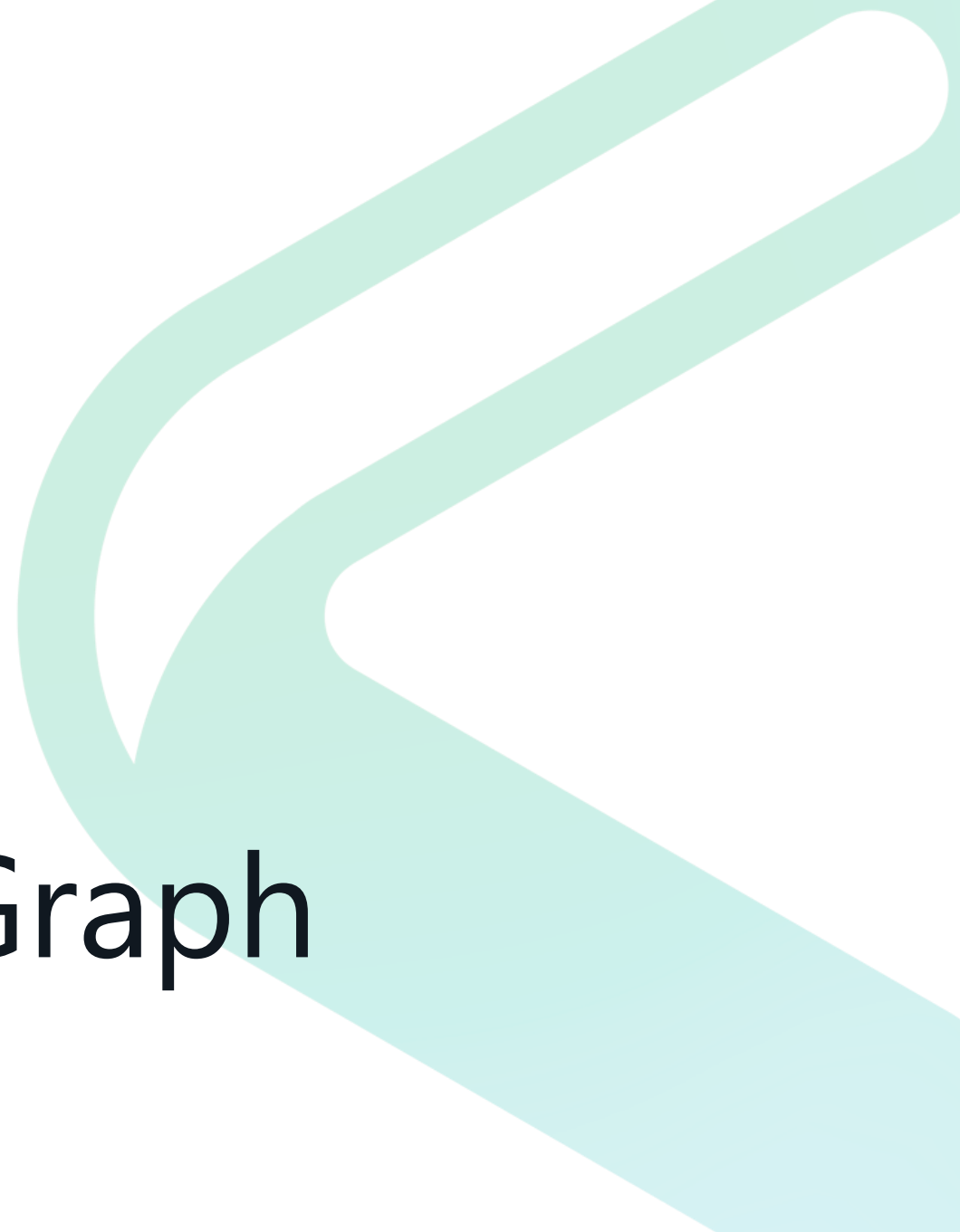


Sergio Govoni

SQL Server 2017 Graph Database



Agenda of this session

- What a Graph Database is
- When and where the graph theory was born
- SQL Server 2017 Graph Database
 - Nodes and Edges
 - The new MATCH function
- How to build a recommendation system for sales
- New T-SQL functions for graph objects

