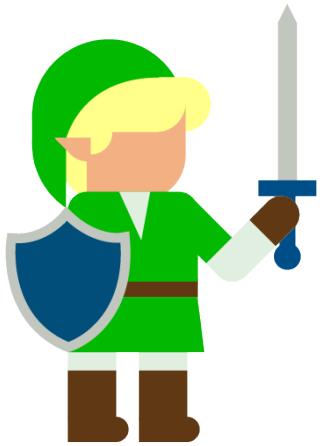
SQLBITS

Enterprise Row Level Security in SQL Server and Power BI

Jamey Johnston



Agenda

- Who am I?
- Row-level Security in MS SQL Server
- Row-level Security in Azure Synapse
- Row-level Security in Power BI
- Questions

Jamey Johnston

- Sr Data Scientist/Engineer
- Microsoft Data Platform MVP
- 30 years of Data Experience
- Prof @ TAMU MS in Analytics
  - http://analytics.stat.tamu.edu
- Semi-Pro Photographer
  - http://jamey.photograhy
- @STATCowboy
- www.STATCowboy.com
- https://github.com/STATCowboy/sqlbits2022-rls-security
  - Download Code Here!

Row Level Security in SQL Server

- RLS allows for controlled access to rows in tables based on attributes of the user executing the query
- 2 Methods or RLS in SQL Server:
  - Filter Based (2005+)
    - SQL Server Security Label Toolkit
    - http://sqlserverlst.codeplex.com/
    - Use views on tables with “labels” to limit access
    - Problem is you have to change the application code and add views (i.e. upgrades are a pain, unsupported applications)
  - Predicate Based (2016+ and Azure)
    - Uses functions and policies to apply predicates to the SQL
    - No application code changes and base database schema left intact (i.e. upgrades not impacted very much by RLS)

Row Level Security: Basic Steps

1. Define Table(s) for RLS
2. Create a new Schema, RLS, for Security Objects
3. Create Table Value Function to define “how” to enforce security on Table
4. Create a Security Policy on the table using the TVF

Table Value Functions

- User defined function that returns a data table
- Powerful alternative to View
  - Expand beyond SELECT and use more powerful T-SQL
- RLS uses them to return a 1 for row matches

```
CREATE FUNCTION RLS.fn_RLSpredicate(@Region AS sysname)
    RETURNS TABLE
WITH SCHEMABINDING
AS
    RETURN SELECT 1 AS fn_RLSpredicate_result
WHERE USER_NAME() = 'VP_US' or @Region = USER_NAME();
GO
```

Security Policy

- Policy that is created to apply the Security Predicate
  - Filter = Select
  - Block = Insert/Update/Delete

```
CREATE SECURITY POLICY Well_HeaderFilter
ADD FILTER PREDICATE RLS.fn_RLSpredicate(Region)
ON dbo.Well_Header
ADD BLOCK PREDICATE RLS.fn_RLSpredicate(Region)
ON dbo.Well_Header AFTER INSERT
GO
```

Why Predicate Based RLS for Business?

- No application code changes and Base database schema left Intact (i.e. upgrades not impacted very much by RLS)
- With ISV applications it is not advisable to change the Schema
- Increased ventures with Internal Partners require row-level granular access to the applications
- RLS allows for the row-level security and eliminates the need for federated/"broken-out" databases/applications

Demo

- Simple RLS Demo in MS SQL Server

Row-level Security in Azure Synapse

Azure Synapse Analytics

- Dedicated SQL Pools
  - Only supports filter predicates
  - **Block predicates aren't currently supported in Azure Synapse.**
  - RLS is supported for External Tables as well!
  - https://docs.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver15
- Serverless SQL Pools
  - https://techcommunity.microsoft.com/t5/azure-synapse-analytics-blog/how-to-implement-row-level-security-in-serverless-sql-pools/ba-p/2354759

