Microsoft Services

SQL Server 2016 and SQL Server 2017 Security Capabilities

Microsoft

Workshop Overview

Time Slot	Key Topics
8:30a-9:00	Intro - Meet and Greet
9:00a-10:00	Key SQL Server Security Capabilities • *Row Level Security* • *Dynamic Data Masking* • *Always Encrypted* • *SQL Server Vulnerability Assessment*
10:15a-11:15	SQL Server 2016 / 2017 – Query Store Features, Auto Tuning, and Adaptive Query Processing
11:30a-12:00	Lunch Break (Over)
12:15p-01:15	SQL Server Machine Learning Services
01:30p-2:30	Azure ML and Databricks

Why SQL Security Intelligence?

No organization is immune to data breaches

- No organization is immune to data breaches and security incidents
- 75% perpetrated by outsiders, while 25% involved internal actors

Common threats

- SQL injection
- Password cracking
- Credential theft/leak
- Privilege abuse

Common regulations

- GDPR (Personal)
- PCI (Payment)
- HIPPA (Health)
- FedRAMP (Government)

Secure your database

1. Discover sensitive data
2. Identify & remediate SQL vulnerabilities
3. Detect & remediate suspicious database activities
4. Meet security regulations requirements

SQL Server Security Capabilities

Layered Security and Defense in Depth

Security Workshop Topics

- SQL Server Security Features Overview
  - Transparent Data Encryption
  - Row Level Security
  - Dynamic Data Masking
  - Always Encrypted
  - Advanced Threat Detection
- General Data Protection Regulation
- SQL Server Management Studio Improvements

SQL Server 2016 / 2017 Mission-Critical (DB Engine)

Performance

Operational Analytics
Insights on operational data; Works with in-memory OLTP and disk-based OLTP

In-memory OLTP Enhancements
Greater T-SQL surface area, terabytes of memory supported, and greater number of parallel CPUs

Live Query Statistics

Query Store
Monitor and optimize query plans

Automatic Database Tuning
Provides insight into potential query performance problems, recommends solutions, and can automatically fix identified problems

DMV Improvements

Adaptive Query Processing
A feature family that introduces a new generation of query processing improvements

Security

Always Encrypted
Sensitive data remains encrypted at all times with ability to query

Row-Level Security
Apply fine-grained access control to table rows

Dynamic Data Masking
Real-time obfuscation of data to prevent unauthorized access

Advanced Threat Detection
Ability to find unusual login patterns, track usage behavior in an auditing database, track SQL injection vulnerability, and more

Other Enhancements
Audit success/failure of database operations

TDE support for storage of in-memory OLTP tables

Enhanced auditing for OLTP with ability to track history of record changes

Availability / Platform

SQL Server 2017 on Linux

Enhanced AlwaysOn
Three synchronous replicas for auto failover across domains

Round robin load balancing of replicas

Automatic failover based on database health

DTC for transactional integrity across database instances with AlwaysOn

Support for SSIS with AlwaysOn

Stretch Database
Archive historical data transparently and securely to Azure

Queries stretch across local data as well as Azure data

Scalability

Machine Learning Services
R Scripting along with Python scripting from the SQL Server Engine

Graph DB Support
For modeling many-to-many relationships

Enhanced Database Caching
Cache data with automatic, multiple TempDB files per instance in multi-core environments

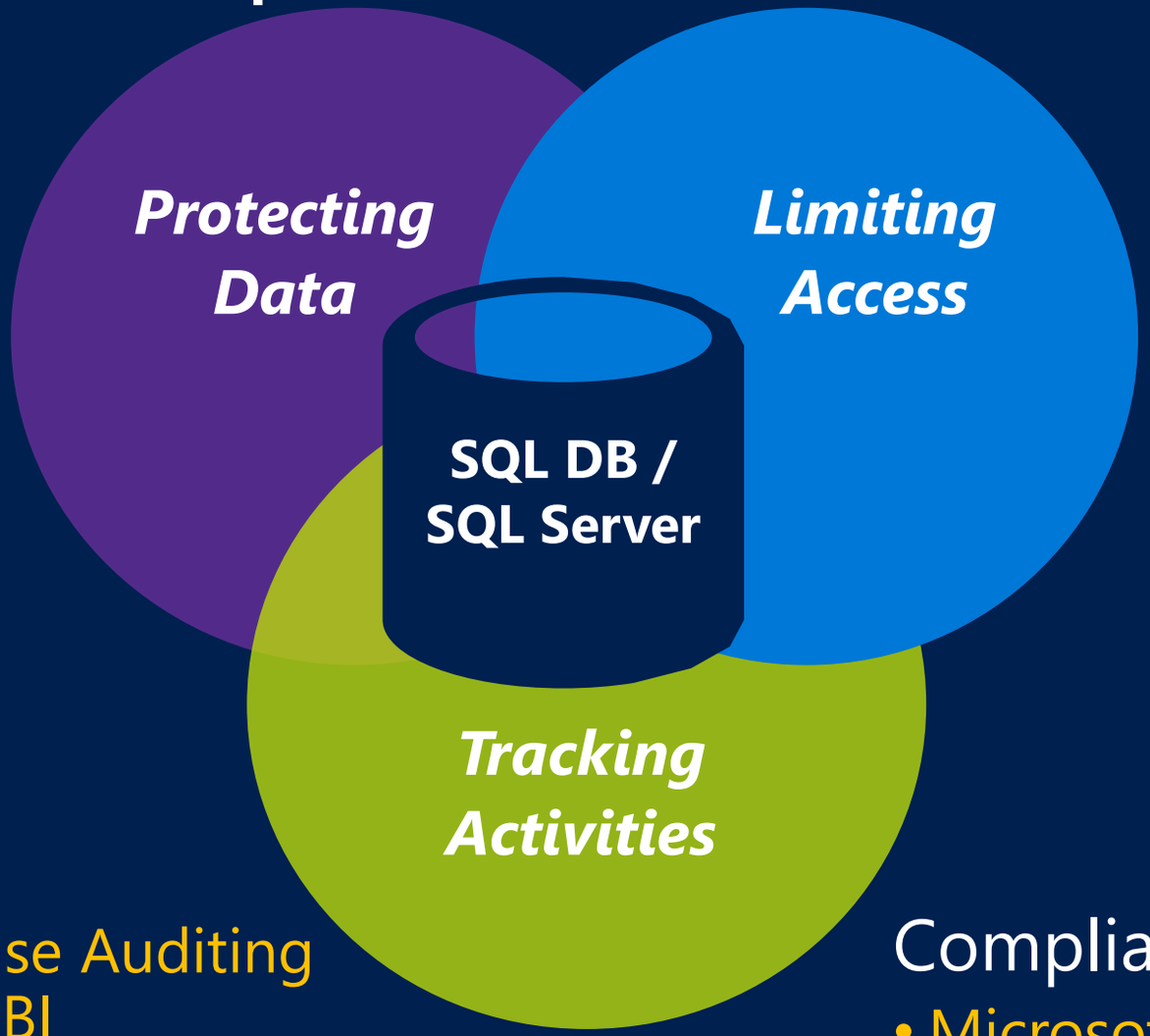
New Programmatic Improvements
New TSQL Functionality, Maintenance Plan Improvements, New ALTER DATABASE Options
Expanded support for JSON data
New PolyBase query engine integrates SQL Server with external data in Hadoop or Azure Blob storage

Temporal Database Support
Query data as points in time

New in SQL Server 2017

Security Landscape

Available
- TDE
- Dynamic Data Masking
- Always Encrypted



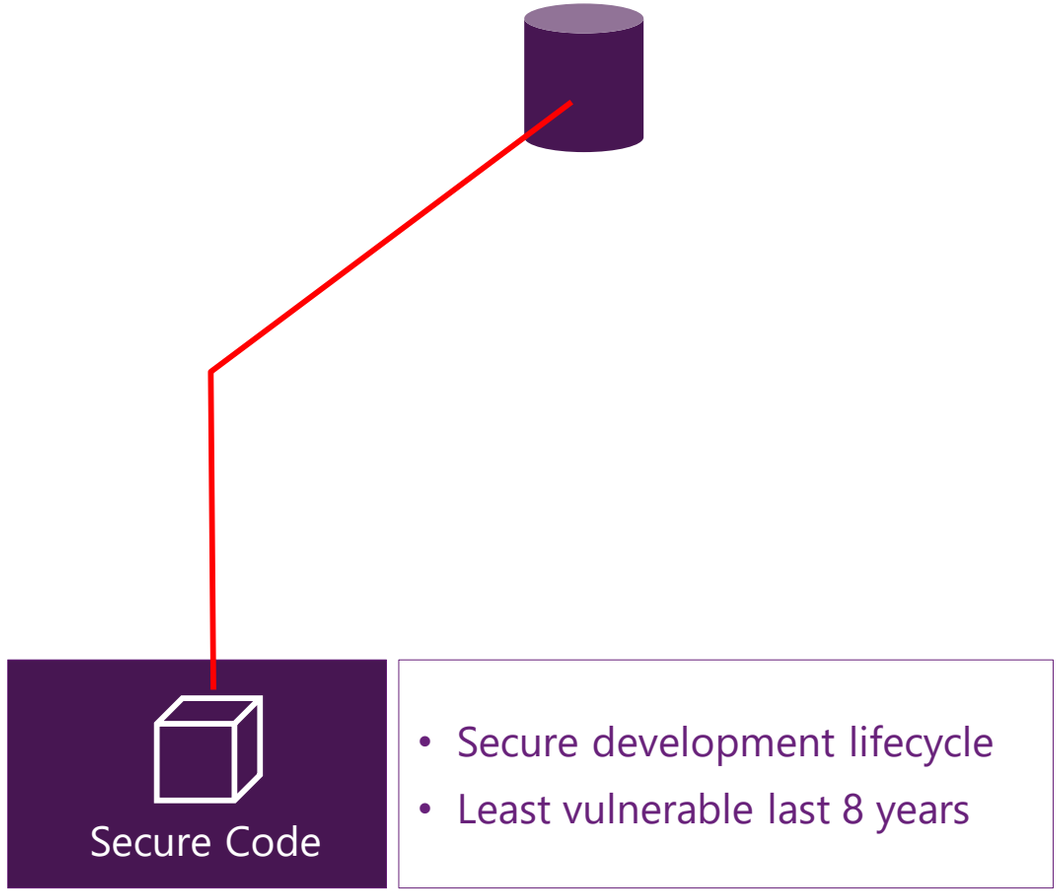
Available
- Row-level Security
- Firewall
- Users & Permissions
- SQL Auth.
- Azure Active Directory Authentication