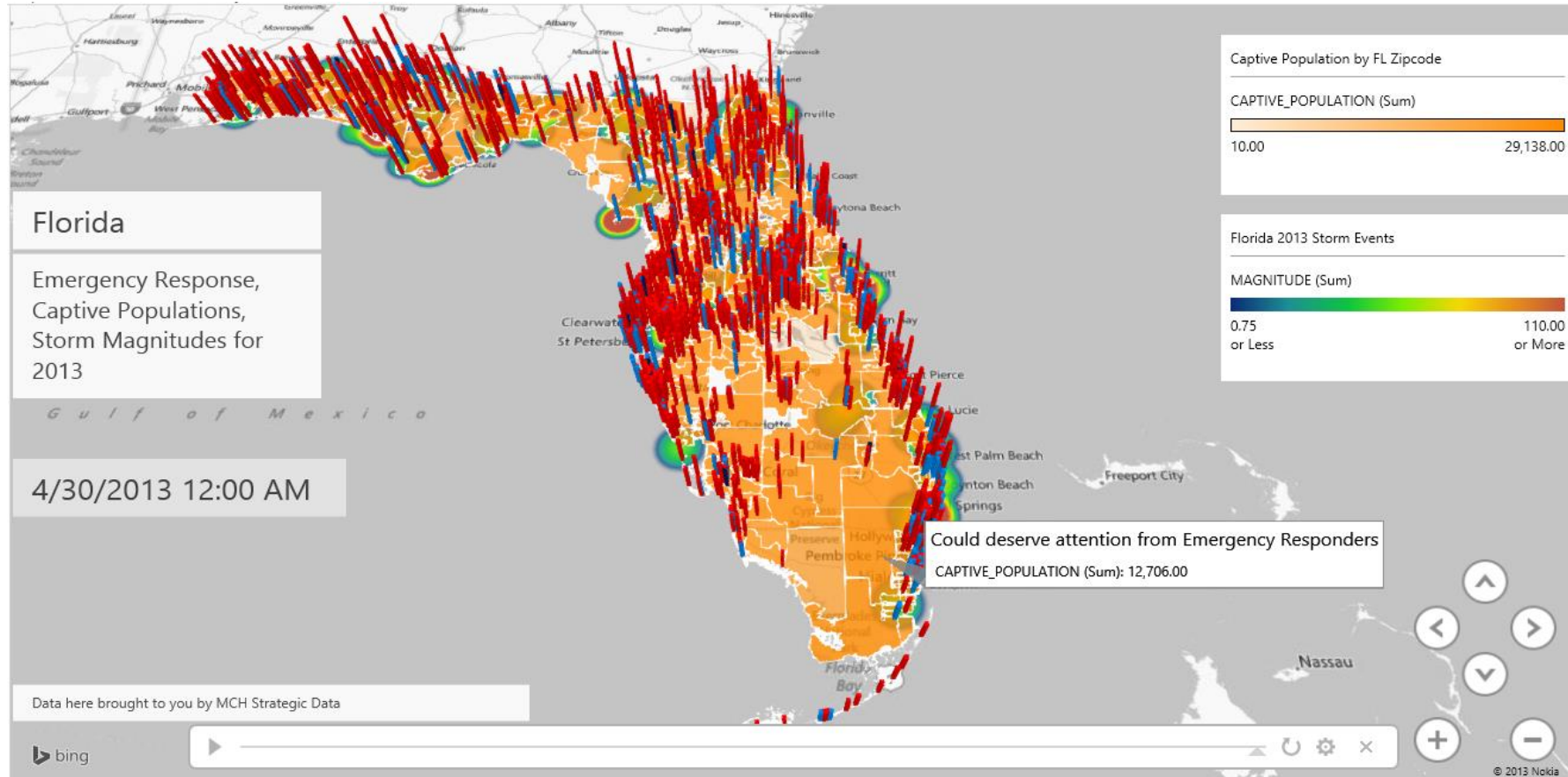
Ambiente di test

- Eseguire SQL Server Management Studio
- Download: <https://bit.ly/2K7Jzgu>
- Connettersi all'istanza SQL Azure
- **Server:** delphi-day-2018.database.windows.net
- **User:** TSQLDDay2018
- **Password:** 94483ycWSmrC22893etWS995!5k88X
- **Database:** WideWorldImporters, AdventureWorks2017

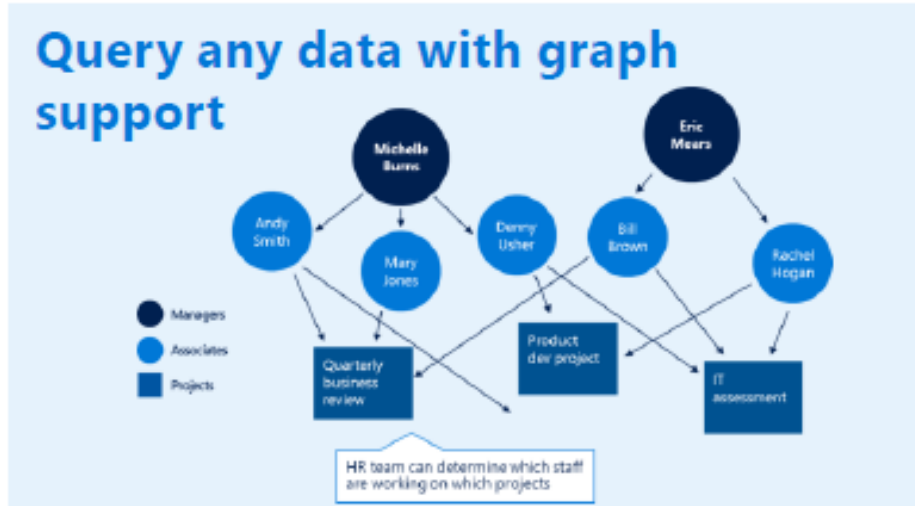


What is a Graph DB?

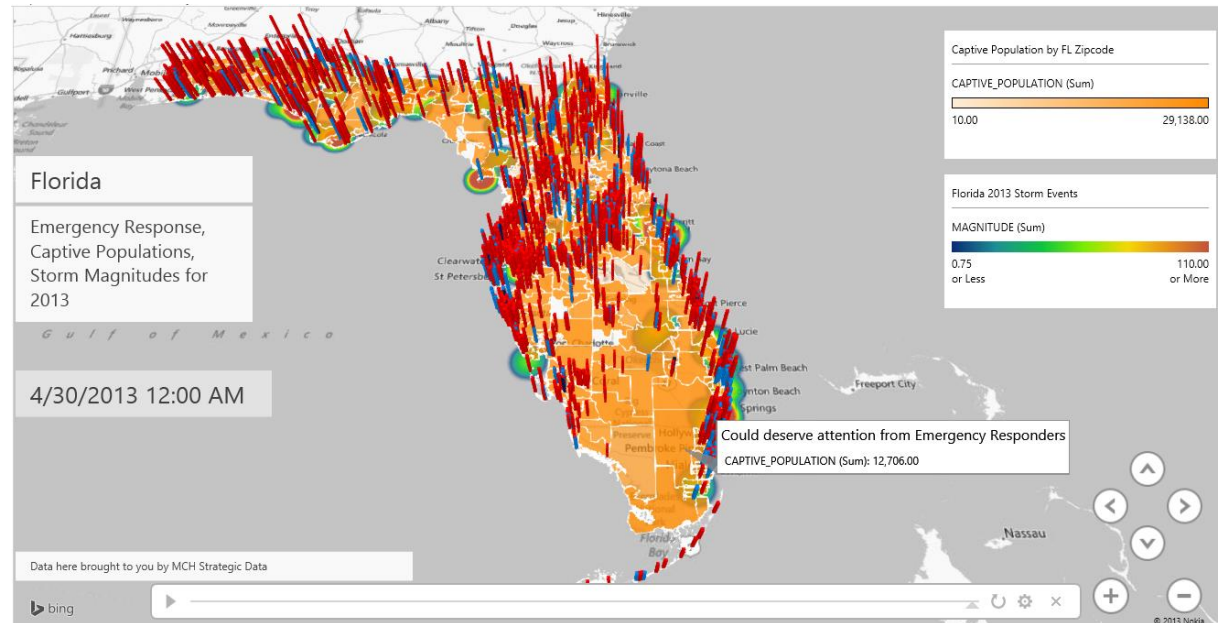
Is this a Graph DB?