Demo

- Simple RLS Demo in Azure Synapse

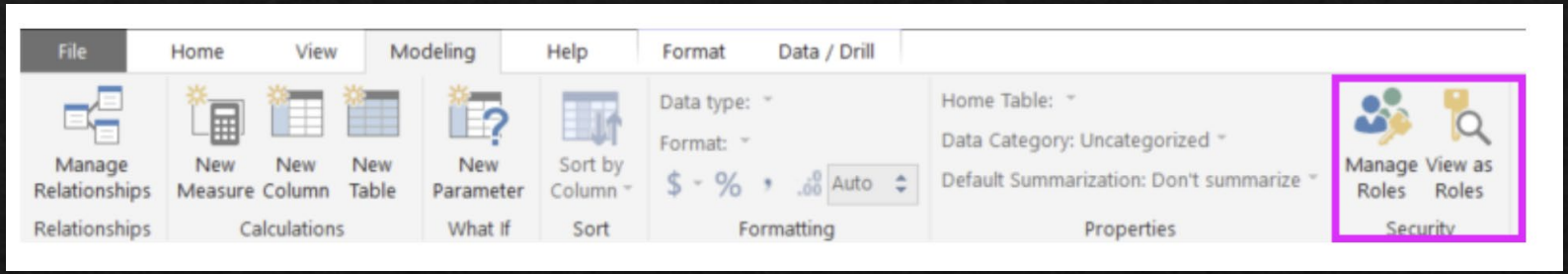
Row-level Security in Power BI

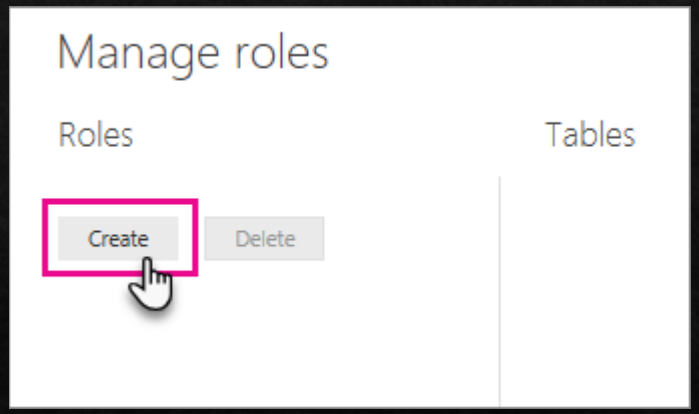
Power BI

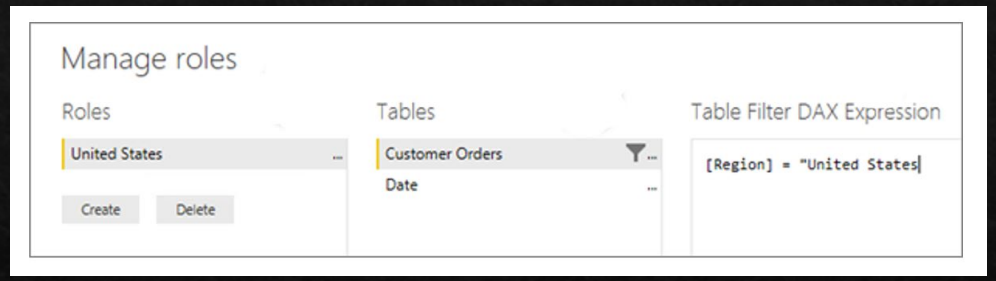
USERNAME() & USERPRINCIPALNAME() in DAX

- The way we determine who is logged into the PowerBI.com Service
- USERNAME() function returns the domain name and username for the currently connected user as DOMAIN\username
- USERPRINCIPALNAME() function returns as username@domain.com

RLS Roles







https://docs.microsoft.com/en-us/power-bi/service-admin-rls

Parent and Child functions

Function	Description
PATH	Returns a delimited text string with the identifiers of all the parents of the current identifier.
PATHCONTAINS	Returns TRUE if the specified *item* exists within the specified *path*.
PATHITEM	Returns the item at the specified *position* from a string resulting from evaluation of a PATH function.
PATHITEMREVERSE	Returns the item at the specified *position* from a string resulting from evaluation of a PATH function.
PATHLENGTH	Returns the number of parents to the specified item in a given PATH result, including self.

