Data Management in SQL

Selected Intermediate SQL Skills

By: Margaret Gonsoulin, PhD

October 31, 2016

Thanks!

- Richard Pham
- Mark Dean
- Andy Kelly
- Hans Nielsen

Poll #1: Your CDW Experience

Rate your level of experience with CDW data on a scale of 1 to 5…

1. Not worked with it at all
2. Have minimal experience with it
3. Have work closely with it for <6 months
4. Have work closely with it for 6 months to 2 years
5. Very experienced with CDW

By the end of this talk,

We hope that a you will:

- Feel better prepared to incorporate some of the "best practices" for working with CDW into their queries
- Be comfortable using temporary tables
- Be capable of "recoding a column" using SQL
- Be able to use "partitions" to select desired records when multiple records are present

Overview of 6 "best practices"…

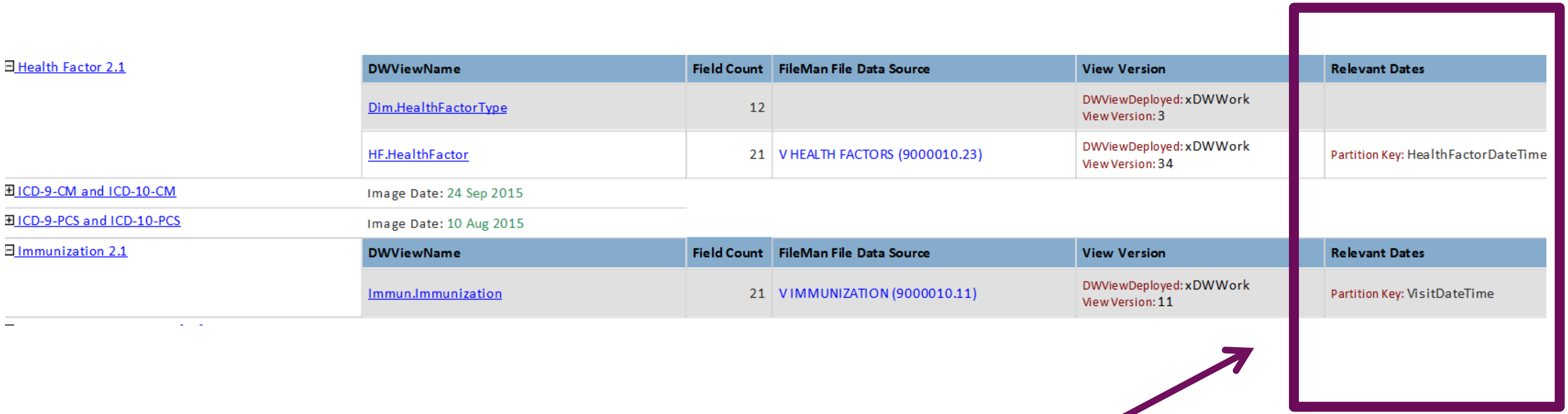
1. Practice with a small amount of data
2. Use "partition dates" or "indexed columns" where possible
3. Convert your CDW date/time fields
4. Use temporary tables until you are sure
5. Explore your estimated query costs before hitting execute
6. Always begin with documentation

1. Practice with small amount of data

- Use TOP # in your SELECT statement to work with a small number of rows
- SELECT only specific columns to work with
- Use the WHERE statement to reduce to a specific station or date range

2. Use “partition keys*” from report

* Partition keys split a single, large table into smaller sub-tables.



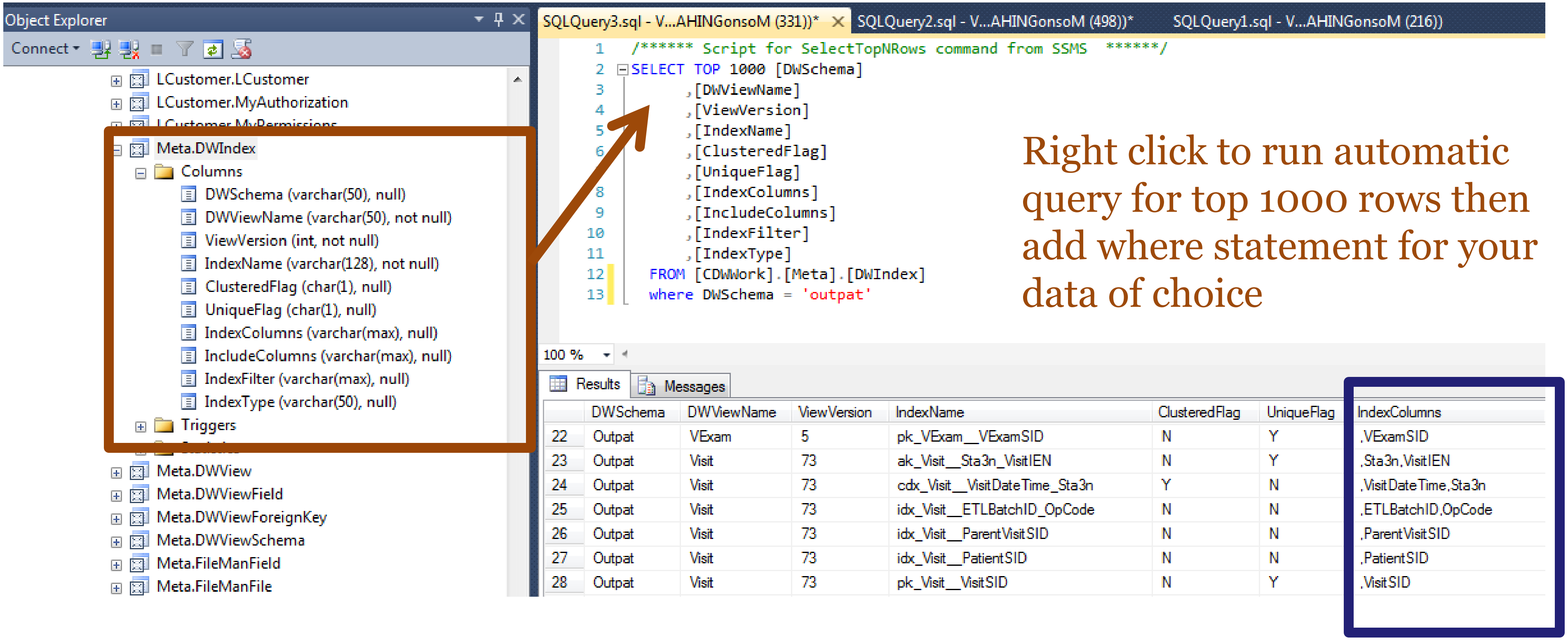
	DWViewName	Field Count	FileMan File Data Source	View Version	Relevant Dates
⊟ Health Factor 2.1	Dim.HealthFactorType	12		DWViewDeployed: xDWWork ViewVersion: 3	
	HF.HealthFactor	21	V HEALTH FACTORS (9000010.23)	DWViewDeployed: xDWWork ViewVersion: 34	Partition Key: HealthFactorDateTime
⊞ ICD-9-CM and ICD-10-CM	Image Date: 24 Sep 2015				
⊞ ICD-9-PCS and ICD-10-PCS	Image Date: 10 Aug 2015				
⊟ Immunization 2.1	DWViewName	Field Count	FileMan File Data Source	View Version	Relevant Dates
	Immun.Immunization	21	V IMMUNIZATION (9000010.11)	DWViewDeployed: xDWWork ViewVersion: 11	Partition Key: VisitDateTime

Look at **Relevant Dates** column in table level metadata report on the CDW SharePoint site

https://vaww.cdw.va.gov/metadata/_layouts/15/ReportServer/RSViewerPage.aspx?rv:RelativeReportUrl=/metadata/Reports/Metadata%20Report.rdl&Source=http%3A%2F%2Fvaww%2Ecdw%2Er02%2Emed%2Eva%2Egov%2Fmetadata%2FReports%2FForms%2FAllItems%2Easpx&DefaultItemOpen=1

2. Or, use "indexed columns*"

*Indexes act as pointers to data in a table, like an index in a book.



Object Explorer
Connect ▾
SQLQuery3.sql - V...AHINGonsoM (331))*
SQLQuery2.sql - V...AHINGonsoM (498))*
SQLQuery1.sql - V...AHINGonsoM (216))
100 %
Results
Messages

- LCustomer.LCustomer
- LCustomer.MyAuthorization
- LCustomer.MyPermissions
- Meta.DWIndex
  - Columns
    - DWSchema (varchar(50), null)
    - DWViewName (varchar(50), not null)
    - ViewVersion (int, not null)
    - IndexName (varchar(128), not null)
    - ClusteredFlag (char(1), null)
    - UniqueFlag (char(1), null)
    - IndexColumns (varchar(max), null)
    - IncludeColumns (varchar(max), null)
    - IndexFilter (varchar(max), null)
    - IndexType (varchar(50), null)
  - Triggers
- Meta.DWView
- Meta.DWViewField
- Meta.DWViewForeignKey
- Meta.DWViewSchema
- Meta.FileManField
- Meta.FileManFile

```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [DWSchema]
      ,[DWViewName]
      ,[ViewVersion]
      ,[IndexName]
      ,[ClusteredFlag]
      ,[UniqueFlag]
      ,[IndexColumns]
      ,[IncludeColumns]
      ,[IndexFilter]
      ,[IndexType]
  FROM [CDWWork].[Meta].[DWIndex]
  where DWSchema = 'outpat'
```

	DWSchema	DWViewName	ViewVersion	IndexName	ClusteredFlag	UniqueFlag	IndexColumns
22	Outpat	VExam	5	pk_VExam__VExamSID	N	Y	,VExamSID
23	Outpat	Visit	73	ak_Visit__Sta3n_VisitIEN	N	Y	,Sta3n,VisitIEN
24	Outpat	Visit	73	cdx_Visit__VisitDateTime_Sta3n	Y	N	,VisitDateTime,Sta3n
25	Outpat	Visit	73	idx_Visit__ETLBatchID_OpCode	N	N	,ETLBatchID,OpCode
26	Outpat	Visit	73	idx_Visit__ParentVisitSID	N	N	,ParentVisitSID
27	Outpat	Visit	73	idx_Visit__PatientSID	N	N	,PatientSID
28	Outpat	Visit	73	pk_Visit__VisitSID	N	Y	,VisitSID