We won't talk about data visualization



- Store and analyze non-hierarchical relationships with **Graph data support**
- **PolyBase** to easily query across SQL Server and data stored in Hadoop
- **Hadoop** combined with **SQL Server** provides value and insight from data lakes



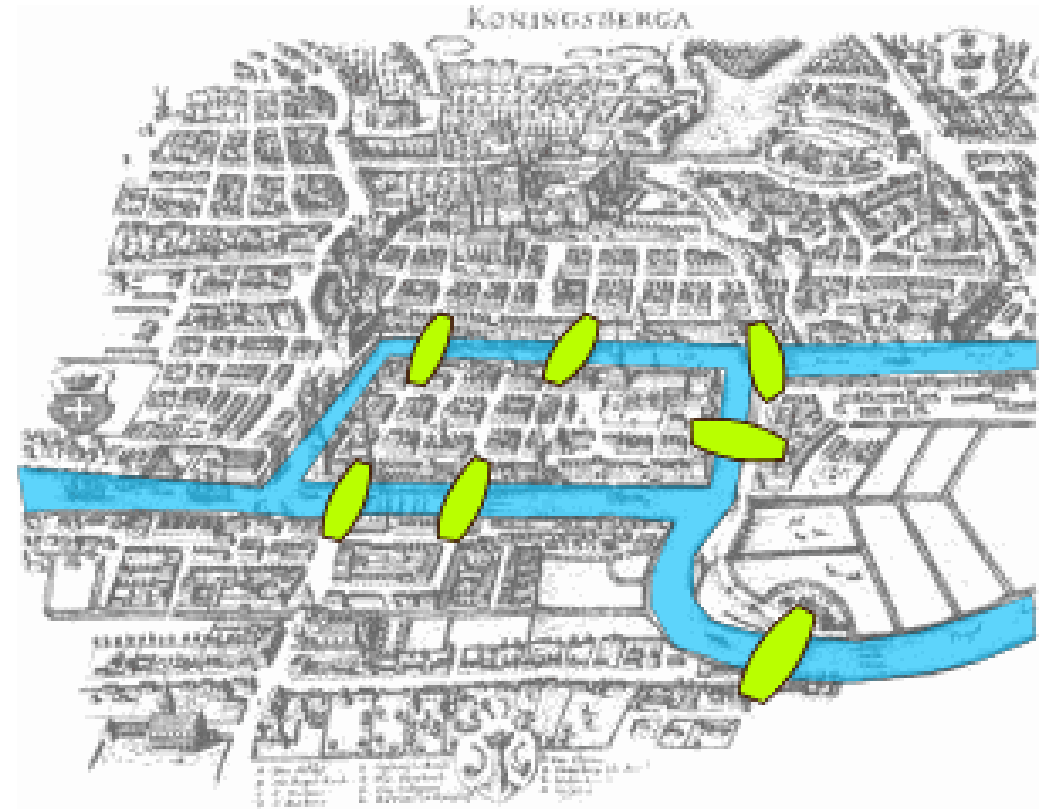
<https://powerbicdn.azureedge.net/mediahandler/blog/legacymedia/8463.Power-Map-for-Excel.png>



When and where was the
graphs theory born?

Graphs theory was born in Konigsberg

- The Seven Bridges of Konigsberg (Kaliningrad, Russian, since 1945) is a historically notable problem in mathematics
- The problem was to invent a walk through the city that would cross each of those bridges once and only once



Leonhard Euler's analysis

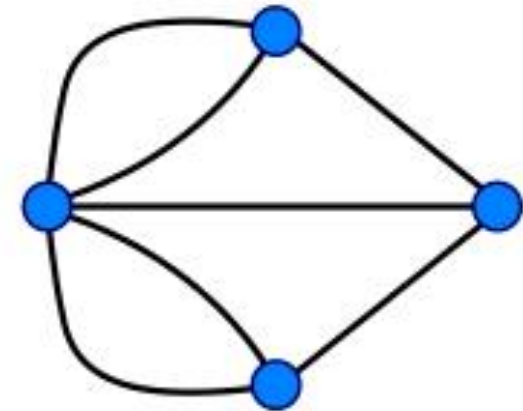
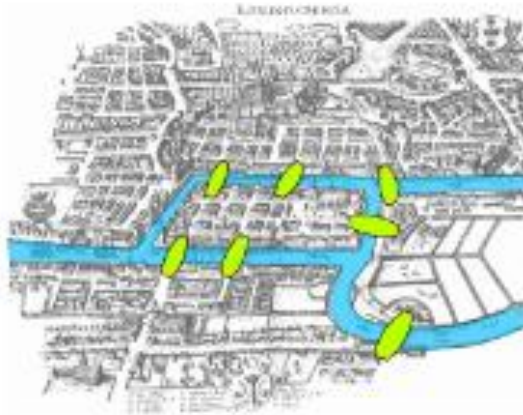


- Euler proved that the problem had no solution!
 - The difficulty he faced was the development of a suitable technique of analysis, and of subsequent tests
- He pointed out that whatever route you choose, the only important thing is the sequence in which you cross the **bridges**
- He replaced each land with an abstract vertex or **node**, and each bridge with an abstract connection, an **edge**

Leonhard Euler's analysis



- Euler's analysis of the Königsberg bridge problem is considered to be the first theorem of Graph Theory



A large, stylized teal graphic on the left side of the page, consisting of several overlapping, rounded, parallel lines that curve from the top-left towards the bottom-right, resembling a thick, curved arrow or a decorative element.