Available
- SQL Database Auditing with Power BI
- Advanced Threat Detection*

Compliance:
- Microsoft Azure Trust Center

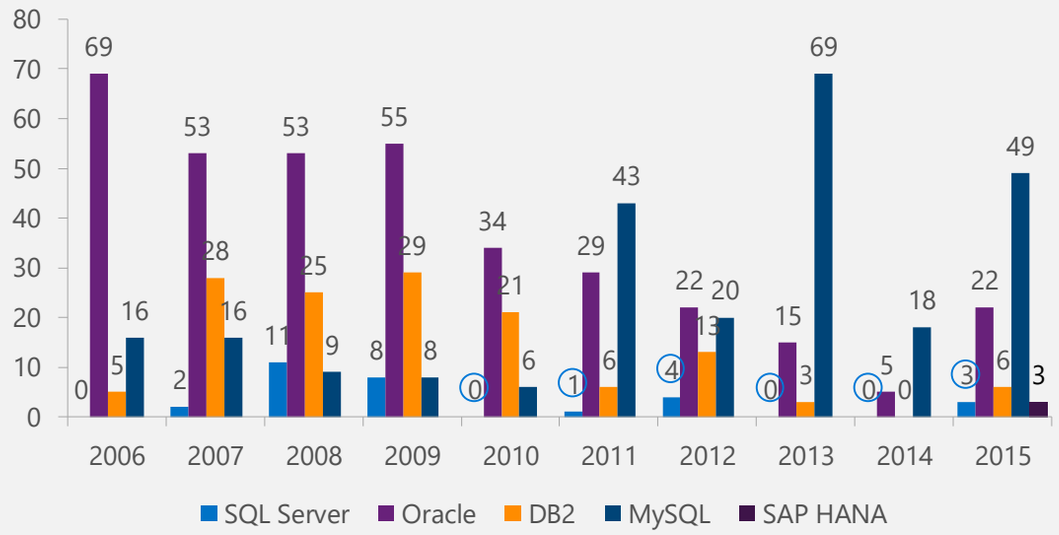
Most Secure Database

Secure core code



- Secure development lifecycle
- Least vulnerable last 8 years

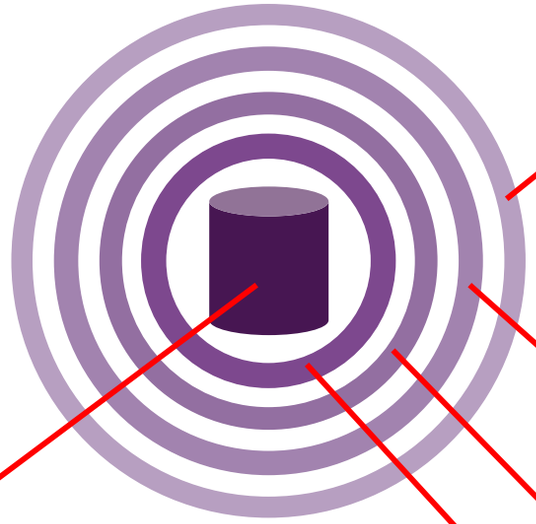
Least vulnerable last 8 years while being **most utilized**



* National Institute of Standards and Technology Comprehensive Vulnerability Database update 10/2015

Most Secure Database

Surrounded by layers of protection



Identity

- SQL Authentication
- Windows Authentication
- Azure Active Directory Auth.

Monitor activity

- SQL Threat Detection
- SQL Server Auditing

Control access

- Row-level Security
- Dynamic Data Masking

Protect data

- Always Encrypted
- Transparent Data Encryption
- Encryption-in-Flight

Secure Code

- Secure development lifecycle
- Least vulnerable last 8 years

Cloud-only

Transparent Data Encryption

Azure SQL DB and SQL Server 2016 / 2017

Encryption@Rest Overview

Provides defense-in-depth against

- Offline attacks
- Online attacks when keys are used as a secondary AuthZ mechanism

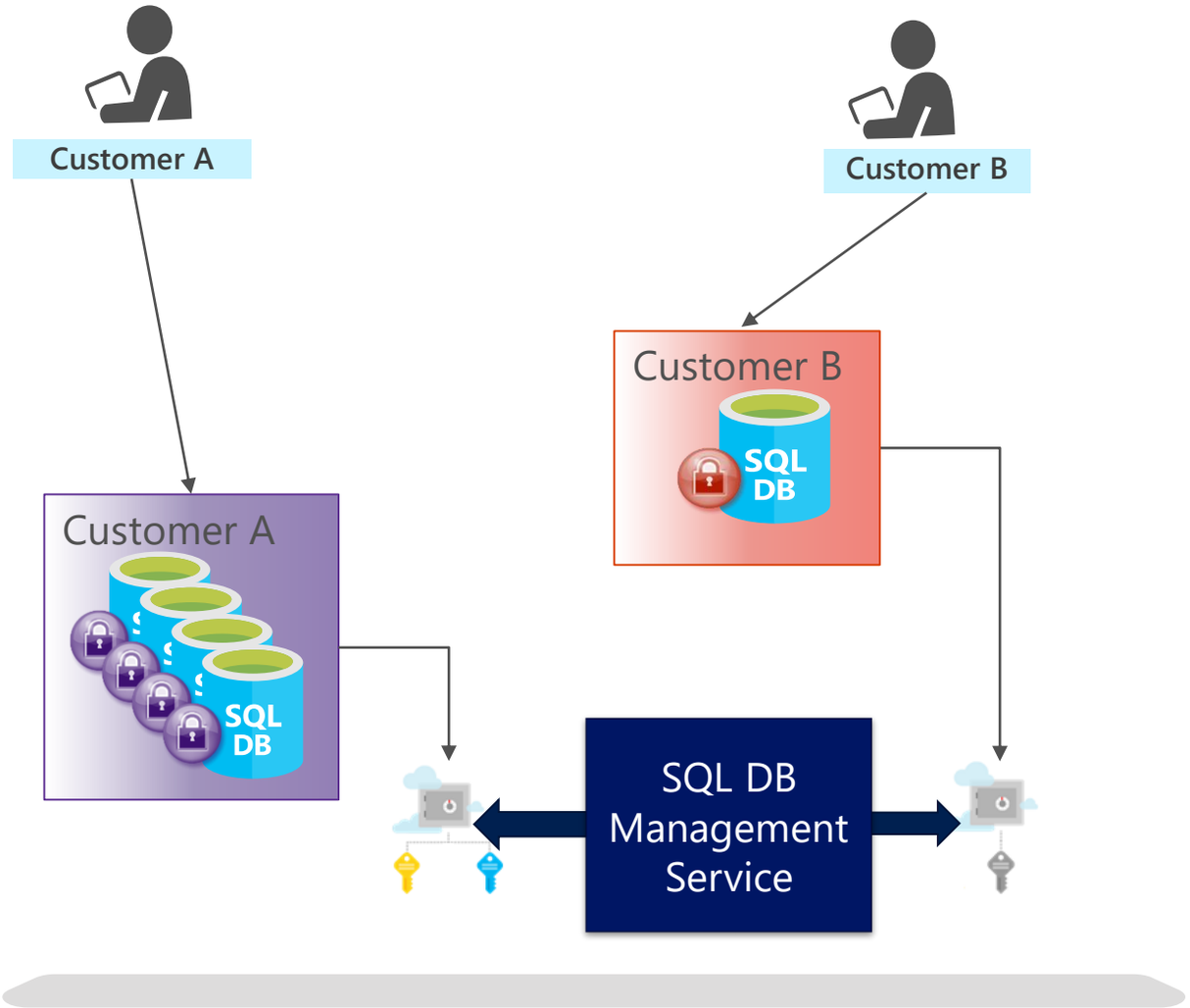
Encryption at-rest is required by certain sovereign laws and certifications

Encryption Models

Encryption Models			
Server Encryption			Client Encryption
Server-side Encryption using service managed keys	Server-side encryption using customer managed keys in Azure KeyVault	Server-side encryption using on-prem customer managed keys	
• Azure services can see decrypted data • Microsoft manages the keys • Full cloud functionality	• Azure services can see decrypted data • Customer controls keys via Azure Key Vault • Full cloud functionality	• Azure services can see decrypted data • Customer controls keys On-prem • Full cloud functionality	• Azure services cannot see decrypted data • Customer keep keys on-premises • REDUCED cloud functionality • Potentially high performance impact

Transparent Data Encryption

How It Works



- On by Choice
  - Protects the user database and all of its backups, Transaction Logs and TempDB
- "2-click" User Experience
  - Alternatively: 2 T-SQL statements
  - Azure SQL DB manages your keys (aka service managed TDE)
- Improved Encryption Performance
  - Using INTEL's AES-NI Hardware Acceleration
- Available on v12 servers, **all** SQL DB's editions

On Premise vs. Azure: Implementing TDE

Key Variables

Number of environments, databases

Key rotation requirements

Key Management Challenges

Provisioning

Rotation

Backup/Restore

DR

Total Effort to Implement:
400+ Hours

**Implementation on SQL Server 2014+*

***Effort Encompasses Multiple Environments*

Azure Implementation

Azure Portal

Powershell

Best Practice:
Enable Transparent Data Encryption on all databases

Row Level Security

Azure SQL DB and SQL Server 2016 / 2017

The Need for Row-Level Security

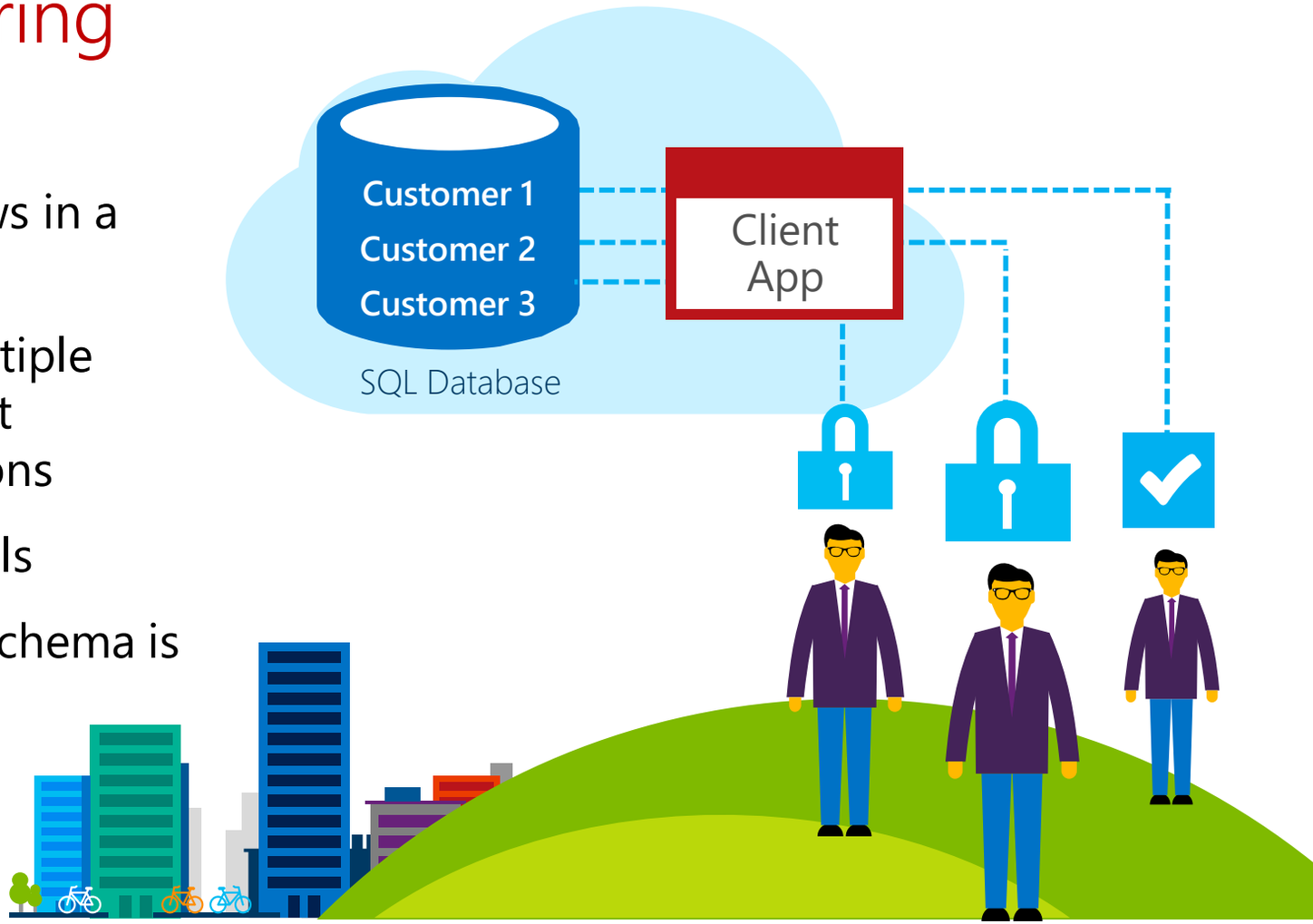
Protect data privacy by ensuring the right access across rows

Fine-grained access control over specific rows in a database table

Help prevent unauthorized access when multiple users share the same tables, or to implement connection filtering in multitenant applications

Administer via SSMS or SQL Server Data Tools

Enforcement logic inside the database and schema is bound to the table



Benefits of Row-Level Security (RLS)

Fine-Grained Access Control

Keeping multitenant databases secure by limiting access by other users who share the same tables

Application Transparency

RLS works transparently at query time, no app changes needed

Compatible with RLS in other leading products

Centralized Security Logic

Enforcement logic resides inside database and is schema-bound to the table it protects providing greater security. Reduced application maintenance and complexity

Store data intended for many consumers in a single database/table while at the same time restricting row-level read and write access based on users' execution context.