https://docs.microsoft.com/en-us/dax/parent-and-child-functions-dax

Demo

- Simple RLS Demo in Power BI

Demo

- Advanced RLS Demo with Hierarchies

RLS with Parent/Child Hierarchies

- Demo will show how an organizational hierarchy and asset hierarchy can be leveraged together to provide RLS on tables using the new predicate based RLS feature in SQL Server 2016 and Azure
- **Important Concepts:**
  - Organization Unit
    - Represents a position in the company (not employee)
    - Security is assigned to the Organization Unit and propagated to the User ID
  - Hierarchy Based Security
    - Allows for inheritance of permissions via the Organization and Asset Hierarchy
      - Do NOT need to assign security to every node in the hierarchy.
      - Child nodes can inherit from Parent Nodes
  - Parent/Child Hierarchy
      - Employee ID / Manager ID - Unary Relationship

Recursive Queries with CTE

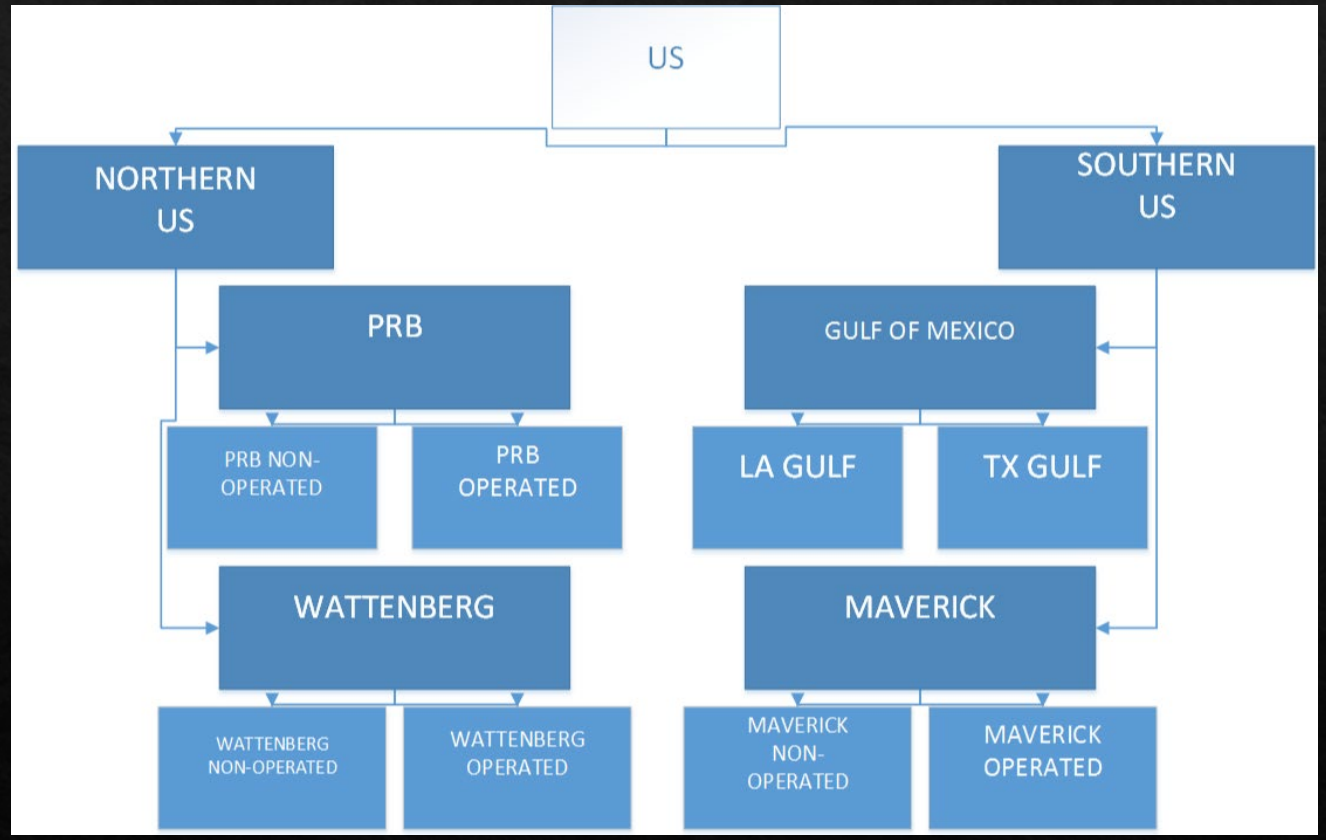
- Use them to query tables with Hierarchical Data