Right click to run automatic query for top 1000 rows then add where statement for your data of choice

3. Convert your CDW date/time fields

See CDW Insights Talk by Andy Kelly on 7/27/2016, on why you should CONVERT your date/time fields in CDW.

```
WHERE VisitDateTime >= CONVERT (DATETIME2(0), '1/1/2015')
AND VisitDateTime <= CONVERT (DATETIME2(0), '1/31/2015')
```

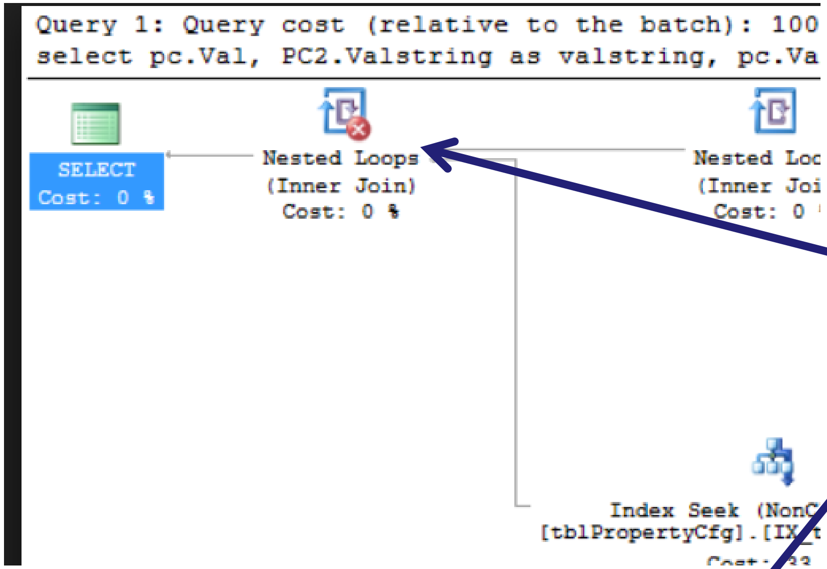
https://vaww.cdw.va.gov/support/Training/cdwinsights/SitePages/Home.aspx

4. Use temporary tables

- Create temporary tables by adding a hashtag to the front of the table name
- It will be erased when you close your query window.

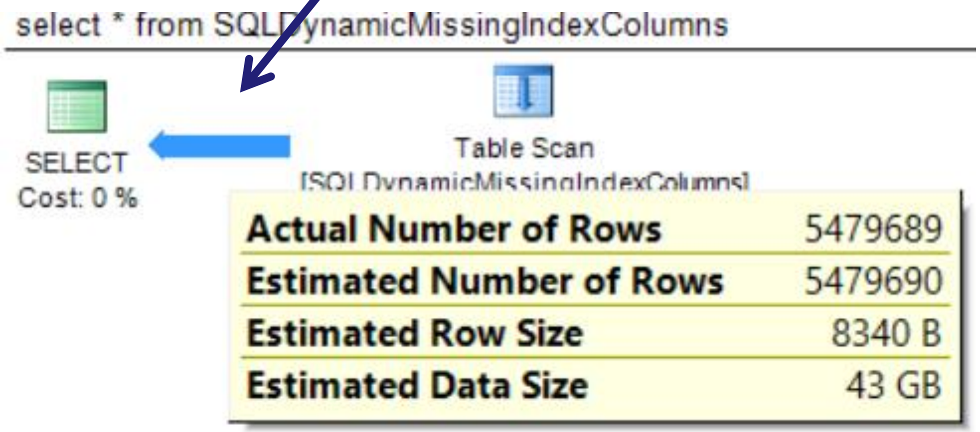
```
SELECT Column1, Column2
INTO #TempTableName
FROM Database.Schema.Table ;
```

5. Check your execution plans



```
Query 1: Query cost (relative to the batch): 100
select pc.Val, PC2.Valstring as valstring, pc.Va
```

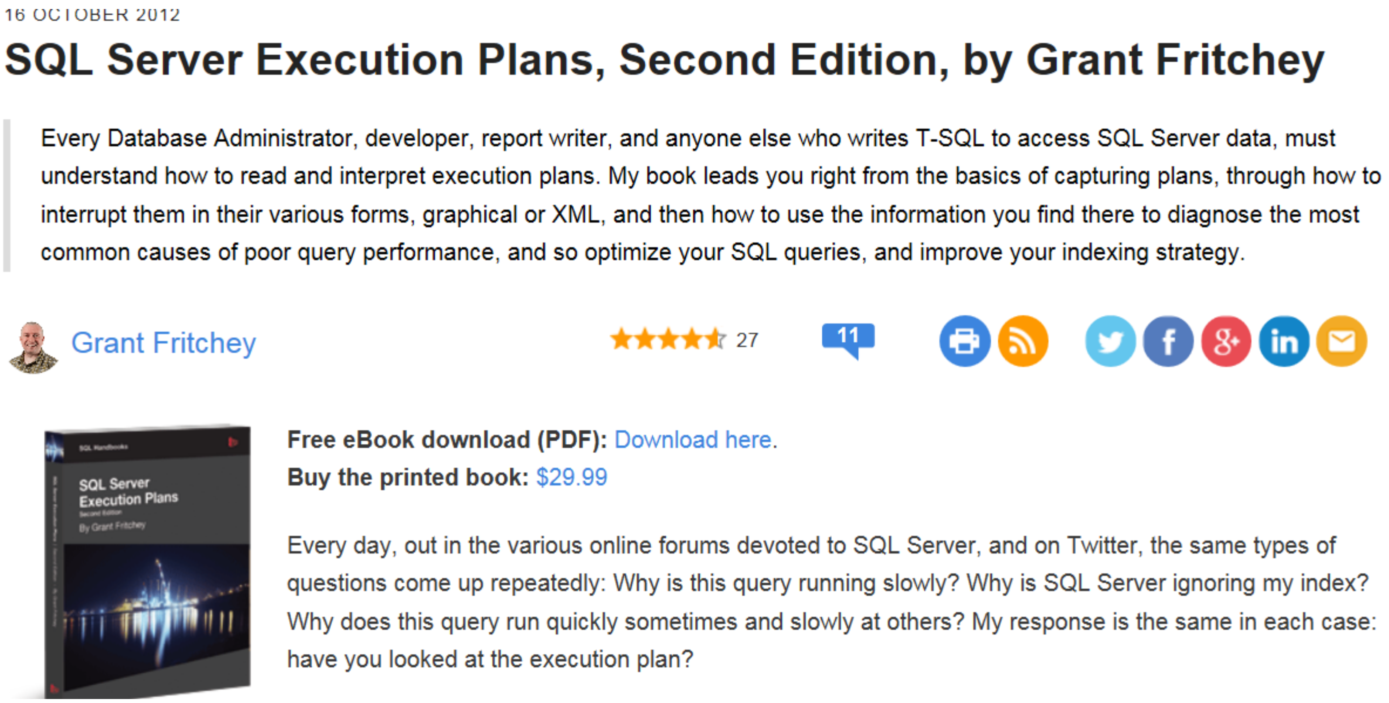
Look out for red "X"s or thick arrows in your execution plan



```
select * from SQLDynamicMissingIndexColumns
```

Actual Number of Rows	5479689
Estimated Number of Rows	5479690
Estimated Row Size	8340 B
Estimated Data Size	43 GB

Read about execution plans…



16 OCTOBER 2012

SQL Server Execution Plans, Second Edition, by Grant Fritchey

> Every Database Administrator, developer, report writer, and anyone else who writes T-SQL to access SQL Server data, must understand how to read and interpret execution plans. My book leads you right from the basics of capturing plans, through how to interrupt them in their various forms, graphical or XML, and then how to use the information you find there to diagnose the most common causes of poor query performance, and so optimize your SQL queries, and improve your indexing strategy.

Grant Fritchey

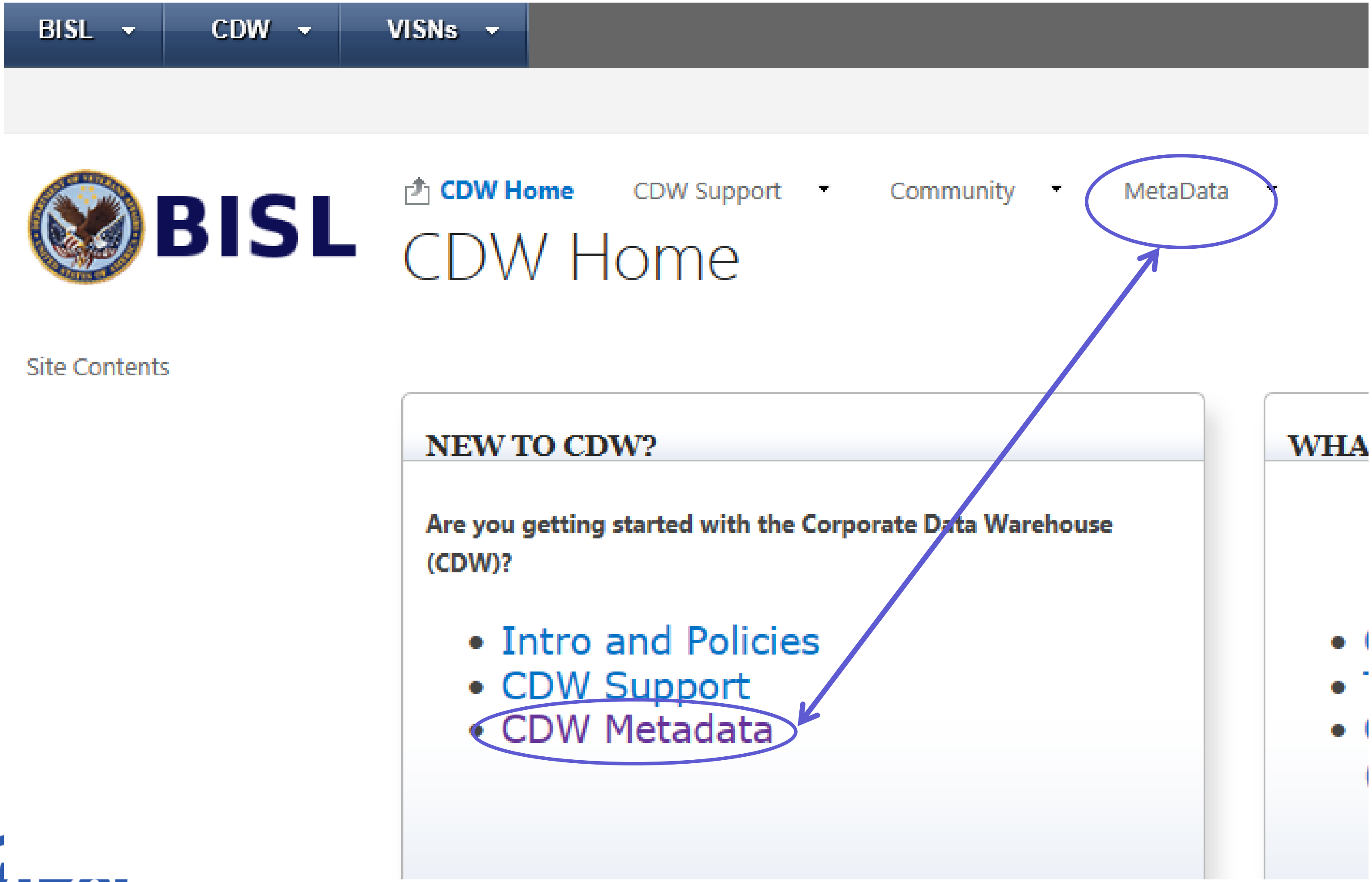
Free eBook download (PDF): Download here.
Buy the printed book: $29.99