Row-Level Security Concepts

Predicate function

User-defined inline table-valued function (iTVF) implementing security logic

Can be arbitrarily complicated, containing joins with other tables

Security predicate

Binds a predicate function to a particular table, applying it for all queries

Two types: filter predicates and blocking predicates

Security policy

Collection of security predicates for managing security across multiple tables

Performance?

Inline functions get optimized to provide comparable performance to views—as if the logic were directly embedded in the original query statement.

```
CREATE SECURITY POLICY mySecurityPolicy
        ADD FILTER PREDICATE dbo.fn_securitypredicate(wing, startTime, endTime)
        ON dbo.patients
```

Configure Row-Level Security

1. Create user accounts to test Row-Level Security

```
USE AdventureWorks2014;
GO
CREATE USER Manager WITHOUT LOGIN;
CREATE USER SalesPerson280 WITHOUT LOGIN;
```

2. Grant read access to users on a required table

```
GRANT SELECT ON Sales.SalesOrderHeader TO Manager;
GRANT SELECT ON Sales.SalesOrderHeader TO SalesPerson280;
```

3. Create a new schema and inline table-valued function

```
CREATE SCHEMA Security;
GO
CREATE FUNCTION Security.fn_securitypredicate(@SalesPersonID AS int)
    RETURNS TABLE
WITH SCHEMABINDING
AS
   RETURN SELECT 1 AS fn_securitypredicate_result WHERE ('SalesPerson' + CAST(@SalesPersonId as VARCHAR(16)) = USER_NAME())
    OR (USER_NAME() = 'Manager');
```

4. Create a security policy, adding the function as both a filter and block predicate on the table

```
CREATE SECURITY POLICY SalesFilter
ADD FILTER PREDICATE Security.fn_securitypredicate(SalesPersonID)
  ON Sales.SalesOrderHeader,
ADD BLOCK PREDICATE Security.fn_securitypredicate(SalesPersonID)
  ON Sales.SalesOrderHeader
WITH (STATE = ON);
```

5. Execute the query to the required table so that each user sees the result (can also alter the security policy to disable)

Row Level Security (RLS) Example

```
CREATE FUNCTION
dbo.fn_securitypredicate(@wing int)
   RETURNS TABLE WITH SCHEMABINDING AS
   return SELECT 1 AS
[fn_securitypredicate_result] FROM
        StaffDuties d INNER JOIN Employees e
        ON (d.EmpId = e.EmpId)
        WHERE e.UserSID = SUSER_SID()
        AND @wing = d.Wing;
CREATE SECURITY POLICY dbo.SecPol
    ADD FILTER PREDICATE
dbo.fn_securitypredicate(Wing) ON Patients
    WITH (STATE = ON)
```

Fine-grained access control over rows in a table based on one or more pre-defined filtering criteria, such as user's role or clearance level in organization

Concepts:

- Predicate Function
- Security Policy

Create a Security Policy

```
--Create a new schema and predicate function, which will use the
--application user ID stored in SESSION_CONTEXT to filter rows.
CREATE SCHEMA Security;
GO
CREATE FUNCTION Security.fn_securitypredicate(@AppUserId int)
RETURNS TABLE
WITH SCHEMABINDING
AS
RETURN SELECT 1 AS fn_securitypredicate_result
WHERE DATABASE_PRINCIPAL_ID() =
DATABASE_PRINCIPAL_ID('AppUser') AND
CAST(SESSION_CONTEXT(N'UserId') AS int) = @AppUserId;
GO

--Create a security policy that adds this function as a filter
--predicate and a block predicate on Sales.
CREATE SECURITY POLICY Security.SalesFilter
ADD FILTER PREDICATE Security.fn_securitypredicate(AppUserId)
  ON dbo.Sales,
ADD BLOCK PREDICATE Security.fn_securitypredicate(AppUserId)
  ON dbo.Sales AFTER INSERT
WITH (STATE = ON);
```

Creates a security policy for row-level security

The following examples demonstrate the use of the CREATE SECURITY POLICY syntax

For an example of a complete security policy scenario, see Row-Level Security

Security Predicates

RLS supports two types of security predicates

- **Filter Predicates** silently filter rows available to read operations (SELECT, UPDATE, and DELETE)
- **Block Predicates** explicitly block write operations (AFTER INSERT, AFTER UPDATE, BEFORE UPDATE, BEFORE DELETE) that violate predicate*

Access to row-level data in table is restricted by security predicate defined as inline table-valued function, which is invoked and enforced by security policy

- For filter predicates, no indication to application that rows have been filtered from result set; if all rows are filtered, a null set will be returned
- For block predicates, any operations that violate predicate will fail with error

How Row-Level Security works

1) Policy Manager creates security predicate and applies it to Patients table
2) Nurse, logged in as Employee ID=100986, queries Patients table
3) Security predicate applies filter in T-SQL, binding the predicate to the Patients table

Nurse
(Employee ID=100986)

Application

```
SELECT Name FROM Patients
```

Database

Security Policy

Filter Predicate:
INNER JOIN...

Patients

Policy Manager

```
CREATE FUNCTION dbo.fn_securitypredicate(@wing int)
    RETURNS TABLE WITH SCHEMABINDING AS
    return SELECT 1 as [fn_securitypredicate_result] FROM
SELECT Name FROM Patients
   SEMIJOIN APPLY dbo.fn_securitypredicate(patients.Wing);
        WHERE e.UserSID = SUSER_SID() AND @wing = d.Wing;
SELECT Patients.Name FROM Patients,
   StaffDuties d INNER JOIN Employees e ON (d.EmpId = e.EmpId)
   WHERE e.UserSID = SUSER_SID() AND Patients.wing = d.Wing;

    WITH (STATE = ON)
```

How Row-Level Security works

4) Filtered rows are returned and nurse sees only the patients on her wing

Nurse
(Employee ID=100986)

Application

Database

Security Policy

Filter Predicate:
INNER JOIN…

Patients

Policy Manager

Patients

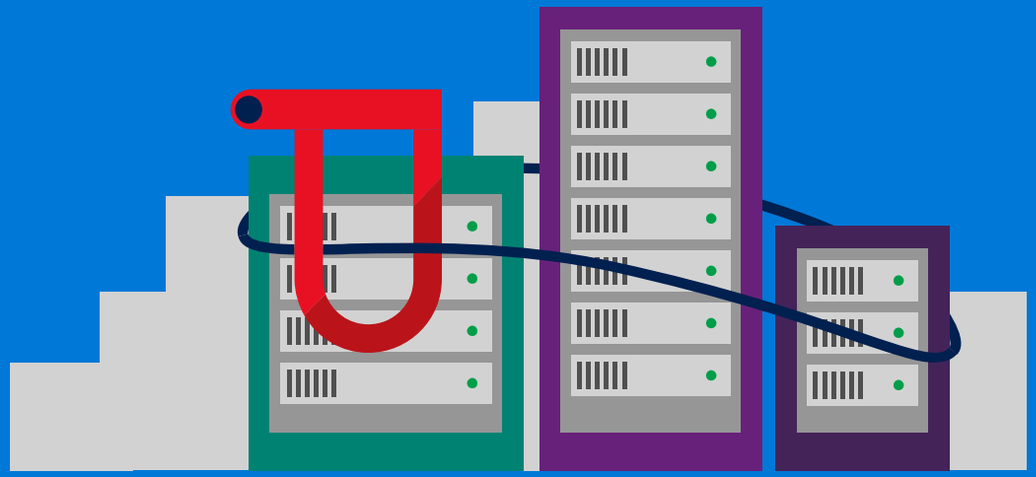
Wing	PatientId	Name
a1	100-207	Jones, Walker
b1	201-677	Smith, Weston
b2	898-045	Jarvis, Anne
a1	009-451	Rickmann, Earnst
b2	922-769	Gray, Jim
a1	801-345	Polka, Dorothy

StaffDuties

EmployeeId	Wing
100986	a1
206985	a2
345906	a3
331225	b1

Demonstration:
Leveraging Row Level Security

Dynamic Data Masking



How Dynamic Data Masking Works

Limit sensitive data exposure by obfuscating data to non-privileged users

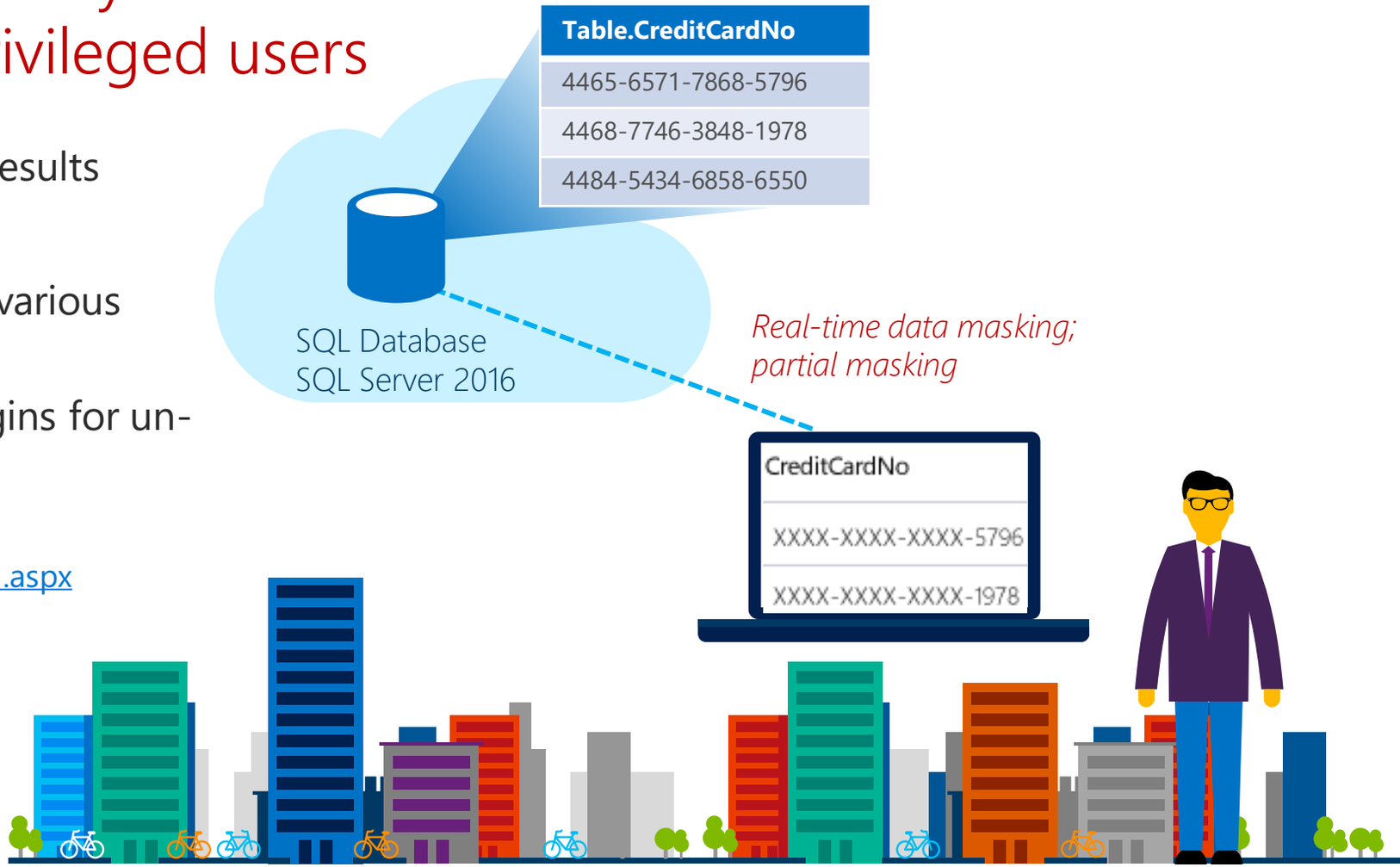
On-the-fly obfuscation of data in query results

