A Developer's Guide to Protecting Your Data Inside of SQL Server

David Berry

@DavidCBerry13

https://github.com/DavidCBerry13/sql-server-security/

Security is a **Team** Sport

Plan for a **defense in depth**

You **cannot** rely on perimeter defenses

In 2018, the **perimeter is everywhere**

What We Will Talk About

Users, Schemas and Roles

Connection Strings

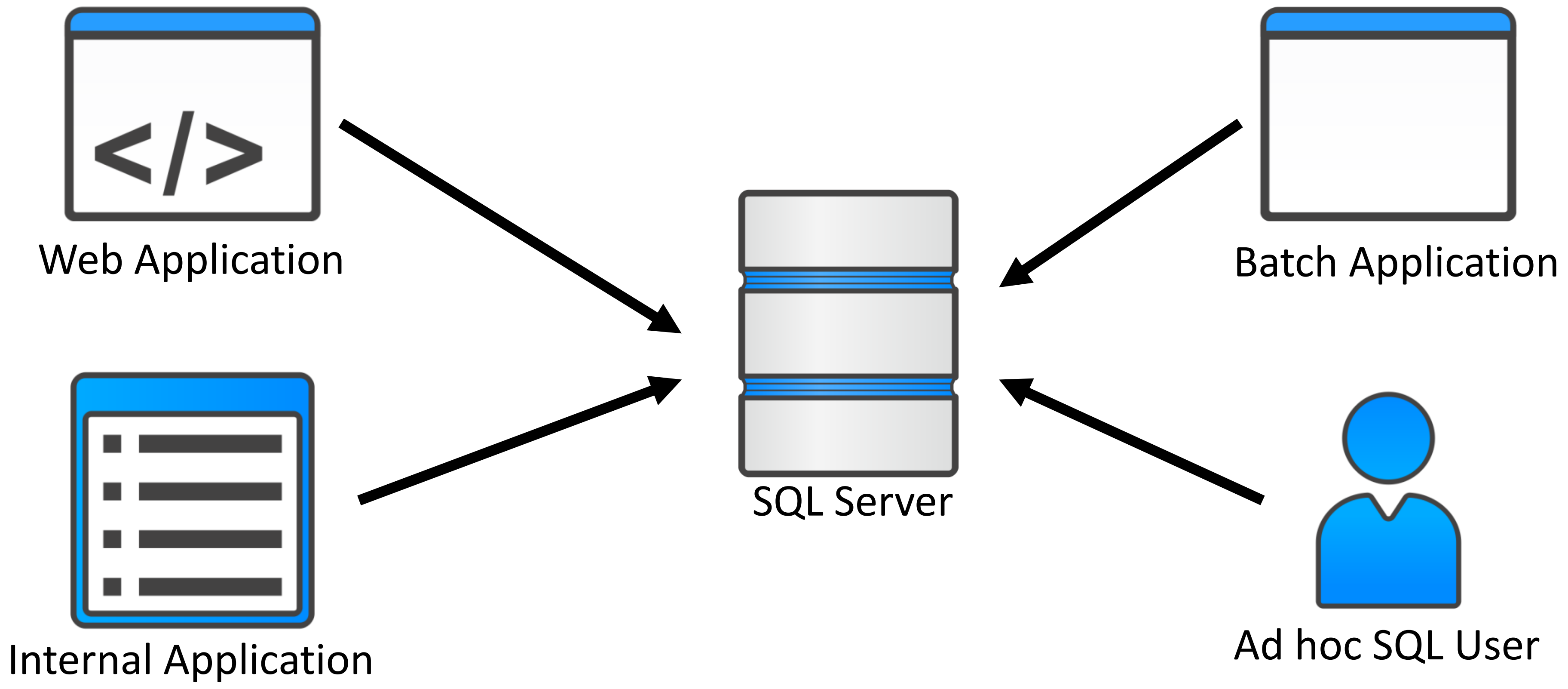
Transport Layer Security

Encrypting Data

SQL Injection

Users, Schemas and Roles

Overused and Overprivileged Users



Problem is when all these users use the same account

Assigning proper users, roles and privileges is your first line of defense

Secure Database User Practices

- Use Windows Authentication
- Have a separate service account for each application
- Use AD groups to For user access
- Have a separate dedicated account to push schema changes
- Do not assign db_owner to any application or user account
- Use database roles to assign permissions

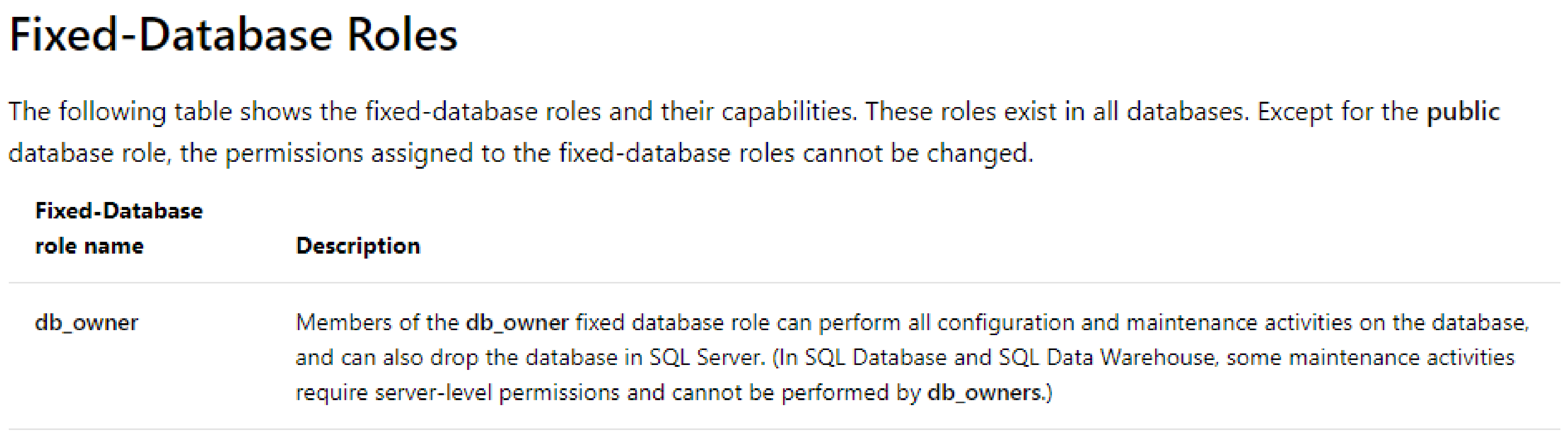
Why Use Windows Authentication

Eliminates an additional password

Take advantage of existing processes

- Especially around employee transfers and termination

db_owner Role



Fixed-Database Roles

The following table shows the fixed-database roles and their capabilities. These roles exist in all databases. Except for the **public** database role, the permissions assigned to the fixed-database roles cannot be changed.

Fixed-Database role name	Description
db_owner	Members of the **db_owner** fixed database role can perform all configuration and maintenance activities on the database, and can also drop the database in SQL Server. (In SQL Database and SQL Data Warehouse, some maintenance activities require server-level permissions and cannot be performed by **db_owners**.)

https://docs.microsoft.com/en-us/sql/relational-databases/security/authentication-access/database-level-roles?view=sql-server-2017

What’s Wrong with the db_owner Role?

Do you really want an application user to be able to create or drop tables?

How about modify your stored procedures?

Or grant and revoke permissions from other users?

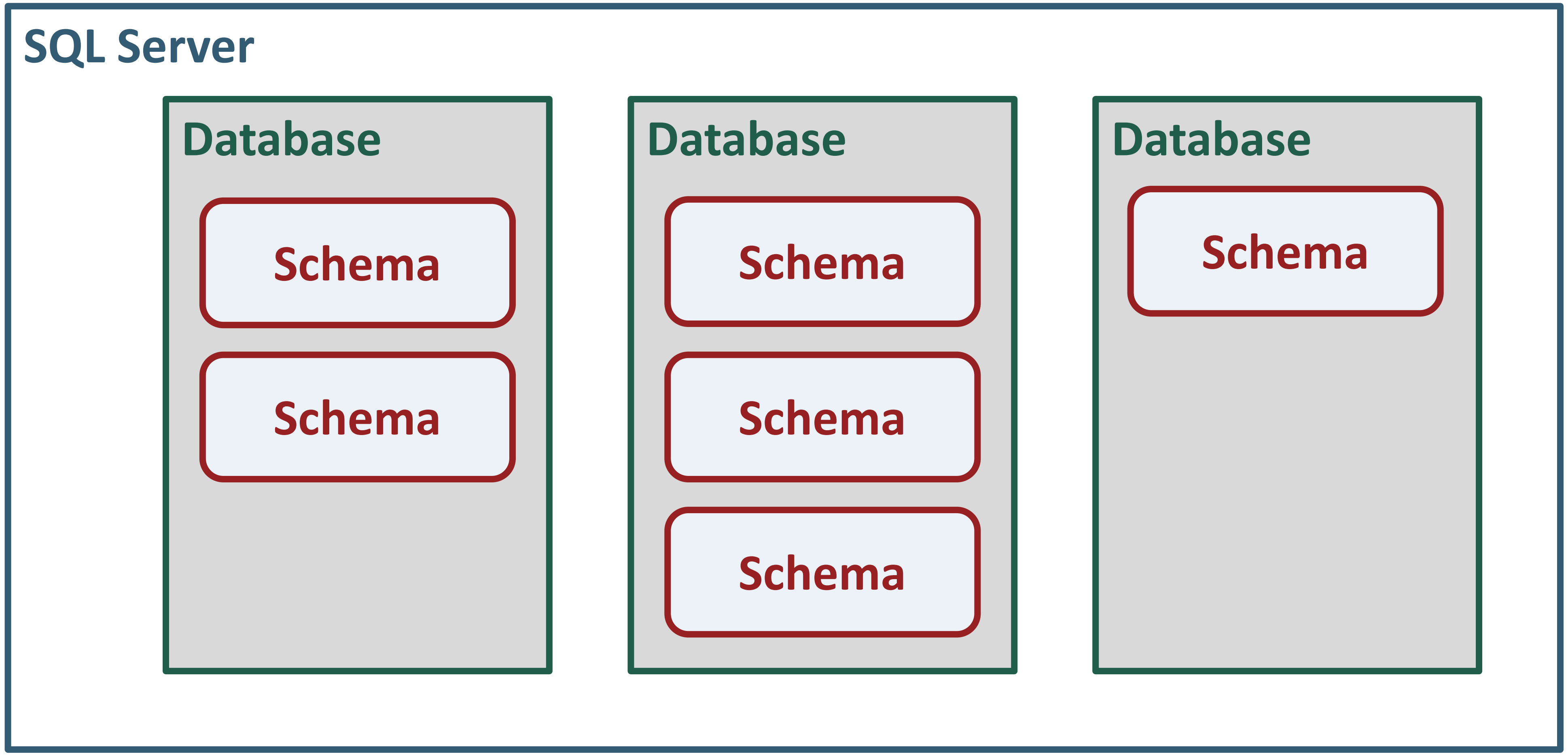
What Is a Database Role

A collection of privileges. Roles help you manage permissions in a database so that you can assign a group or privileges together