Every day, out in the various online forums devoted to SQL Server, and on Twitter, the same types of questions come up repeatedly: Why is this query running slowly? Why is SQL Server ignoring my index? Why does this query run quickly sometimes and slowly at others? My response is the same in each case: have you looked at the execution plan?

https://www.simple-talk.com/books/sql-books/sql-server-execution-plans-second-edition-by-grant-fritchey/

6. Always begin with documentation

a. You might use CDW metadata



CDW Home

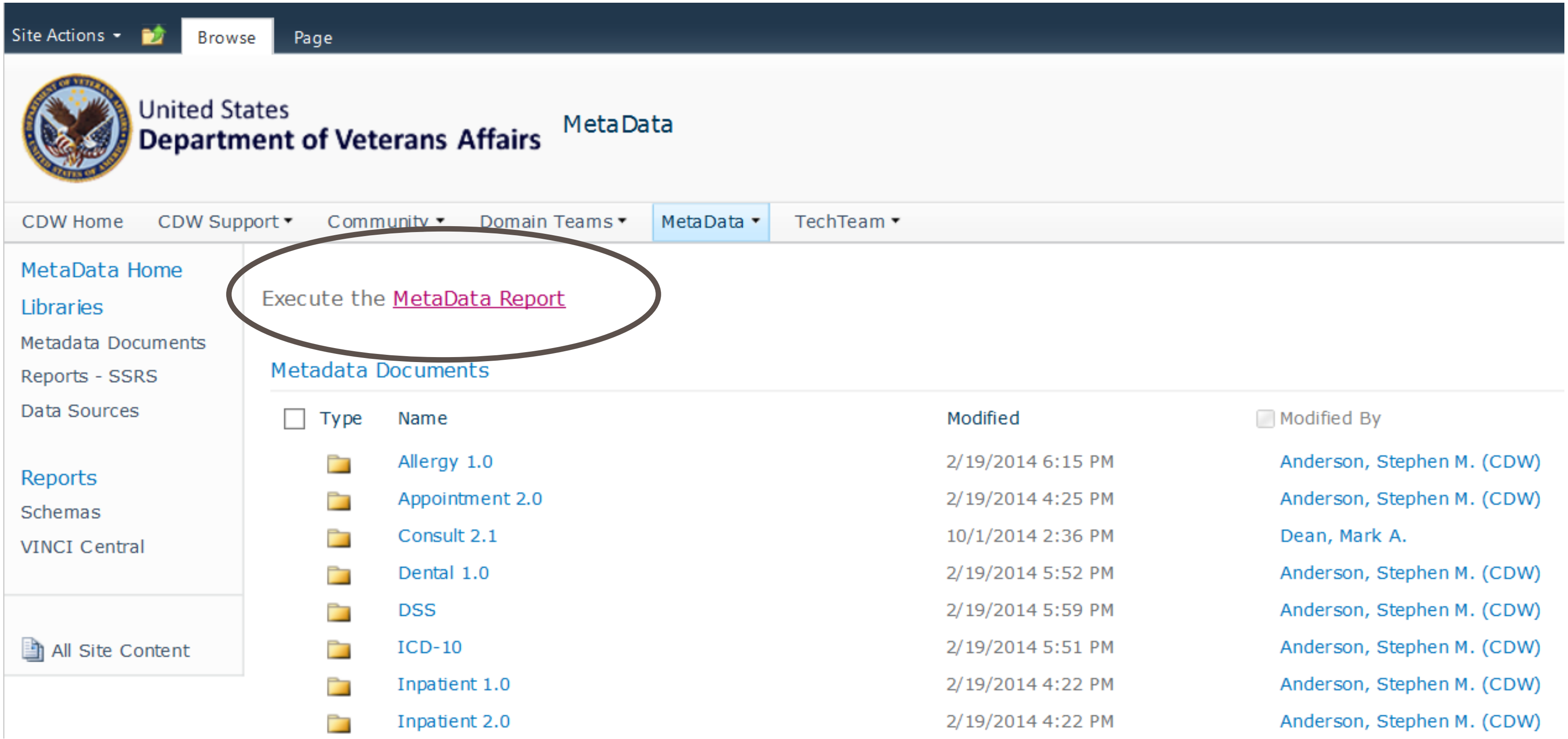
NEW TO CDW?

Are you getting started with the Corporate Data Warehouse (CDW)?

- Intro and Policies
- CDW Support
- CDW Metadata

https://vaww.cdw.va.gov/Pages/CDWHome.aspx

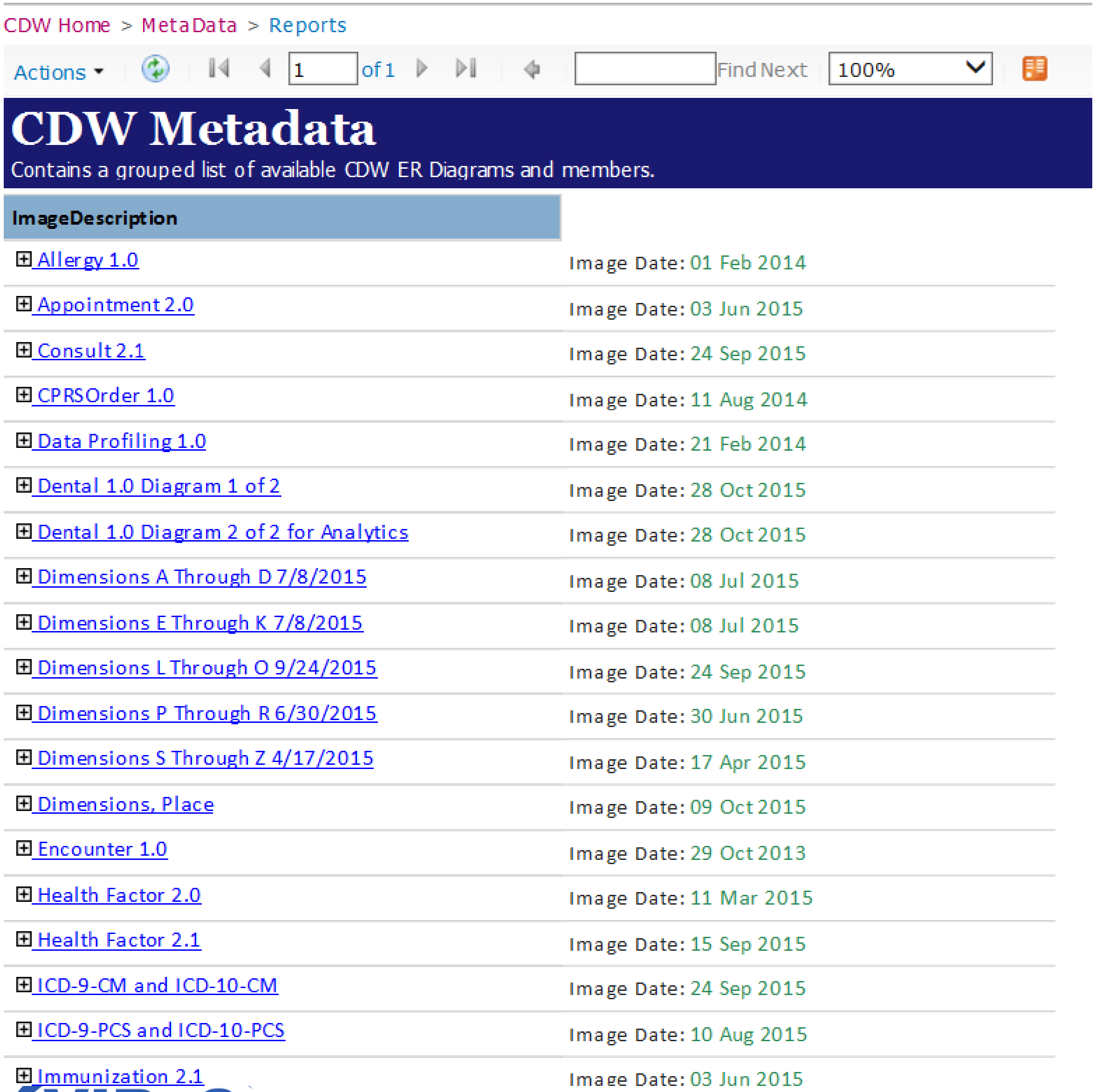
Click “execute the metadata report”



Execute the MetaData Report

Metadata Documents

□ Type	Name	Modified	Modified By
	Allergy 1.0	2/19/2014 6:15 PM	Anderson, Stephen M. (CDW)
	Appointment 2.0	2/19/2014 4:25 PM	Anderson, Stephen M. (CDW)
	Consult 2.1	10/1/2014 2:36 PM	Dean, Mark A.
	Dental 1.0	2/19/2014 5:52 PM	Anderson, Stephen M. (CDW)
	DSS	2/19/2014 5:59 PM	Anderson, Stephen M. (CDW)
	ICD-10	2/19/2014 5:51 PM	Anderson, Stephen M. (CDW)
	Inpatient 1.0	2/19/2014 4:22 PM	Anderson, Stephen M. (CDW)
	Inpatient 2.0	2/19/2014 4:22 PM	Anderson, Stephen M. (CDW)