Policy-driven on the table and column

Multiple masking functions available for various sensitive data categories

Flexibility to define a set of privileged logins for un-masked data access

By default, database owner is unmasked
https://msdn.microsoft.com/en-us/library/mt130841.aspx



Benefits of Dynamic Data Masking

Regulatory Compliance

A strong demand for applications to meet **privacy standards** recommended by regulating authorities

Sensitive Data Protection

Protects against unauthorized access to sensitive data in the application, and against exposure to developers or DBAs who need access to the production database

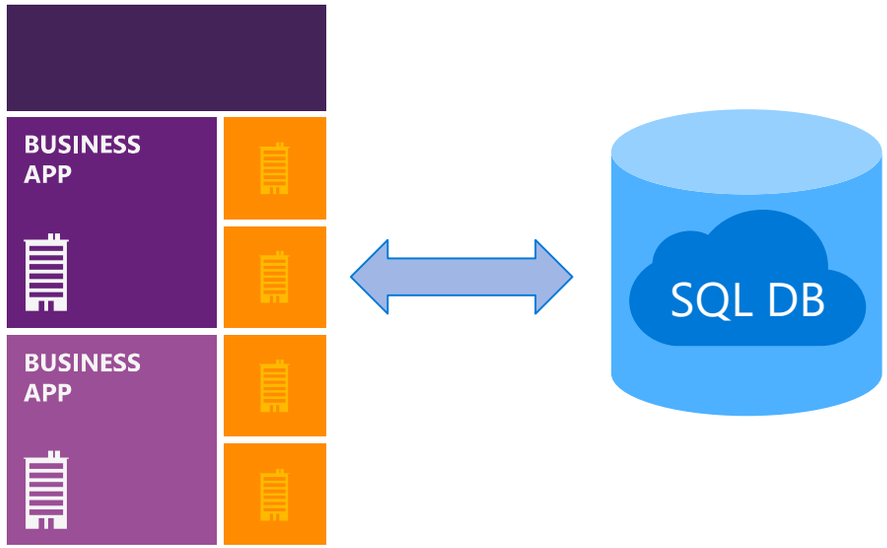
Agility and Transparency

Data is masked on the fly, with underlying data in the database remaining intact. Transparent to the application and applied according to user privilege

Limit access to sensitive data by defining policies to obfuscate specific database fields, without affecting the integrity of the database.

How Dynamic Data Masking Works

1) Security officer defines Employee table masking policy – T-SQL queries
2) Dynamic data masking obfuscates sensitive data in Employee table
3) Authorized users access Employee table data in SQL Server results




```
ALTER TABLE [Employee] ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'partial(0,"XXX-XX-",2)’)

ALTER TABLE [Employee] ALTER COLUMN [Email]
ADD MASKED WITH (FUNCTION = ‘EMAIL()’)

ALTER TABLE [Employee] ALTER COLUMN [Salary]
ADD MASKED WITH (FUNCTION = ‘RANDOM(1,20000)’)

GRANT UNMASK to hrsupervisor
```

Security Officer

```
SELECT [Name],
       [SocialSecurityNumber],
       [Email],
       [Salary]
FROM [Employee]
```

non-privileged login

	First Name	Social Security Number	Email	Salary
1	LILA	XXX-XX-XX37	lXX@XXXX.net	8940
2	JAMIE	XXX-XX-XX14	jXX@XXXX.com	19582
3	SHELLEY	XXX-XX-XX28	sXX@XXXX.net	3713
4	MARCELLA	XXX-XX-XX65	mXX@XXXX.net	11572
5	GILBERT	XXX-XX-XX87	gXX@XXXX.net	4487

hrsupervisor login

	First Name	Social Security Num...	Email	Salary
1	LILA	758-10-9637	lila.barnett@comcast.net	1012794
2	JAMIE	113-29-4314	jamie.brown@ntlworld.com	1025713
3	SHELLEY	550-72-2028	shelley.lynn@charter.net	1040131
4	MARCELLA	903-94-5665	marcella.estrada@comcast.net	1040753
5	GILBERT	376-79-4787	gilbert.juarez@verizon.net	1041308

Most secure database

Can be Combined with Row Level Security

4) Data masking obscures columns, row level security filters rows


```
ALTER TABLE [Employee] ALTER COLUMN [SocialSecurityNumber]
ADD MASKED WITH (FUNCTION = 'partial(0,"XXX-XX-",2)’)

ALTER TABLE [Employee] ALTER COLUMN [Email]
ADD MASKED WITH (FUNCTION = ‘EMAIL()’)

ALTER TABLE [Employee] ALTER COLUMN [Salary]
ADD MASKED WITH (FUNCTION = ‘RANDOM(1,20000)’)

GRANT UNMASK to hrsupervisor
```

Security Officer

```
SELECT [Name],
       [SocialSecurityNumber],
       [Email],
       [Salary]
FROM [Employee]
```

non-privileged login

	First Name	Social Security Number	Email	Salary
1	LILA	XXX-XX-XX37	lXX@XXXX.net	8940
2	JAMIE	XXX-XX-XX14	jXX@XXXX.com	19582
3	SHELLEY	XXX-XX-XX28	sXX@XXXX.net	3713
4	MARCELLA	XXX-XX-XX65	mXX@XXXX.net	11572
5	GILBERT	XXX-XX-XX87	gXX@XXXX.net	4487

hrsupervisor login

	First Name	Social Security Num...	Email	Salary
1	LILA	758-10-9637	lila.barnett@comcast.net	1012794
2	JAMIE	113-29-4314	jamie.brown@ntlworld.com	1025713
3	SHELLEY	550-72-2028	shelley.lynn@charter.net	1040131
4	MARCELLA	903-94-5665	marcella.estrada@comcast.net	1040753
5	GILBERT	376-79-4787	gilbert.juarez@verizon.net	1041308

Querying for Masked Columns

Use **sys.masked_columns** view to query for table columns that have a masking function applied to them.

This view inherits from **sys.columns** view. It returns all columns in **sys.columns** view, plus **is_masked** and **masking_function** columns—indicating if a column is masked, and if so, what masking function is defined.

This view only shows columns on which there is a masking function applied.

```
SELECT c.name, tbl.name AS table_name,
c.is_masked, c.masking_function
FROM sys.masked_columns AS c
JOIN sys.tables AS tbl
    ON c.[object_id] = tbl.[object_id]
WHERE is_masked = 1;
```

Configure Dynamic Data Masking

Use an ALTER TABLE statement to add a masking function to the required column in the table

```
USE AdventureWorks2014;
GO
ALTER TABLE Person.EmailAddress
ALTER COLUMN EmailAddress
ADD MASKED WITH (FUNCTION = 'email()');
```

Create a new user with SELECT permission on the table, and then execute a query to view masked data

```
CREATE USER TestUser WITHOUT LOGIN;
GRANT SELECT ON Person.EmailAddress TO TestUser;
```

Verify that the masking function changes the required column with a masked field

```
EXECUTE AS USER = 'TestUser';
SELECT EmailAddressID, EmailAddress FROM Person.EmailAddress;
REVERT;
```

On Premise vs. Azure: Implementing Data Masking

Implementation DDM

Requires a custom solution or SQL Server 2016

Implementation Challenges

Deployment challenges

"Trusting" engineers

Auditability

Est. Effort to Implement: 40 Hours

**Implementation on SQL Server 2014*

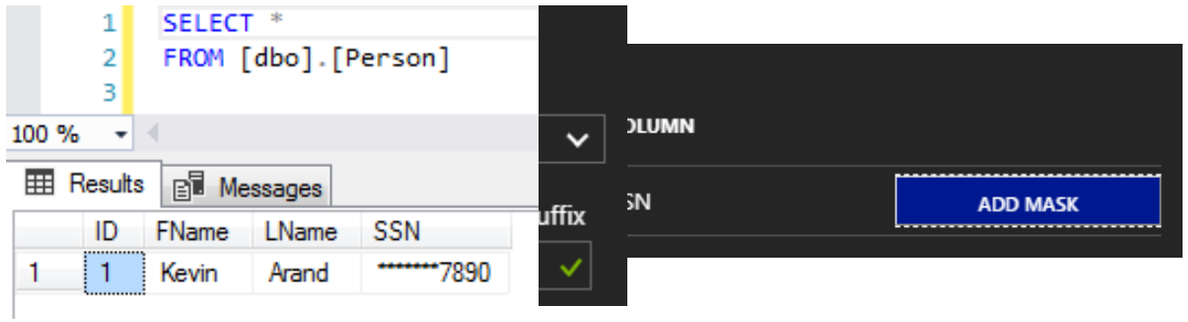
***Effort encompasses multiple environments*

Azure Implementation

Powershell

Rest API

Azure Portal

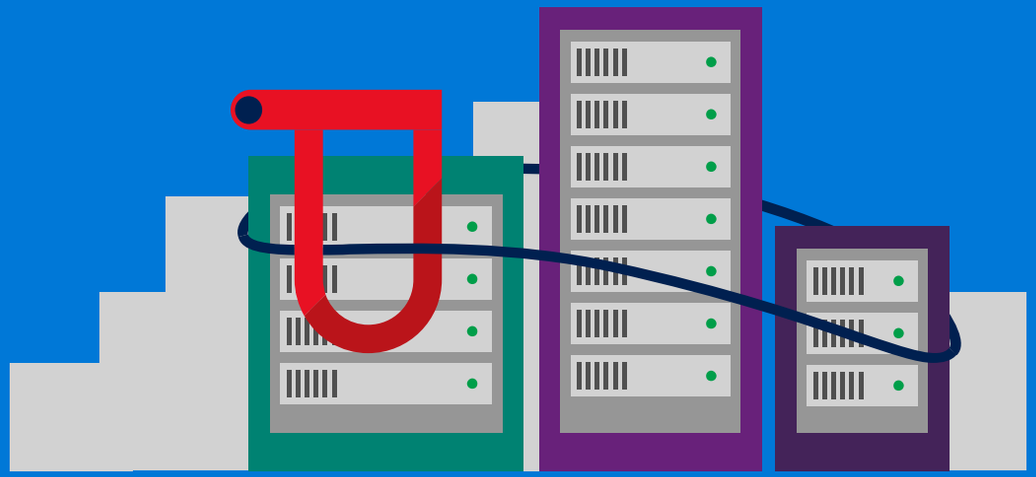


Dynamic Data Masking is built into the Azure Portal Giving Column Based Recommendations

Demonstration: Leveraging Dynamic Data Masking

- SQL Server 2016 / 2017
- Azure SQL DB Portal

Always Encrypted in SQL Server 2016 / 2017



The Need for Always Encrypted

Prevents Data Disclosure

Client-side encryption of sensitive data using keys that are *never* given to the database system

Queries on Encrypted Data

Support for equality comparison, including join, group by, and distinct operators

