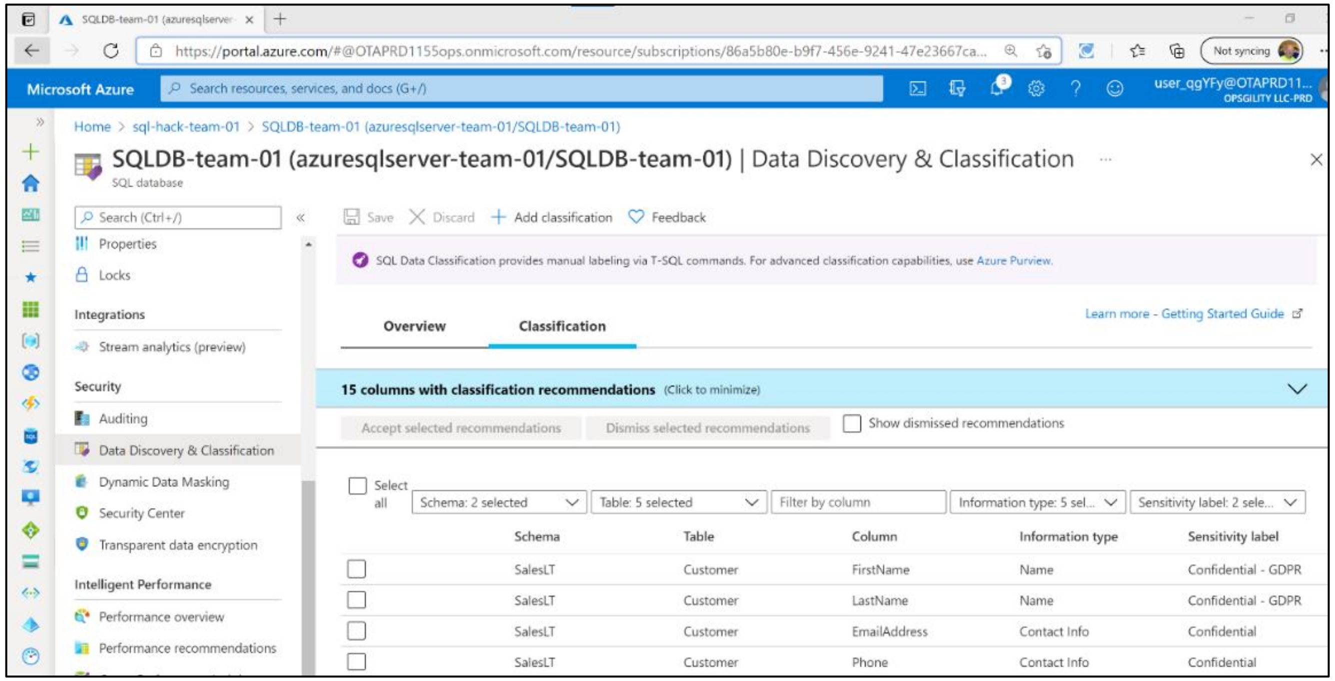
Unique information types: 0

Label distribution: 0 COLUMNS

Information type distribution: 0 COLUMNS

The recommendations show the name of the schema, table and column with intelligent information type classification and sensitivity recommendations.

As can be seen the **Customer** table in the **SalesLT** schema contains the columns **FirstName** and **LastName**. The initial data classification scan has identified that the **Information type** of these columns from a data classification perspective is **Name** and the **Sensitivity Label** for these columns is recommended to be **Confidential – GDPR**.


SQLDB-team-01 (azuresqlserver
https://portal.azure.com/#@OTAPRD1155ops.onmicrosoft.com/resource/subscriptions/86a5b80e-b9f7-456e-9241-47e23667ca...
Not syncing
Microsoft Azure
Search resources, services, and docs (G+/)
user_qgYFy@OTAPRD11...
OPSGILITY LLC-PRD
Home > sql-hack-team-01 > SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01)
SQL database
Search (Ctrl+/)
Properties
Locks
Integrations
Stream analytics (preview)
Security
Auditing
Data Discovery & Classification
Dynamic Data Masking
Security Center
Transparent data encryption
Intelligent Performance
Performance overview
Performance recommendations
Save
Discard
Add classification
Feedback
Learn more - Getting Started Guide
Overview
Classification
Accept selected recommendations
Dismiss selected recommendations
Show dismissed recommendations
Select all
Schema: 2 selected
Table: 5 selected
Filter by column
Information type: 5 sel...
Sensitivity label: 2 sele...

SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) | Data Discovery & Classification

SQL Data Classification provides manual labeling via T-SQL commands. For advanced classification capabilities, use Azure Purview.

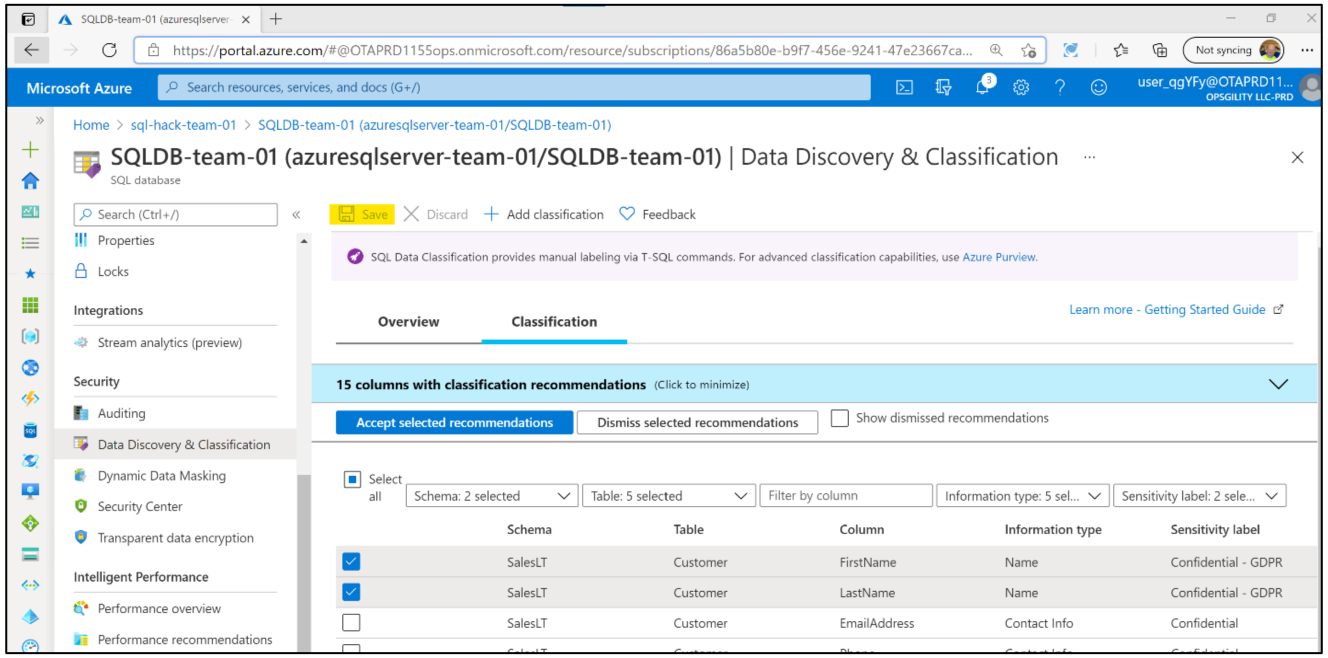
15 columns with classification recommendations (Click to minimize)

Schema	Table	Column	Information type	Sensitivity label
SalesLT	Customer	FirstName	Name	Confidential - GDPR
SalesLT	Customer	LastName	Name	Confidential - GDPR
SalesLT	Customer	EmailAddress	Contact Info	Confidential
SalesLT	Customer	Phone	Contact Info	Confidential

4. Select the **FirstName** and **LastName** classification recommendations by selecting the recommendation rows, click **Accept selected recommendations** and then click **Save**.

5. Click the **Overview** tab on the Data Discovery & Classification report to look at the saved data classifications.

There are now two columns classified from the Customer table with the information type of **Name** and the sensitivity label **Confidential – GDPR**.