Creating Database Roles

```
CREATE ROLE Operations;



GRANT SELECT ON Customers TO Operations;
GRANT SELECT ON Addresses TO Operations;
GRANT SELECT ON CustomerAddresses TO Operations;
GRANT SELECT ON Orders TO Operations;
GRANT SELECT ON OrderItems TO Operations;




ALTER ROLE Operations
    ADD MEMBER [MyDomain\OperationsTeam];
```

Schemas

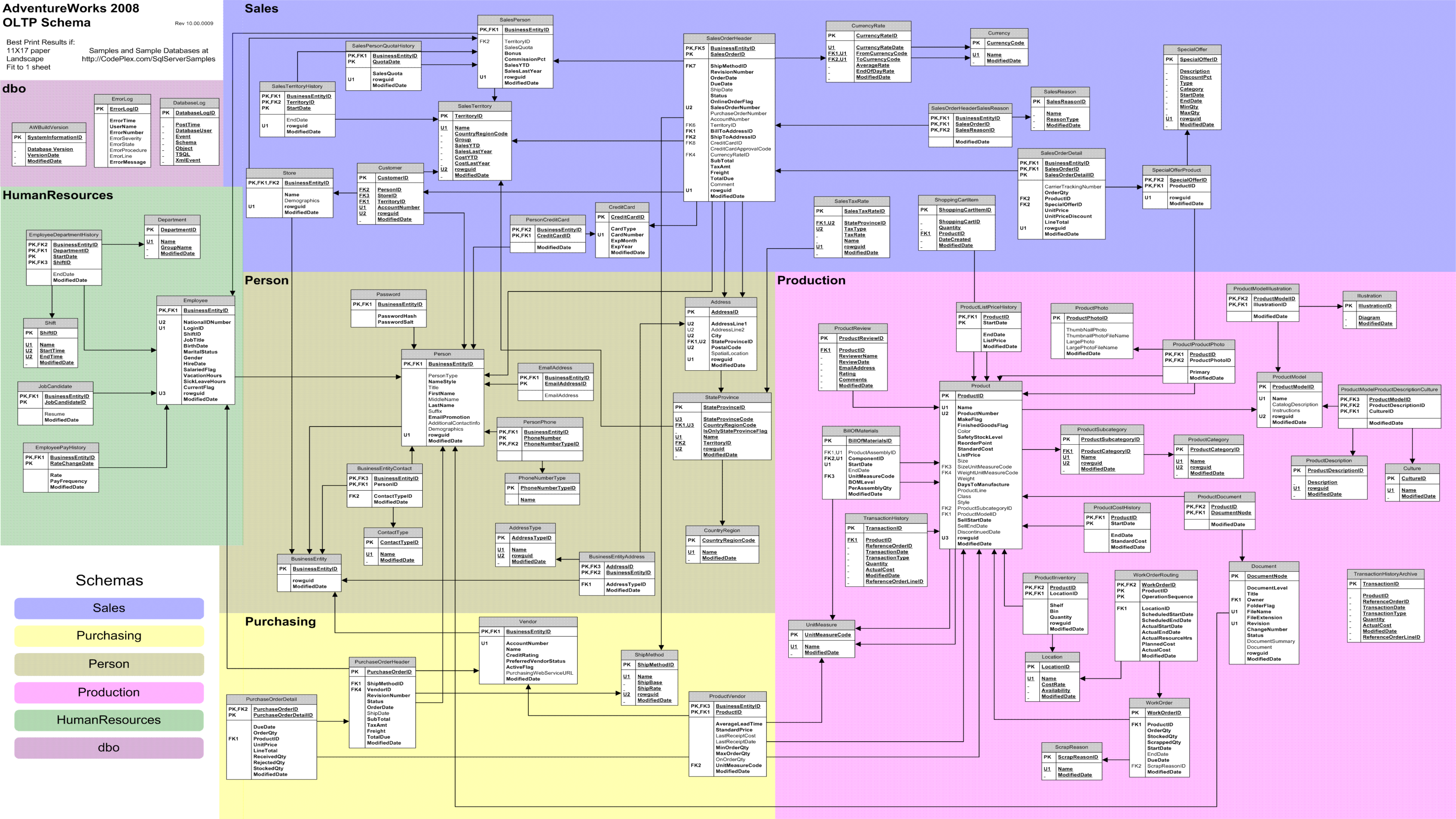

SQL Server
Database
Schema
Schema
Database
Schema
Schema
Schema
Database
Schema

AdventureWorks 2008 OLTP Schema

Rev 10.00.0009

Best Print Results if:
11X17 paper
Landscape
Fit to 1 sheet

Samples and Sample Databases at
http://CodePlex.com/SqlServerSamples



Sales
dbo
HumanResources
Person
Production
Purchasing
AWBuildVersion: PK SystemInformationID; Database Version, VersionDate, ModifiedDate
ErrorLog: PK ErrorLogID; ErrorTime, UserName, ErrorNumber, ErrorSeverity, ErrorState, ErrorProcedure, ErrorLine, ErrorMessage
DatabaseLog: PK DatabaseLogID; PostTime, DatabaseUser, Event, Schema, Object, TSQL, XmlEvent
SalesPerson: PK,FK1 BusinessEntityID; FK2 TerritoryID, SalesQuota, Bonus, CommissionPct, SalesYTD, SalesLastYear, U1 rowguid, ModifiedDate
SalesPersonQuotaHistory: PK,FK1 BusinessEntityID; PK QuotaDate; SalesQuota, U1 rowguid, ModifiedDate
SalesTerritoryHistory: PK,FK1 BusinessEntityID; PK,FK2 TerritoryID; PK StartDate; EndDate, U1 rowguid, ModifiedDate
SalesTerritory: PK TerritoryID; U1 Name, CountryRegionCode, Group, SalesYTD, SalesLastYear, CostYTD, CostLastYear, U2 rowguid, ModifiedDate
Store: PK,FK1,FK2 BusinessEntityID; Name, Demographics, U1 rowguid, ModifiedDate
Customer: PK CustomerID; FK2 PersonID, FK3 StoreID, FK1 TerritoryID, U1 AccountNumber, U2 rowguid, ModifiedDate
SalesOrderHeader: PK,FK5 BusinessEntityID; PK SalesOrderID; FK7 ShipMethodID, RevisionNumber, OrderDate, DueDate, ShipDate, Status, OnlineOrderFlag, U2 SalesOrderNumber, PurchaseOrderNumber, AccountNumber, FK6 TerritoryID, FK1 BillToAddressID, FK2 ShipToAddressID, FK8 CreditCardID, CreditCardApprovalCode, FK4 CurrencyRateID, SubTotal, TaxAmt, Freight, TotalDue, Comment, U1 rowguid, ModifiedDate
CurrencyRate: PK CurrencyRateID; U1 CurrencyRateDate, FK1,U1 FromCurrencyCode, FK2,U1 ToCurrencyCode, AverageRate, EndOfDayRate, ModifiedDate
Currency: PK CurrencyCode; U1 Name, ModifiedDate
SpecialOffer: PK SpecialOfferID; Description, DiscountPct, Type, Category, StartDate, EndDate, MinQty, MaxQty, U1 rowguid, ModifiedDate
SalesReason: PK SalesReasonID; Name, ReasonType, ModifiedDate
SalesOrderHeaderSalesReason: PK,FK1 BusinessEntityID; PK,FK1 SalesOrderID; PK,FK2 SalesReasonID; ModifiedDate
SalesOrderDetail: PK,FK1 BusinessEntityID; PK,FK1 SalesOrderID; PK SalesOrderDetailID; CarrierTrackingNumber, OrderQty, FK2 ProductID, FK2 SpecialOfferID, UnitPrice, UnitPriceDiscount, LineTotal, U1 rowguid, ModifiedDate
SpecialOfferProduct: PK,FK2 SpecialOfferID; PK,FK1 ProductID; U1 rowguid, ModifiedDate
CreditCard: PK CreditCardID; U1 CardType, CardNumber, ExpMonth, ExpYear, ModifiedDate
PersonCreditCard: PK,FK2 BusinessEntityID; PK,FK1 CreditCardID; ModifiedDate
SalesTaxRate: PK SalesTaxRateID; FK1,U2 StateProvinceID, U2 TaxType, TaxRate, Name, U1 rowguid, ModifiedDate
ShoppingCartItem: PK ShoppingCartItemID; ShoppingCartID, Quantity, FK1 ProductID, DateCreated, ModifiedDate
Department: PK DepartmentID; U1 Name, GroupName, ModifiedDate
EmployeeDepartmentHistory: PK,FK2 BusinessEntityID; PK,FK1 DepartmentID; PK StartDate; PK,FK3 ShiftID; EndDate, ModifiedDate
Shift: PK ShiftID; U1 Name, U2 StartTime, U2 EndTime, ModifiedDate
Employee: PK,FK1 BusinessEntityID; U2 NationalIDNumber, U1 LoginID, ShiftID, JobTitle, BirthDate, MaritalStatus, Gender, HireDate, SalariedFlag, VacationHours, SickLeaveHours, CurrentFlag, U3 rowguid, ModifiedDate
JobCandidate: PK,FK1 BusinessEntityID; PK JobCandidateID; Resume, ModifiedDate
EmployeePayHistory: PK,FK1 BusinessEntityID; PK RateChangeDate; Rate, PayFrequency, ModifiedDate
Password: PK,FK1 BusinessEntityID; PasswordHash, PasswordSalt
Person: PK,FK1 BusinessEntityID; PersonType, NameStyle, Title, FirstName, MiddleName, LastName, Suffix, EmailPromotion, AdditionalContactInfo, Demographics, U1 rowguid, ModifiedDate
EmailAddress: PK,FK1 BusinessEntityID; PK EmailAddressID; EmailAddress
PersonPhone: PK,FK1 BusinessEntityID; PK PhoneNumber; PK,FK2 PhoneNumberTypeID
PhoneNumberType: PK PhoneNumberTypeID; Name
BusinessEntityContact: PK,FK3 BusinessEntityID; PK,FK1 PersonID; FK2 ContactTypeID, ModifiedDate
ContactType: PK ContactTypeID; U1 Name, ModifiedDate
AddressType: PK AddressTypeID; U1 Name, U2 rowguid, ModifiedDate
BusinessEntityAddress: PK,FK3 AddressID; PK,FK2 BusinessEntityID; FK1 AddressTypeID, ModifiedDate
BusinessEntity: PK BusinessEntityID; rowguid, ModifiedDate
Address: PK AddressID; U2 AddressLine1, U2 AddressLine2, U2 City, FK1,U2 StateProvinceID, U2 PostalCode, SpatialLocation, U1 rowguid, ModifiedDate
StateProvince: PK StateProvinceID; U3 StateProvinceCode, FK1,U3 CountryRegionCode, IsOnlyStateProvinceFlag, U1 Name, FK2 TerritoryID, U2 rowguid, ModifiedDate
CountryRegion: PK CountryRegionCode; U1 Name, ModifiedDate
Vendor: PK,FK1 BusinessEntityID; U1 AccountNumber, Name, CreditRating, PreferredVendorStatus, ActiveFlag, PurchasingWebServiceURL, ModifiedDate
ShipMethod: PK ShipMethodID; U1 Name, ShipBase, ShipRate, U2 rowguid, ModifiedDate
PurchaseOrderHeader: PK PurchaseOrderID; FK1 ShipMethodID, FK4 VendorID, RevisionNumber, Status, OrderDate, ShipDate, SubTotal, TaxAmt, Freight, TotalDue, ModifiedDate
PurchaseOrderDetail: PK,FK2 PurchaseOrderID; PK PurchaseOrderDetailID; DueDate, OrderQty, FK1 ProductID, UnitPrice, LineTotal, ReceivedQty, RejectedQty, StockedQty, ModifiedDate
ProductVendor: PK,FK3 BusinessEntityID; PK,FK1 ProductID; AverageLeadTime, StandardPrice, LastReceiptCost, LastReceiptDate, MinOrderQty, MaxOrderQty, OnOrderQty, FK2 UnitMeasureCode, ModifiedDate
ProductReview: PK ProductReviewID; FK1 ProductID, ReviewerName, ReviewDate, EmailAddress, Rating, Comments, ModifiedDate
ProductListPriceHistory: PK,FK1 ProductID; PK StartDate; EndDate, ListPrice, ModifiedDate
ProductPhoto: PK ProductPhotoID; ThumbNailPhoto, ThumbnailPhotoFileName, LargePhoto, LargePhotoFileName, ModifiedDate
ProductModelIllustration: PK,FK2 ProductModelID; PK,FK1 IllustrationID; ModifiedDate
Illustration: PK IllustrationID; Diagram, ModifiedDate
ProductProductPhoto: PK,FK1 ProductID; PK,FK2 ProductPhotoID; Primary, ModifiedDate
ProductModel: PK ProductModelID; U1 Name, CatalogDescription, Instructions, U2 rowguid, ModifiedDate
ProductModelProductDescriptionCulture: PK,FK3 ProductModelID; PK,FK2 ProductDescriptionID; PK,FK1 CultureID; ModifiedDate
Product: PK ProductID; U1 Name, U2 ProductNumber, MakeFlag, FinishedGoodsFlag, Color, SafetyStockLevel, ReorderPoint, StandardCost, ListPrice, Size, FK3 SizeUnitMeasureCode, FK4 WeightUnitMeasureCode, Weight, DaysToManufacture, ProductLine, Class, Style, FK2 ProductSubcategoryID, FK1 ProductModelID, SellStartDate, SellEndDate, DiscontinuedDate, U3 rowguid, ModifiedDate
BillOfMaterials: PK BillOfMaterialsID; FK1,U1 ProductAssemblyID, FK2,U1 ComponentID, U1 StartDate, EndDate, FK3 UnitMeasureCode, BOMLevel, PerAssemblyQty, ModifiedDate
ProductSubcategory: PK ProductSubcategoryID; FK1 ProductCategoryID, U1 Name, U2 rowguid, ModifiedDate
ProductCategory: PK ProductCategoryID; U1 Name, U2 rowguid, ModifiedDate
ProductDescription: PK ProductDescriptionID; Description, U1 rowguid, ModifiedDate
Culture: PK CultureID; U1 Name, ModifiedDate
TransactionHistory: PK TransactionID; FK1 ProductID, ReferenceOrderID, TransactionDate, TransactionType, Quantity, ActualCost, ModifiedDate, ReferenceOrderLineID
ProductCostHistory: PK,FK1 ProductID; PK StartDate; EndDate, StandardCost, ModifiedDate
ProductDocument: PK,FK2 ProductID; PK,FK1 DocumentNode; ModifiedDate
UnitMeasure: PK UnitMeasureCode; U1 Name, ModifiedDate
ProductInventory: PK,FK2 ProductID; PK,FK1 LocationID; Shelf, Bin, Quantity, rowguid, ModifiedDate
Location: PK LocationID; U1 Name, CostRate, Availability, ModifiedDate
WorkOrderRouting: PK,FK2 WorkOrderID; PK ProductID; PK OperationSequence; FK1 LocationID, ScheduledStartDate, ScheduledEndDate, ActualStartDate, ActualEndDate, ActualResourceHrs, PlannedCost, ActualCost, ModifiedDate
Document: PK DocumentNode; DocumentLevel, Title, FK1 Owner, FolderFlag, U1 FileName, FileExtension, U1 Revision, ChangeNumber, Status, DocumentSummary, Document, rowguid, ModifiedDate
TransactionHistoryArchive: PK TransactionID; ProductID, ReferenceOrderID, TransactionDate, TransactionType, Quantity, ActualCost, ModifiedDate, ReferenceOrderLineID
WorkOrder: PK WorkOrderID; FK1 ProductID, OrderQty, StockedQty, ScrappedQty, StartDate, EndDate, DueDate, FK2 ScrapReasonID, ModifiedDate
ScrapReason: PK ScrapReasonID; U1 Name, ModifiedDate