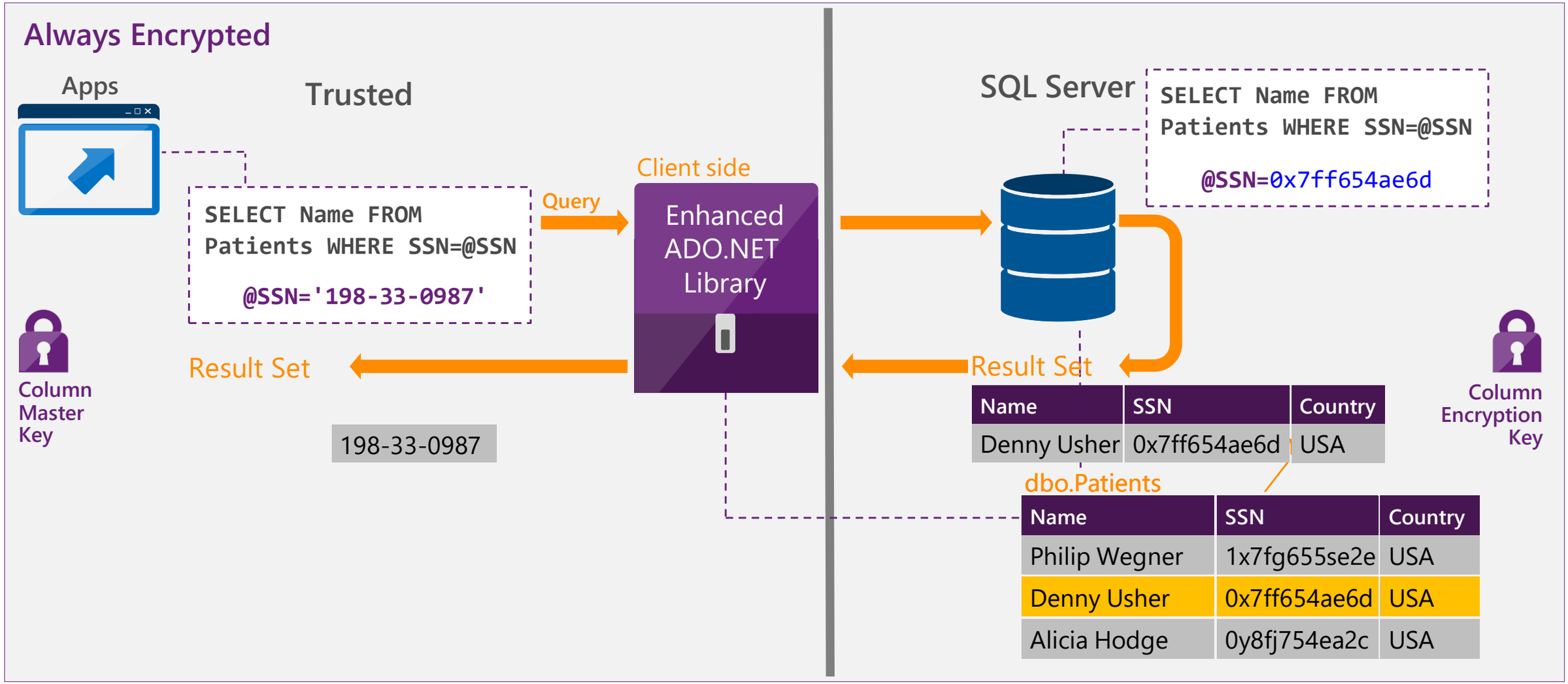
Application Transparency

Minimal application changes via server and client library enhancements

Allows customers to securely store sensitive data outside of their trust boundary.
Data remains protected from high-privileged, yet unauthorized users.

Protect your Data at Rest and In-Motion

without impacting database performance



Most secure database

Types of Encryption for Always Encrypted

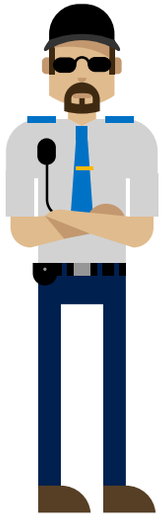
```
Randomized Encryption
   Encrypt('123-45-6789') = 0x17cfd50a
   Repeat: Encrypt('123-45-6789') = 0x9b1fcf32
   Allows for transparent retrieval of encrypted
   data but NO operations
   More secure

Deterministic Encryption
   Encrypt('123-45-6789') = 0x85a55d3f
   Repeat: Encrypt('123-45-6789') = 0x85a55d3f
   Allows for transparent retrieval of encrypted
   data AND equality comparison
   (i.e. in WHERE clauses and Joins, DISTINCT,
   GROUP BY)
```

Two Types of Encryption:

Randomized Encryption uses a method that encrypts data in a less predictable manner

Deterministic Encryption uses a method which always generates the same encrypted value for any given plaintext value

Key Provisioning



Security Officer

1. Generate CEKs and master key

Column encryption key (CEK)

Column master key (CMK)

2. Encrypt CEK

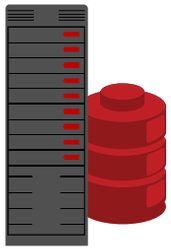
Encrypted CEK

3. Store master key securely

CMK store:
Certificate store
HSM
Azure Key Vault
…

CMK

4. Upload encrypted CEK to DB



Encrypted CEK Database

Always Encrypted T-SQL

```
CREATE COLUMN MASTER KEY MyCMK
WITH ( KEY_STORE_PROVIDER_NAME = ‘MSSQL_CERTIFICATE_STORE’,
KEY_PATH = ‘Current User / Personal /
f2260f28909d21c642a3d8e0b45a830e79a12420’ );
```

```
CREATE COLUMN ENCRYPTION KEY MyCEK
WITH VALUES
( COLUMN_MASTER_KEY = MyCMK,
ALGORITHM = ‘RSA_OAEP’,
ENCRYPTED_VALUE = ‘0x017000_64003);
```

```
CREATE TABLE Customers (
Customers nvarchar(60) COLLATE Latin1_General_BIN2 ENCRYPTED
WITH (COLUMN_ENCRYPTED_KEY = MyCEK,
ENCRYPTION_TYPE = RANDOMIZED, ALGORITHM = ‘AEAD_AES_256_CBC_HMAC_SHA_256’),

SSN varchar(11) COLLATE Latin1_General_BIN2 ENCRYPTED
WITH (COLUMN_ENCRYPTED_KEY = MyCEK,
ENCRYPTION_TYPE = DETERMINISTIC, ALGORITHM = ‘AEAD_AES_256_CBC_HMAC_SHA_256’), Age int NULL );
```

Advanced Threat Detection

Auditing & Threat Detection

Monitor and detect suspicious activities on your databases, and streamline compliance-related tasks.

Regulatory compliance

Auditing helps enterprise customers meet stringent regulatory requirements and security standards (e.g. PCI-DSS, HIPAA)

Intelligence of the cloud

Proprietary algorithms work around-the-clock to build a behavioral profile of your database, and identify anomalous activities and potential threats.

Investigate and mitigate

React and respond to threats in real-time, via email alerts and the Azure portal. Enterprise-grade security is easier than ever before.

Status: Auditing Generally Available, Threat Detection

Detects suspicious database activities

Threat Detection

HIGH 1 MEDIUM 3

DESCRIPTION	COUNT	DETECTED BY	DATE	STATE	SEVERITY
Potential SQL Injection	1	Microsoft	03/28/17	Active	High
Someone logged on to your SQL server from an an u...	1	Microsoft	03/29/17	Active	Medium
Logon by an unfamiliar principal	1	Microsoft	03/29/17	Active	Medium
A possible vulnerability to SQL Injection	1	Microsoft	03/28/17	Active	Medium

- ✓ Just turn it ON
- ✓ Detects potential vulnerabilities and SQL injection attacks
- ✓ Detects unusual behavior activities
- ✓ Actionable alerts which recommend how to investigate & remediate

*It costs $15/server/month , first 60 days for free.

Demonstration:
Azure Advanced Threat Detection

- SQL Server Injection Attacks
- Alerts and Monitoring

The General Data Protection Regulation (GDPR)

What are the key changes with the GDPR?

Personal privacy

Individuals have the right to:

- Access their personal data
- Correct errors in their personal data
- Erase their personal data
- Object to processing of their personal data
- Export personal data

Controls and notifications

Organizations will need to:

- Protect personal data using appropriate security
- Notify authorities within 72 hours of breaches
- Obtain appropriate consents for processing data
- Keep records detailing data processing

Transparent policies

Organizations are required to:

- Provide clear notice of data collection
- Outline processing purposes and use cases
- Define data retention and deletion policies

IT and training

Organizations will need to:

- Train privacy personnel & employees
- Audit and update data policies
- Employ a Data Protection Officer (if required)
- Create & manage compliant vendor contracts

Preparing for GDPR compliance

Questions for leading your preparation:

Do you know **WHERE** your data resides and who has **ACCESS** to that data?

Do you **CONTROL** who has access to your data and how it is **USED** based on risk assessment in **REAL-TIME**?

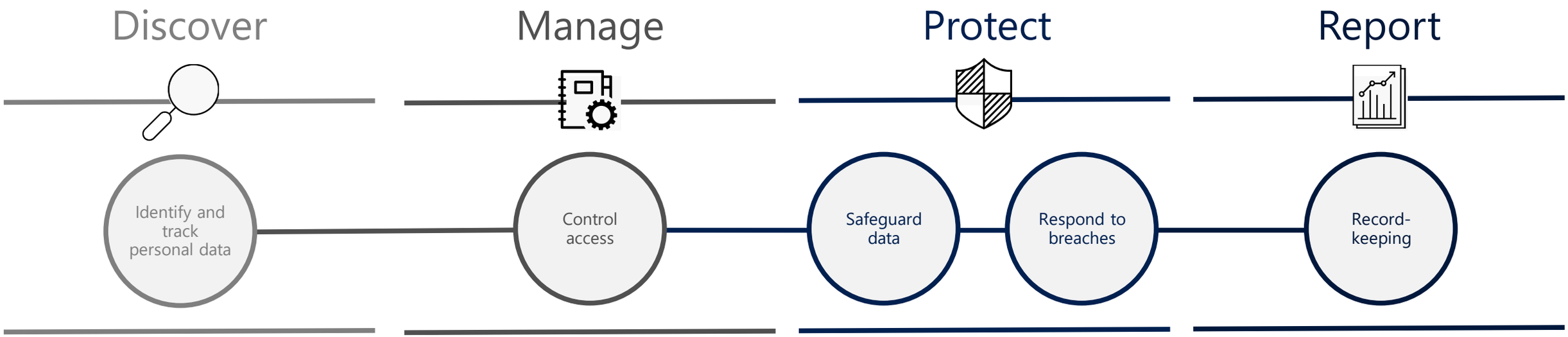
Can you **CLASSIFY**, **PROTECT** and apply **POLICY-driven** actions to your data, on devices, between apps, in any location, at rest and in transit?

Can you automatically **DETECT** a data or identity breach? Are you able to **RESPOND** adequately to a breach?

Do you continuously **REVIEW** and **UPDATE** your data protection **POLICIES** and **PRACTICES**?

SOLUTIONS TO HELP YOU
PREPARE FOR THE GDPR

How does SQL Server help today?



- Easily query databases to uncover personal data
- Tag data with sensitivity labels using Extended Properties

- Securely authenticate to your database and apply granular authorization policies
- Restrict access to users using Dynamic Data Masking and Row-Level Security

- Encrypt data whether at rest, in transit or in client applications
- Track and log database events to identify potential threats or security violations
- Use continuously learning algorithms to identify unusual or suspicious activity

- Track and report on all database activities with granularly configurable auditing

Guide to enhancing privacy and addressing GDPR requirements with the Microsoft SQL platform
http://download.microsoft.com/download/4/9/4/4948194B-A613-49ED-90A5-5144313549AB/microsoft-sql-and-the-gdpr.pdf

GDPR Translated to SQL Technologies…

1 **Discover**	Inventory personal data in database systems Review access model, understand the attack surface area Track data flows and map data lineage	T-SQL Queries, Full Text search Data Classification new! Vulnerability Assessment new!
2 **Manage**	Manage authentication and authorization mechanisms Properly configure database firewall Limit application access according to authorization principles	Windows auth, Azure AD auth, role-base security, etc. Azure SQL Firewall DDM, RLS
3 **Protect**	Encryption of data at rest, in motion, in use Maintain records and audits of all database activities Detect data breach and respond accordingly Ensure business continuity	TLS, TDE, Always Encrypted Auditing Threat Detection Always On, Active Geo-Replication
4 **Report**	Maintain audit records of database activities Continuously assess and analyze security measures	Auditing, Temporal tables Vulnerability Assessment new!