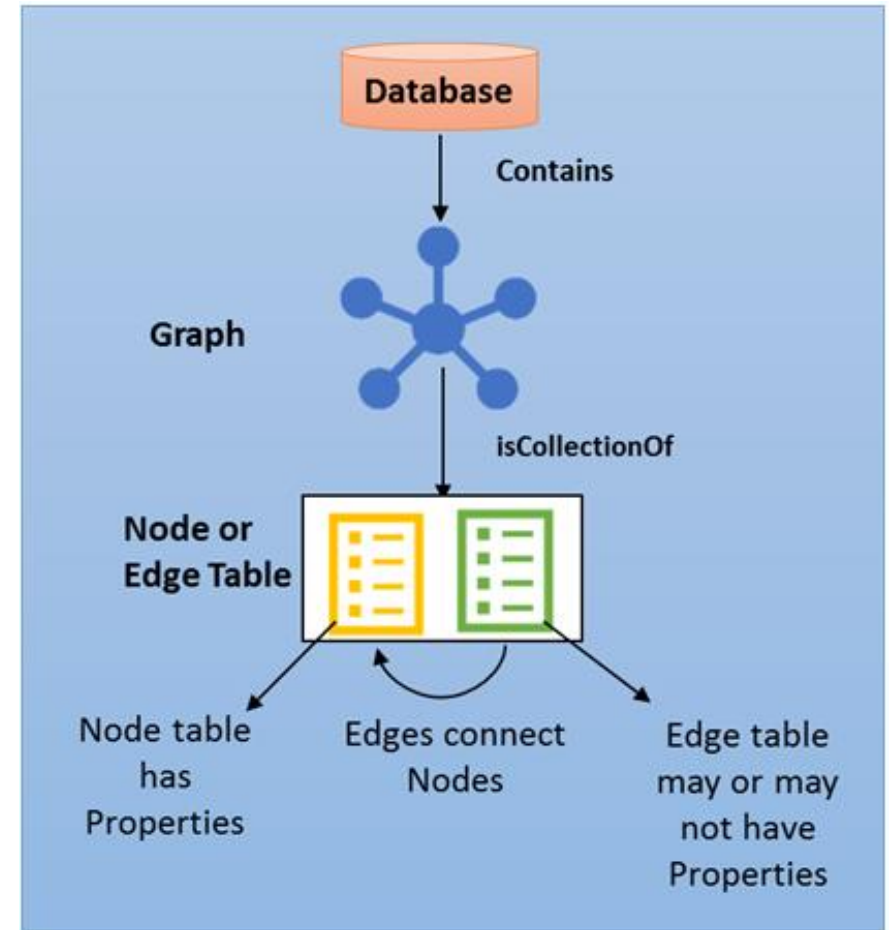
SQL Graph Database

Why SQL Graph Database?

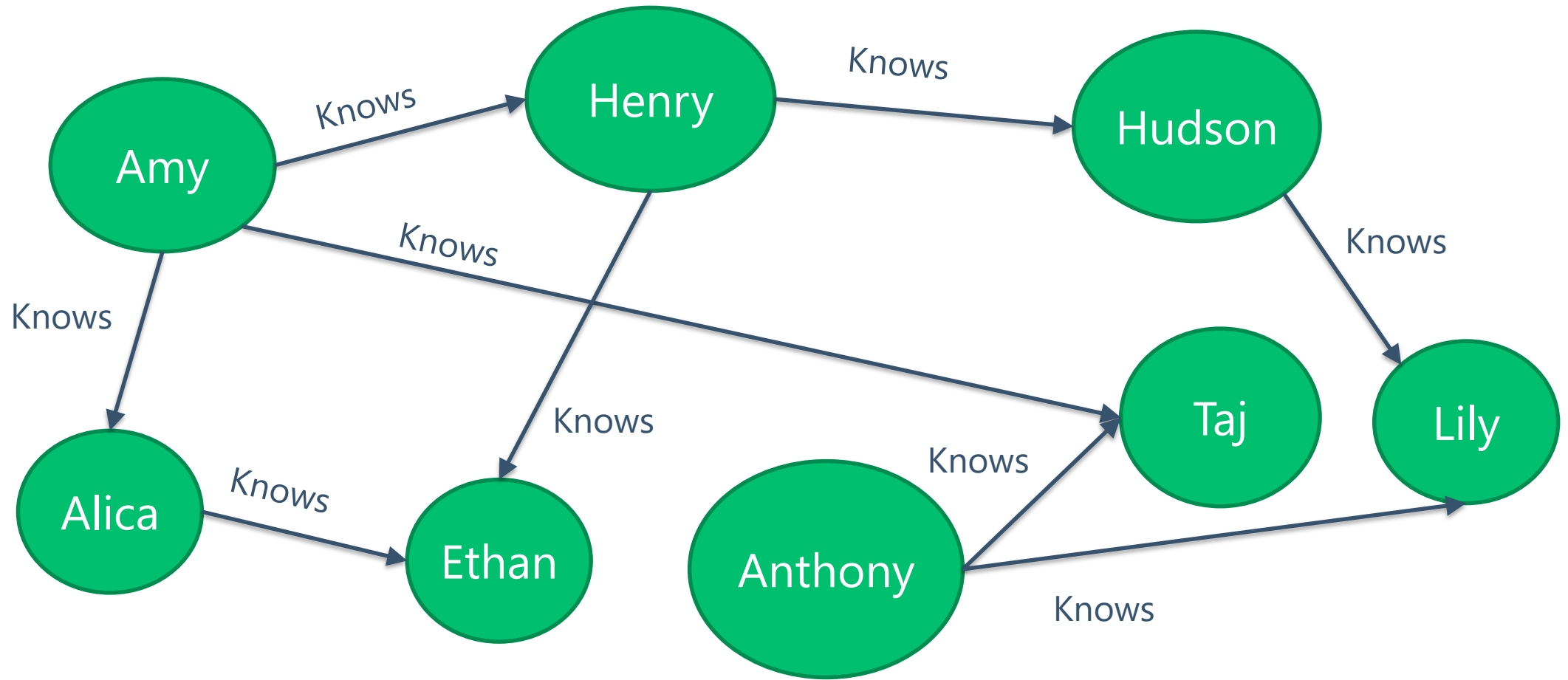
- There is nothing you can do with a graph database, which cannot be done using a relational database
- A graph database can express certain kind of queries more easily than a relational database
- RDBMS is optimized for aggregated data, GDB is optimized for highly connected data