Schemas

- Sales
- Purchasing
- Person
- Production
- HumanResources
- dbo

Other Access Control Mechanisms

Views

- Limit what columns or what rows can be seen
- Views can be inserted and updated in many cases
- ORM support is still a challenge

Stored Procedures

- Specify exactly what a user can do
- Use to eliminate ad-hoc data access

Connection Strings

Securing Your Connection String

Use Windows Authentication

Eliminates the password in your connection string

Never store connection strings in code

These can be easily reverse engineered

Use Key Vault For SQL Logins

If you must use SQL logins, store the password somewhere secure

Transport Layer Security

Little Known Facts

Traditional SQL Server	The connection between your application and SQL Server is ***not encrypted*** by default
SQL Azure	The connection between your application and SQL Server ***is encrypted*** by default

What Does Transport Layer Security Give Us

Encrypts our data in motion

Verifies who we are talking to

Connection Strings to Encrypt Data

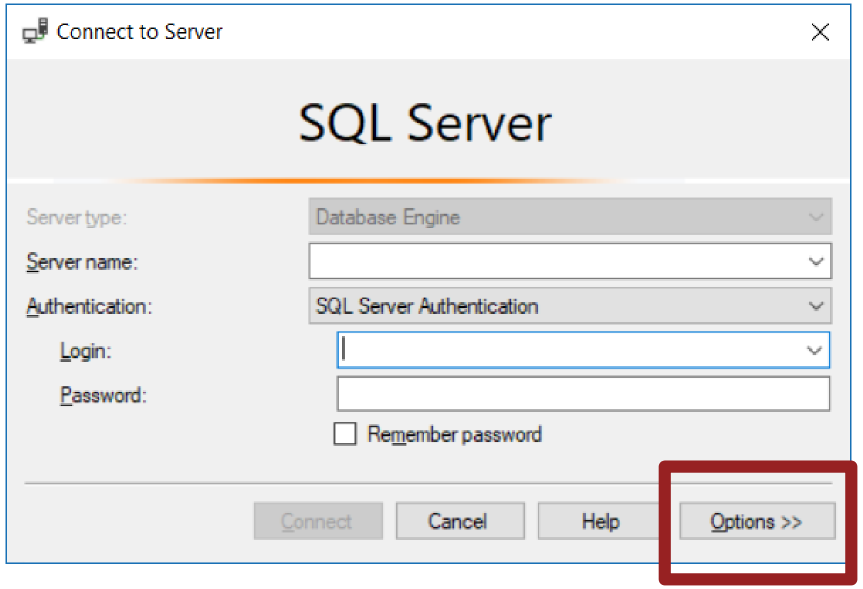
Encrypt Data and Allow a Self Signed Certificate

```
Server=<server>\<instance>;Database=<database>;Integrated
Security=true;Encrypt=true;TrustServerCertificate=True
```

Encrypt Data and Validate Server Certificate

```
Server=<server>\<instance>;Database=<database>;Integrated
Security=true;Encrypt=true;TrustServerCertificate=false
```

Decrypting Data from SSMS



Connect to Server

SQL Server

Server type: Database Engine

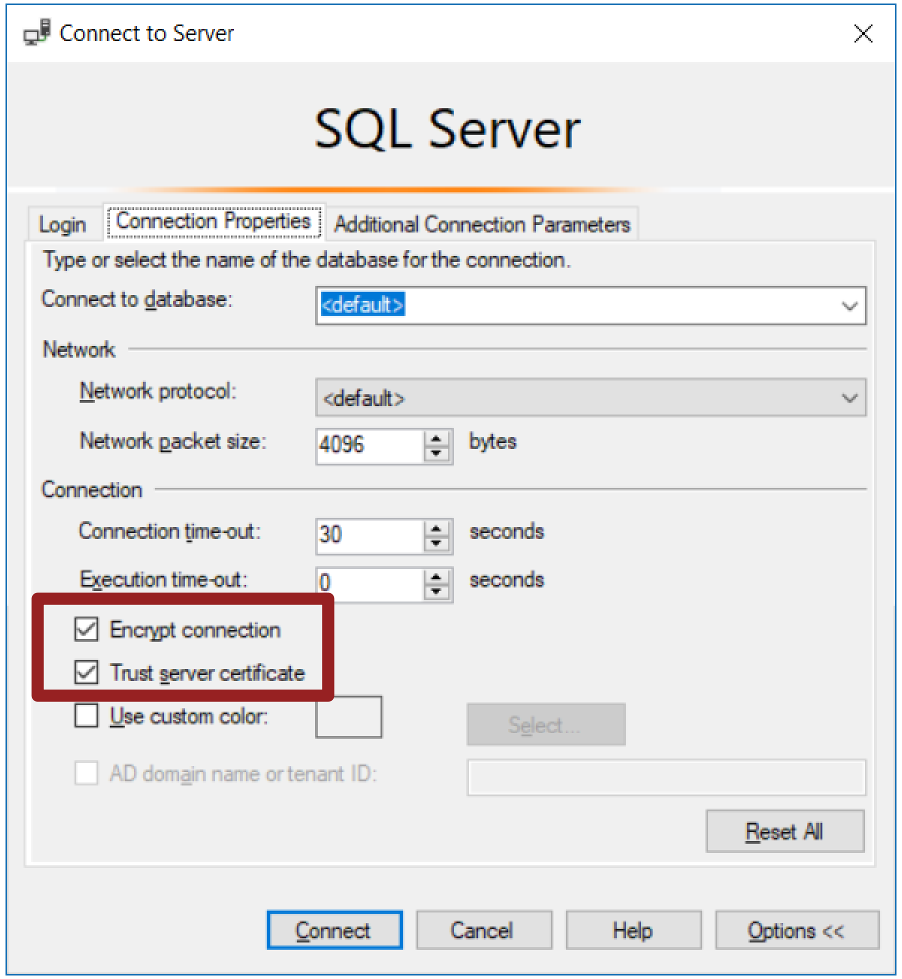
Server name:

Authentication: SQL Server Authentication

Login:

Password:

☐ Remember password



Connect to Server

SQL Server

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

Connect to database: <default>

Network

Network protocol: <default>

Network packet size: 4096 bytes

Connection

Connection time-out: 30 seconds

Execution time-out: 0 seconds

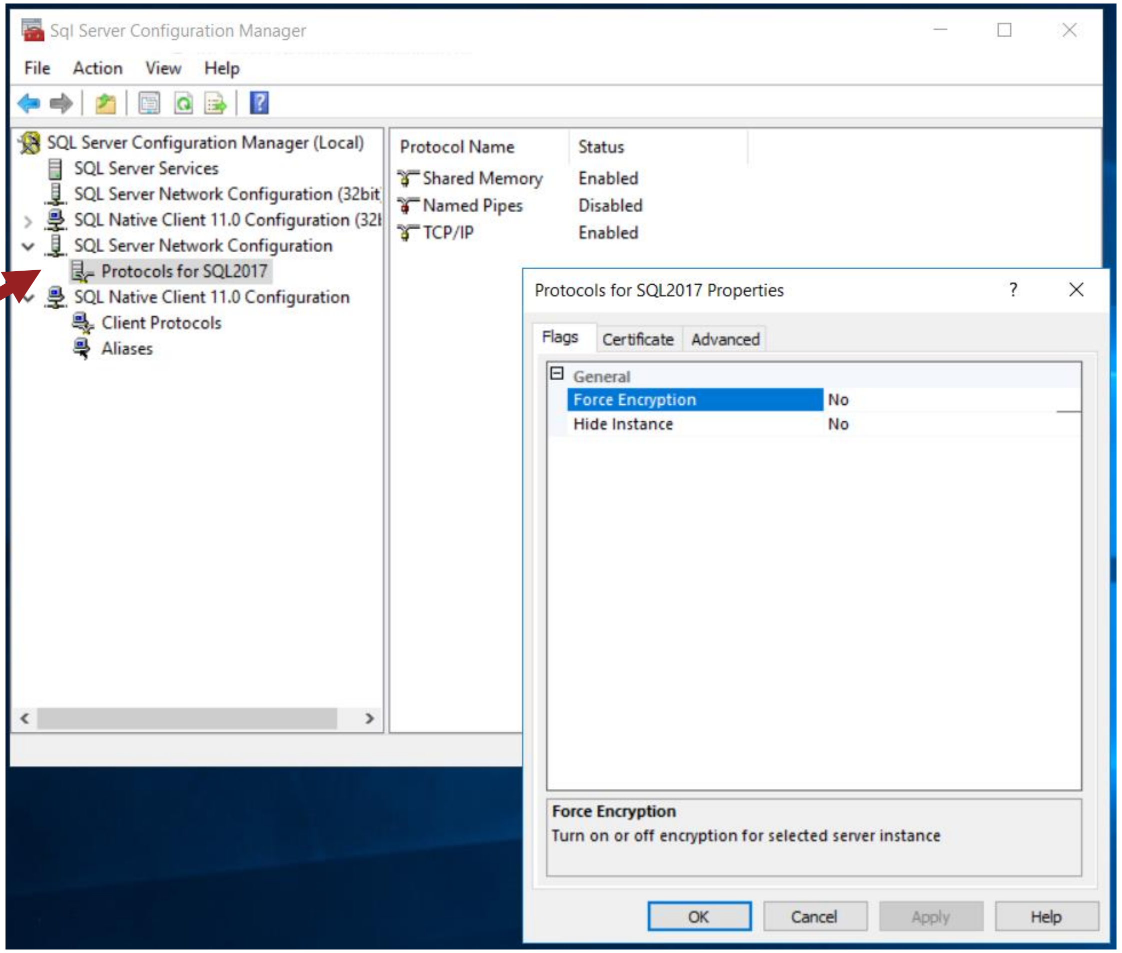
☑ Encrypt connection

☑ Trust server certificate

☐ Use custom color:

☐ AD domain name or tenant ID:

Configuring SQL Server to Force TLS


Sql Server Configuration Manager
File Action View Help
SQL Server Configuration Manager (Local)
SQL Server Services
SQL Server Network Configuration (32bit
SQL Native Client 11.0 Configuration (32
SQL Server Network Configuration
Protocols for SQL2017
SQL Native Client 11.0 Configuration
Client Protocols
Aliases
Protocols for SQL2017 Properties
Flags
Certificate
Advanced
General
OK
Cancel
Apply
Help

Protocol Name	Status
Shared Memory	Enabled
Named Pipes	Disabled
TCP/IP	Enabled

Force Encryption	No
Hide Instance	No

Force Encryption
Turn on or off encryption for selected server instance

Right Click then
Select "Properties"

Full instructions at: https://www.sqlshack.com/how-to-set-and-use-encrypted-sql-server-connections/

TLS Strategy

Set Encrypt=true for you connection strings

Work towards installing a verifiable certificate

Discuss requiring encrypted connections with your DBA team

Database Encryption

Transparent Data Encryption

Encrypts the data files of SQL Server (data at rest)

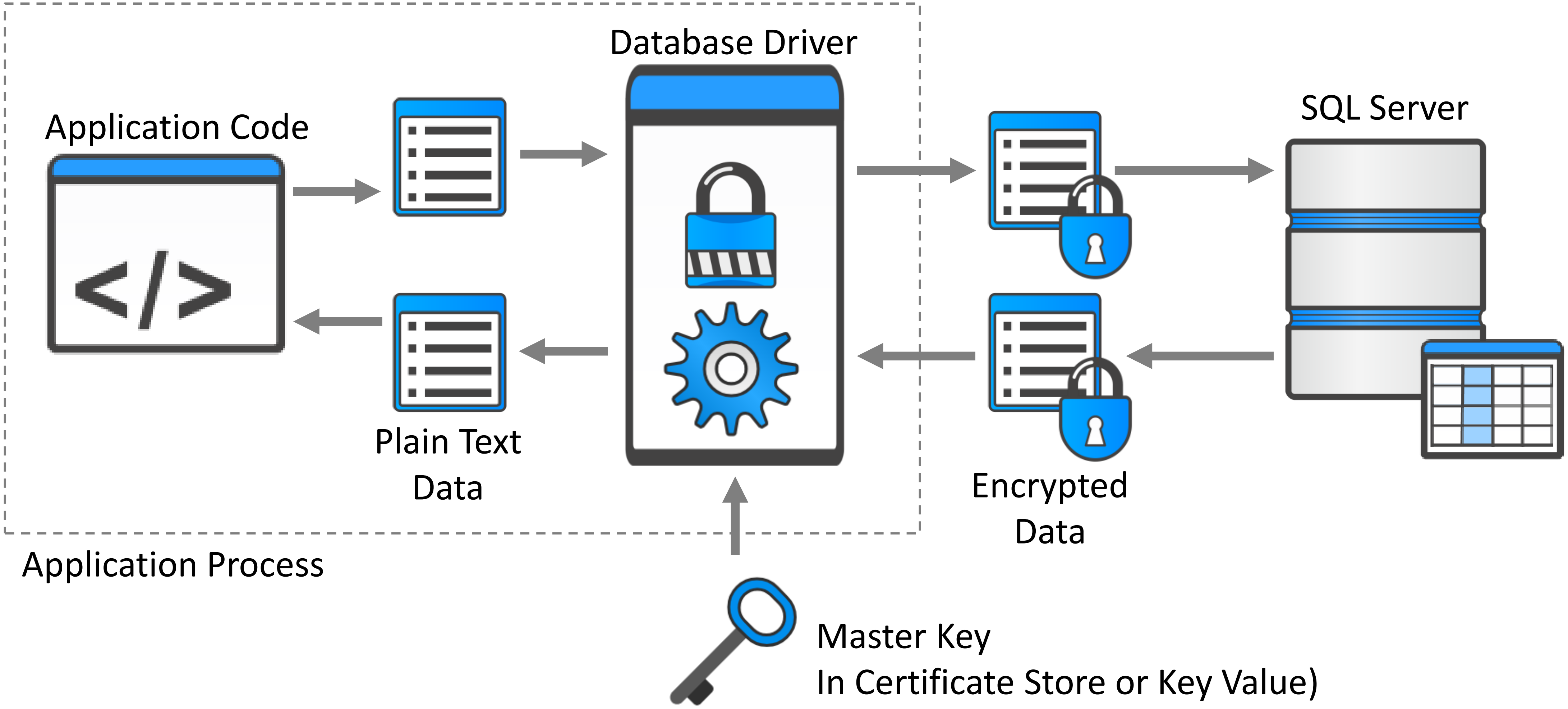
Protects you in case an attacker gets your data files or backups

Always Encrypted

Designed to protect data at a column level

Available in SQL Server 2016 and later

How Always Encrypted Works



What You Can/Cannot Encrypt

Can Be Encrypted

- VARCHAR/NVARCHAR
- INT/SMALLINT/BIGINT
- FLOAT/DOUBLE
- DECIMAL
- MONEY/SMALLMONEY
- DATETIME/DATE/TIME

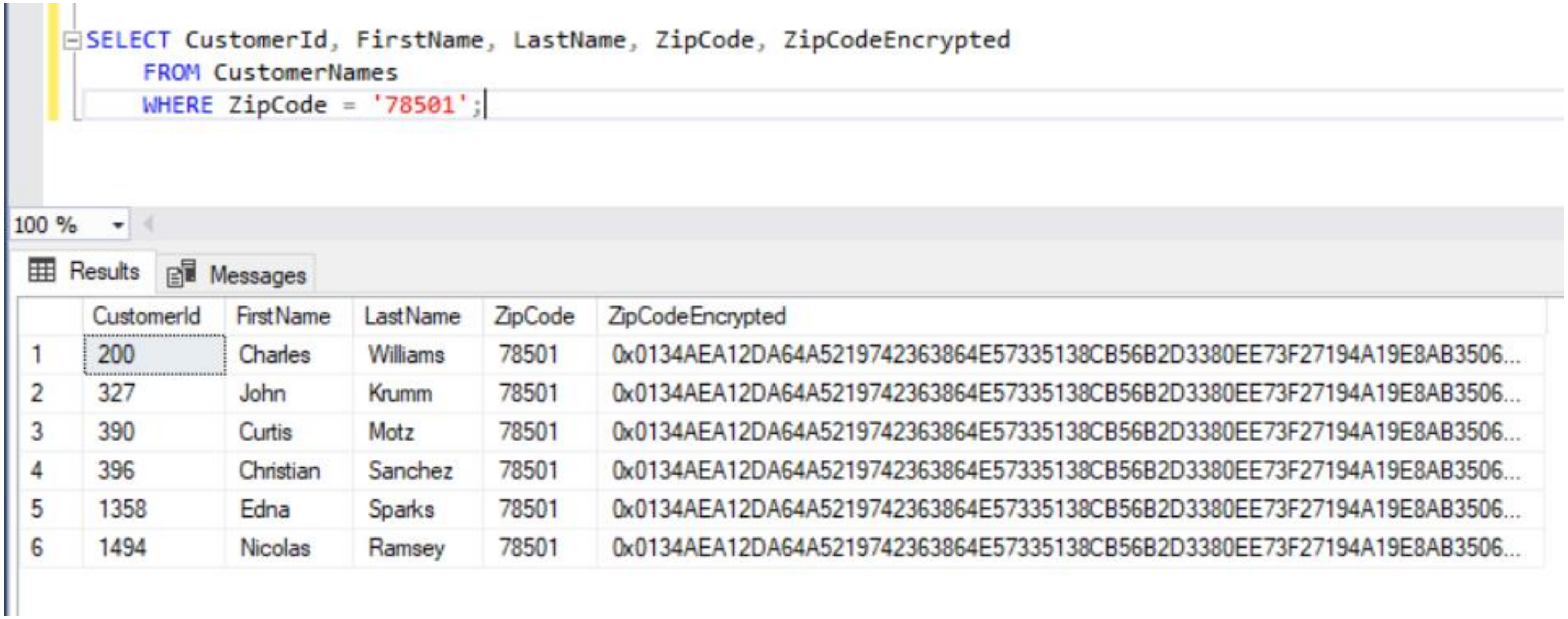
Cannot Be Encrypted

- XML
- SQL_VARIANT
- IMAGE
- GEOGRAPHY
- GEOMETRY

Encryption Types

Deterministic	Randomized
• Always generates the same encrypted value for a plain text value • Column can be used in equality comparisons, joins, group by operations and indexes • Not suitable for cardinality columns because the attacker can guess values based on probabilities	• Different encrypted values will be generated for the same plain text input • Column cannot be used in equality comparisons, joins, group by operations or indexes