SQLDB-team-01 (azuresqlserver
https://portal.azure.com/#@OTAPRD1155ops.onmicrosoft.com/resource/subscriptions/86a5b80e-b9f7-456e-9241-47e23667ca...
Not syncing
Microsoft Azure
Search resources, services, and docs (G+/)
user_qgYFy@OTAPRD11...
OPSGILITY LLC-PRD
Home > sql-hack-team-01 > SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01)

SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) | Data Discovery & Classification

SQL database

Search (Ctrl+/)
Properties
Locks
Integrations
Stream analytics (preview)
Security
Auditing
Data Discovery & Classification
Dynamic Data Masking
Security Center
Transparent data encryption
Intelligent Performance
Performance overview
Performance recommendations
Save
Discard
Add classification
Feedback

SQL Data Classification provides manual labeling via T-SQL commands. For advanced classification capabilities, use Azure Purview.

Learn more - Getting Started Guide
Overview
Classification

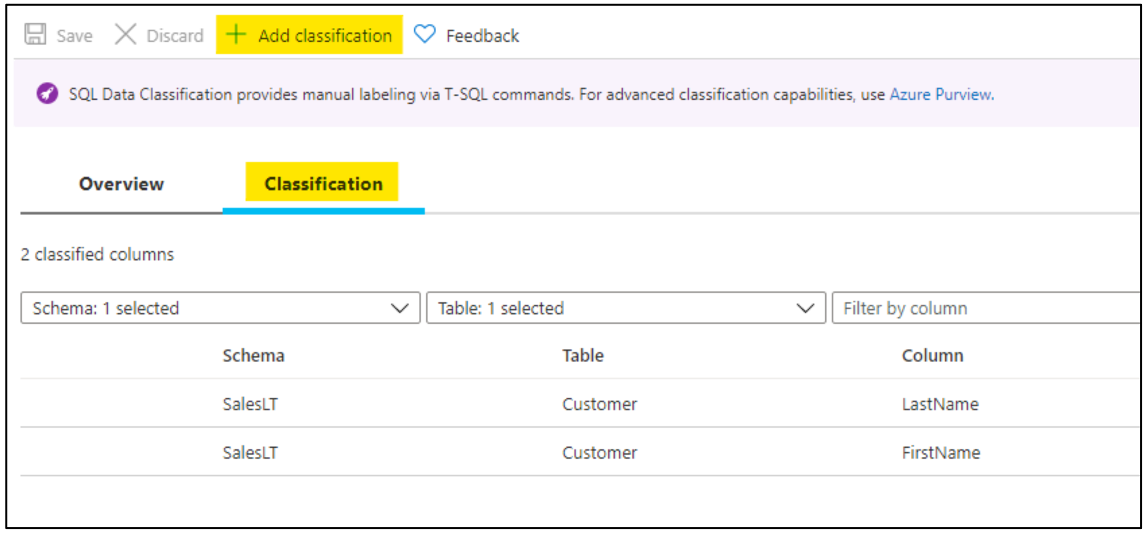
15 columns with classification recommendations (Click to minimize)

Accept selected recommendations
Dismiss selected recommendations
Show dismissed recommendations
Select all
Schema: 2 selected
Table: 5 selected
Filter by column
Information type: 5 sel...
Sensitivity label: 2 sele...

	Schema	Table	Column	Information type	Sensitivity label
✓	SalesLT	Customer	FirstName	Name	Confidential - GDPR
✓	SalesLT	Customer	LastName	Name	Confidential - GDPR
☐	SalesLT	Customer	EmailAddress	Contact Info	Confidential

Now let's add a custom data classification which is not based on the auto recommendations.

6. Switch back to the **Classification** tab at the top of the report click "**+ Add classification**".



SQL Data Classification provides manual labeling via T-SQL commands. For advanced classification capabilities, use Azure Purview.

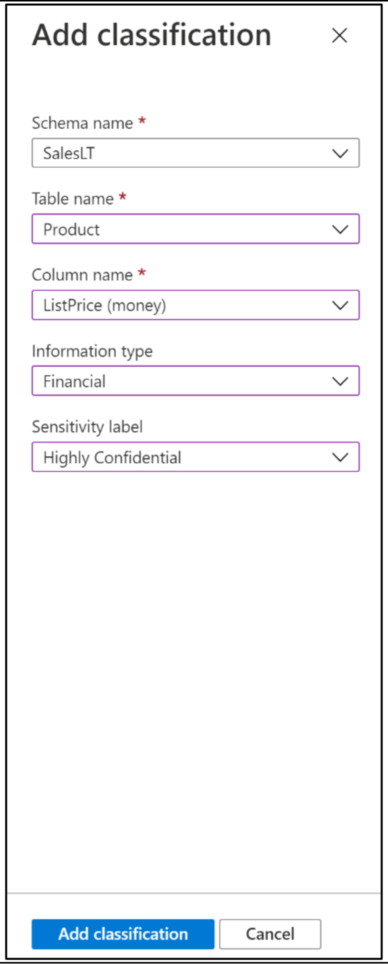
2 classified columns

Schema	Table	Column
SalesLT	Customer	LastName
SalesLT	Customer	FirstName

7. On the **Add Classification** blade on the far right of the screen set the following values and then click **Add Classification** and then **Save** to save your new classification.

8. Click the **Overview** tab to look at the saved data classifications.

Schema name:	**SalesLT**
Table name:	**Product**
Column name:	**ListPrice**
Information type:	**Financial**
Sensitivity Label:	**Highly Confidential**
Click	**Add Classification**
Click	**Save**


Add classification

Schema name * SalesLT

Table name * Product

Column name * ListPrice (money)

Information type Financial

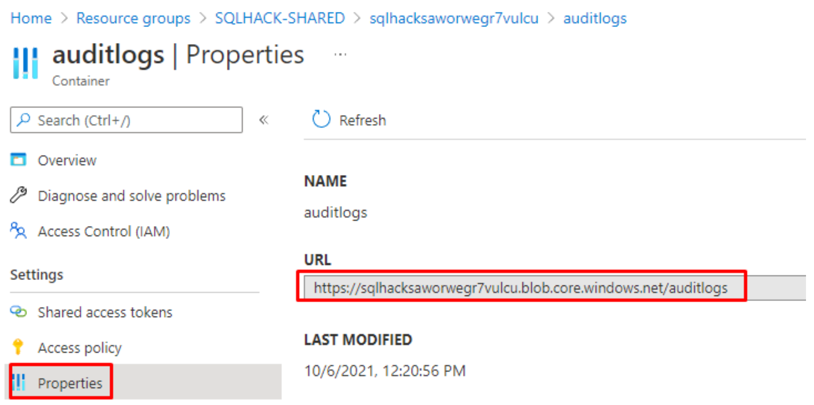
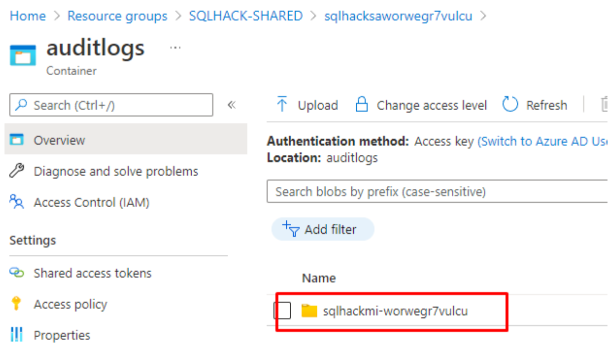
Sensitivity label Highly Confidential

9. Open SQL Server Management Studio, connect to the shared SQL Managed Instance and open a new TSQL query window connected to your **TEAMXX_TenantDataDb** database 10. Run the SELECT statements opposite against your **TEAMXX_TenantDataDb** database.	`-- 1 Data Discovery & Classification` `SELECT` `c.FirstName` `,c.LastName` `,c.*` `FROM SalesLT.Customer c;` `SELECT` `p.ListPrice` `FROM SalesLT.Product p;`	
Nothing out of the ordinary happens - two simple result sets should be returned containing the FirstName, LastName and ListPrice columns.	***REMEMBER: Data Discovery and Classification is not a security mechanism – it's a data tagging and management tool.***	
Now let's see how classifying columns is actually very useful when used in conjunction with the SQL Auditing that we configured earlier.		

Query the Audit Log Directly using TSQL

Although we can view audit logs though Management Studio, it is also possible to query them directly using TSQL and a system function called `sys.fn_get_audit_file` to see what has been captured.