SQL Graph Database: Overview

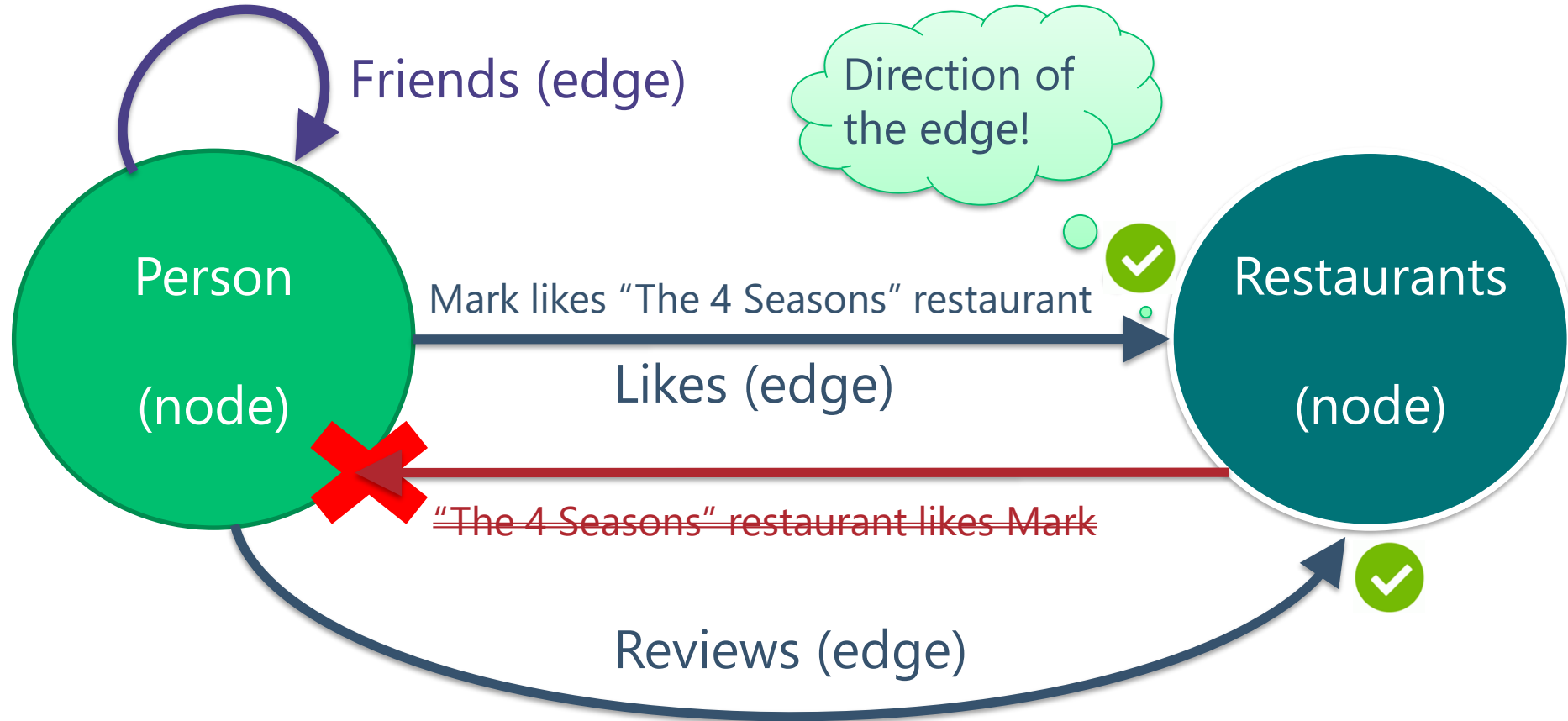
- SQL Graph is fully integrated in the SQL Engine
 - No third parts tools are needed
- Only one logical graph can be created per database
- A graph is a collection of **nodes** and **edges** tables that can be created under any schema
 - Two new table types
 - New T-SQL function **MATCH**



Social graph



Sample graph schema

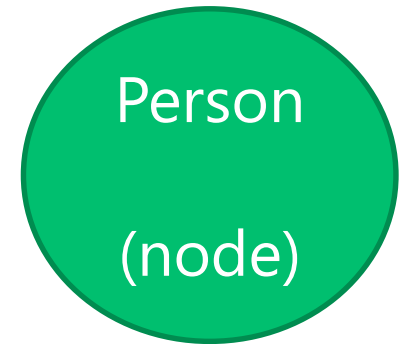


Graph objects: Node

- A node table represents an entity in a Graph DB
- Every time a node is created, in addition to the user defined columns, the Engine will create an implicit column named **\$node_id**
 - It uniquely identifies a given node in the database
 - It contains a combination of the `object_id` of the node and an internally *bigint* stored in an hidden column named `graph_id`