Deterministic Encryption Example



```
SELECT CustomerId, FirstName, LastName, ZipCode, ZipCodeEncrypted
    FROM CustomerNames
    WHERE ZipCode = '78501';
```

	CustomerId	FirstName	LastName	ZipCode	ZipCodeEncrypted
1	200	Charles	Williams	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...
2	327	John	Krumm	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...
3	390	Curtis	Motz	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...
4	396	Christian	Sanchez	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...
5	1358	Edna	Sparks	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...
6	1494	Nicolas	Ramsey	78501	0x0134AEA12DA64A5219742363864E57335138CB56B2D3380EE73F27194A19E8AB3506...

Connection Strings For Always Encrypted

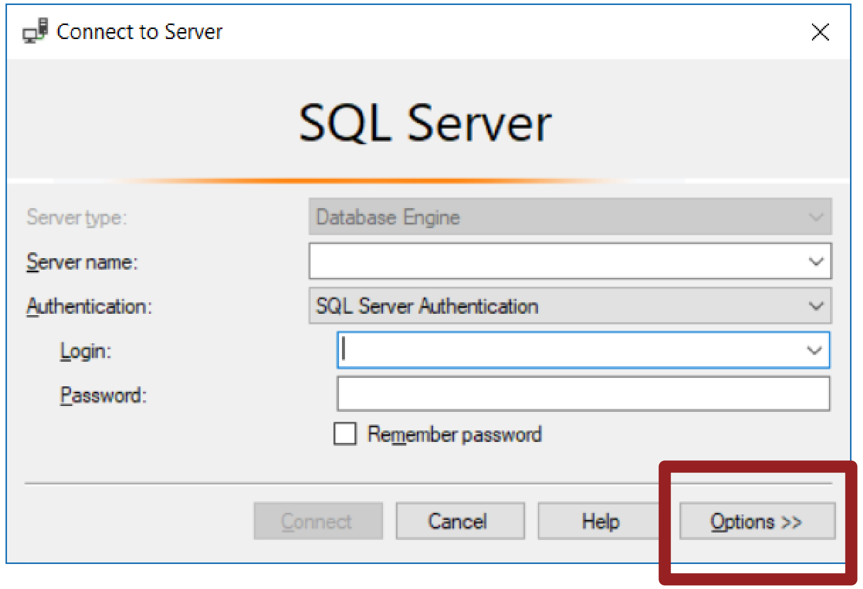
Encrypt Data and Allow a Self Signed Certificate

```
Server=<server>\<instance>;Database=<database>;Integrated
Security=true;Encrypt=true;TrustServerCertificate=True;
Column Encryption Setting=enabled
```

Using Azure Key Vault Provider

```
// https://docs.microsoft.com/en-us/azure/sql-database/sql-database-always-encrypted-azure-key-vault
// Required NuGet Packages
//     Microsoft.SqlServer.Management.AlwaysEncrypted.AzureKeyVaultProvider
//     Microsoft.IdentityModel.Clients.ActiveDirectory

private static ClientCredential _clientCredential;

static void InitializeAzureKeyVaultProvider()
{
    _clientCredential = new ClientCredential(applicationId, clientKey);

    SqlColumnEncryptionAzureKeyVaultProvider azureKeyVaultProvider =
      new SqlColumnEncryptionAzureKeyVaultProvider(GetToken);

    Dictionary<string, SqlColumnEncryptionKeyStoreProvider> providers =
          new Dictionary<string, SqlColumnEncryptionKeyStoreProvider>();

    providers.Add(SqlColumnEncryptionAzureKeyVaultProvider.ProviderName, azureKeyVaultProvider);
    SqlConnection.RegisterColumnEncryptionKeyStoreProviders(providers);
}

public async static Task<string> GetToken(string authority, string resource, string scope)
{
    var authContext = new AuthenticationContext(authority);
    AuthenticationResult result = await authContext.AcquireTokenAsync(resource, _clientCredential);

    if (result == null)
        throw new InvalidOperationException("Failed to obtain the access token");
    return result.AccessToken;
}
```

Encrypting Connection from SSMS



Connect to Server

SQL Server

Server type: Database Engine

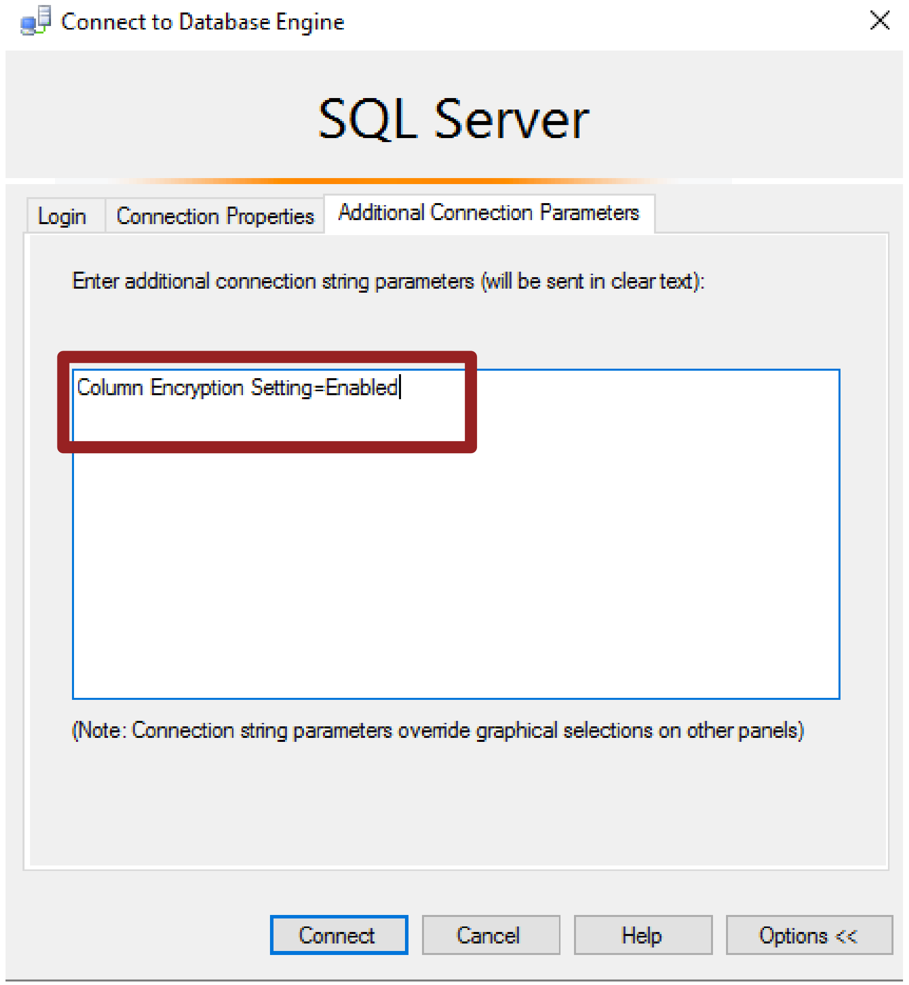
Server name:

Authentication: SQL Server Authentication

Login:

Password:

☐ Remember password



Connect to Database Engine

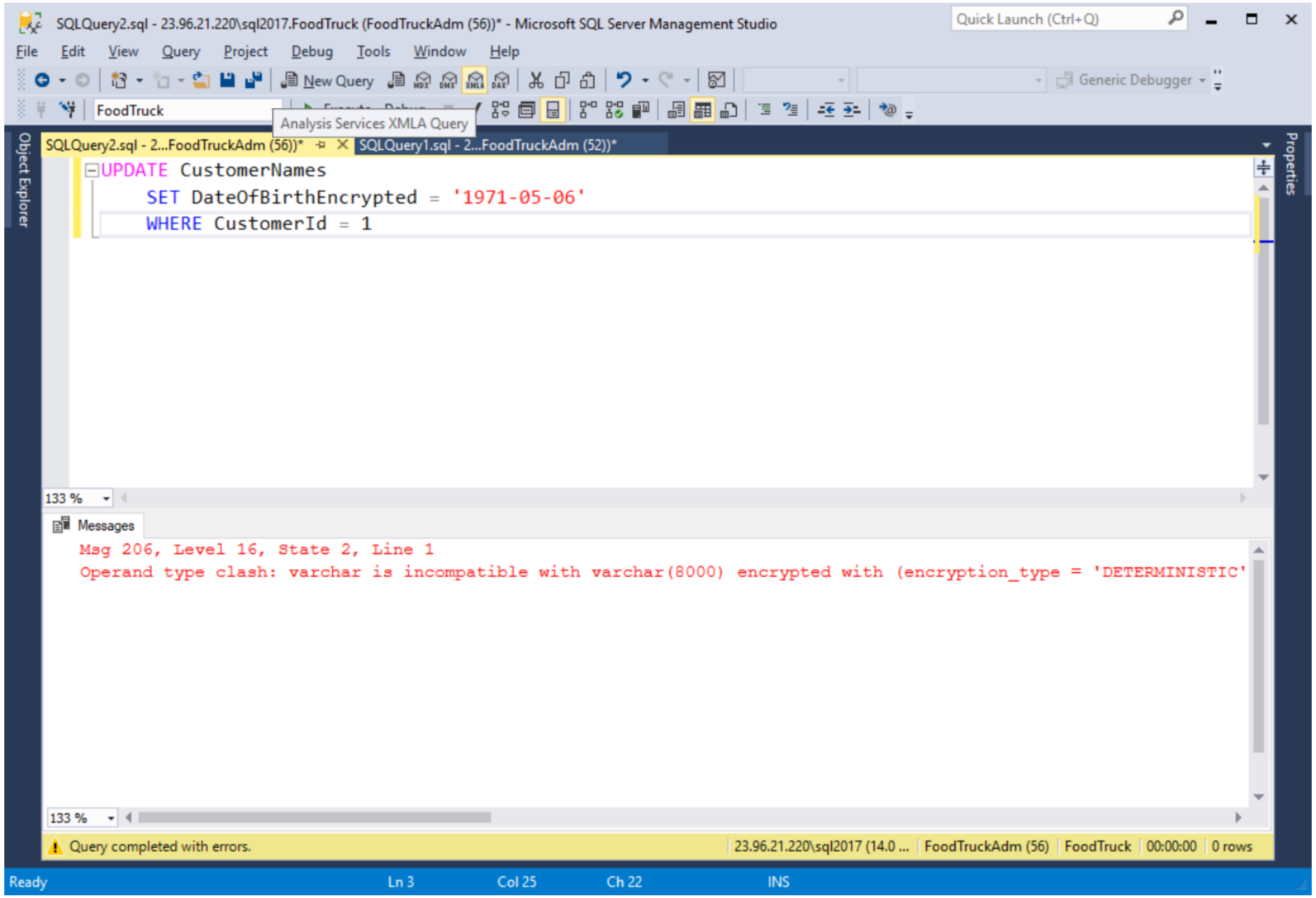
SQL Server

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

```
Column Encryption Setting=Enabled
```