Narrative	Screenshot/Code	Notes

1. Open a new query window in SQL Server Management Studio on the team VM. Copy the SQL statement opossite and paste it into the query window.
2. Update the script with the storage container URL and folder of the audit logs
To get the URL of the storage account and the folder to use in the script,
⇨ In Azure portal, in the SQLHACK-SHARED resource group, search and click on the storage account resource. In the left blade, click on containers, then click on “auditlogs” container.
On the left, click on “Properties” and copy the URL property value. Past it in the TSQL script.
Also, copy the name of the folder present in the container and append it to the URL in the script.
	`--Query the SQL Audit log stored in blob storage`
	`SELECT * FROM sys.fn_get_audit_file`
	`('<<URL FOR THE STORAGE ACCOUNT auditlogs...sqlmi_auditlog FOLDER>>',default,default)`
	`WHERE database_name = 'TEAMxx_TenantDataDb'`
	Home > Resource groups > SQLHACK-SHARED > sqlhacksaworwegr7vulcu > auditlogs
	auditlogs | Properties
	Container
	NAME: auditlogs
	URL: https://sqlhacksaworwegr7vulcu.blob.core.windows.net/auditlogs
	LAST MODIFIED: 10/6/2021, 12:20:56 PM
	 Search (Ctrl+/)
	Refresh
	Overview
	Diagnose and solve problems
	Access Control (IAM)
	Settings
	Shared access tokens
	Access policy
	Properties
	Home > Resource groups > SQLHACK-SHARED > sqlhacksaworwegr7vulcu >
	auditlogs
	Container
	Authentication method: Access key (Switch to Azure AD Us
	Location: auditlogs
	Name
	sqlhackmi-worwegr7vulcu
	 Search (Ctrl+/)
	Upload
	Change access level
	Refresh
	Overview
	Diagnose and solve problems
	Access Control (IAM)
	Settings
	Shared access tokens
	Access policy
	Properties
	Search blobs by prefix (case-sensitive)
	Add filter	

a. Once the query has been modified and executed, review the columns: • statement • data_sensitivity_information	

5. LAB 3 Part 1: Azure Defender for SQL – Vulnerability Assessment

When provisioning an Azure SQL Managed Instance or an Azure SQL Database logical server there is the option to enable the security feature Azure Defender for SQL.

This security feature offers two security components:

- Vulnerability Assessments
- Advanced Threat Protection

This first part of the lab will focus on Vulnerability Assessments, Part 2 will deal with Advanced Threat Protection.

Vulnerability Assessment

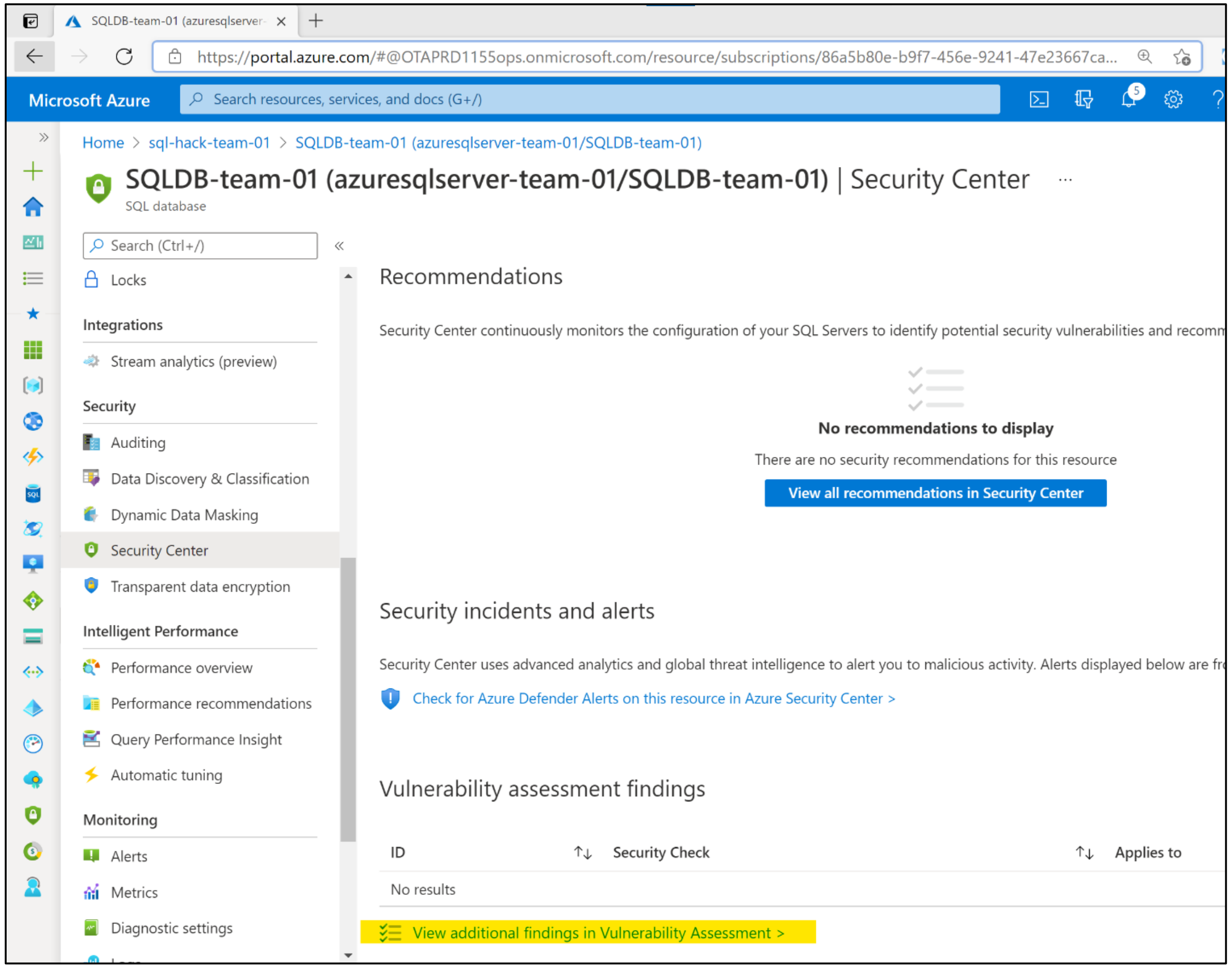
A Vulnerability Assessment is an output position (or report) from a vulnerability scan.

A Vulnerability Assessment scan is the application of SQL Server best practices based on a rules engine, the goal being to improve the security posture of your Azure SQL Managed Instance or Azure SQL Database. The first scan will produce the initial vulnerability scan baseline. The first scan happens automatically once a database is deployed.

More details on Azure SQL vulnerability assessments can be found here:

https://docs.microsoft.com/en-us/azure/azure-sql/database/sql-vulnerability-assessment

Narrative	Screenshot/Code	Notes
1. In the Azure portal navigate to the shared SQL Managed Instance.		
2. Scroll down the Overview screen until you see the list of databases and click on your **TEAMXX_TenantDataDB** database.		