https://technet.microsoft.com/en-us/library/ms186243(v=sql.105).aspx

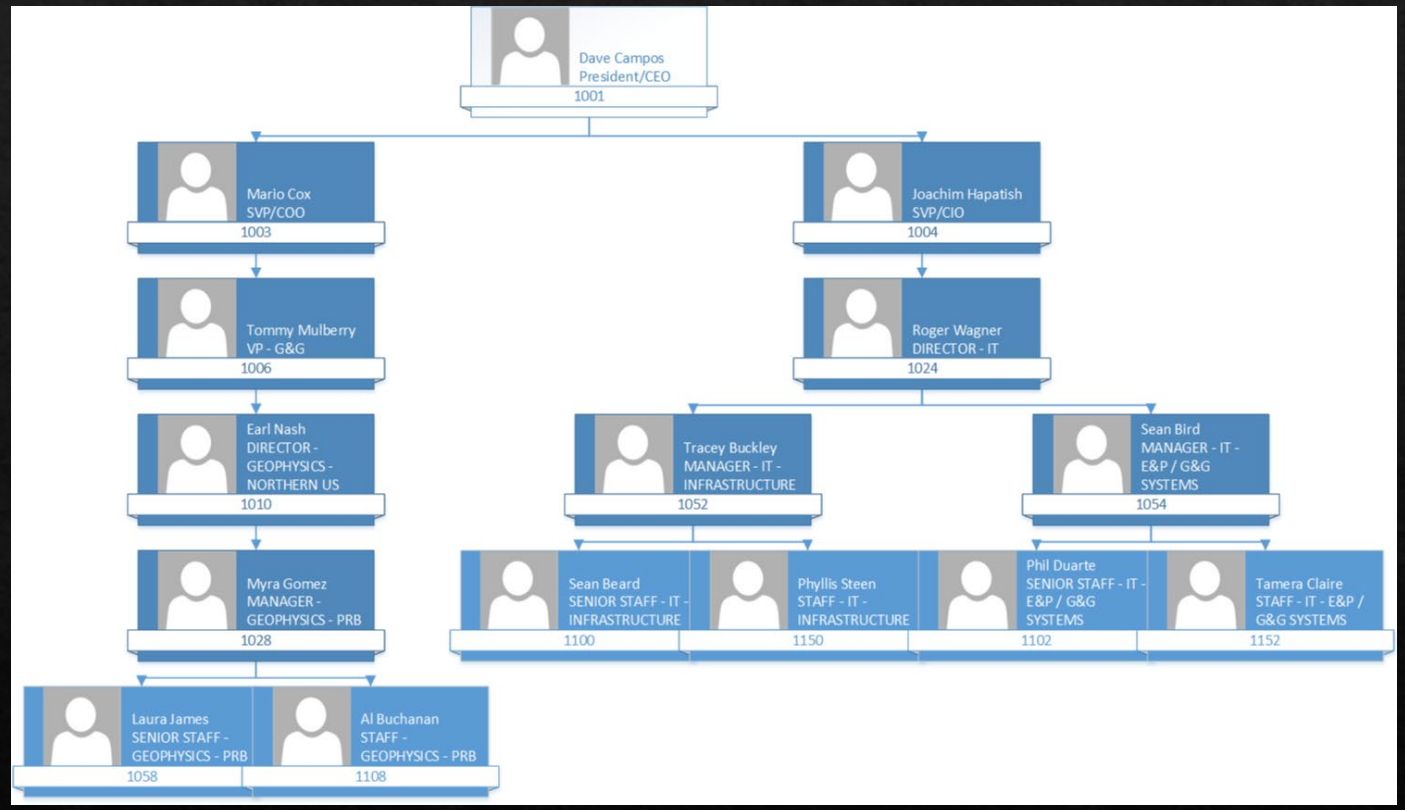
```
WITH cte_name ( column_name [,...n] )
AS(
CTE_query_definition –- Anchor member is defined.
UNION ALL
CTE_query_definition –- Recursive member is defined referencing cte_name.
)
-- Statement using the CTE
SELECT * FROM cte_name
```