CDW Metadata

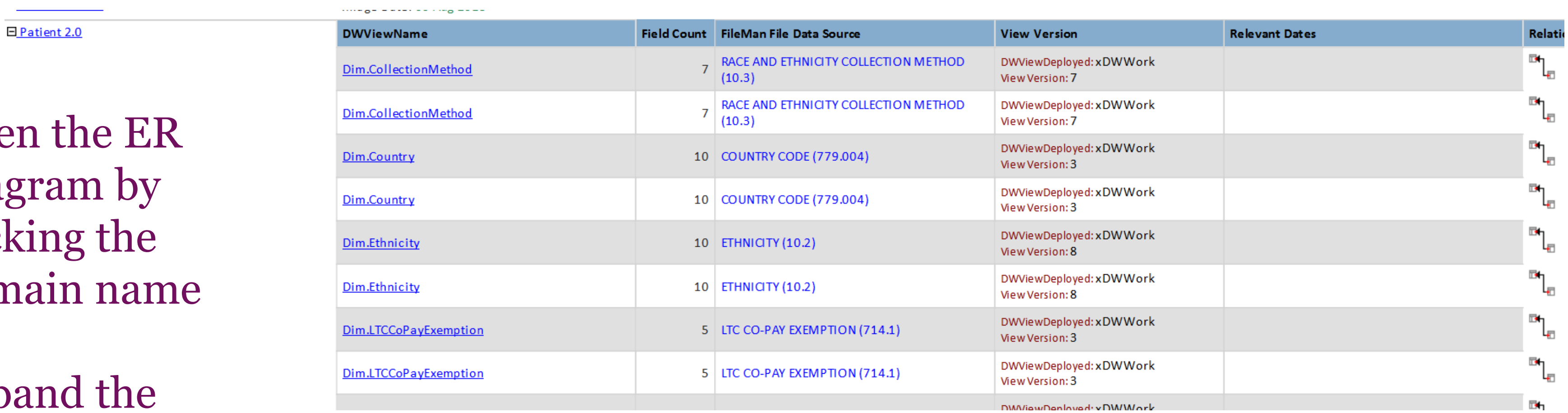
Contains a grouped list of available CDW ER Diagrams and members.

ImageDescription	
⊞ Allergy 1.0	Image Date: 01 Feb 2014
⊞ Appointment 2.0	Image Date: 03 Jun 2015
⊞ Consult 2.1	Image Date: 24 Sep 2015
⊞ CPRSOrder 1.0	Image Date: 11 Aug 2014
⊞ Data Profiling 1.0	Image Date: 21 Feb 2014
⊞ Dental 1.0 Diagram 1 of 2	Image Date: 28 Oct 2015
⊞ Dental 1.0 Diagram 2 of 2 for Analytics	Image Date: 28 Oct 2015
⊞ Dimensions A Through D 7/8/2015	Image Date: 08 Jul 2015
⊞ Dimensions E Through K 7/8/2015	Image Date: 08 Jul 2015
⊞ Dimensions L Through O 9/24/2015	Image Date: 24 Sep 2015
⊞ Dimensions P Through R 6/30/2015	Image Date: 30 Jun 2015
⊞ Dimensions S Through Z 4/17/2015	Image Date: 17 Apr 2015
⊞ Dimensions, Place	Image Date: 09 Oct 2015
⊞ Encounter 1.0	Image Date: 29 Oct 2013
⊞ Health Factor 2.0	Image Date: 11 Mar 2015
⊞ Health Factor 2.1	Image Date: 15 Sep 2015
⊞ ICD-9-CM and ICD-10-CM	Image Date: 24 Sep 2015
⊞ ICD-9-PCS and ICD-10-PCS	Image Date: 10 Aug 2015
⊞ Immunization 2.1	Image Date: 03 Jun 2015

CDW Metadata Report

scroll down to find additional domains

Table-level documentation

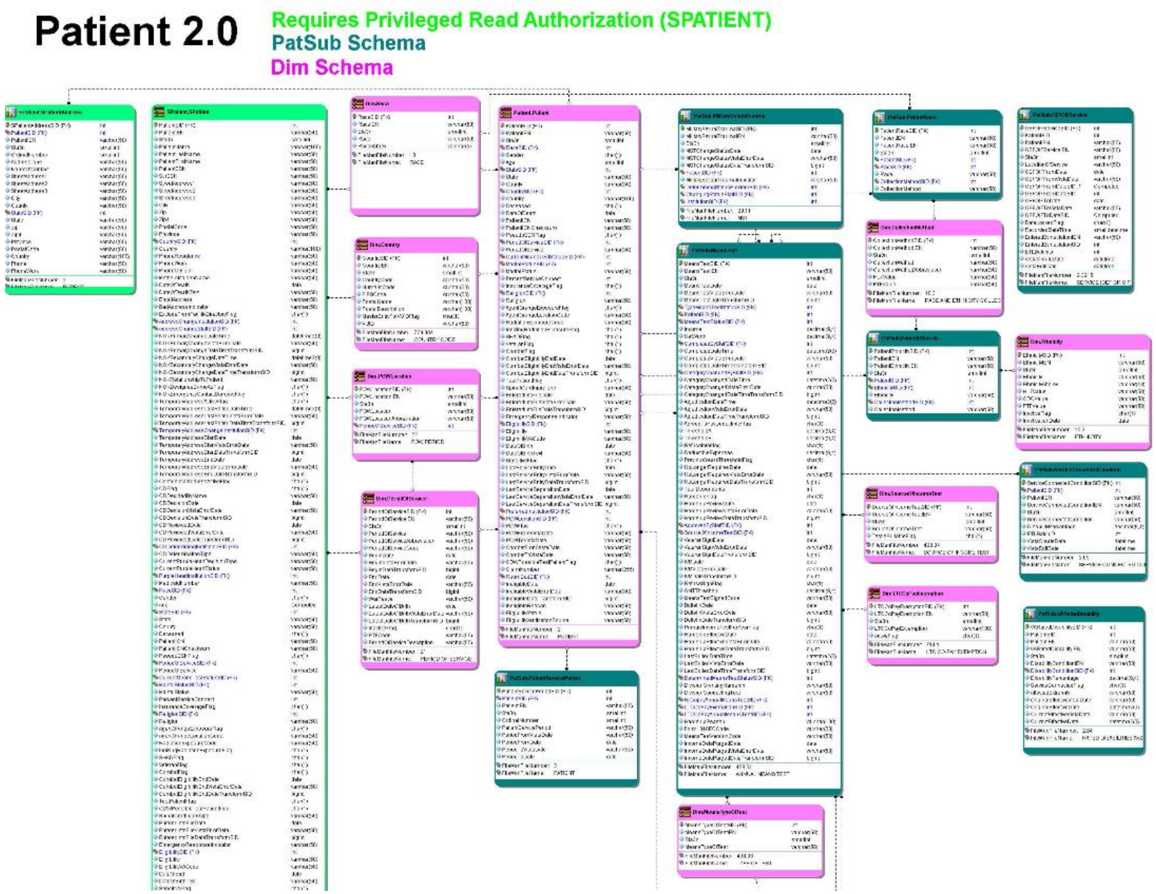


⊟ Patient 2.0

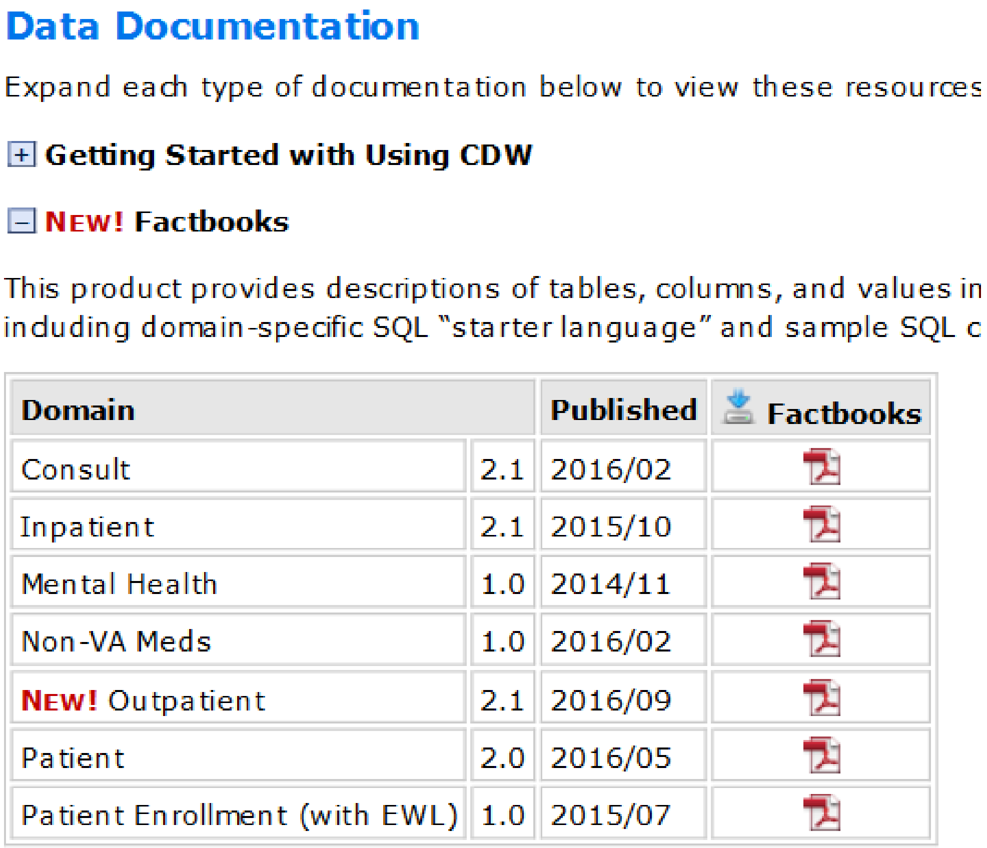
DWViewName	Field Count	FileMan File Data Source	View Version	Relevant Dates	Relati
Dim.CollectionMethod	7	RACE AND ETHNICITY COLLECTION METHOD (10.3)	DWViewDeployed: xDWWork View Version: 7		
Dim.CollectionMethod	7	RACE AND ETHNICITY COLLECTION METHOD (10.3)	DWViewDeployed: xDWWork View Version: 7		
Dim.Country	10	COUNTRY CODE (779.004)	DWViewDeployed: xDWWork View Version: 3		
Dim.Country	10	COUNTRY CODE (779.004)	DWViewDeployed: xDWWork View Version: 3		
Dim.Ethnicity	10	ETHNICITY (10.2)	DWViewDeployed: xDWWork View Version: 8		
Dim.Ethnicity	10	ETHNICITY (10.2)	DWViewDeployed: xDWWork View Version: 8		
Dim.LTCCoPayExemption	5	LTC CO-PAY EXEMPTION (714.1)	DWViewDeployed: xDWWork View Version: 3		
Dim.LTCCoPayExemption	5	LTC CO-PAY EXEMPTION (714.1)	DWViewDeployed: xDWWork View Version: 3		
			DWViewDeployed: xDWWork		