3. In the **TEAMXX_TenantDataDB** database screen. On the left hand blade click **Security Center** in the Security section 4. Scroll down the "Security center" screen to the bottom and click the "**View additional findings in Vulnerability Assessment >**" link	

SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) | Security Center

SQL database

Recommendations

Security Center continuously monitors the configuration of your SQL Servers to identify potential security vulnerabilities and recomm

No recommendations to display

There are no security recommendations for this resource

Security incidents and alerts

Security Center uses advanced analytics and global threat intelligence to alert you to malicious activity. Alerts displayed below are fr

Check for Azure Defender Alerts on this resource in Azure Security Center >

Vulnerability assessment findings

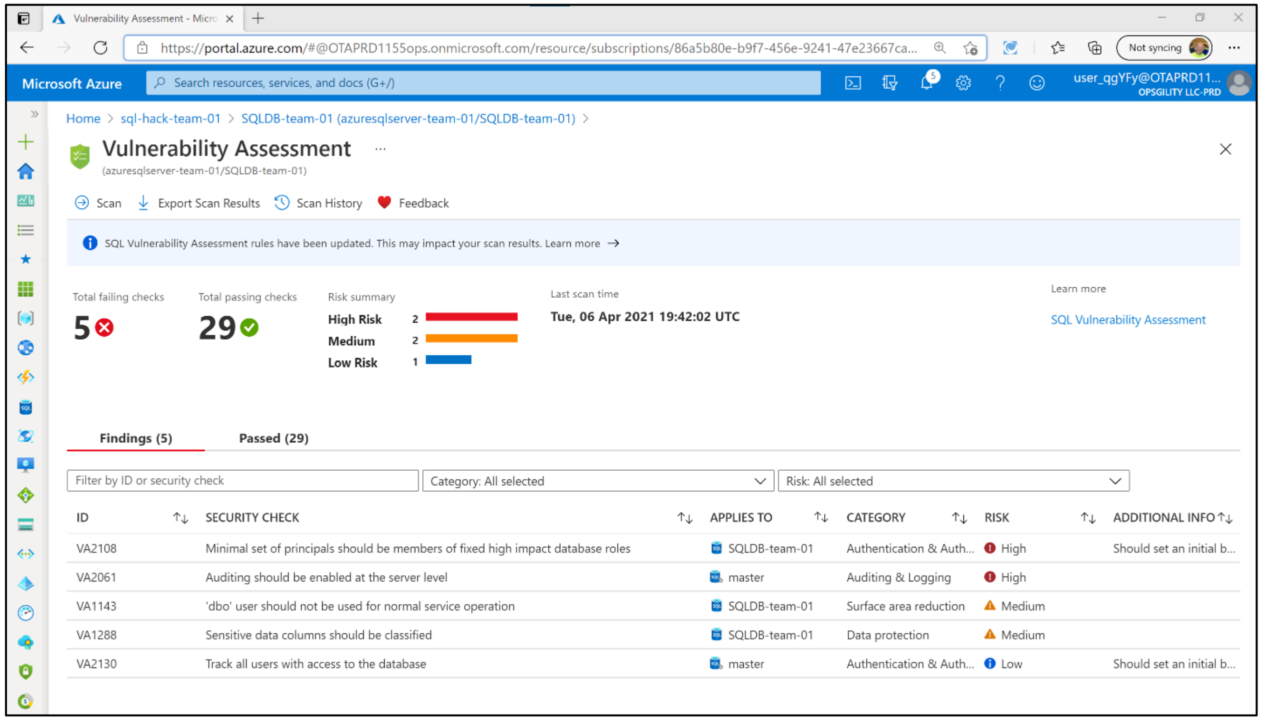
ID	Security Check	Applies to
No results		

View additional findings in Vulnerability Assessment >

Microsoft

The "Vulnerability Assessment" page can be used to run a scan, view scan history and will show the number of checks that have been passed and failed for the last scan with failed checks listed in the table below.

5. Run a scan if prompted to do so which should only take a few minutes.


Vulnerability Assessment - Micro
https://portal.azure.com/#@OTAPRD1155ops.onmicrosoft.com/resource/subscriptions/86a5b80e-b9f7-456e-9241-47e23667ca...
Not syncing
Microsoft Azure
Search resources, services, and docs (G+/)
user_qgYFy@OTAPRD11...
OPSGILITY LLC-PRD
Home > sql-hack-team-01 > SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) >
Scan
Export Scan Results
Scan History
Feedback

Vulnerability Assessment

(azuresqlserver-team-01/SQLDB-team-01)

SQL Vulnerability Assessment rules have been updated. This may impact your scan results. Learn more →

Total failing checks: 5

Total passing checks: 29

Risk summary:

High Risk	2
Medium	2
Low Risk	1

Last scan time: Tue, 06 Apr 2021 19:42:02 UTC

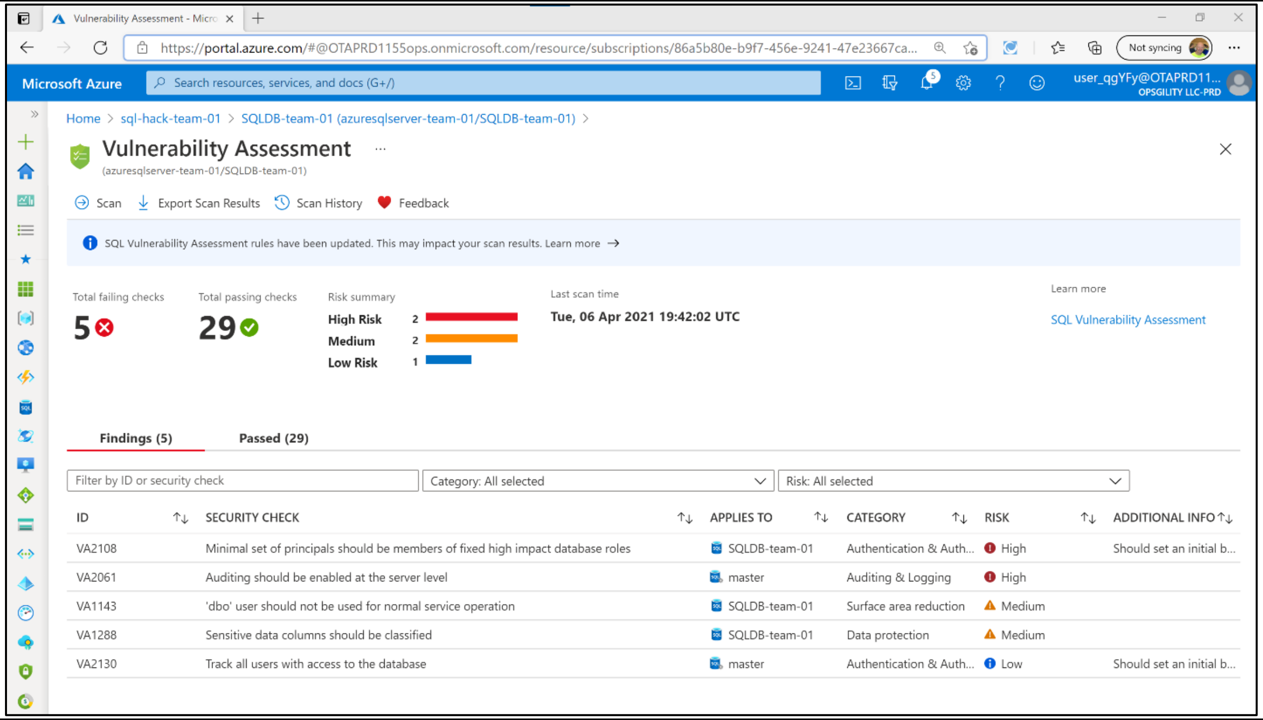
Learn more: SQL Vulnerability Assessment

Findings (5) | Passed (29)

Filter by ID or security check | Category: All selected | Risk: All selected

ID	SECURITY CHECK	APPLIES TO	CATEGORY	RISK	ADDITIONAL INFO
VA2108	Minimal set of principals should be members of fixed high impact database roles	SQLDB-team-01	Authentication & Auth...	High	Should set an initial b...
VA2061	Auditing should be enabled at the server level	master	Auditing & Logging	High	
VA1143	'dbo' user should not be used for normal service operation	SQLDB-team-01	Surface area reduction	Medium	
VA1288	Sensitive data columns should be classified	SQLDB-team-01	Data protection	Medium	
VA2130	Track all users with access to the database	master	Authentication & Auth...	Low	Should set an initial b...

6. Review the lists of passed and failed checks. Notice that the report is specific to database you ran the scan for but does also include events against the system database and therefore flag server configuration issues.



Vulnerability Assessment - Micro

https://portal.azure.com/#@OTAPRD1155ops.onmicrosoft.com/resource/subscriptions/86a5b80e-b9f7-456e-9241-47e23667ca...