Asset Hierarchy

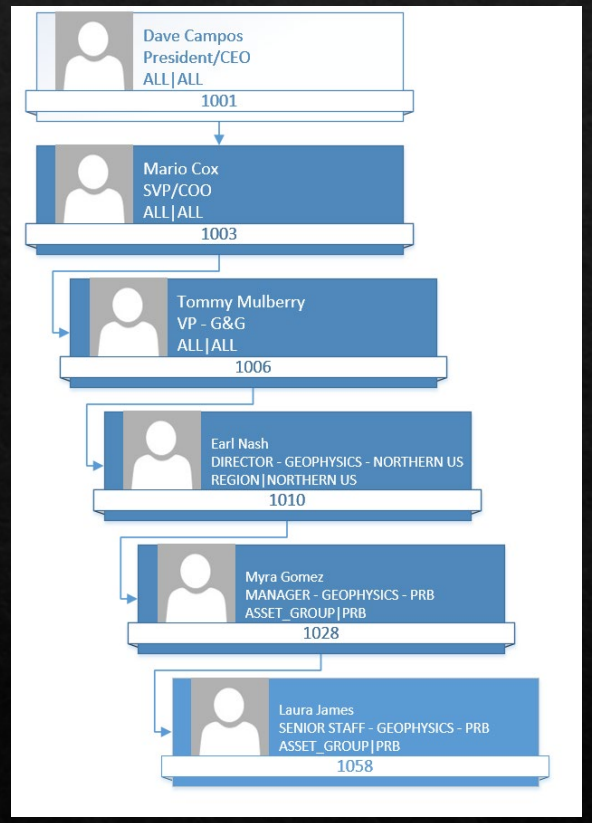
Snapshot of the Asset Hierarchy



Organizational Hierarchy



Hierarchies and RLS



insert into [SEC_ASSET_MAP] values (100001, 'ALL', 'ALL');