DDL Extension: Create a node table

```
CREATE TABLE Nodes.Person
(
  PersonID INTEGER IDENTITY(1, 1) NOT NULL PRIMARY KEY
  ,FirstName NVARCHAR(50) NOT NULL
  ,LastName NVARCHAR(50) NOT NULL
) AS NODE;
```



Node properties

\$node_id	PersonID	FirstName	LastName
{"type":"node","schema":"dbo","table":"Person","id":0}	1	Sergio	Govoni
{"type":"node","schema":"dbo","table":"Person","id":1}	2	Gianluca	Sartori
{"type":"node","schema":"dbo","table":"Person","id":2}	3	Danilo	Dominici
{"type":"node","schema":"dbo","table":"Person","id":3}	4	Gianluca	Hotz

Graph objects: Edge

- An edge table
 - Represents a relation in a graph
 - May or may not have any user defined attributes
- Edges are always directed and connected with two nodes
- In the first release, constraints are not available on the edge table, so an edge table can connect any two nodes on the graph

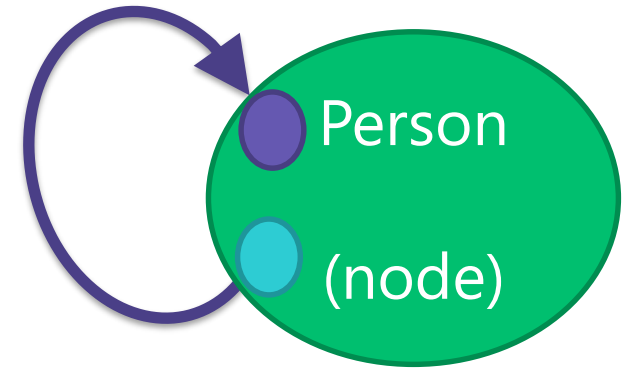
Graph objects: Edge

- Every time an edge table is created, in addition to the user defined columns, the Engine will create three implicit columns
 - **\$edge_id** is a combination of the object_id of the edge and an internally bigint stored in an hidden column named graph_id
 - **\$from_id** stores the \$node_id of the node where the edge starts from
 - **\$to_id** stores the \$node_id of the node at which the edge ends

DDL Extension: Create an edge table

```
CREATE TABLE Edges.Friends  
(  
  StartDate DATETIME NOT NULL  
)  
AS EDGE;
```

Friends (edge)



Nodes connected by this edge

Edge property

\$edge_id	\$from_id	\$to_id	StartDate
{"type":"edge","table":"Friends","id":0}	{"type":"node","table":"Person","id":0}	{"type":"node","table":"Person","id":1}	01/01/2017
{"type":"edge","table":"Friends","id":1}	{"type":"node","table":"Person","id":1}	{"type":"node","table":"Person","id":2}	08/17/2017
{"type":"edge","table":"Friends","id":2}	{"type":"node","table":"Person","id":0}	{"type":"node","table":"Person","id":2}	08/21/2017

Query Extension: The MATCH clause

- Starting with SQL Server 2017, the MATCH clause allows you to specify the search pattern for a graph schema
- It can be used only
 - With graph node and edge tables
 - In SELECT statements, as a part of the WHERE clause
- Nowadays, OR and NOT operators are not supported in the MATCH pattern

Query Extension: The MATCH clause

- Search pattern goes through one node to another by an edge, in the direction provided by the arrow
- Edge names or aliases are provided inside parenthesis
- Node names or aliases appear at the two ends of the arrow



