using System;

using System.Data.SqlClient;

using System.Text;

using System.Collections.Generic;

using System.Threading.Tasks;

public class SystemPromptGenerator

{

private readonly string \_connectionString;

public SystemPromptGenerator(string connectionString)

{

\_connectionString = connectionString;

}

public async Task<string> GenerateSystemPromptForTableAsync(string baseTableName)

{

var tables = await GetRelatedTablesAsync(baseTableName);

var promptBuilder = new StringBuilder();

promptBuilder.AppendLine("You are an assistant that helps users query a database with a specific table and its related tables. " +

"Your role is to assist with queries involving data retrieval and filtering across these tables. Here is the database structure and relationships:");

foreach (var table in tables)

{

promptBuilder.AppendLine();

promptBuilder.AppendLine($"- \*\*{table.Name} Table\*\*");

promptBuilder.AppendLine($" Purpose: Stores information related to {table.Name.ToLower()}.");

promptBuilder.AppendLine(" Columns:");

foreach (var column in table.Columns)

{

promptBuilder.AppendLine($" - {column.Name} ({column.Type}){(column.IsForeignKey ? $" - Foreign Key to {column.ForeignKeyTable}" : "")}");

}

}

promptBuilder.AppendLine();

promptBuilder.AppendLine("If a user query mentions any entity or topic outside of this schema, respond with: " +

"\"I can only assist with queries related to the specified tables.\"");

return promptBuilder.ToString();

}

private async Task<List<Table>> GetRelatedTablesAsync(string baseTableName)

{

var tables = new Dictionary<string, Table>();

using (var connection = new SqlConnection(\_connectionString))

{

await connection.OpenAsync();

// Query to retrieve base table and all linked tables via foreign keys

var query = @"

WITH TableRelations AS (

SELECT

t.TABLE\_NAME AS TableName,

c.COLUMN\_NAME AS ColumnName,

c.DATA\_TYPE AS DataType,

fk.REFERENCED\_TABLE\_NAME AS ReferencedTable

FROM

INFORMATION\_SCHEMA.TABLES t

JOIN

INFORMATION\_SCHEMA.COLUMNS c ON t.TABLE\_NAME = c.TABLE\_NAME

LEFT JOIN

(SELECT

fkc.TABLE\_NAME, fkc.COLUMN\_NAME, rc.UNIQUE\_CONSTRAINT\_TABLE\_NAME AS REFERENCED\_TABLE\_NAME

FROM

INFORMATION\_SCHEMA.REFERENTIAL\_CONSTRAINTS rc

JOIN

INFORMATION\_SCHEMA.KEY\_COLUMN\_USAGE fkc ON rc.CONSTRAINT\_NAME = fkc.CONSTRAINT\_NAME) fk

ON c.TABLE\_NAME = fk.TABLE\_NAME AND c.COLUMN\_NAME = fk.COLUMN\_NAME

WHERE

t.TABLE\_TYPE = 'BASE TABLE'

AND (t.TABLE\_NAME = @BaseTableName OR fk.REFERENCED\_TABLE\_NAME = @BaseTableName)

)

SELECT \* FROM TableRelations

ORDER BY TableName";

using (var command = new SqlCommand(query, connection))

{

command.Parameters.AddWithValue("@BaseTableName", baseTableName);

using (var reader = await command.ExecuteReaderAsync())

{

while (await reader.ReadAsync())

{

var tableName = reader.GetString(0);

var columnName = reader.GetString(1);

var dataType = reader.GetString(2);

var referencedTable = reader.IsDBNull(3) ? null : reader.GetString(3);

if (!tables.ContainsKey(tableName))

{

tables[tableName] = new Table { Name = tableName };

}

tables[tableName].Columns.Add(new Column

{

Name = columnName,

Type = dataType,

IsForeignKey = referencedTable != null,

ForeignKeyTable = referencedTable

});

}

}

}

}

return new List<Table>(tables.Values);

}

}

public class Table

{

public string Name { get; set; }

public List<Column> Columns { get; set; } = new List<Column>();

}

public class Column

{

public string Name { get; set; }

public string Type { get; set; }

public bool IsForeignKey { get; set; }

public string ForeignKeyTable { get; set; }

}

// Usage Example

public static async Task Main()

{

var connectionString = "your-database-connection-string";

var baseTableName = "Trades"; // Replace with the main table of interest

var generator = new SystemPromptGenerator(connectionString);

string systemPrompt = await generator.GenerateSystemPromptForTableAsync(baseTableName);

Console.WriteLine(systemPrompt);

}