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

SSMS Functionality Changes


SQLQuery2.sql - 23.96.21.220\sql2017.FoodTruck (FoodTruckAdm (56))* - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File Edit View Query Project Debug Tools Window Help
New Query
Generic Debugger
FoodTruck
Analysis Services XMLA Query
SQLQuery2.sql - 2...FoodTruckAdm (56))*
SQLQuery1.sql - 2...FoodTruckAdm (52))*
Object Explorer
Properties
133 %
Messages
133 %

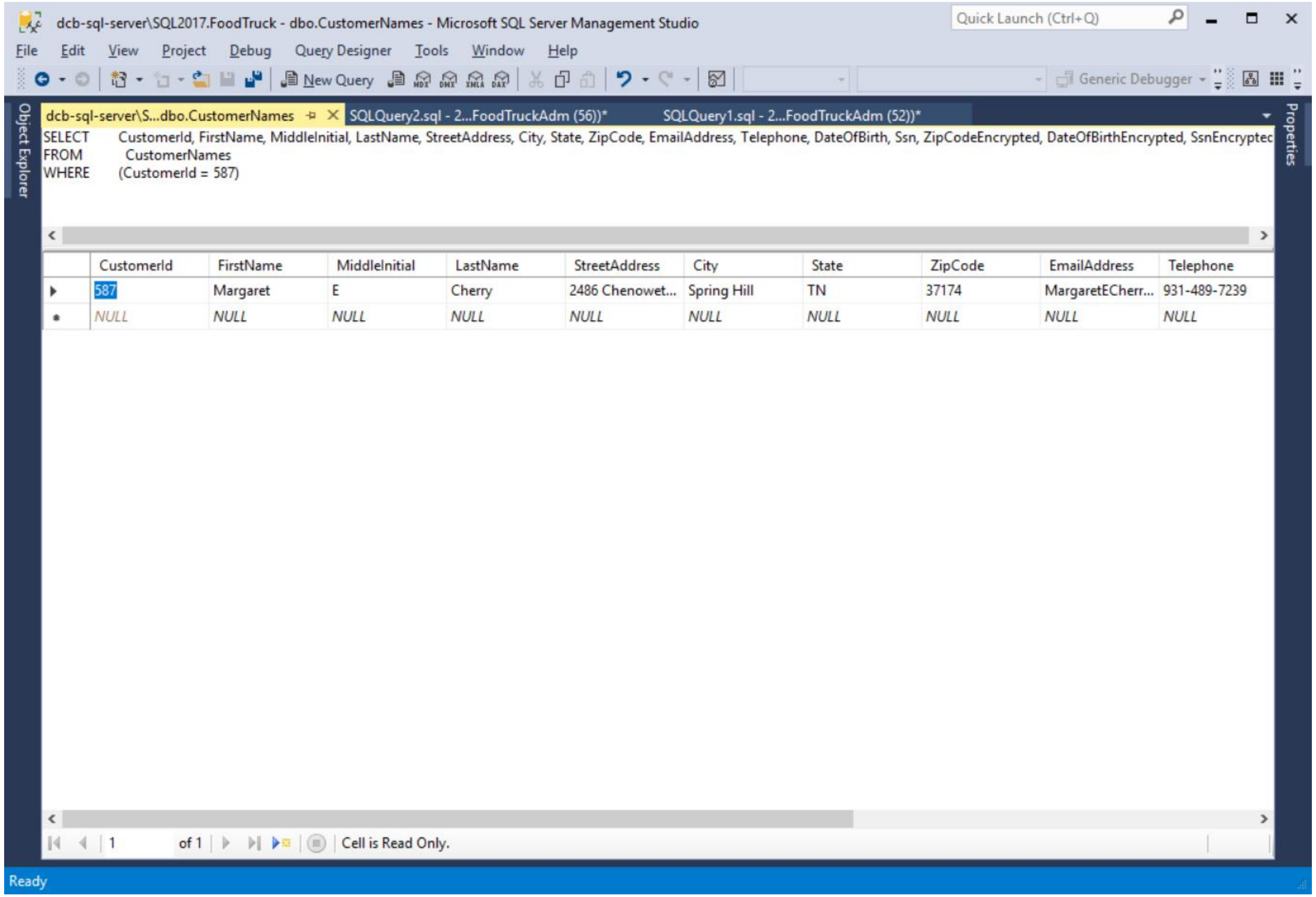
```sql
UPDATE CustomerNames
    SET DateOfBirthEncrypted = '1971-05-06'
    WHERE CustomerId = 1
```

```
Msg 206, Level 16, State 2, Line 1
Operand type clash: varchar is incompatible with varchar(8000) encrypted with (encryption_type = 'DETERMINISTIC'
```

Query completed with errors. | 23.96.21.220\sql2017 (14.0 ... | FoodTruckAdm (56) | FoodTruck | 00:00:00 | 0 rows

Ready Ln 3 Col 25 Ch 22 INS

Editing Encrypted Data in SSMS


dcb-sql-server\SQL2017.FoodTruck - dbo.CustomerNames - Microsoft SQL Server Management Studio
Quick Launch (Ctrl+Q)
File
Edit
View
Project
Debug
Query Designer
Tools
Window
Help
New Query
Generic Debugger
dcb-sql-server\S...dbo.CustomerNames
SQLQuery2.sql - 2...FoodTruckAdm (56))*
SQLQuery1.sql - 2...FoodTruckAdm (52))*
Object Explorer
Properties

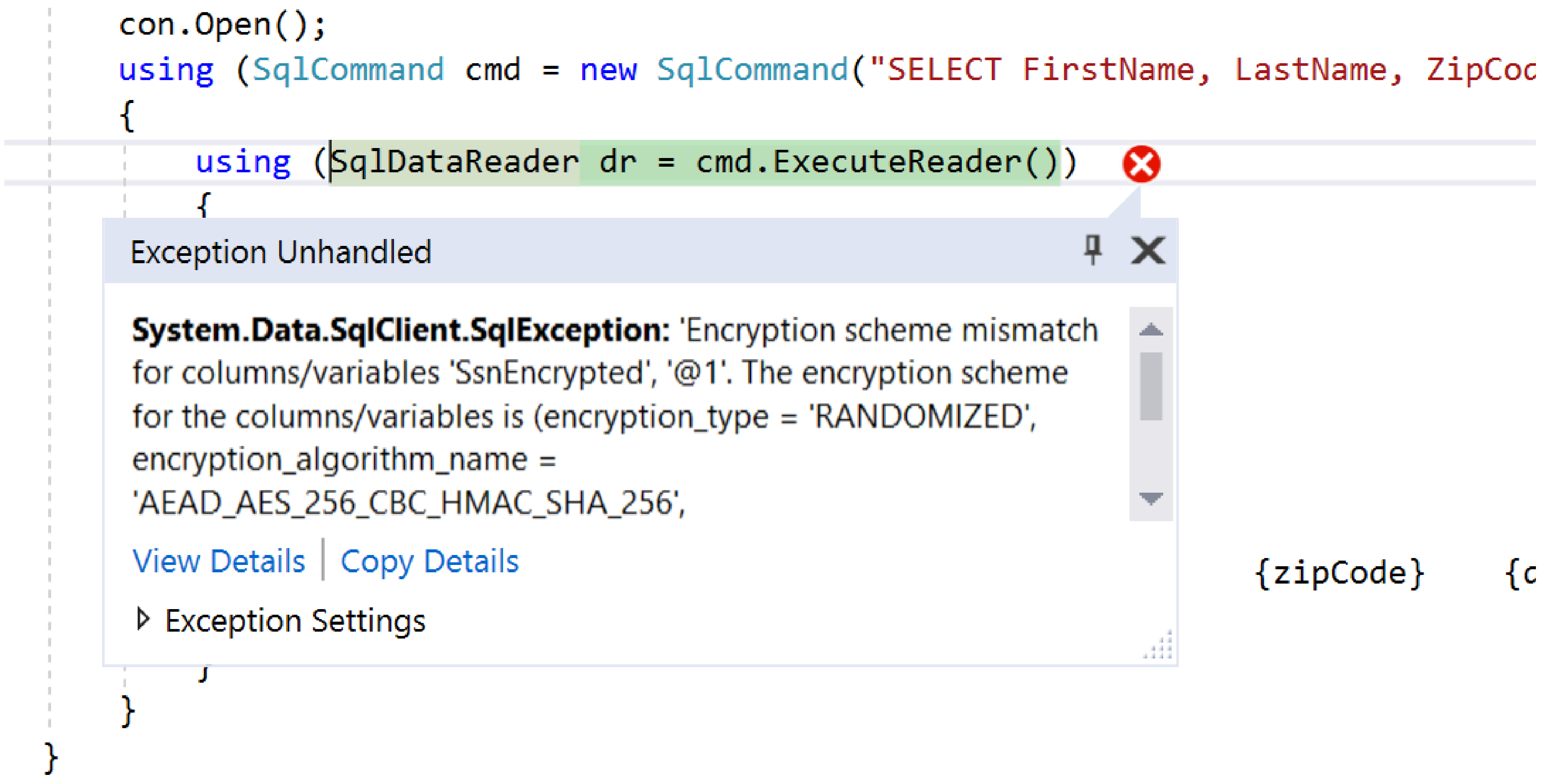
```
SELECT    CustomerId, FirstName, MiddleInitial, LastName, StreetAddress, City, State, ZipCode, EmailAddress, Telephone, DateOfBirth, Ssn, ZipCodeEncrypted, DateOfBirthEncrypted, SsnEncryptec
FROM       CustomerNames
WHERE     (CustomerId = 587)
```

	CustomerId	FirstName	MiddleInitial	LastName	StreetAddress	City	State	ZipCode	EmailAddress	Telephone
▶	587	Margaret	E	Cherry	2486 Chenowet...	Spring Hill	TN	37174	MargaretECherr...	931-489-7239
●	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

1 of 1 Cell is Read Only.

Ready

What Happens When Searching Randomized Data?