Open the ER Diagram by clicking the domain name or Expand the table list by hitting the plus sign

ER Diagram for Patient Domain



Or, a VIReC *Factbook*



Data Documentation

Expand each type of documentation below to view these resources

⊞ **Getting Started with Using CDW**

⊟ **NEW! Factbooks**

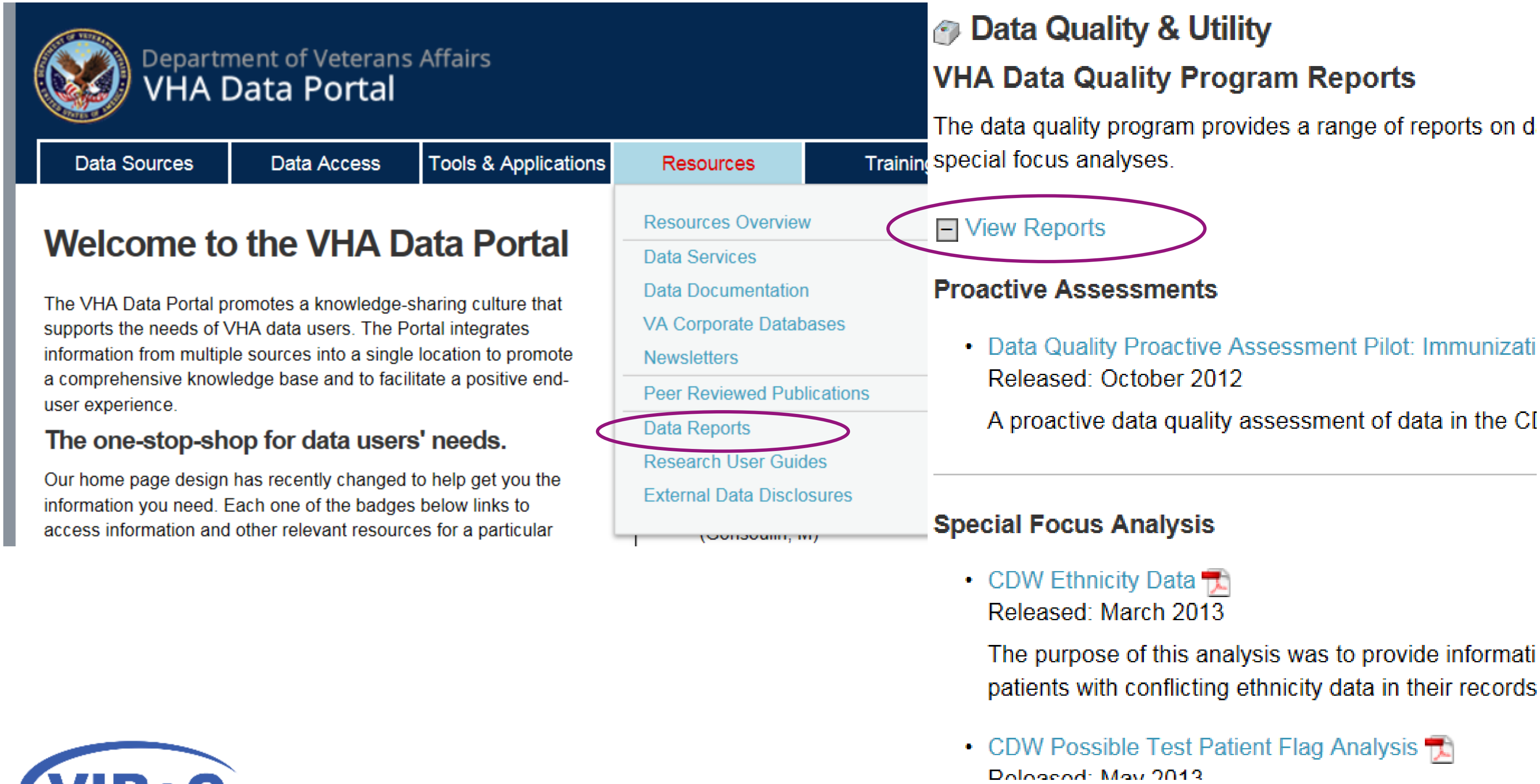
This product provides descriptions of tables, columns, and values in including domain-specific SQL "starter language" and sample SQL c

Domain		Published	Factbooks
Consult	2.1	2016/02	📄
Inpatient	2.1	2015/10	📄
Mental Health	1.0	2014/11	📄
Non-VA Meds	1.0	2016/02	📄
NEW! Outpatient	2.1	2016/09	📄
Patient	2.0	2016/05	📄
Patient Enrollment (with EWL)	1.0	2015/07	📄

VIReC
RESEARCHERS' GUIDE TO VA DATA

http://vaww.virec.research.va.gov/CDW/Documentation.htm

Data Quality Report



Department of Veterans Affairs
VHA Data Portal

Welcome to the VHA Data Portal

The VHA Data Portal promotes a knowledge-sharing culture that supports the needs of VHA data users. The Portal integrates information from multiple sources into a single location to promote a comprehensive knowledge base and to facilitate a positive end-user experience.

The one-stop-shop for data users' needs.

Our home page design has recently changed to help get you the information you need. Each one of the badges below links to access information and other relevant resources for a particular

Data Quality & Utility

VHA Data Quality Program Reports

The data quality program provides a range of reports on d
special focus analyses.

⊟ View Reports

Proactive Assessments

- Data Quality Proactive Assessment Pilot: Immunizati
  Released: October 2012
  A proactive data quality assessment of data in the CI

Special Focus Analysis

- CDW Ethnicity Data
  Released: March 2013
  The purpose of this analysis was to provide informati
  patients with conflicting ethnicity data in their records
- CDW Possible Test Patient Flag Analysis
  Released: May 2013

http://vaww.vhadataportal.med.va.gov/

Practice Problems…

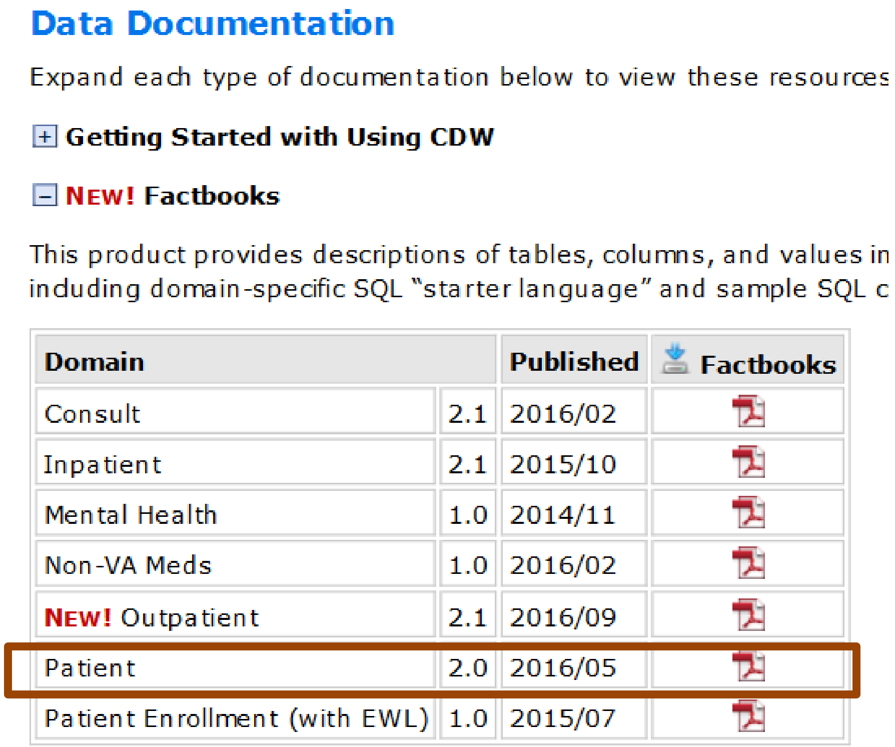
1. **Create a temporary table of patients on which to practice your SQL technique**
2. Recode marital status
3. Find the most recent visit for a cohort of patients

Let’s start with a small temporary table of patient data

- We'll use SQL code:
  - SELECT specific columns
  - TOP 10000 (a small number of rows)
  - INTO #TempTableName
  - And, a WHERE clause to specify only those patient records with characteristics that lead us to believe that this is a legitimate/complete record