Microsoft Azure

Search resources, services, and docs (G+/)

user_qgYFy@OTAPRD11... OPSGILITY LLC-PRD

Home > sql-hack-team-01 > SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) >

Vulnerability Assessment
(azuresqlserver-team-01/SQLDB-team-01)

Scan | Export Scan Results | Scan History | Feedback

SQL Vulnerability Assessment rules have been updated. This may impact your scan results. Learn more →

Total failing checks: 5
Total passing checks: 29

Risk summary
High Risk 2
Medium 2
Low Risk 1

Last scan time: Tue, 06 Apr 2021 19:42:02 UTC

Learn more: SQL Vulnerability Assessment

Findings (5) | Passed (29)

Filter by ID or security check | Category: All selected | Risk: All selected

ID	SECURITY CHECK	APPLIES TO	CATEGORY	RISK	ADDITIONAL INFO
VA2108	Minimal set of principals should be members of fixed high impact database roles	SQLDB-team-01	Authentication & Auth...	High	Should set an initial b...
VA2061	Auditing should be enabled at the server level	master	Auditing & Logging	High	
VA1143	'dbo' user should not be used for normal service operation	SQLDB-team-01	Surface area reduction	Medium	
VA1288	Sensitive data columns should be classified	SQLDB-team-01	Data protection	Medium	
VA2130	Track all users with access to the database	master	Authentication & Auth...	Low	Should set an initial b...

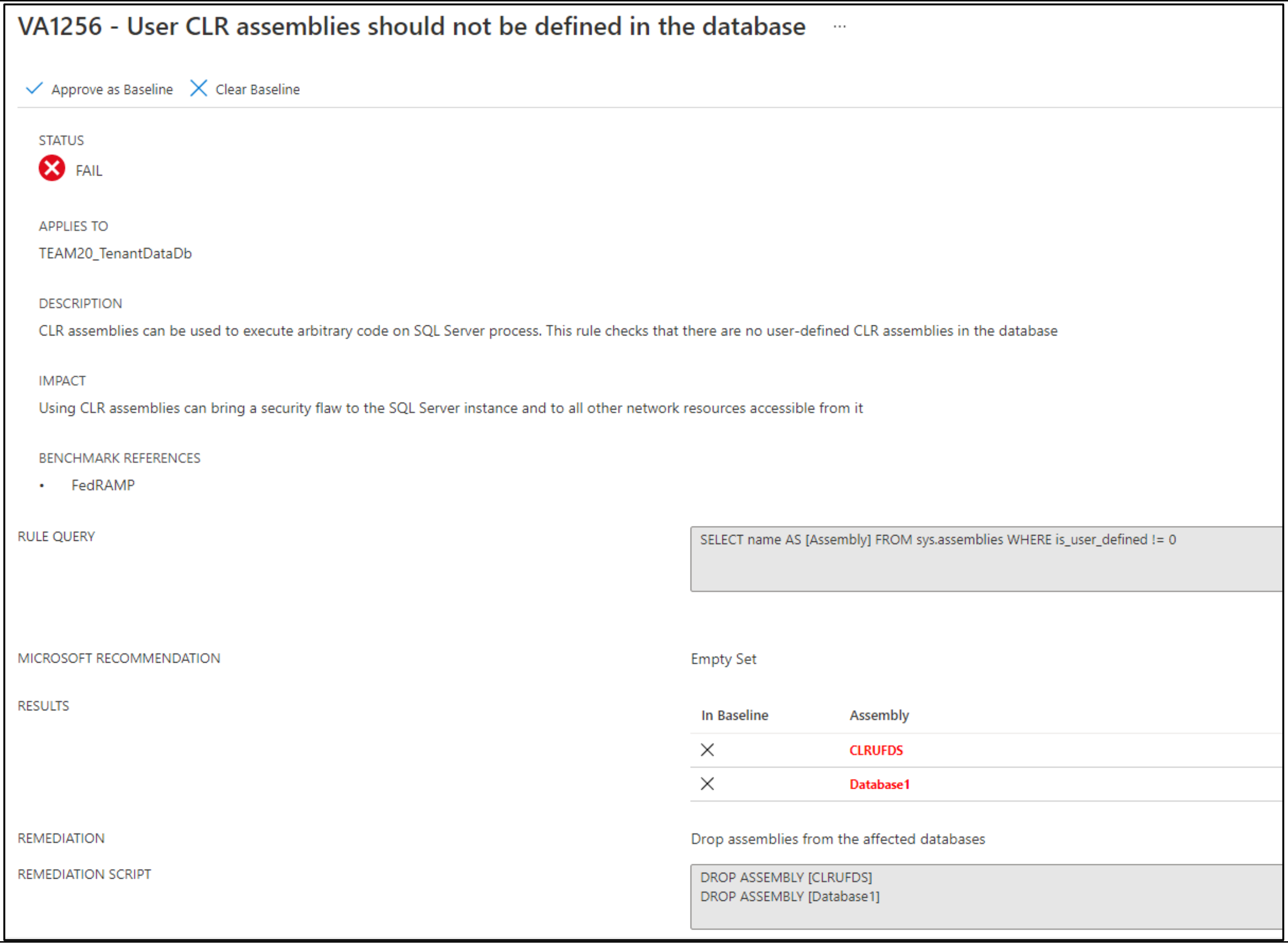
7. In the **Findings** tab, which lists the failed checks, click on finding:

ID	**Security Check**
VA1256	User CLR assemblies should not be defined in the database

8. Note the detailed report lists the rule's details, the offending CLRs and a remediation script to remove them.

However, because these 2 CLRs are an integral part of our migrated legacy application we need to keep them.

But equally we don't want them to be continuously flagged as an issue in the Vulnerability Assessment reports. To do this we can add exceptions to the Vulnerability Assessment's "baseline" position.


VA1256 - User CLR assemblies should not be defined in the database ···

STATUS
✖ FAIL

APPLIES TO
TEAM20_TenantDataDb

DESCRIPTION
CLR assemblies can be used to execute arbitrary code on SQL Server process. This rule checks that there are no user-defined CLR assemblies in the database

IMPACT
Using CLR assemblies can bring a security flaw to the SQL Server instance and to all other network resources accessible from it

BENCHMARK REFERENCES
- FedRAMP

RULE QUERY

```
SELECT name AS [Assembly] FROM sys.assemblies WHERE is_user_defined != 0
```

MICROSOFT RECOMMENDATION
Empty Set

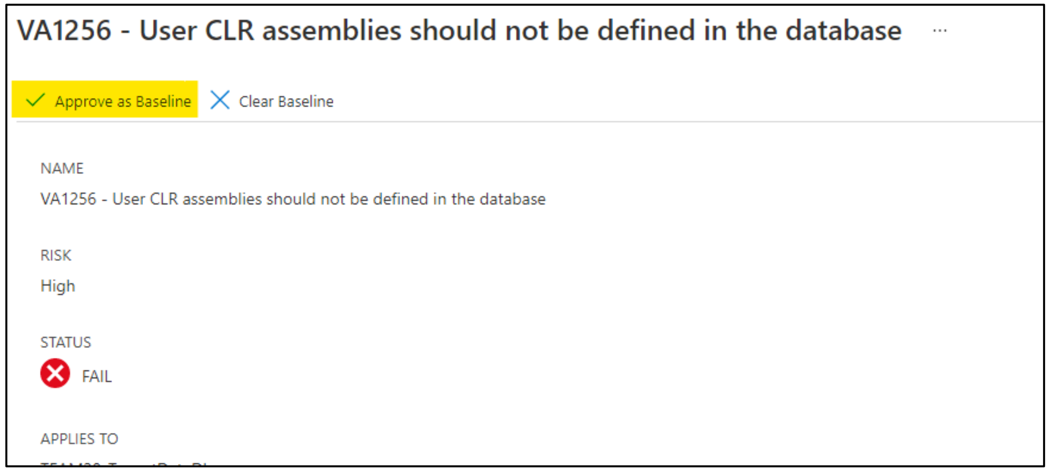
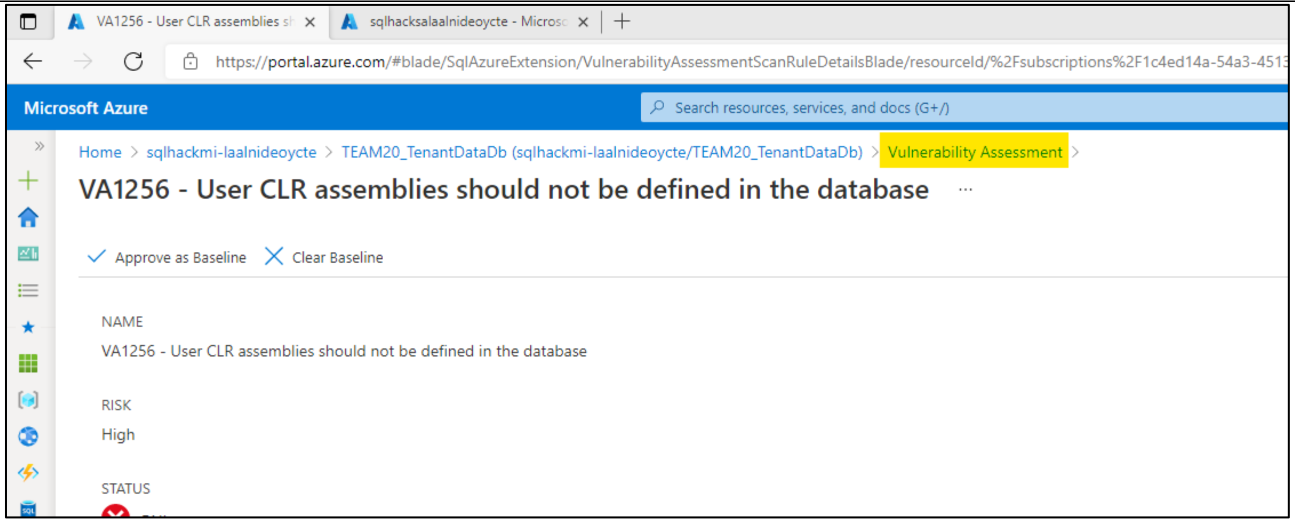
RESULTS