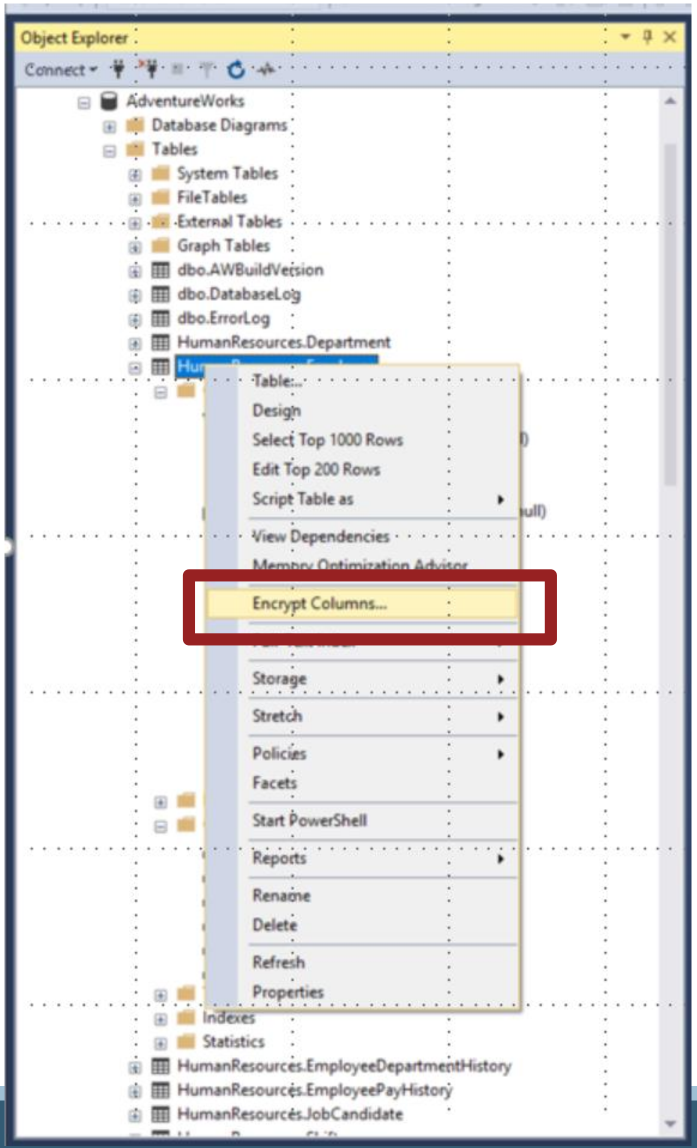
```
    con.Open();
    using (SqlCommand cmd = new SqlCommand("SELECT FirstName, LastName, ZipCoc
    {
        using (SqlDataReader dr = cmd.ExecuteReader())
        {
```

Exception Unhandled

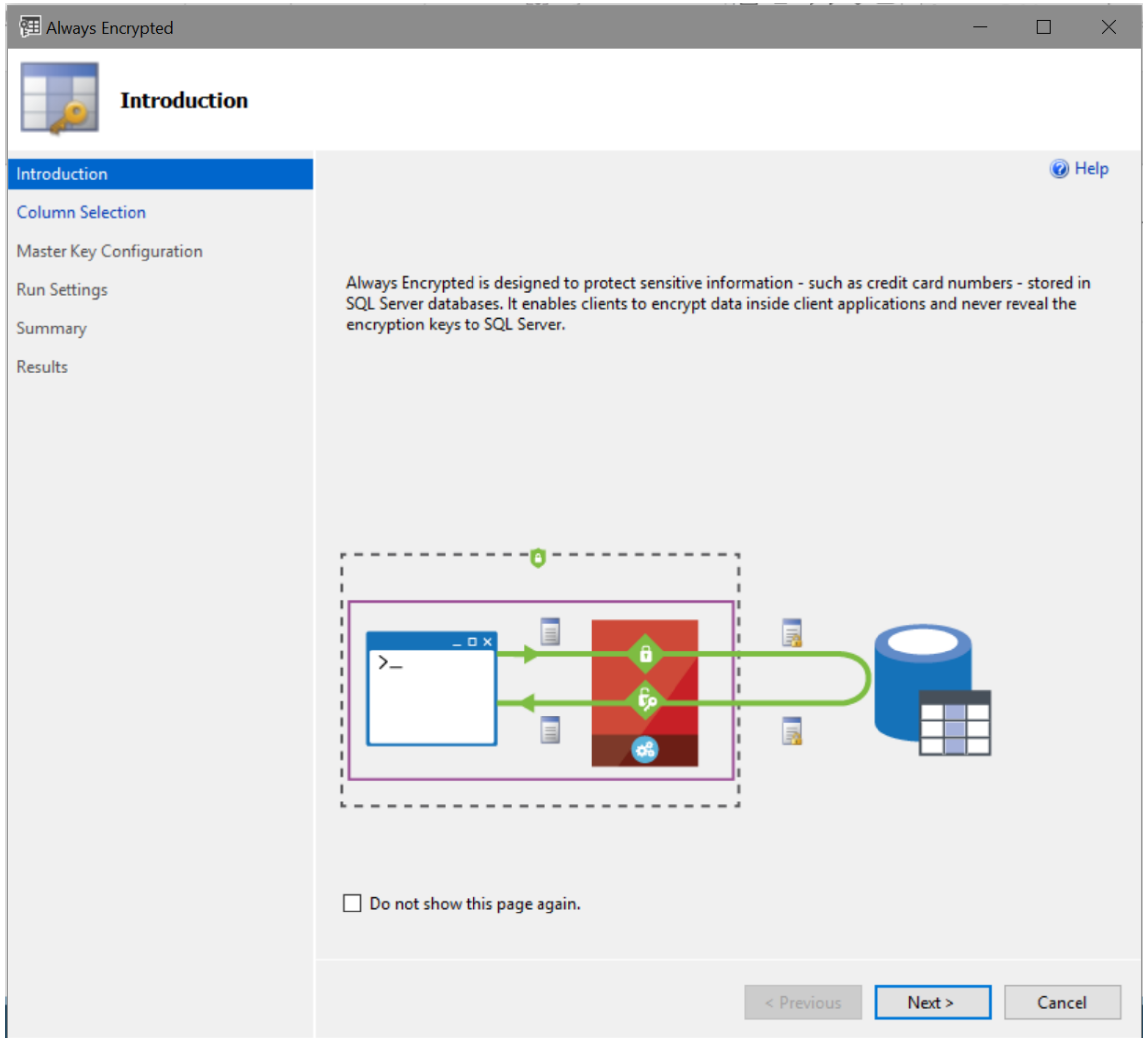
System.Data.SqlClient.SqlException: 'Encryption scheme mismatch for columns/variables 'SsnEncrypted', '@1'. The encryption scheme for the columns/variables is (encryption_type = 'RANDOMIZED', encryption_algorithm_name = 'AEAD_AES_256_CBC_HMAC_SHA_256',

```
{zipCode}    {c
        }
    }
}
```

Always Encrypted Wizard



Always Encrypted Wizard



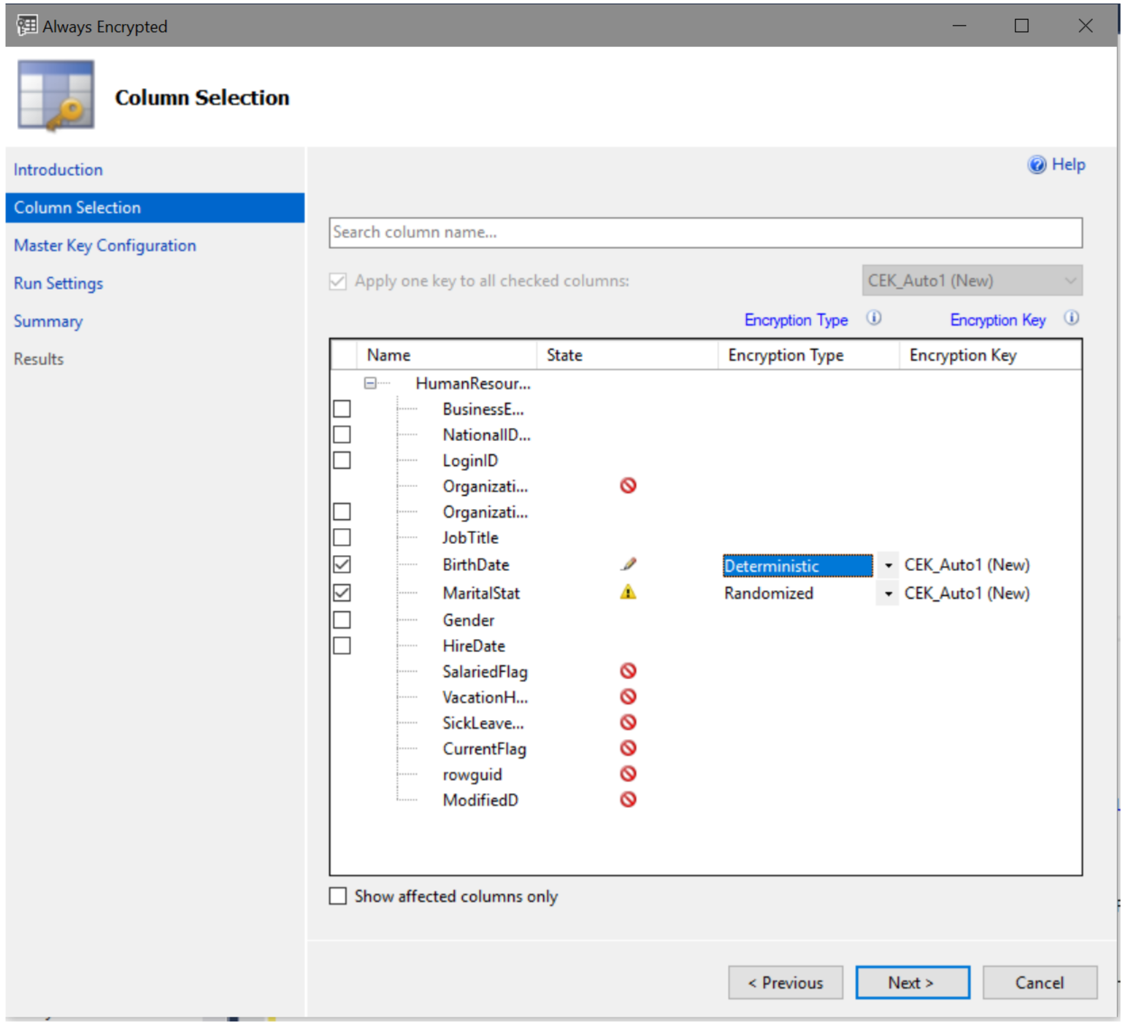
Always Encrypted

Introduction

Always Encrypted is designed to protect sensitive information - such as credit card numbers - stored in SQL Server databases. It enables clients to encrypt data inside client applications and never reveal the encryption keys to SQL Server.

☐ Do not show this page again.

Selecting Columns to be Encrypted



Always Encrypted

Column Selection

- Introduction
- Column Selection
- Master Key Configuration
- Run Settings
- Summary
- Results

Help

Search column name...