VIReC *Patient Domain Factbook*

b) you might use the VIReC Patient Factbook



Data Documentation

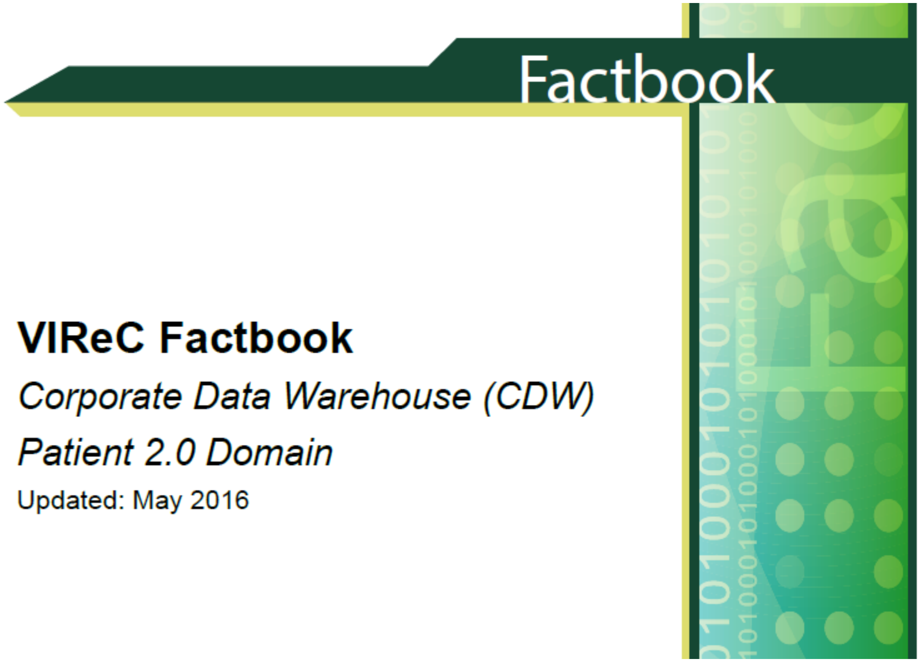
Expand each type of documentation below to view these resources

⊞ **Getting Started with Using CDW**

⊟ **NEW! Factbooks**

This product provides descriptions of tables, columns, and values in
including domain-specific SQL "starter language" and sample SQL c

Domain		**Published**	**Factbooks**
Consult	2.1	2016/02	📄
Inpatient	2.1	2015/10	📄
Mental Health	1.0	2014/11	📄
Non-VA Meds	1.0	2016/02	📄
NEW! Outpatient	2.1	2016/09	📄
Patient	2.0	2016/05	📄
Patient Enrollment (with EWL)	1.0	2015/07	📄



Factbook

VIReC Factbook

Corporate Data Warehouse (CDW)

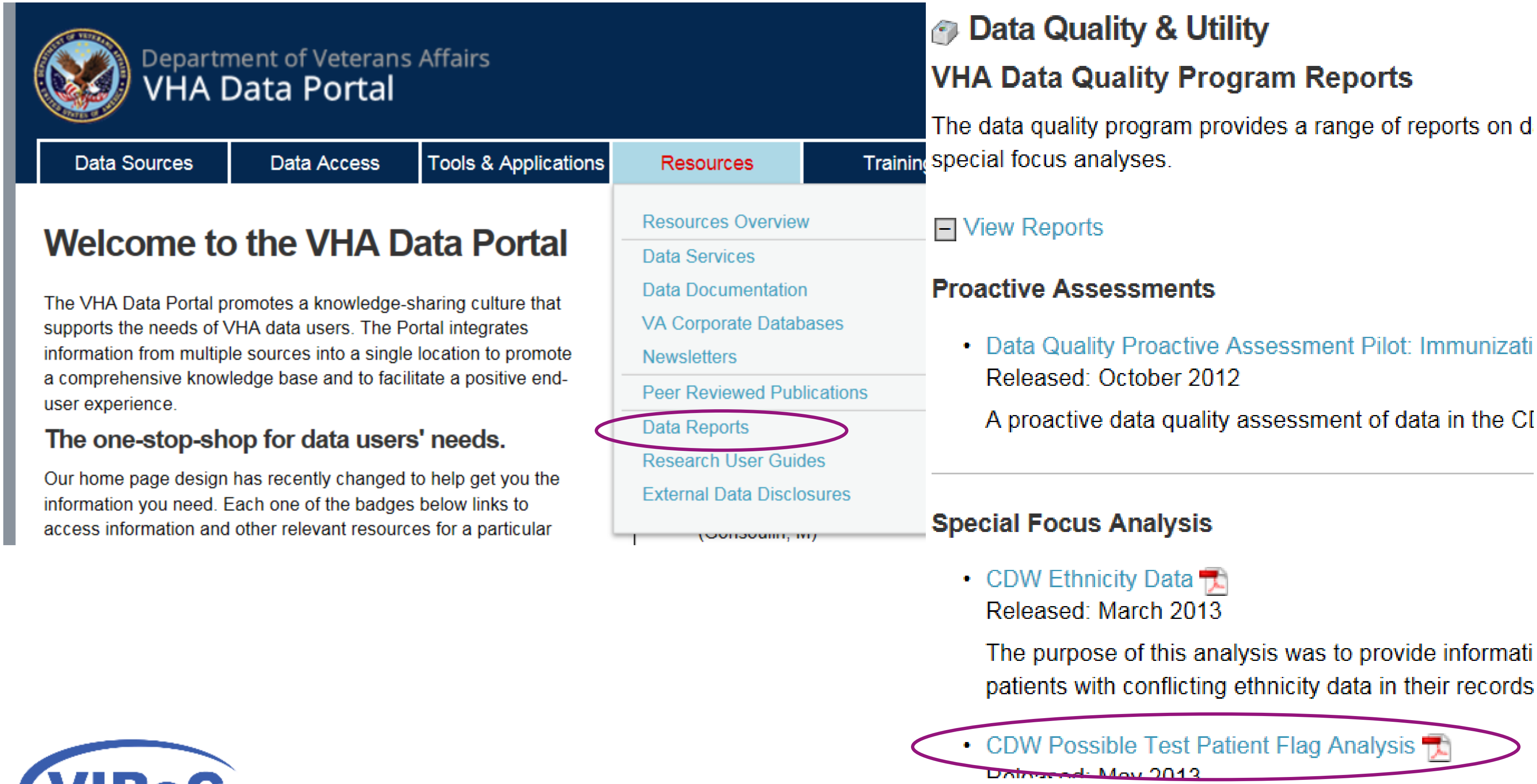
Patient 2.0 Domain

Updated: May 2016

http://vaww.virec.research.va.gov/CDW/Documentation.htm

Data Quality Report on Patient data

c) CDW Possible Test Patients Report might be useful



Welcome to the VHA Data Portal

The VHA Data Portal promotes a knowledge-sharing culture that supports the needs of VHA data users. The Portal integrates information from multiple sources into a single location to promote a comprehensive knowledge base and to facilitate a positive end-user experience.

The one-stop-shop for data users' needs.

Our home page design has recently changed to help get you the information you need. Each one of the badges below links to access information and other relevant resources for a particular

Data Quality & Utility

VHA Data Quality Program Reports

The data quality program provides a range of reports on d special focus analyses.

⊟ View Reports

Proactive Assessments

- Data Quality Proactive Assessment Pilot: Immunizati
  Released: October 2012
  A proactive data quality assessment of data in the CI

Special Focus Analysis

- CDW Ethnicity Data
  Released: March 2013
  The purpose of this analysis was to provide informati patients with conflicting ethnicity data in their records
- CDW Possible Test Patient Flag Analysis
  Released: May 2013

VIReC RESEARCHERS' GUIDE TO VA DATA

http://vaww.vhadataportal.med.va.gov/

A query to create our temp table of patient data

```
SELECT TOP 10000
       PatientSID,
       PatientICN,
       MaritalStatus
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

Let’s Break it down, SELECT

```
SELECT TOP 10000
       PatientSID,
       PatientICN,
       MaritalStatus
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

- The select statement begins every basic query
- TOP 10,000 specifies the number of rows that will be kept out of the Patient.Patient table
- PatientSID, PatientICN & MaritalStatus are the three columns that I chose to include in the output table

Let’s Break it down, INTO

```
SELECT TOP 10000
       PatientSID,
       PatientICN
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

- The INTO clause alerts SSMS that the name of a table will follow, and the query results should be stored there
- The hash or number sign (#) indicates that this table is a temporary table that should be erased when you close the query window
- “PracticeCohort” is the name that I chose for this temporary table

*this statement may be used to create a non-temporary table (e.g., dflt.cohort), but you cannot use to insert data into an existing table.

Let’s Break it down, FROM

```
SELECT TOP 10000
       PatientSID,
       PatientICN,
       MaritalStatus
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

- The FROM clause specifies the table that SQL Server should use when trying to find the selected columns utilized in the query
- CDWWork is the name of the database
- The first “Patient” is the schema for this fact table
- The second “Patient” is the name of this fact table