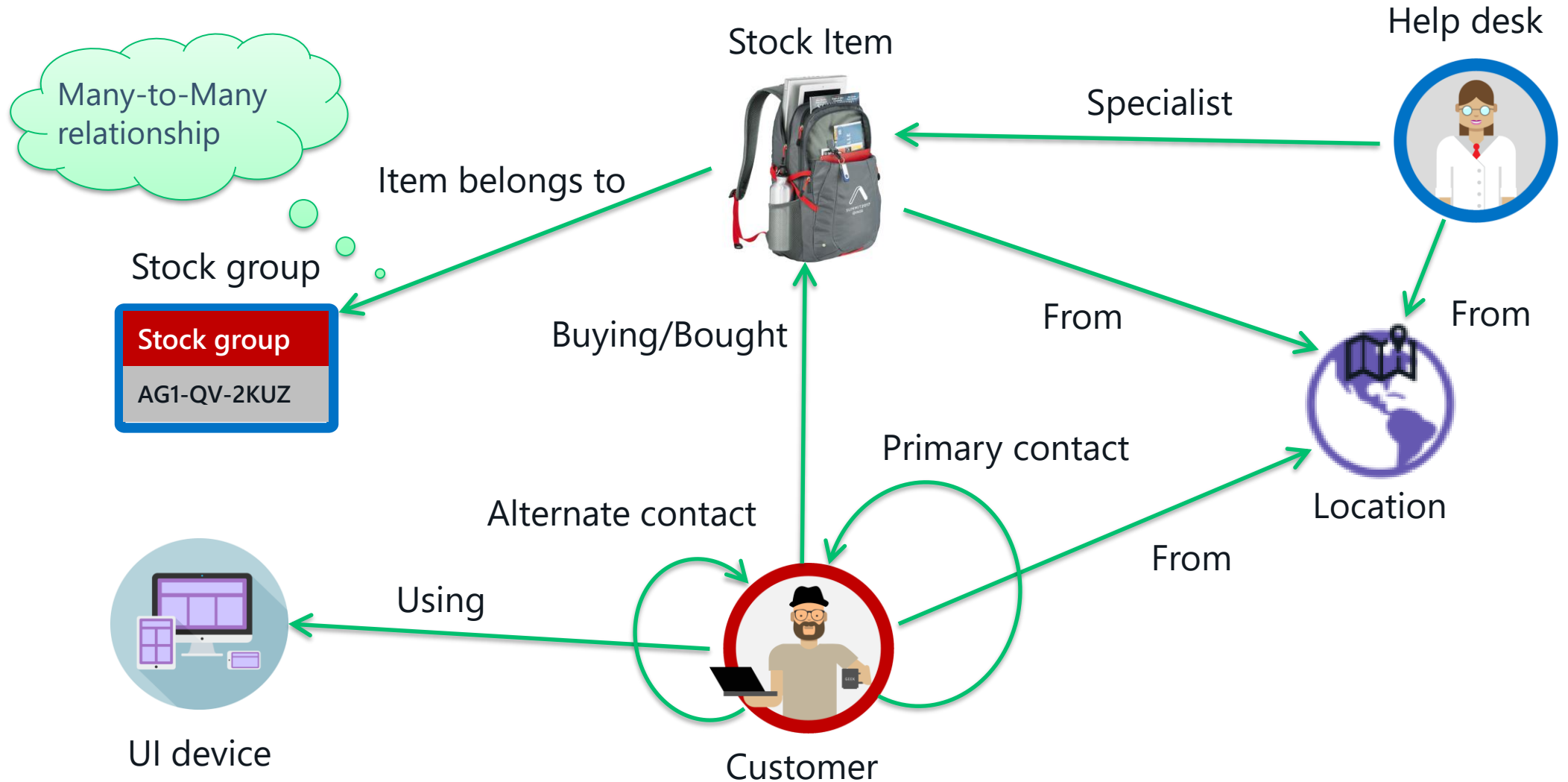
DEMO

Nodes, Edges and the MATCH function

A decorative graphic on the left side of the slide, consisting of several overlapping, curved teal shapes that resemble stylized arrows or abstract lines pointing towards the right.

Sales Recommendation System

Sales Recommendation: Scenario



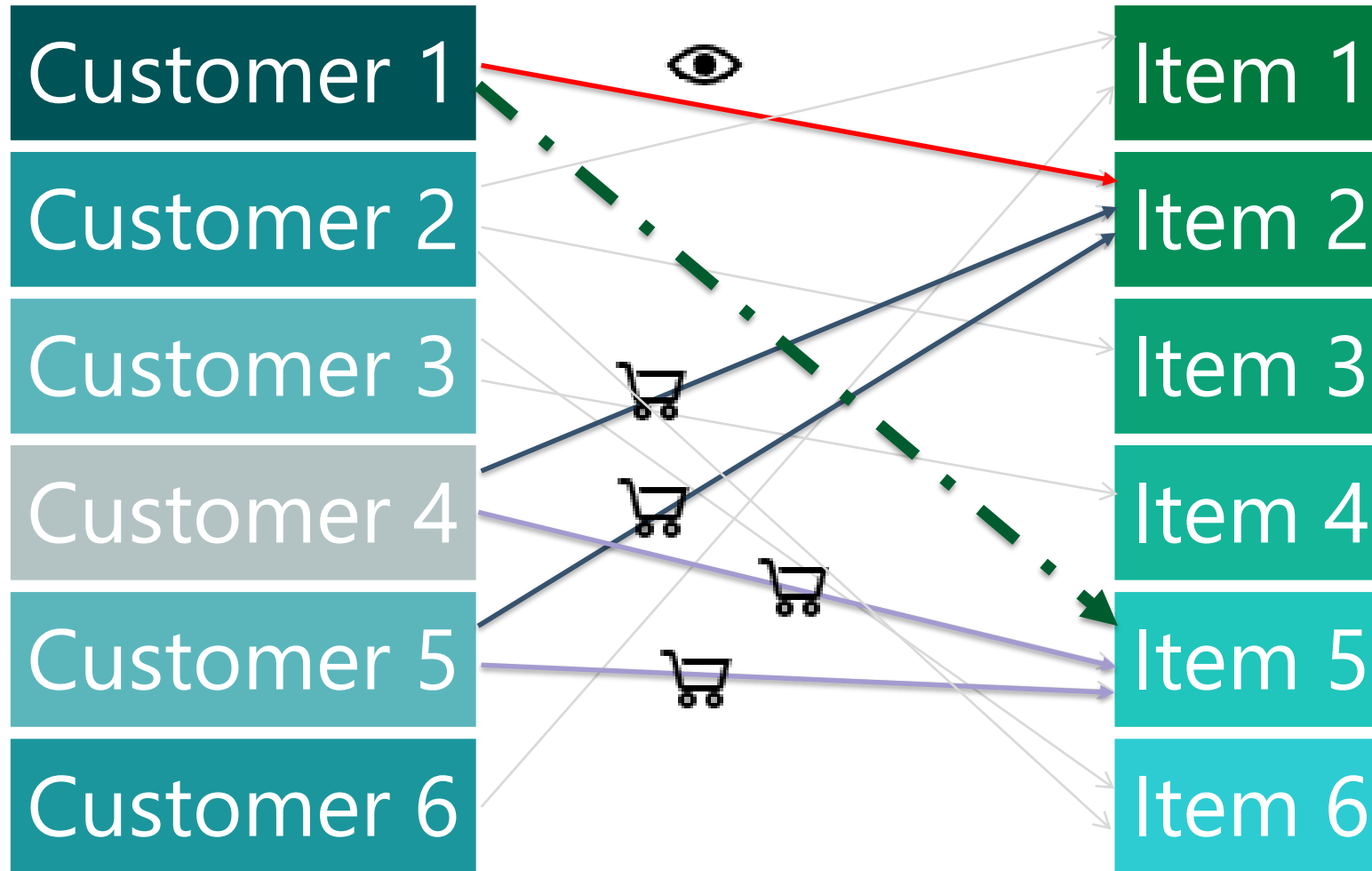
Sales Recommendation: Requirements

- Supposing to have a user connected to your e-commerce, this user is looking for a product or he/she has just bought a product
- Our goal is to find products similar to that one he/she has bought, based on other customers' behavior
 - Find products that are recommended for another one