In Baseline	Assembly
✕	CLRUFDS
✕	Database1

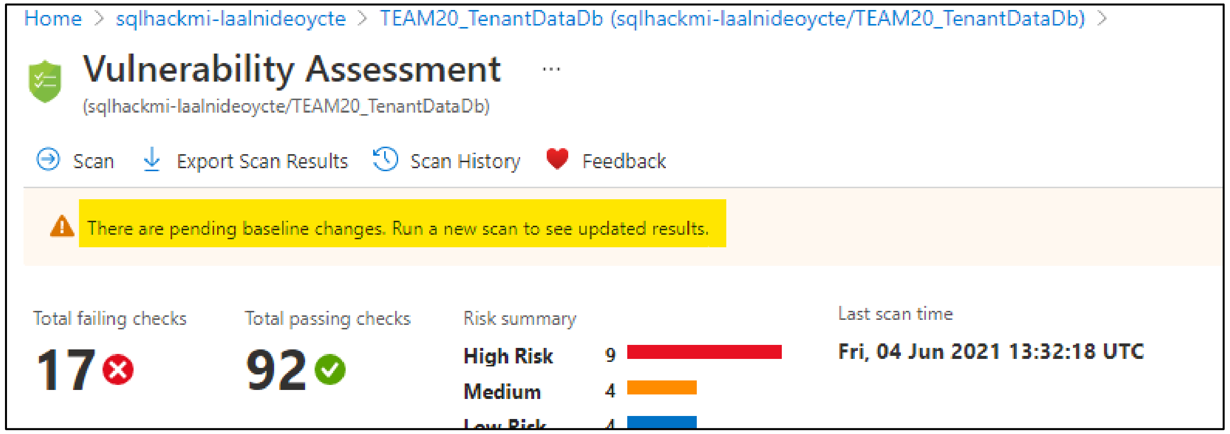
REMEDIATION
Drop assemblies from the affected databases

REMEDIATION SCRIPT

```
DROP ASSEMBLY [CLRUFDS]
DROP ASSEMBLY [Database1]
```

9. On the details page for V1256 click **Approve as Baseline** and select **Yes** in the warning message. Approving as the baseline will update the Vulnerability Assessment rules engine to accept the current CLR Assemblies as allowable and set a new baseline position for the rule.	 VA1256 - User CLR assemblies should not be defined in the database NAME VA1256 - User CLR assemblies should not be defined in the database RISK High STATUS FAIL APPLIES TO
10. Navigate back to the Vulnerability Assessment summary page by clicking the "**Vulnerability Assessment**" link at the top of the portal page	 https://portal.azure.com/#blade/SqlAzureExtension/VulnerabilityAssessmentScanRuleDetailsBlade/resourceId/%2Fsubscriptions%2F1c4ed14a-54a3-4513 Home > sqlhackmi-laalnideoycte > TEAM20_TenantDataDb (sqlhackmi-laalnideoycte/TEAM20_TenantDataDb) > Vulnerability Assessment > VA1256 - User CLR assemblies should not be defined in the database NAME VA1256 - User CLR assemblies should not be defined in the database RISK High STATUS

11. Once back at the Vulnerability Assessment summary page there will be a warning that the baseline has been updated and a new scan is needed.



Home > sqlhackmi-laalnideoycte > TEAM20_TenantDataDb (sqlhackmi-laalnideoycte/TEAM20_TenantDataDb) >

Vulnerability Assessment ...
(sqlhackmi-laalnideoycte/TEAM20_TenantDataDb)

Scan | Export Scan Results | Scan History | Feedback

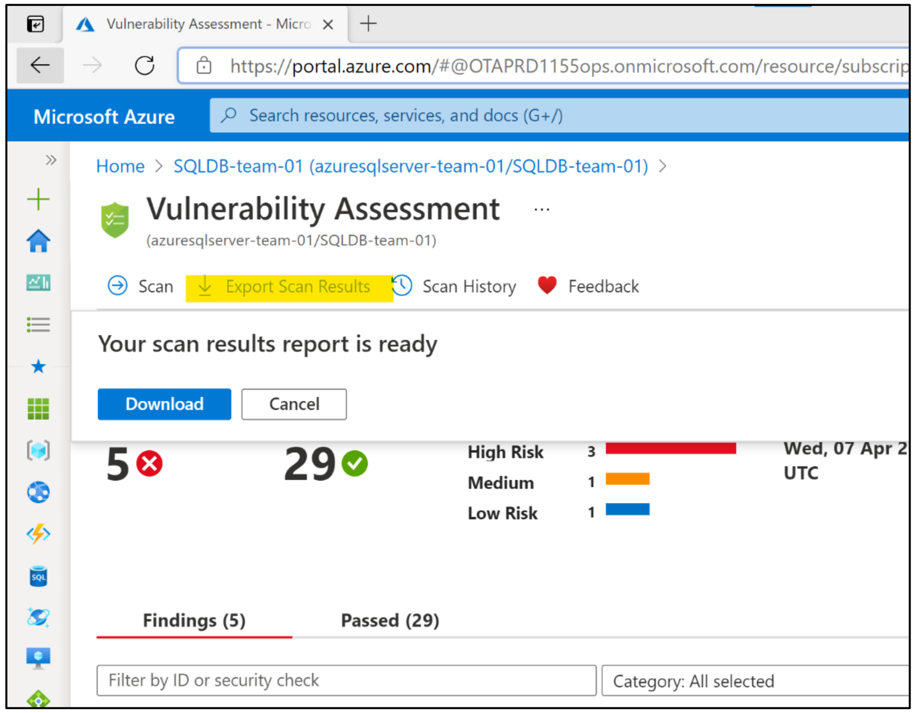
⚠ There are pending baseline changes. Run a new scan to see updated results.

Total failing checks: 17 ✗
Total passing checks: 92 ✓
Risk summary: High Risk 9, Medium 4, Low Risk 4
Last scan time: Fri, 04 Jun 2021 13:32:18 UTC

12. Click the **Scan** button to run a manual scan which will take a about a minute. Once the scan completes the finding VA1256 will be removed from the Findings list.

When making changes to a Vulnerability Assessment baseline it may be necessary for compliance reasons to export a Scan Findings report to show the security posture of the Azure SQL Database in relation to the amended baseline.

To export the results of a scan to reflect the current baseline click "**Export Scan Results**" at the top of the portal screen:



Vulnerability Assessment - Micro
https://portal.azure.com/#@OTAPRD1155ops.onmicrosoft.com/resource/subscrip

Microsoft Azure | Search resources, services, and docs (G+/)

Home > SQLDB-team-01 (azuresqlserver-team-01/SQLDB-team-01) >

Vulnerability Assessment ...
(azuresqlserver-team-01/SQLDB-team-01)

Scan | Export Scan Results | Scan History | Feedback

Your scan results report is ready

Download | Cancel

5 ✗ 29 ✓ High Risk 3, Medium 1, Low Risk 1 — Wed, 07 Apr 2 UTC

Findings (5) | Passed (29)

Filter by ID or security check | Category: All selected

NOTE: Excel is *not* installed on your lab VMs *so you will have to copy the report to your own desktop to have a look at it.*

6. LAB 3 Part 2: Azure Defender for SQL – Advanced Threat Protection

The other security component of Azure Defender for SQL is Advanced Threat Protection.