Inherits from CEO

Inherits from SVP who Inherits from CEO

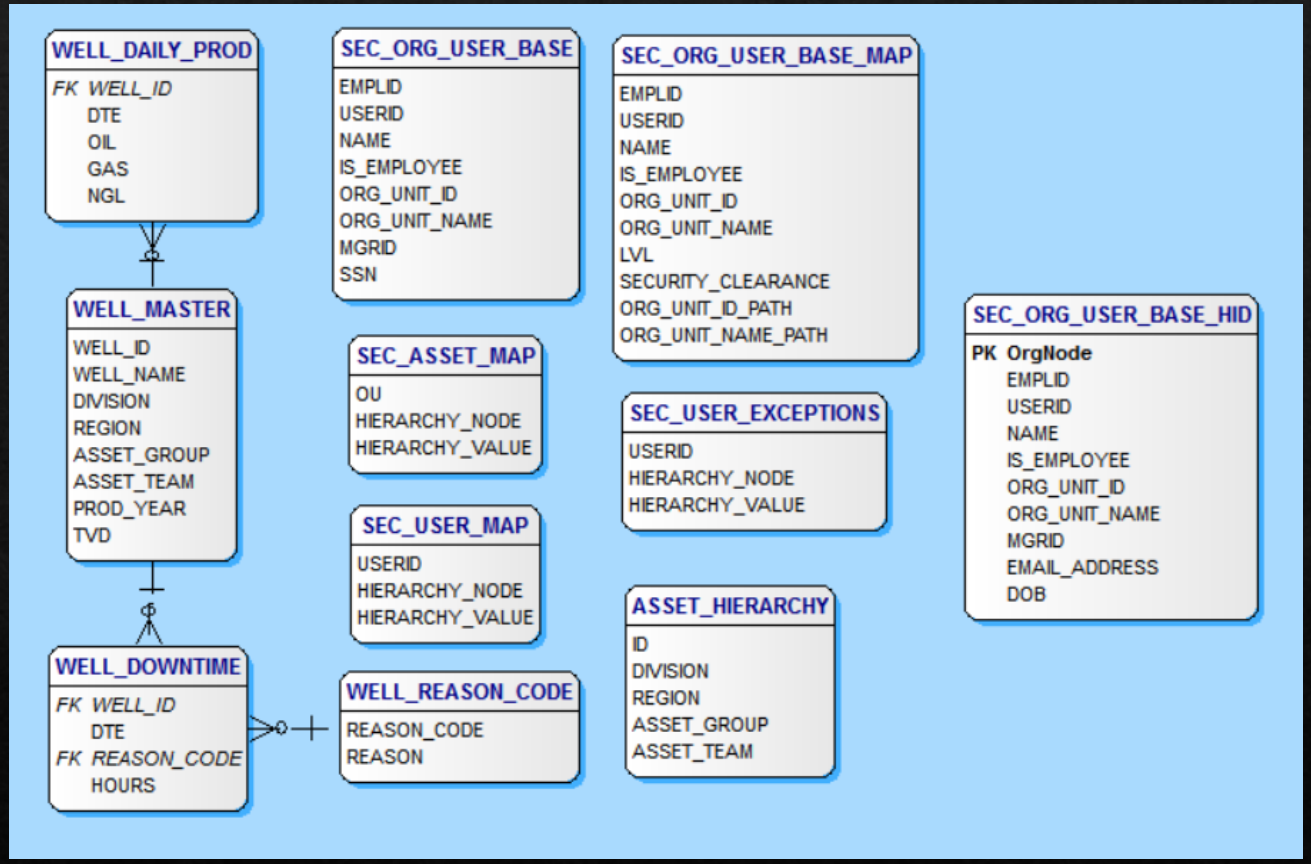
insert into [SEC_ASSET_MAP] values (100010, 'REGION', 'NORTHERN US');

insert into [SEC_ASSET_MAP] values (100028, 'ASSET_GROUP', 'PRB');

Inherits from Manger

Security Record for Every Employee is NOT Required!

Demo ERD


WELL_DAILY_PROD
FK WELL_ID
DTE
OIL
GAS
NGL
WELL_MASTER
WELL_ID
WELL_NAME
DIVISION
REGION
ASSET_GROUP
ASSET_TEAM
PROD_YEAR
TVD
WELL_DOWNTIME
FK WELL_ID
DTE
FK REASON_CODE
HOURS
SEC_ORG_USER_BASE
EMPLID
USERID
NAME
IS_EMPLOYEE
ORG_UNIT_ID
ORG_UNIT_NAME
MGRID
SSN
SEC_ASSET_MAP
OU
HIERARCHY_NODE
HIERARCHY_VALUE
SEC_USER_MAP
USERID
HIERARCHY_NODE
HIERARCHY_VALUE
WELL_REASON_CODE
REASON_CODE
REASON
SEC_ORG_USER_BASE_MAP
EMPLID
USERID
NAME
IS_EMPLOYEE
ORG_UNIT_ID
ORG_UNIT_NAME
LVL
SECURITY_CLEARANCE
ORG_UNIT_ID_PATH
ORG_UNIT_NAME_PATH
SEC_USER_EXCEPTIONS
USERID
HIERARCHY_NODE
HIERARCHY_VALUE
ASSET_HIERARCHY
ID
DIVISION
REGION
ASSET_GROUP
ASSET_TEAM
SEC_ORG_USER_BASE_HID
PK OrgNode
EMPLID
USERID
NAME
IS_EMPLOYEE
ORG_UNIT_ID
ORG_UNIT_NAME
MGRID
EMAIL_ADDRESS
DOB

Questions?

Thank you for attending!

- @STATCowboy
- http://STATCowboy.com
- Download Demos
  - https://github.com/STATCowboy/sqlbits2022-rls-security

Feedback

https://sqlb.it/?7014