Translated to SQL Technologies…

1 Discover

Inventory personal data in database systems

Review access model, understand the attack surface area

Track data flows and map data lineage

T-SQL Queries, Full Text search

Data classification new!

Vulnerability Assessment new!

Query editor

Run Login Open query

```
SELECT product_id FROM products WHERE
CONTAINS(prod_desc, "Underwater music player"
OR
FORMSOF(THESAURUS, 'water music')) AND ...
```

Data classification

Save Cancel Add column View report Column labels Feedback

We have found 10 columns that you could classify. Click here to view them.

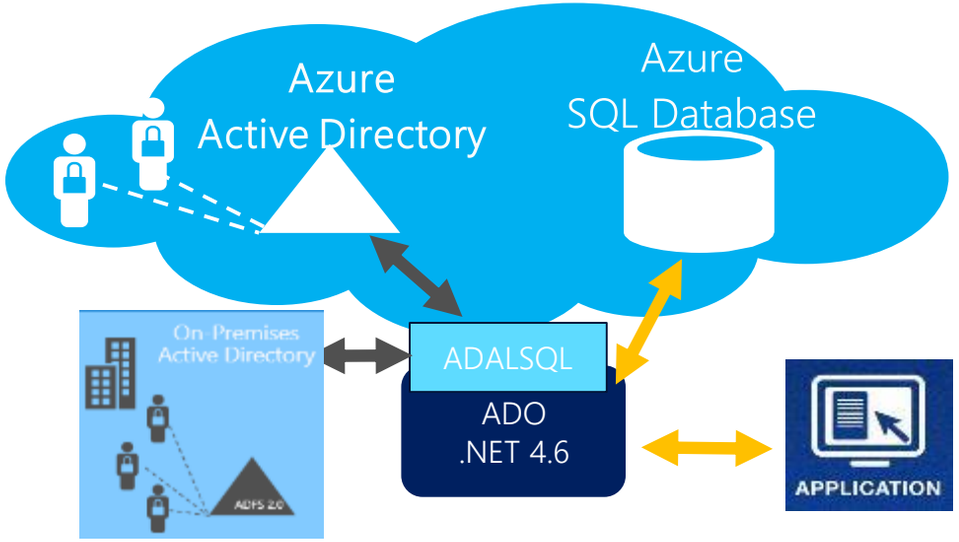
5 Columns classified

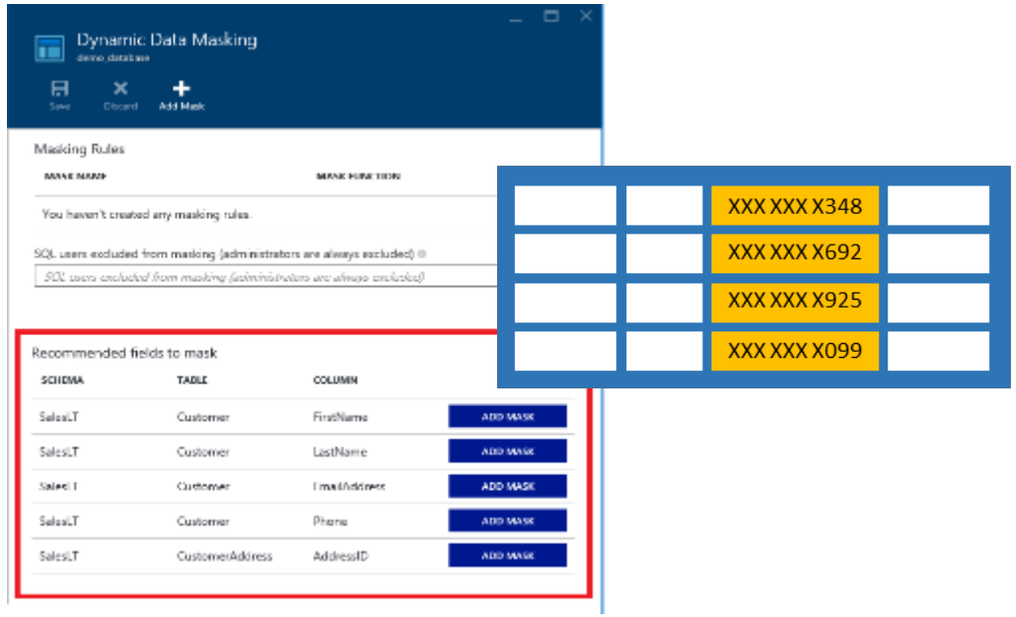
Filter by name... Schemas: All Tables: All Sensitivity: All

SCHEMA	TABLE	COLUMN	INFO TYPE	SENSITIVITY LABEL
dbo	sqli_creditcards	ccNumber	Credit card	Sensitive
dbo	sqli_customers	FirstName	Name	Public
dbo	sqli_customers	LastName	Name	PII
dbo	sqli_customers	Email	Contact info	PII

Translated to SQL Technologies…

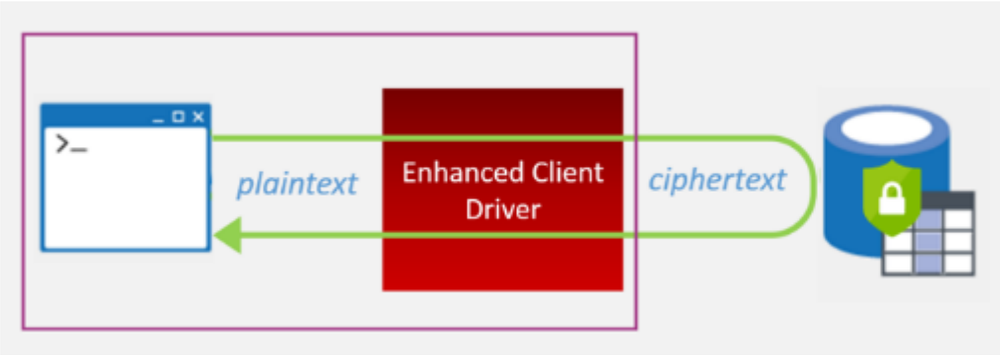
2 **Manage**	Manage authentication and authorization mechanisms	Windows authentication, Azure AD auth, role-base security…
	Properly configure database firewall	Azure SQL Firewall
	Limit application access according to authorization principles	Dynamic Data Masking, Row-Level Security

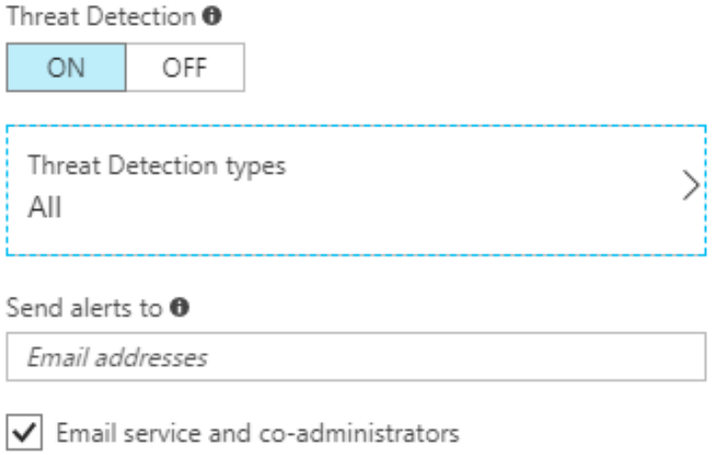




Translated to SQL Technologies…

3 **Protect**	Encryption of data at rest, in motion, in use Maintain records and audits of all database activities Detect data breach and respond accordingly Ensure business continuity	TLS, TDE, Always Encrypted Auditing, Threat Detection Always On, Active Geo-Replication





Translated to SQL Technologies…

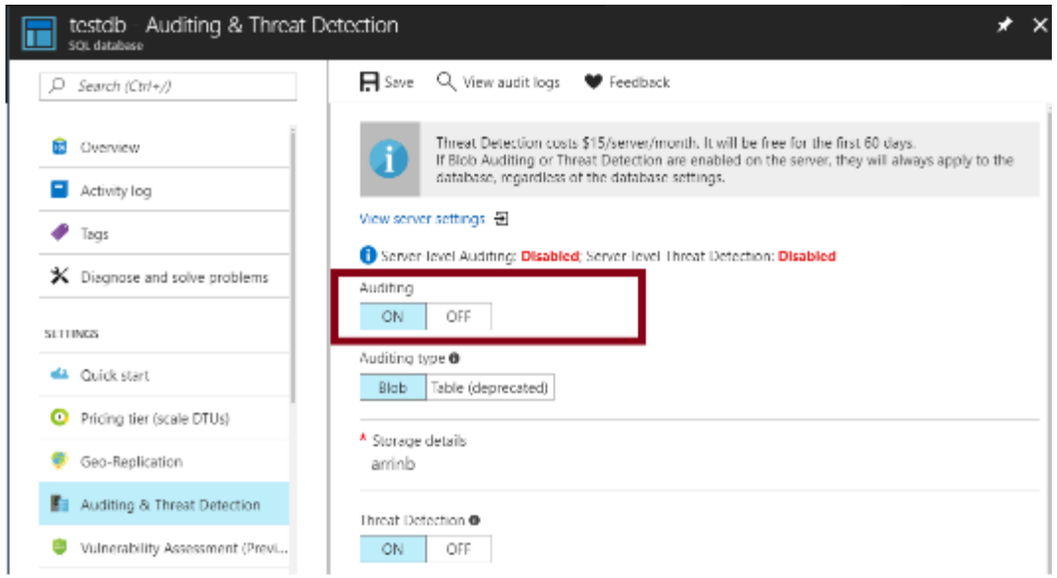
4 Report

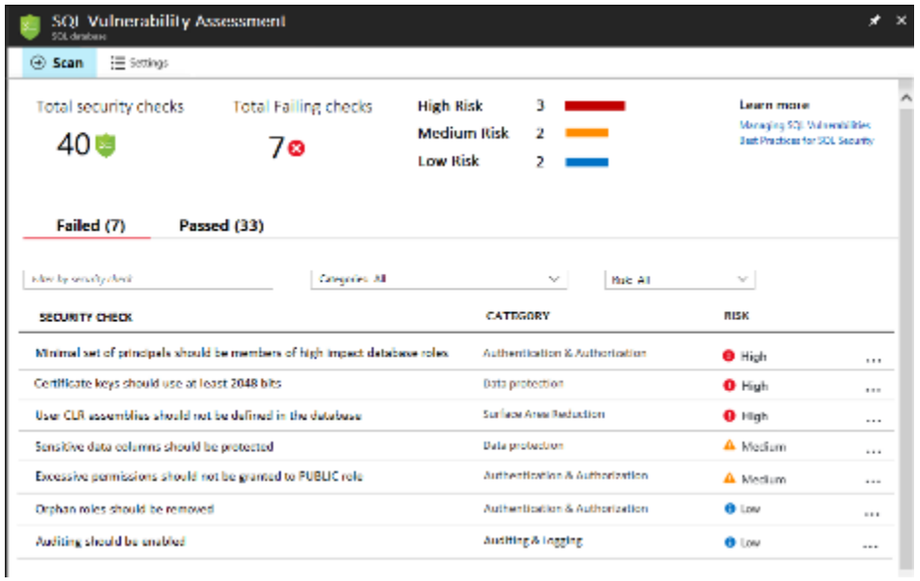
Maintain audit records of database activities

Continuously assess and analyze security measures

Auditing, Temporal tables

Vulnerability Assessment new!