Sales Recommendation: The algorithm

1. Identify the customer and the product he/she is purchasing
2. Identify the other customers who have purchased the same item he/she is looking for
3. Find the other products that customers, at step two, have purchased
4. Recommend to the current customer the top items from the previous step, ordered by the number of times they were purchased


Sales Recommendation System





DEMO

Build a recommendation system
using SQL Graph



New T-SQL functions for graph objects

New T-SQL functions for graph objects

- Node functions related
 - OBJECT_ID_FROM_NODE_ID(\$node_id)
 - GRAPH_ID_FROM_NODE_ID(\$node_id)
 - NODE_ID_FROM_PARTS(object_id, graph_id)
- Edge functions related
 - OBJECT_ID_FROM_EDGE_ID(\$edge_id)
 - GRAPH_ID_FROM_EDGE_ID(\$edge_id)
 - EDGE_ID_FROM_PARTS(object_id, graph_id)



DEMO

New T-SQL functions for graph objects

Known issues

- Today we can
 - Insert duplicates in an edge
 - Delete a \$node_id with relationships in edges table
- Update the columns \$from_id and \$to_id is not allowed, you have to use DELETE/INSERT

Limitations

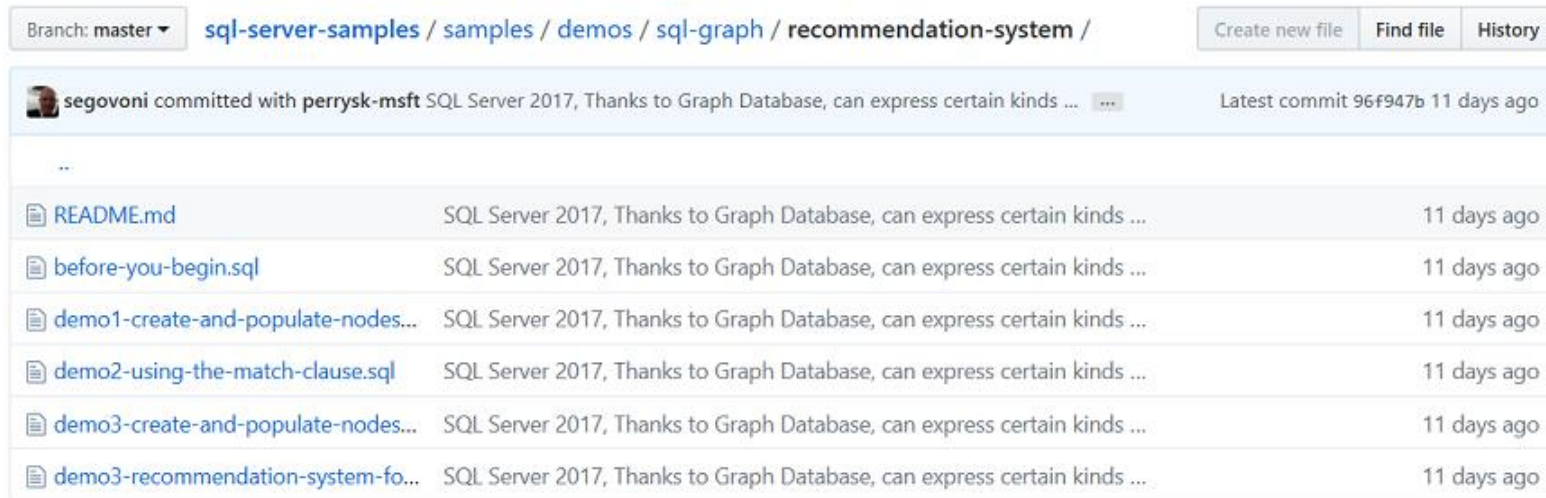
- Today we can't define a node or an edge table as
 - Local or global temporary table
 - In-Memory optimized table
 - Table type or a table variable
 - System versioned temporal table
- Submit cross database queries that involve graph objects are not allowed

In the road map (from PASS Summit 2017)

- Shortest path
- Heterogeneous associations
 - Find restaurants or stock items that one likes
- Language enhancements MERGE DML
- Edge constraints

SQL Graph Database on GitHub

- Some weeks ago, Microsoft merged the pull request I have done on the GitHub repository [Microsoft/sql-server-samples](https://github.com/Microsoft/sql-server-samples)



The screenshot shows the GitHub interface for the repository 'sql-server-samples' at the path 'samples / demos / sql-graph / recommendation-system /'. The current branch is 'master'. The commit history table shows a commit by 'segovoni' on '11 days ago' with the message 'SQL Server 2017, Thanks to Graph Database, can express certain kinds ...'. The table lists several files that were added in this commit, all dated '11 days ago':

File Name	Commit Message	Time
..		
README.md	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago
before-you-begin.sql	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago
demo1-create-and-populate-nodes...	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago
demo2-using-the-match-clause.sql	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago
demo3-create-and-populate-nodes...	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago
demo3-recommendation-system-fo...	SQL Server 2017, Thanks to Graph Database, can express certain kinds ...	11 days ago

Summary

- RDBMS is optimized for aggregated data, GDB is optimized for highly connected data
- You can evaluate to use SQL Graph DB when your application has
 - Hierarchical data with multiple parents
 - Complex many-to-many relationships
 - To analyze interconnected data and relationships

Some useful resources

- [Graph processing with SQL Server and Azure SQL Database](#)
- [SQL Graph Architecture](#)
- [SQL Graph in SQL Server 2017](#)
- [Arvind Shyamsundar's blog](#)



Questions?



[#DelphiDay](#)

[@segovoni](#)

Thanks for attending this event!