☑ Apply one key to all checked columns: CEK_Auto1 (New)

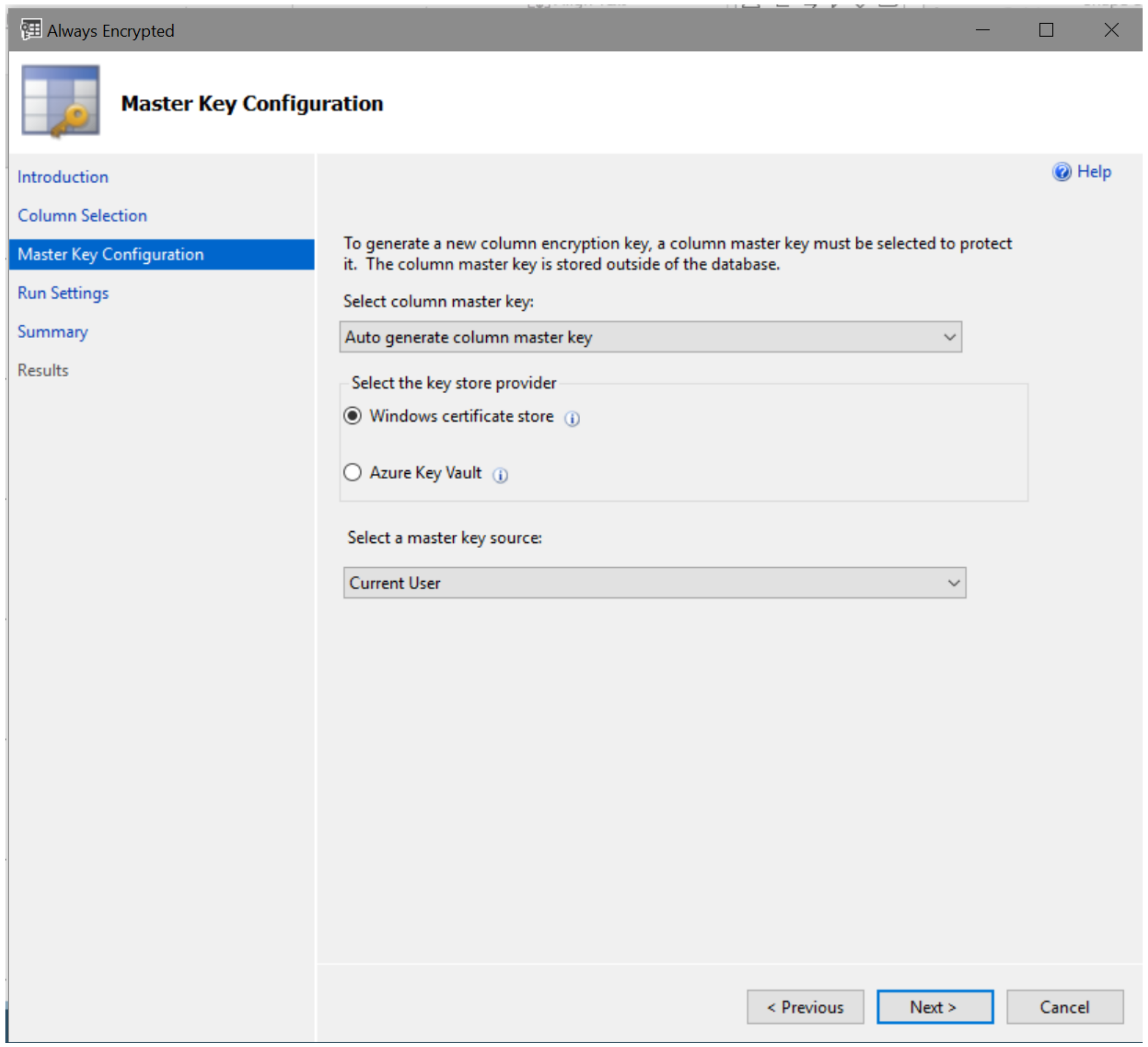
Encryption Type ⓘ Encryption Key ⓘ

	Name	State	Encryption Type	Encryption Key
	HumanResour...			
☐	BusinessE...			
☐	NationalID...			
☐	LoginID			
	Organizati...	🚫		
☐	Organizati...			
☐	JobTitle			
☑	BirthDate	✎	Deterministic	CEK_Auto1 (New)
☑	MaritalStat	⚠	Randomized	CEK_Auto1 (New)
☐	Gender			
☐	HireDate			
	SalariedFlag	🚫		
	VacationH...	🚫		
	SickLeave...	🚫		
	CurrentFlag	🚫		
	rowguid	🚫		
	ModifiedD	🚫		

☐ Show affected columns only

< Previous | Next > | Cancel

Creating a Master Encryption Key



Master Key Configuration

To generate a new column encryption key, a column master key must be selected to protect it. The column master key is stored outside of the database.

Select column master key: Auto generate column master key

Select the key store provider

(●) Windows certificate store

( ) Azure Key Vault

Select a master key source: Current User

Always Encrypted Columns DDL

```
CREATE TABLE [HumanResources].[Employee]
(
    [BusinessEntityID] [int] NOT NULL,
    [NationalIDNumber] [nvarchar](15) NOT NULL,
    [LoginID] [nvarchar](256) NOT NULL,
    [OrganizationNode] [hierarchyid] NULL,
    [OrganizationLevel]  AS ([OrganizationNode].[GetLevel]()),
    [JobTitle] [nvarchar](50) NOT NULL,
    [BirthDate] [date] ENCRYPTED WITH (COLUMN_ENCRYPTION_KEY = [CEK_Auto1],
        ENCRYPTION_TYPE = Deterministic, ALGORITHM = 'AEAD_AES_256_CBC_HMAC_SHA_256') NOT NULL,
     [MaritalStatus] [nchar](1) NOT NULL,
    [Gender] [nchar](1) COLLATE Latin1_General_BIN2 ENCRYPTED WITH
        (COLUMN_ENCRYPTION_KEY = [CEK_Auto1],
        ENCRYPTION_TYPE = Randomized, ALGORITHM = 'AEAD_AES_256_CBC_HMAC_SHA_256') NOT NULL,
    [HireDate] [date] NOT NULL,
    [SalariedFlag] [dbo].[Flag] NOT NULL,
    CONSTRAINT [PK_Employee_BusinessEntityID]
        PRIMARY KEY (BusinessEntityID)
)
GO
```

Always Encrypted Search Implications

- To be searched, columns must use deterministic encryption
- You can only search on the full value (no range searches or LIKE clauses)
- Consider storing a subset of the field to be searched (last 4 of social)

Always Encrypted Documentation

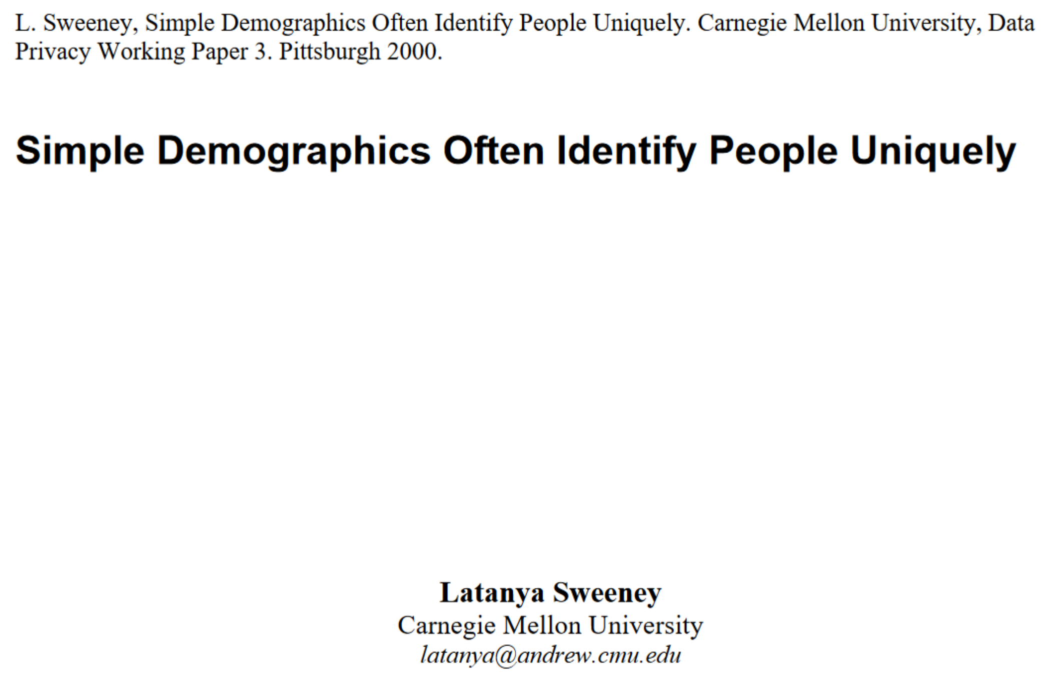
Microsoft Documentation

https://docs.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-database-engine

Simple Talk Article

https://www.red-gate.com/simple-talk/sql/database-administration/sql-server-encryption-always-encrypted/

Simple Demographics Often Identify People Uniquely



L. Sweeney, Simple Demographics Often Identify People Uniquely. Carnegie Mellon University, Data Privacy Working Paper 3. Pittsburgh 2000.

Simple Demographics Often Identify People Uniquely

Latanya Sweeney
Carnegie Mellon University
latanya@andrew.cmu.edu

https://dataprivacylab.org/projects/identifiability/paper1.pdf

SQL Injection

Injection attacks are still ranked #1 on the OWASP top ten list of security vulnerabilities

But we are actually seeing SQL Injection decline

Vulnerable Code

```
String sql = @"SELECT CustomerId, FirstName, LastName, Ssn
                FROM Customers WHERE CustomerId = '" + customerId + "'";

using (SqlCommand cmd = new SqlCommand(sql, dbConnection))
{
    using (SqlDataReader reader = cmd.ExecuteReader())
    {
        if (reader.Read())
        {
            customer = new Customer()
            {
                CustomerId = reader.GetString(0),
                FirstName = reader.GetString(1),
                LastName = reader.GetString(2),
                Ssn = reader.GetString(3)
            };
        }
    }
}
```

Safe Code

```
String sql = @"SELECT CustomerId, FirstName, LastName, Ssn
                FROM Customers WHERE CustomerId = @customerId";

using (SqlCommand cmd = new SqlCommand(sql, dbConnection))
{
    cmd.Parameters.Add(“@customerId”, customerId);
    using (SqlDataReader reader = cmd.ExecuteReader())
    {
        if (reader.Read())
        {
            customer = new Customer()
            {
                CustomerId = reader.GetString(0),
                FirstName = reader.GetString(1),
                LastName = reader.GetString(2),
                Ssn = reader.GetString(3)
            };
        }
    }
}
```

Defeating SQL Injection

Parameterize Your SQL

Don’t try to escape your own SQL strings. There are too many scenarios for us to cover

Use an ORM

Modern ORMs automatically parameterize SQL statements for you

Scan Your Code

Code analysis in Visual Studio or tools like Sonarqube and Puma Scan can detect vulnerable SQL

Final Thoughts

Thank You

David Berry
@DavidCBerry13
https://github.com/DavidCBerry13/sql-server-security/