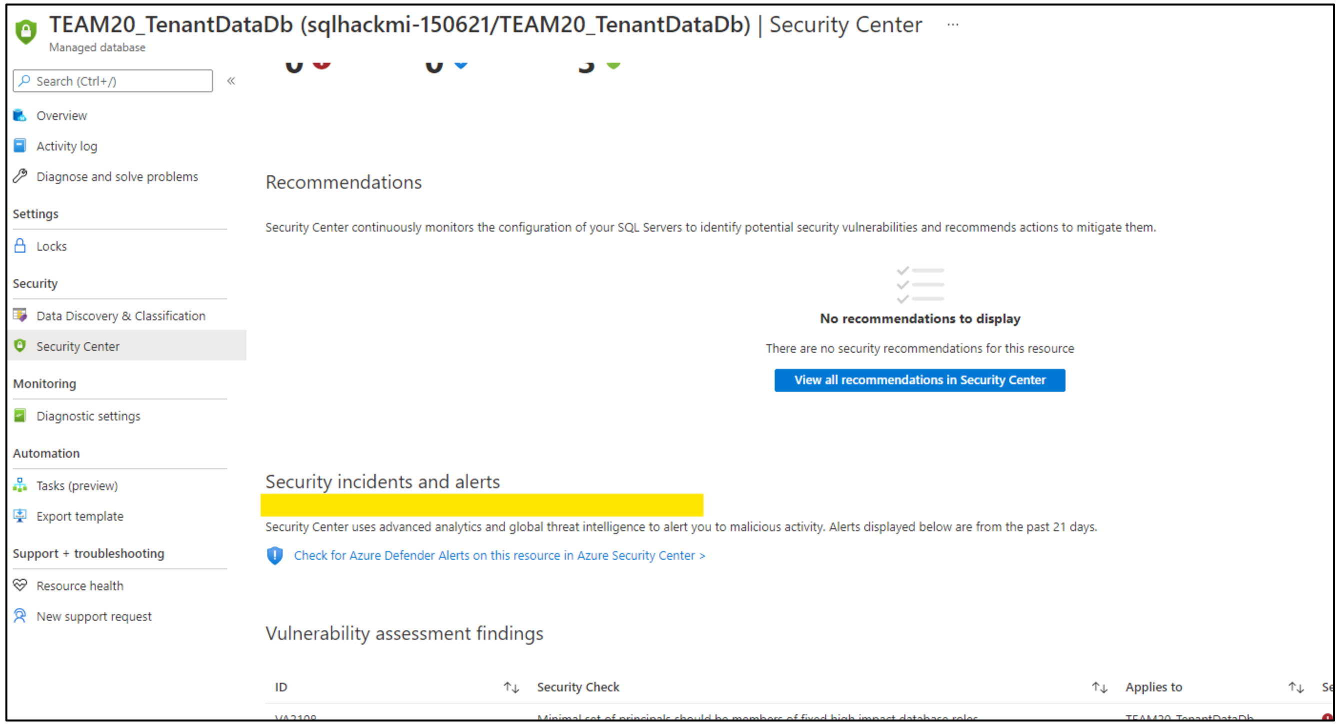
Advanced Threat Protection provides a layer of security that can detect and respond to potential threats as they occur by providing security alerts on anomalous activities. Alerts can be generated based on suspicious database activities, potential vulnerabilities, and SQL injection attacks, as well as anomalous database access and queries patterns.

More information in Azure Defender for SQL – Advanced Threat Protection can be found here:

https://docs.microsoft.com/en-us/azure/azure-sql/database/threat-detection-overview

Advanced Threat Protection

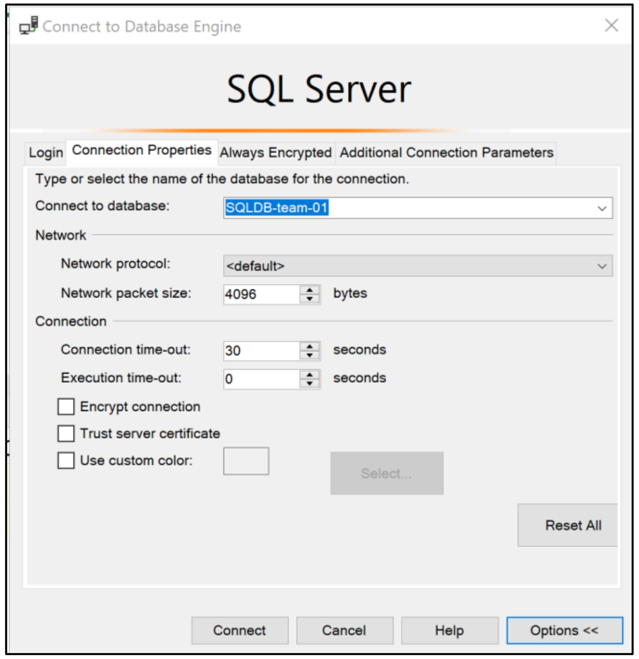
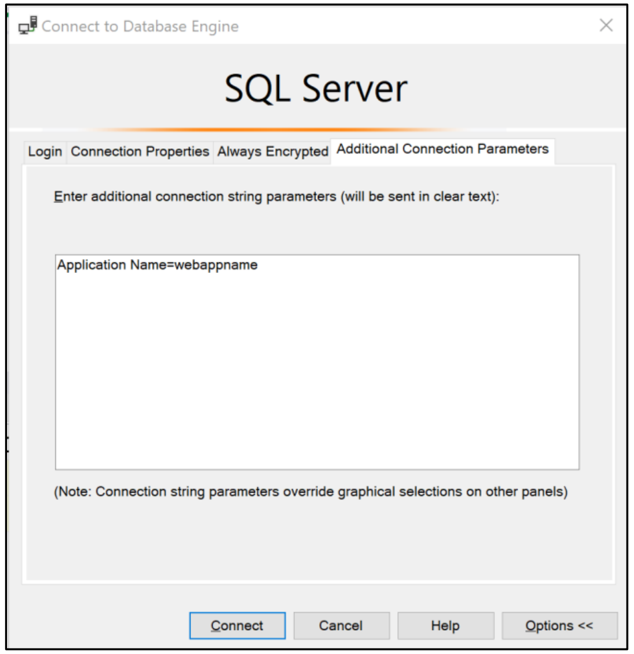
Narrative	Screenshot/Code	Notes
1. In the Azure portal navigate to the shared SQL Managed Instance.		
2. Scroll down the Overview screen until you see the list of databases and click on your **TEAMXX_TenantDataDB** database.		

3. In the **TEAMXX_TenantDataDB** database screen, on the left-hand blade click **Security Center** in the Security section 4. Scroll down to the **Security incidents and alters heading** – note no incidents or alerts are listed:	 TEAM20_TenantDataDb (sqlhackmi-150621/TEAM20_TenantDataDb) | Security Center Managed database Recommendations Security Center continuously monitors the configuration of your SQL Servers to identify potential security vulnerabilities and recommends actions to mitigate them. No recommendations to display There are no security recommendations for this resource Security incidents and alerts Security Center uses advanced analytics and global threat intelligence to alert you to malicious activity. Alerts displayed below are from the past 21 days. Check for Azure Defender Alerts on this resource in Azure Security Center > Vulnerability assessment findings	
5. On the team VM, open a new query window in SQL Server Management Studio connected to your **TEAMXX_TenantDataDB** database.		

6. To simulate a potential SQL injection query copy the following SELECT into the new query window ***BUT DON'T RUN IT YET***:	`--Advanced Threat Protection`
	`SELECT *`
	`FROM sys.databases`
	`WHERE database_id like '' or 1 = 1 -- ' and family = 'test1';`
		Notice that the logic in the WHERE clause will always equate to true and the positioning of single-quotes including in the comment represents a potential SQL injection vulnerability

7. Before running the query change the connection properties as show opposite using the **Query\Connection\Change Connection**… menu in SSMS.