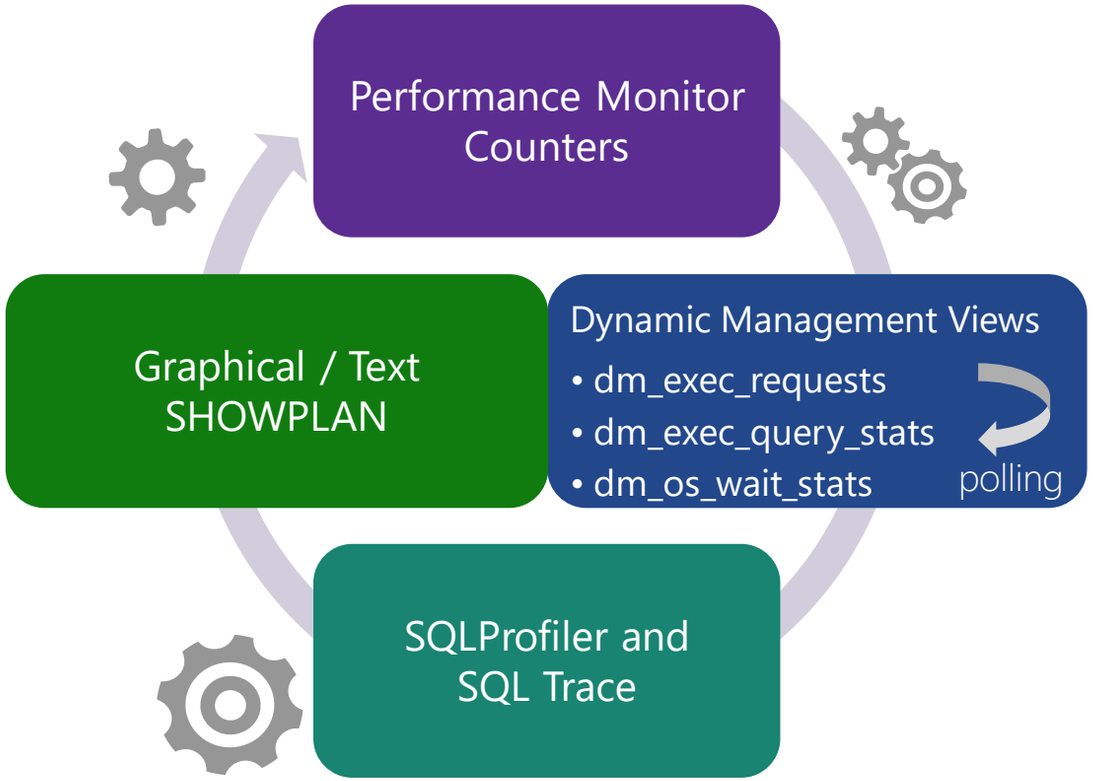


SQL Server Management Studio

New Features in SSMS 17.4+

SQL Server 2016/2017 Monitoring and Tooling

Traditional Troubleshooting

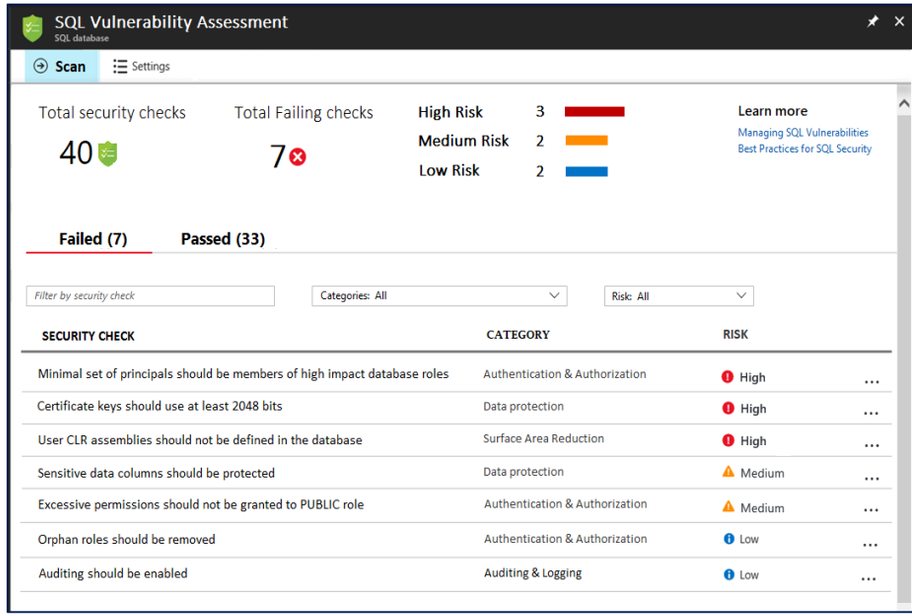


Full SQLTrace Parity+ since 2012

- Extended Events is scalable
- Query Store is persisted and improving
- Performance Dashboard Reports
- Live Query Statistics
- Lightweight Query Profiling
- Expanded Query Plan Diagnostics
- SSMS Dump Analysis (Preview)
- SQL Server Vulnerability Assessment
- SQL Data Discovery and Classification

Why Vulnerability Assessment?

Your first stop to track and improve the security of SQL database



Get Visibility

Discover sensitive data and potential security vulnerabilities

Remediate

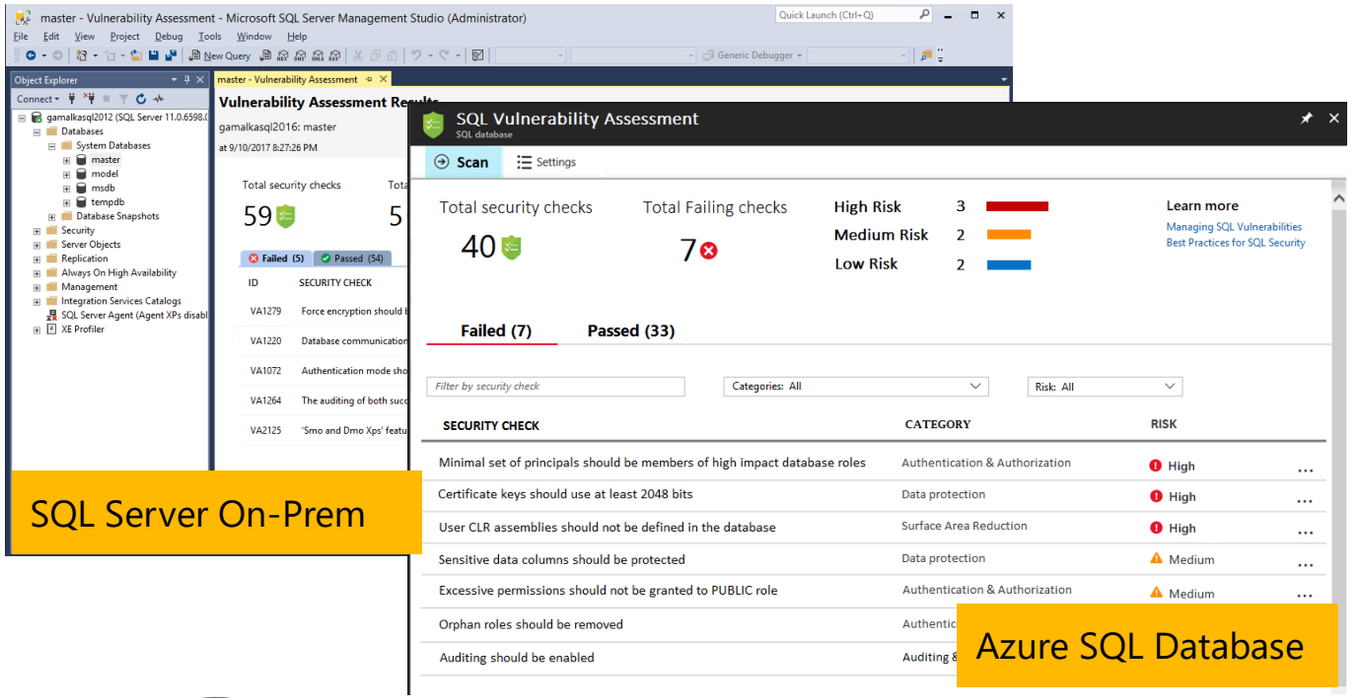
Actionable steps to help remediate any gaps and harden your security defenses

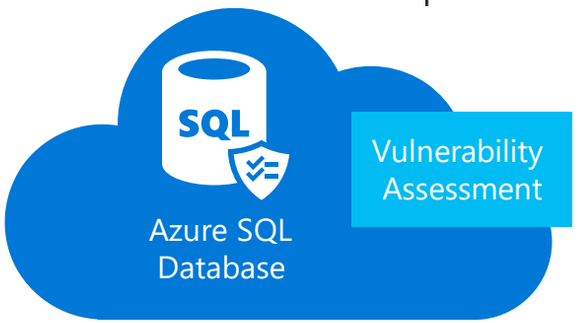
Customize

Tune the assessment to your specific environment with configurable policies so you can focus on deviations

SQL Server Vulnerability Assessment (17.4+)

A One-Stop-Shop to Track and Improve your SQL Security State





Identifies , tracks , resolves SQL security vulnerabilities

- **Just run a scan**
- **Discover sensitive data that is not protected**
- **Identify & remediate security misconfigurations**
- **Coherent report that helps meet compliance requirements**
- **SQL Server 2017** and **Azure SQL Database**

How Does VA Work?

1. Run a scan.
Scanning Service built-in to Azure SQL DB

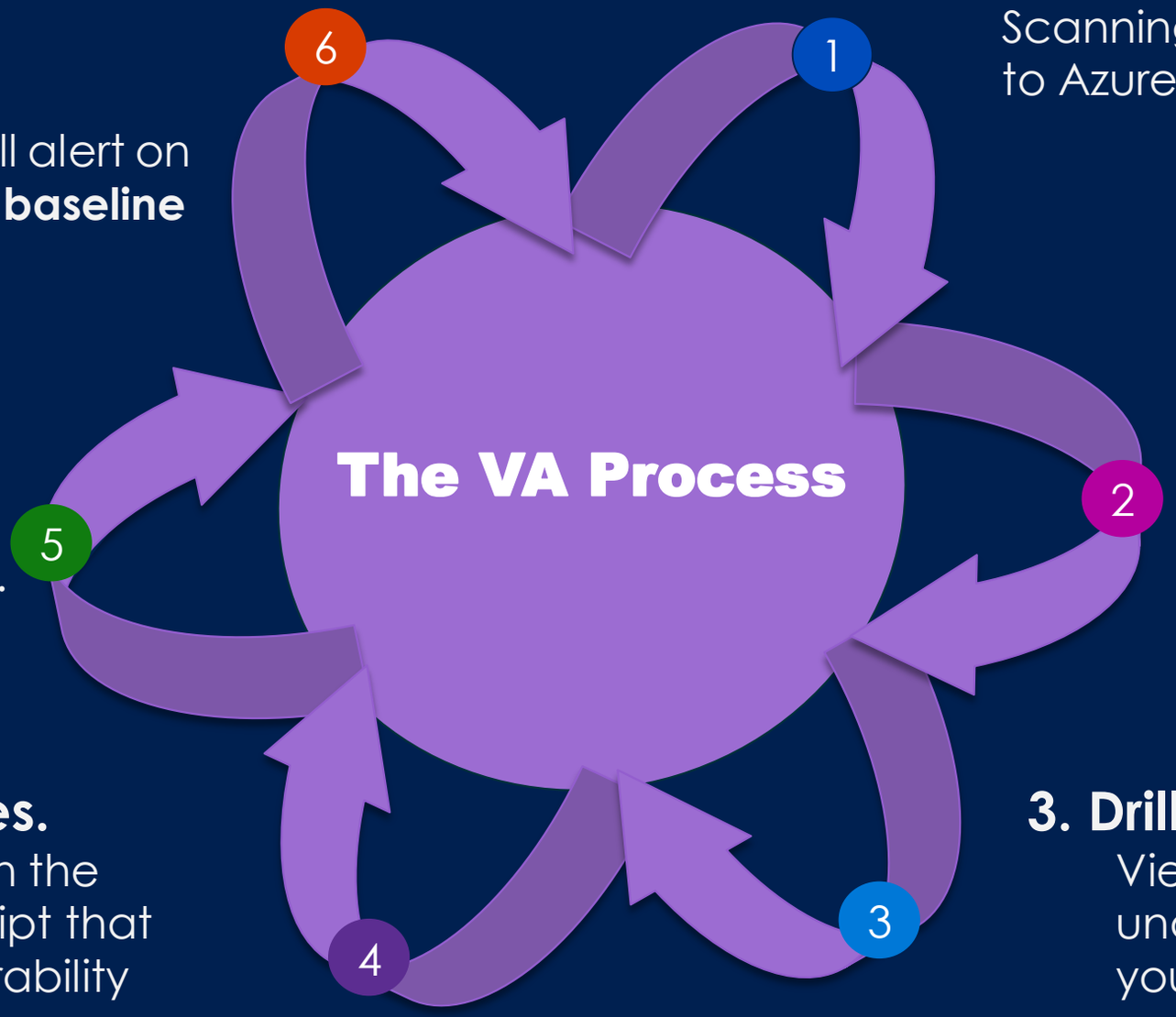
2. View a report.
All-up assessment of security state