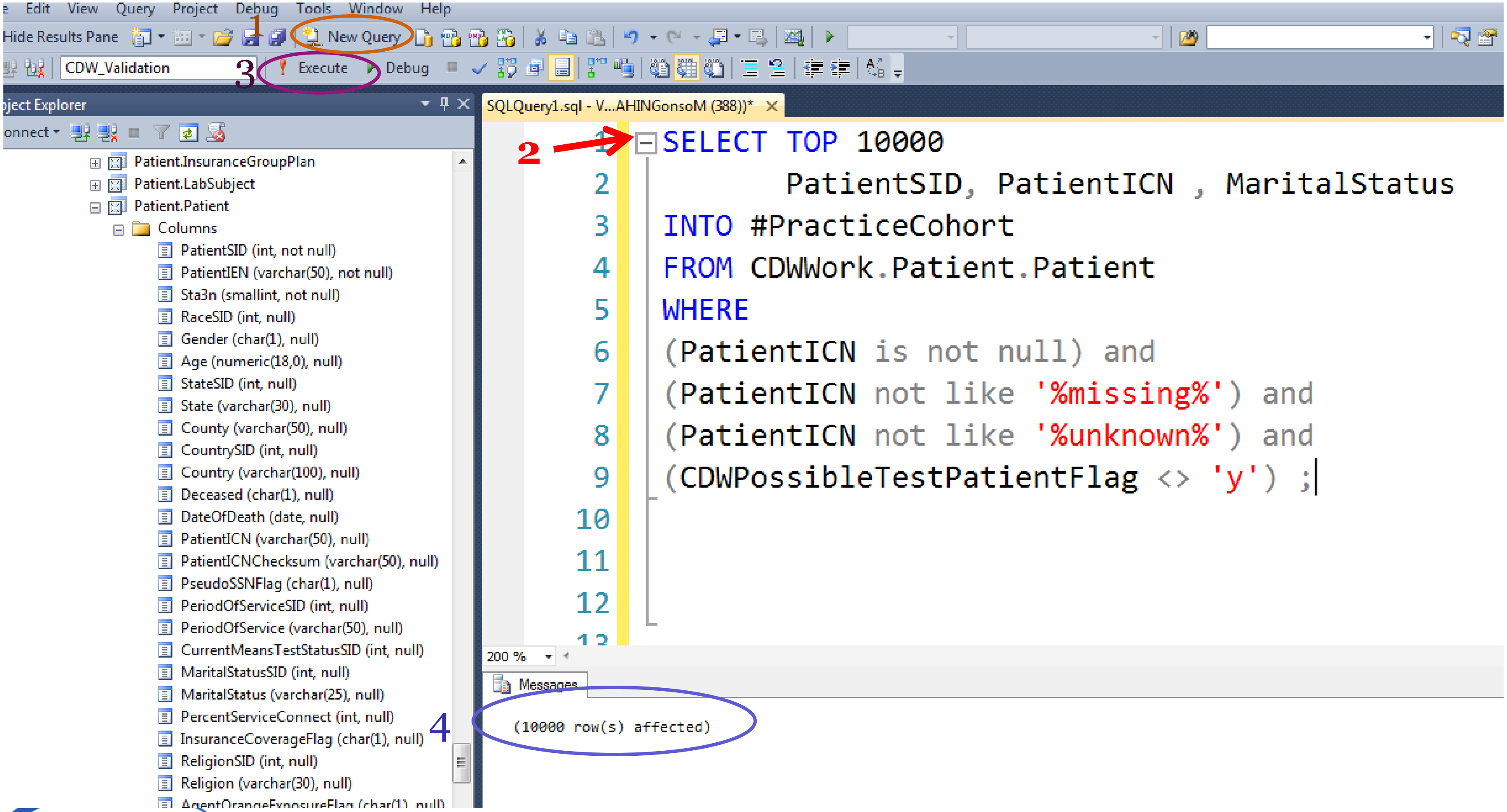
Let’s Break it down, WHERE

```
SELECT TOP 10000
       PatientSID,
       PatientICN,
       MaritalStatus
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

- The WHERE clause is used to specify the conditions under which a row of data will be selected for use in this query
- Using the Data Quality reports on patient identifiers, I choose to exclude any null, missing or unknown PatientICNs and any records marked as possible test patients.
- The % signs are wild cards that allow for any characters to precede or follow these terms.

VIReC
RESEARCHERS' GUIDE TO VA DATA

Run query in SSMS


e Edit View Query Project Debug Tools Window Help
Hide Results Pane
New Query
CDW_Validation
Execute
Debug
1
2
3
4
ject Explorer
onnect
Patient.InsuranceGroupPlan
Patient.LabSubject
Patient.Patient
Columns
PatientSID (int, not null)
PatientIEN (varchar(50), not null)
Sta3n (smallint, not null)
RaceSID (int, null)
Gender (char(1), null)
Age (numeric(18,0), null)
StateSID (int, null)
State (varchar(30), null)
County (varchar(50), null)
CountrySID (int, null)
Country (varchar(100), null)
Deceased (char(1), null)
DateOfDeath (date, null)
PatientICN (varchar(50), null)
PatientICNChecksum (varchar(50), null)
PseudoSSNFlag (char(1), null)
PeriodOfServiceSID (int, null)
PeriodOfService (varchar(50), null)
CurrentMeansTestStatusSID (int, null)
MaritalStatusSID (int, null)
MaritalStatus (varchar(25), null)
PercentServiceConnect (int, null)
InsuranceCoverageFlag (char(1), null)
ReligionSID (int, null)
Religion (varchar(30), null)
AgentOrangeExposureFlag (char(1), null)
SQLQuery1.sql - V...AHINGonsoM (388))*
200 %
Messages

```sql
SELECT TOP 10000
       PatientSID, PatientICN , MaritalStatus
INTO #PracticeCohort
FROM CDWWork.Patient.Patient
WHERE
(PatientICN is not null) and
(PatientICN not like '%missing%') and
(PatientICN not like '%unknown%') and
(CDWPossibleTestPatientFlag <> 'y') ;
```

```
(10000 row(s) affected)
```

Poll #2: About You

Which of the following best describes your role in the VA? (Check all that apply)

❑ Research Investigator / PI

❑Career Development Awardee

❑Data Manager / Analyst

❑Project Coordinator

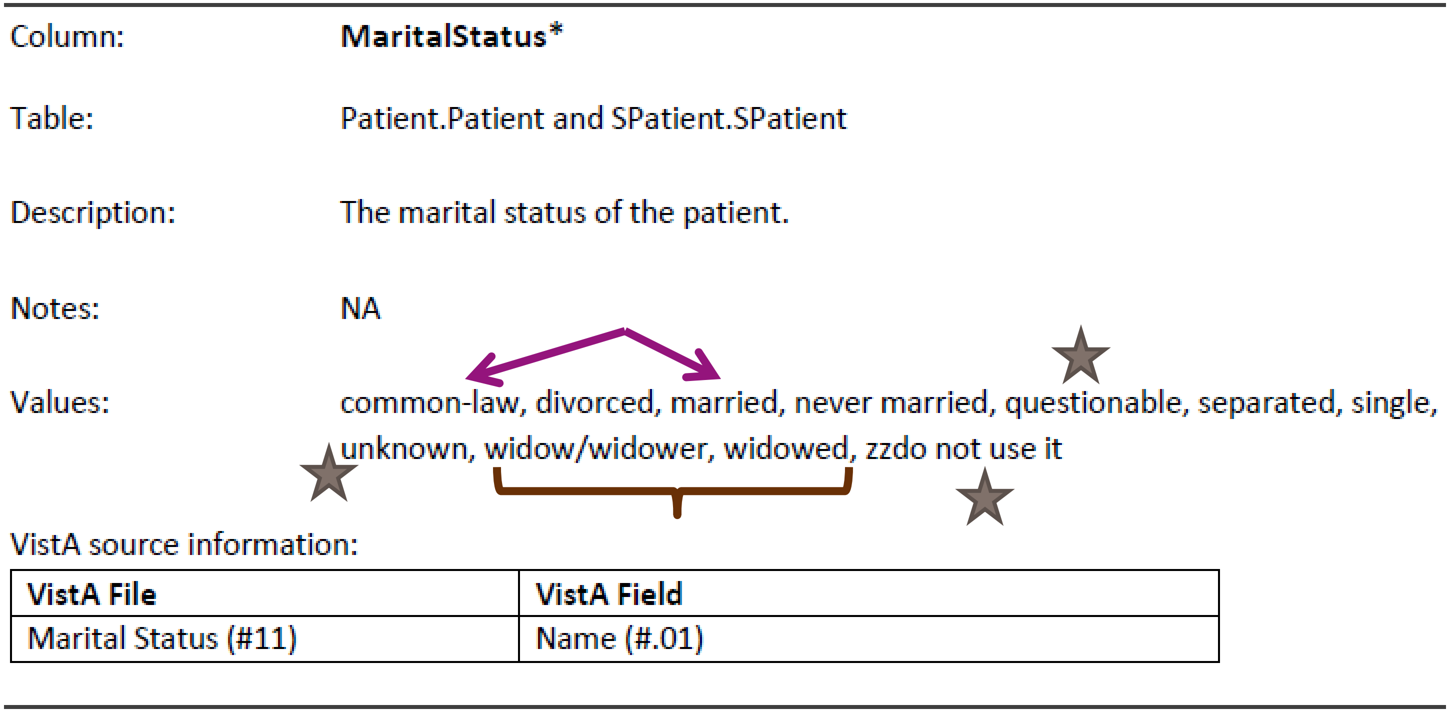
❑Operations / Partnered Research / QI

❑Other

Practice Problems…

1. Create a temporary table of patients on which to practice your SQL technique
2. **Recode marital status**
3. Find the most recent visit for a cohort of patients

Factbook Entry for Marital Status



Column: **MaritalStatus***

Table: Patient.Patient and SPatient.SPatient

Description: The marital status of the patient.

Notes: NA

Values: common-law, divorced, married, never married, questionable, separated, single, unknown, widow/widower, widowed, zzdo not use it

VistA source information:

VistA File	VistA Field
Marital Status (#11)	Name (#.01)

Recoding in SQL

```
 SELECT PatientSID , PatientICN , MaritalStatus ,
       CASE
       WHEN MaritalStatus like 'Married' THEN 'Married'
       WHEN MaritalStatus like 'Common-Law' THEN 'Married'
       WHEN MaritalStatus like 'Separated' THEN 'Separated'
       WHEN MaritalStatus like 'Divorced' THEN 'Divorced'
       WHEN MaritalStatus like 'Widow/Widower' THEN 'Widowed'
       WHEN MaritalStatus like 'Widowed' THEN 'Widowed'
       WHEN MaritalStatus like 'Single%' THEN 'Single'
       WHEN MaritalStatus like 'Never Married' THEN 'Never Married'
       ELSE 'Missing'
       END AS MaritalStatusRecode
INTO #MaritalRecode
FROM #PracticeCohort;
```

Let’s break it down, SELECT

```
SELECT PatientSID , PatientICN ,
MaritalStatus ,
       CASE
       ……………
       ELSE 'Missing'
       END AS MaritalStatusRecode
INTO #MaritalRecode
FROM #PracticeCohort;
```

- The SELECT clause reads four columns into the result set:
  - PatientSID
  - PatientICN
  - MaritalStatus
  - And, it creates a new recoded column called MaritalStatusRecode using the CASE expression…

Let’s break it down, CASE