8. Click **Connect**

Specify the name of the team Azure SQL Database: **TEAMXX_TenantDataDB**	On "Additional Connection Parameters add a connection string option to specify the application name: **Application Name=webappname**
	

Connect to Database Engine

SQL Server

Login | Connection Properties | Always Encrypted | Additional Connection Parameters

Type or select the name of the database for the connection.

Connect to database: SQLDB-team-01

Network

Network protocol: <default>

Network packet size: 4096 bytes

Connection

Connection time-out: 30 seconds

Execution time-out: 0 seconds

☐ Encrypt connection

☐ Trust server certificate

☐ Use custom color:

Select... | Reset All

Connect | Cancel | Help | Options <<

Connect to Database Engine

SQL Server

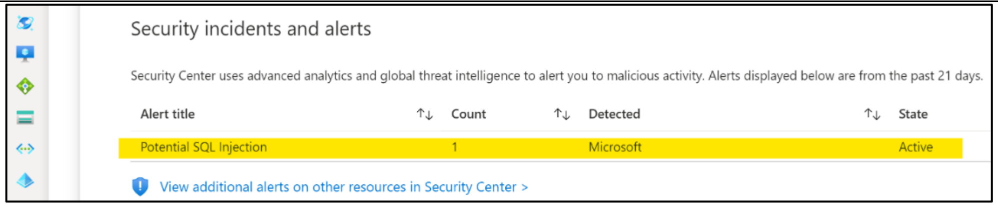
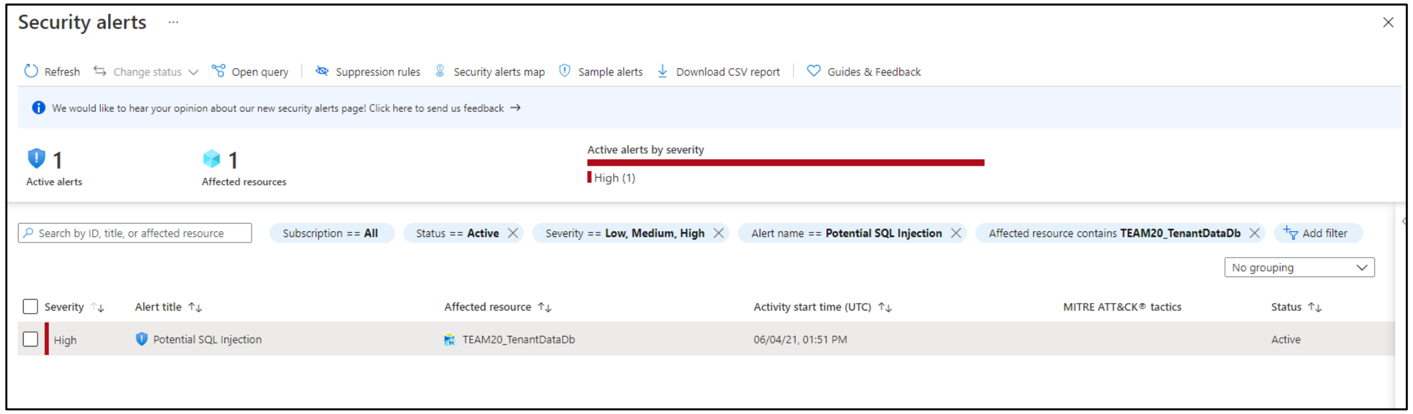
Login | Connection Properties | Always Encrypted | Additional Connection Parameters

Enter additional connection string parameters (will be sent in clear text):

Application Name=webappname

(Note: Connection string parameters override graphical selections on other panels)

Connect | Cancel | Help | Options <<

9. Run the query. It will return a list of databases on the server.		
10. Back in the Azure Portal **Security Center** screen, after a few minutes an Alert should be generated:	 Security incidents and alerts Security Center uses advanced analytics and global threat intelligence to alert you to malicious activity. Alerts displayed below are from the past 21 days. Alert title / Count / Detected / State Potential SQL Injection / 1 / Microsoft / Active View additional alerts on other resources in Security Center >	**NOTE:** It might take up to 10mins for the alert to appear in the portal
11. Once the Later appears click on it to see the details. Depending on the progress of other teams you may see multiple entries in the details table.	 Security alerts We would like to hear your opinion about our new security alerts page! Click here to send us feedback → 1 Active alerts · 1 Affected resources · Active alerts by severity: High (1) Search by ID, title, or affected resource · Subscription == **All** · Status == **Active** · Severity == **Low, Medium, High** · Alert name == **Potential SQL Injection** · Affected resource contains **TEAM20_TenantDataDb** · Add filter · No grouping Severity / Alert title / Affected resource / Activity start time (UTC) / MITRE ATT&CK® tactics / Status High / Potential SQL Injection / TEAM20_TenantDataDb / 06/04/21, 01:51 PM / / Active	

12. Try clicking on the Alert. Note that you can drill further into the alert to see more details, get explanations and links to documentation on the alert and even advice on how negate and remediate the problem.		

7. LAB 4: Information Protection using Dynamic Data Masking

In this lab we'll cover information protection. These capabilities offer inner security layers that can be used to protect data. We'll explore how the Dynamic Data Masking feature is part of SQL Server's "secure by default" posture and can protect information by limiting sensitive data exposure by masking it to non-privileged users thus helping to prevent unauthorized access.

Administrators can designate how much of the sensitive data to reveal with minimal impact on the application layer. It's a SQL policy-based security feature (meaning permissions are applied using DDL statements) that hides the sensitive data in the result set of a query over designated database fields, while the data in the database remains unchanged.

For example, a service representative at a call center might identify a caller by confirming several characters of their email address, but the complete email address shouldn't be revealed to the service representative. A masking rule can be defined that masks all the email address in the result set of any query.

More details on Dynamic Data Masking can be found here:
https://docs.microsoft.com/en-us/azure/azure-sql/database/dynamic-data-masking-overview

Narrative	Screenshot/Code	Notes
For databases hosted on SQL Managed Instance, Dynamic Data Masking needs to be configured via TSQL.		
1. In SQL Server Management Studio open a new query window connected to your **TEAMXX_TenantDataDB** database.		

2. To mask the email address column in the customer table	`-- Replace XX with your team number`
	`USE TEAMXX_TenantDataDb;`	

we need to change the column definition by running this SQL:

```
-- Alter column definition to mask [EmailAddress] data
ALTER TABLE [SalesLT].[Customer]
ALTER COLUMN [EmailAddress] VARCHAR(50) MASKED WITH (FUNCTION = 'email()');
```

Note that we used a built-in email masking function.

For details on built in masking functions and how to create custom masks see this documentation: [Dynamic Data Masking - SQL Server | Microsoft Docs]

Now lets test the masking to see what affect it has.

1. Open a new query window in SQL Server Management Studio. Copy the SQL statements below and paste them into the query window.

One of the main advantages of Dynamic Data Masking is that because the masking/unmasking is performed by the SQL Server engine, masked data will appear masked in ***any*** client application without the need to make application code changes.

NOTE: ***The statements below are separate steps - run each step individually and look at the results after each one.***

```
USE TEAMXX_TenantDataDb; -- Replace XX with your team number


-- STEP 1: SELECT performed by a member of db_owner or sysadmin
role to show plain text
SELECT TOP 100 c.EmailAddress, c.* FROM SalesLT.Customer c;


-- STEP 2: Create new database user and give them SELECT
permission on [Customer]
CREATE USER TestUser WITHOUT LOGIN;
GRANT SELECT ON [SalesLT].[Customer] to TestUser;
```

Note that once defined a mask is applied by default – you have to be assigned the UNMASK permission to see any masked data as plain text. But UNMASK is a global permission that applies to *all* masked columns – you can’t mask/unmask columns individually.

	-- STEP 3: SELECT columns from the customer table as the test user
	EXECUTE AS USER = 'TestUser';
	SELECT TOP 100 c.EmailAddress, c.* FROM SalesLT.Customer c;
	REVERT;
	
	-- STEP 4: Grant unmask privilege to the test user
	GRANT UNMASK TO TestUser;
	
	-- STEP 5: Select from the table again as test user
	EXECUTE AS USER = 'TestUser';
	SELECT c.EmailAddress, c.* FROM SalesLT.Customer c;
	REVERT;	