3. Drill-down to results.
View detailed results and understand how it impacts your DB security

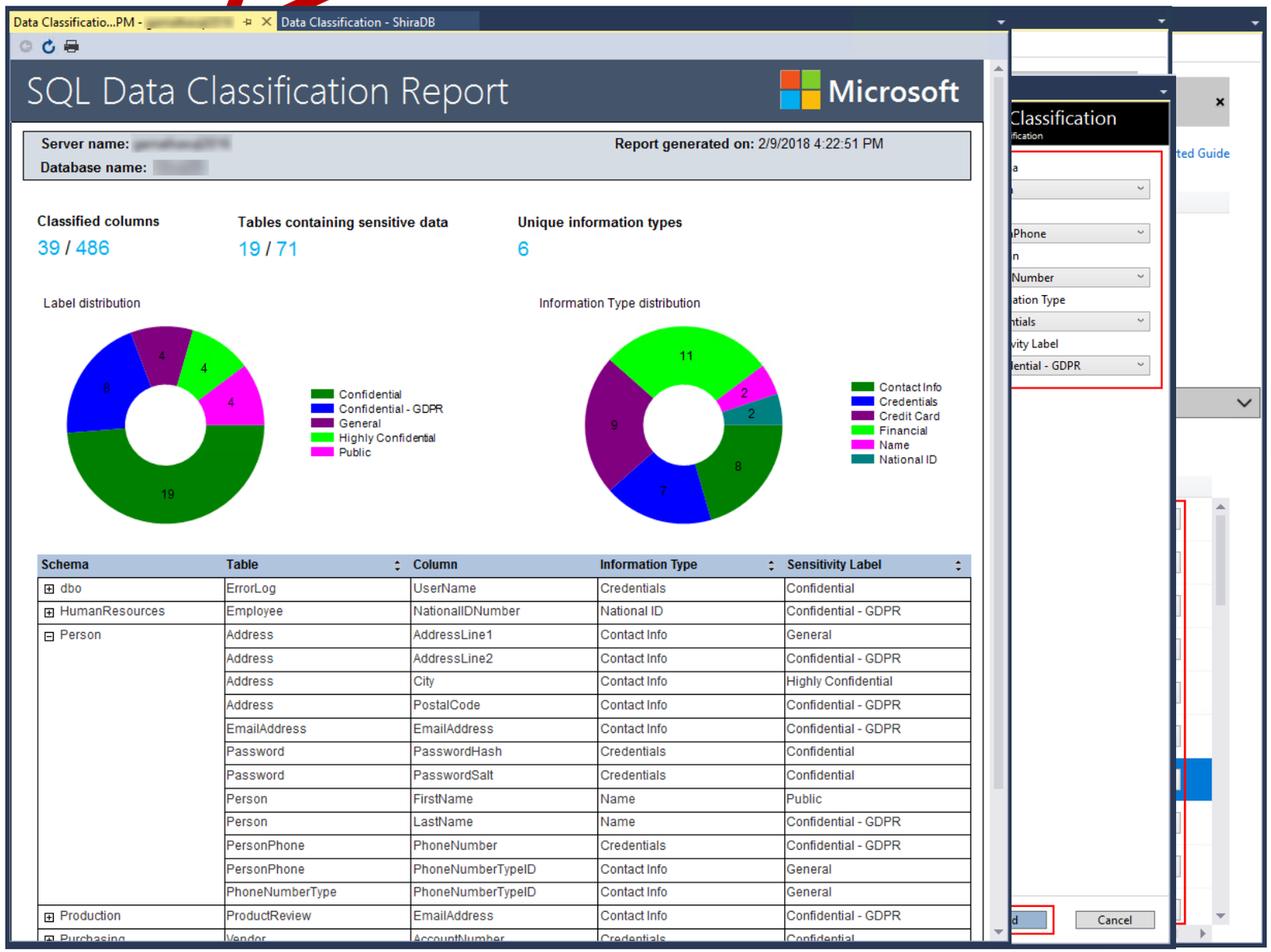
4. Remediate issues.
Directly from within the report, e.g. run script that resolves the vulnerability

5. Set a BASELINE.
Customize scan requirements based on **your** environment.

6. Detect deviations.
Subsequent scans will alert on **deviations from your baseline**



SQL Data Discovery and Classification (17.5+)



- New tool built into SQL Server Management Studio (SSMS)
- For **discovering**, **classifying**, **labeling** & **reporting** the sensitive data
  - (Financial, healthcare, PII, etc.)
- Helping meet data privacy standards and regulatory compliance requirements, such as GDPR.
- Controlling access to and hardening the security of databases/columns containing highly sensitive data.
- Data Discovery & Classification is **supported for SQL Server 2008 and later**.

Demonstration:
SQL Server Management Studio Improvements (17.5)

- SQL Server Vulnerability Assessment
- SQL Data Discovery and Classification

SQL Server

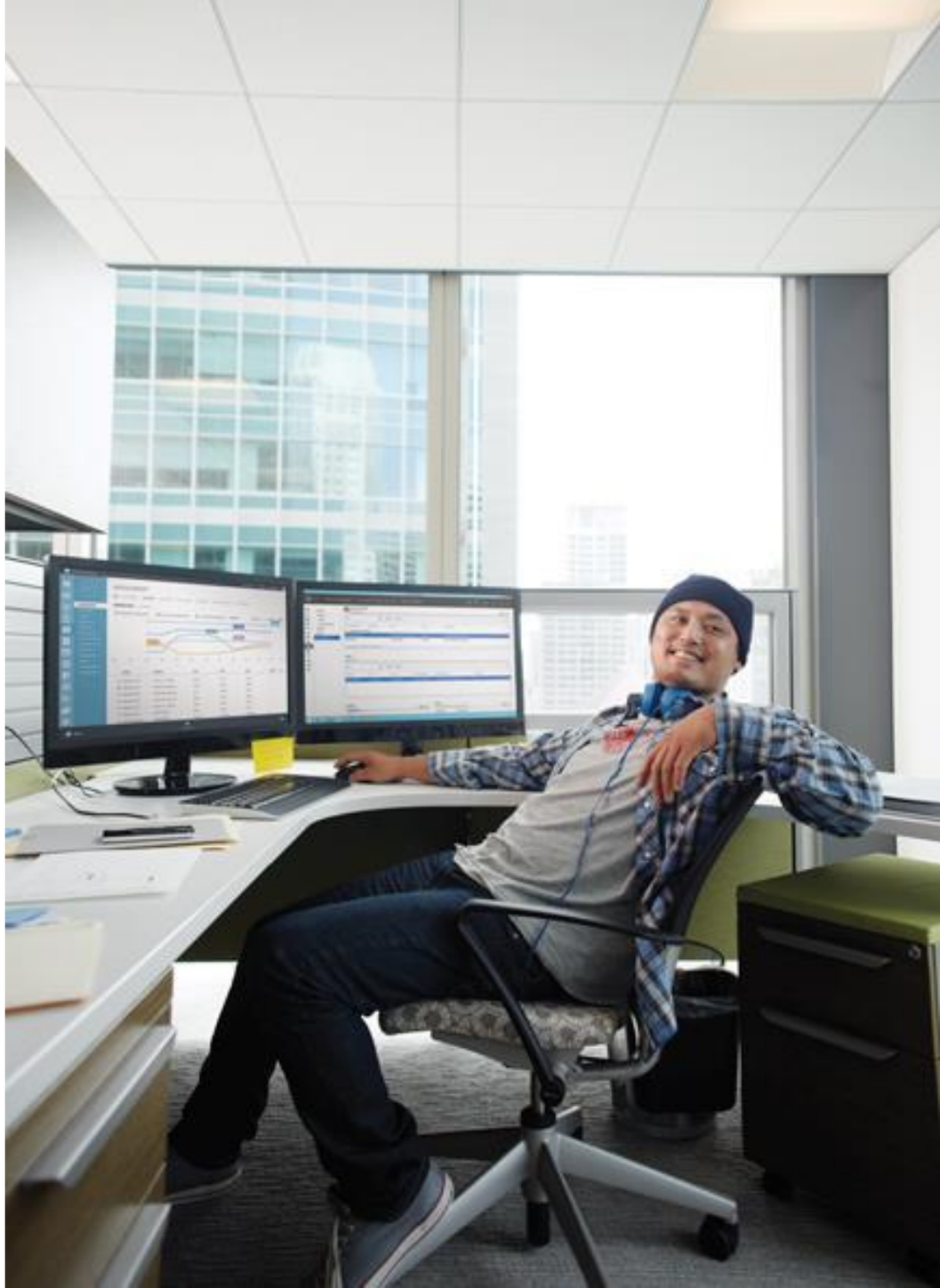
Protect the data inside your databases with controls for managing access and authorization at several levels

Discover

- SQL Query Language

Protect

- Azure SQL Database firewall
- SQL Server authentication
- Dynamic Data Masking (DDM)
- Row-Level Security (RLS)
- Transparent Data Encryption
- Always Encrypted
- Auditing for SQL Database and SQL Server audit
- SQL Database Threat Detection



Protect your Windows devices against Spectre and Meltdown

Guidance varies between OEM / Device manufacturers, products, and roles (Examples:)

- Azure Stack guidance:
  KB4073418: Azure stack guidance to protect against the speculative execution side-channel vulnerabilities
- SQL Server guidance:
  KB4073225: SQL Server Guidance to protect against speculative execution side-channel vulnerabilities
- SCCM guidance:
  Additional guidance to mitigate speculative execution side-channel vulnerabilities
- IT Pro Guidance:
  Windows Client Guidance for IT Pros to protect against speculative execution side-channel vulnerabilities
- Windows for Business blog:
  Windows Analytics now helps assess Meltdown and Spectre protections
- Consumer Guidance:
  Protecting your device against chip-related security vulnerabilities

References

- SpectreAttacks: Exploiting Speculative Execution
  https://spectreattack.com/spectre.pdf
- Meltdown
  https://meltdownattack.com/meltdown.pdf
- Protect SQL Server from attacks on Spectre and Meltdown side-channel vulnerabilities
  https://support.microsoft.com/en-us/help/4073225/guidance-protect-sql-server-against-spectre-meltdown
- PowerShell Script to patch Meltdown/Spectre Exploits for Windows Server
  https://gallery.technet.microsoft.com/scriptcenter/Meltdown-Spectre-Script-3cd11f26
- Cloud Cybersecurity in Healthcare: Thoughts on Spectre & Meltdown
  https://enterprise.microsoft.com/en-us/articles/industries/health/cloud-cybersecurity-in-healthcare-thoughts-on-spectre-meltdown/
- SQL Whitepaper guiding customers (SQL and GDPR Guide)
  https://aka.ms/gdprsqlwhitepaper
- SQL Server Security Blog
  http://blogs.msdn.microsoft.com/sqlsecurity/
- SQL Server Security | Microsoft Docs
  https://www.microsoft.com/GDPR/
  https://www.gdprbenchmark.com/

Microsoft