```
CASE
WHEN MaritalStatus like 'Married' THEN 'Married'
WHEN MaritalStatus like 'Common-Law' THEN 'Married'
WHEN MaritalStatus like 'Separated' THEN 'Separated'
WHEN MaritalStatus like 'Divorced' THEN 'Divorced'
WHEN MaritalStatus like 'Widow/Widower' THEN 'Widowed'
WHEN MaritalStatus like 'Widowed' THEN 'Widowed'
WHEN MaritalStatus like 'Single%' THEN 'Single'
WHEN MaritalStatus like 'Never Married' THEN 'Never
Married'
ELSE 'Missing'
END AS MaritalStatusRecode
```

- The CASE expression is used in the SELECT clause*.
- It recodes the values of the column MaritalStatus to collapse values of “common-law” into married and the values related to widowed.
- Non-standard values such as “zz do not use it” are collapsed into ‘missing’ using ELSE
- The recoded column is given the name MaritalStatusRecode using END AS

*the CASE expression may be used in SELECT, WHERE, HAVING or ORDER BY clauses

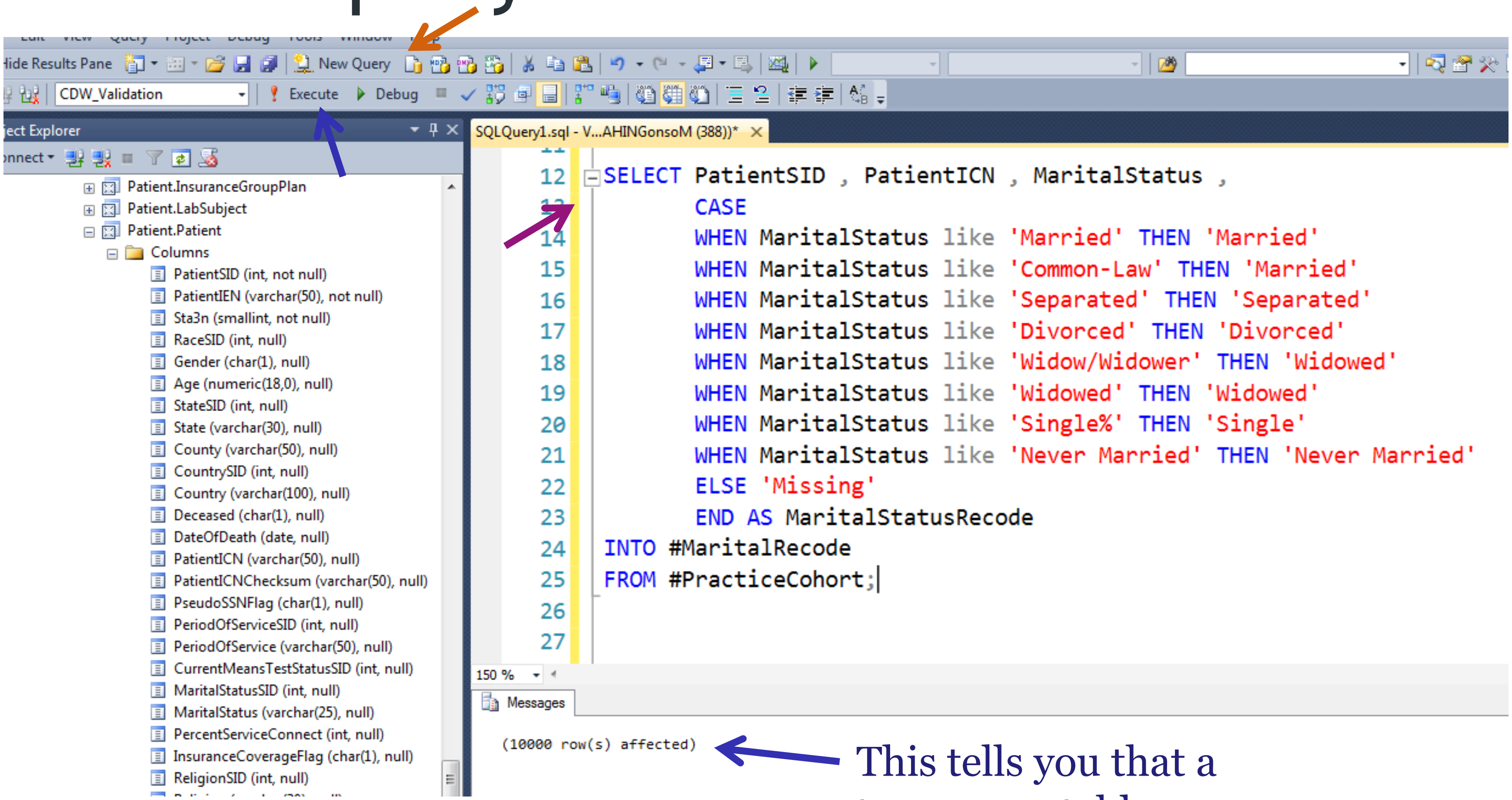
Let’s break it down, INTO & FROM

```
SELECT PatientSID , PatientICN ,
MaritalStatus ,
       CASE
       …
       ELSE 'Missing'
       END AS MaritalStatusRecode
INTO #MaritalRecode
FROM #PracticeCohort;
```

- The SELECT INTO statement* indicates that the result set should be stored in a temporary table called #MaritalRecode
- The FROM clause selects our temporary practice cohort as the source of the columns we are working with in this query.

*this statement may be used to create a non-temporary table (e.g., dflt.cohort), but you cannot use to insert data into an existing table.

Run the query in SSMS


Hide Results Pane
New Query
CDW_Validation
Execute
Debug
Object Explorer
Connect
SQLQuery1.sql - V...AHINGonsoM (388))*
Patient.InsuranceGroupPlan
Patient.LabSubject
Patient.Patient
Columns
PatientSID (int, not null)
PatientIEN (varchar(50), not null)
Sta3n (smallint, not null)
RaceSID (int, null)
Gender (char(1), null)
Age (numeric(18,0), null)
StateSID (int, null)
State (varchar(30), null)
County (varchar(50), null)
CountrySID (int, null)
Country (varchar(100), null)
Deceased (char(1), null)
DateOfDeath (date, null)
PatientICN (varchar(50), null)
PatientICNChecksum (varchar(50), null)
PseudoSSNFlag (char(1), null)
PeriodOfServiceSID (int, null)
PeriodOfService (varchar(50), null)
CurrentMeansTestStatusSID (int, null)
MaritalStatusSID (int, null)
MaritalStatus (varchar(25), null)
PercentServiceConnect (int, null)
InsuranceCoverageFlag (char(1), null)
ReligionSID (int, null)
150 %
Messages

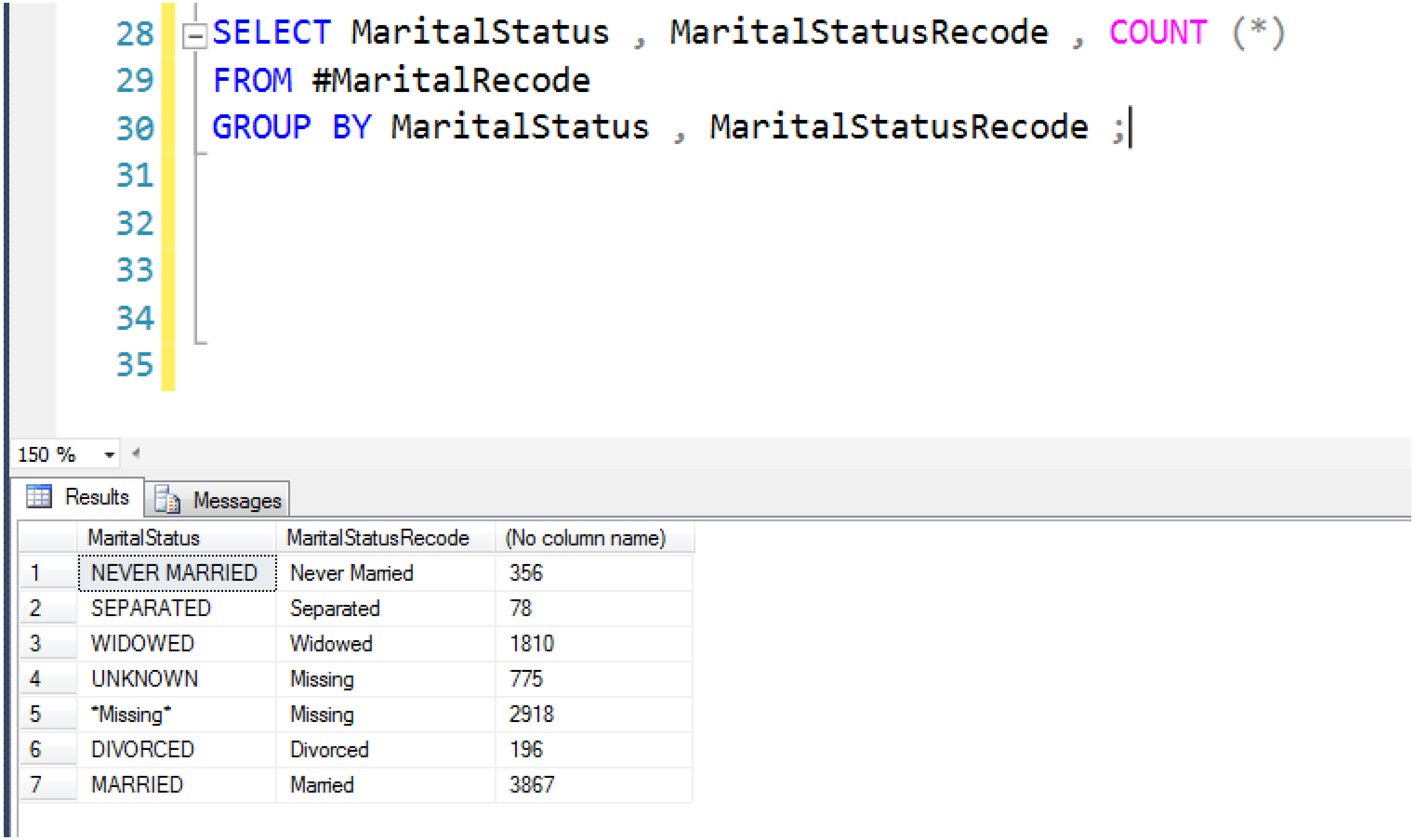
```sql
SELECT PatientSID , PatientICN , MaritalStatus ,
        CASE
        WHEN MaritalStatus like 'Married' THEN 'Married'
        WHEN MaritalStatus like 'Common-Law' THEN 'Married'
        WHEN MaritalStatus like 'Separated' THEN 'Separated'
        WHEN MaritalStatus like 'Divorced' THEN 'Divorced'
        WHEN MaritalStatus like 'Widow/Widower' THEN 'Widowed'
        WHEN MaritalStatus like 'Widowed' THEN 'Widowed'
        WHEN MaritalStatus like 'Single%' THEN 'Single'
        WHEN MaritalStatus like 'Never Married' THEN 'Never Married'
        ELSE 'Missing'
        END AS MaritalStatusRecode
INTO #MaritalRecode
FROM #PracticeCohort;
```

```
(10000 row(s) affected)
```

This tells you that a temporary table was created, and it has 10,000 rows in it

VIReC
RESEARCHERS' GUIDE TO VA DATA

A quick check of the recode



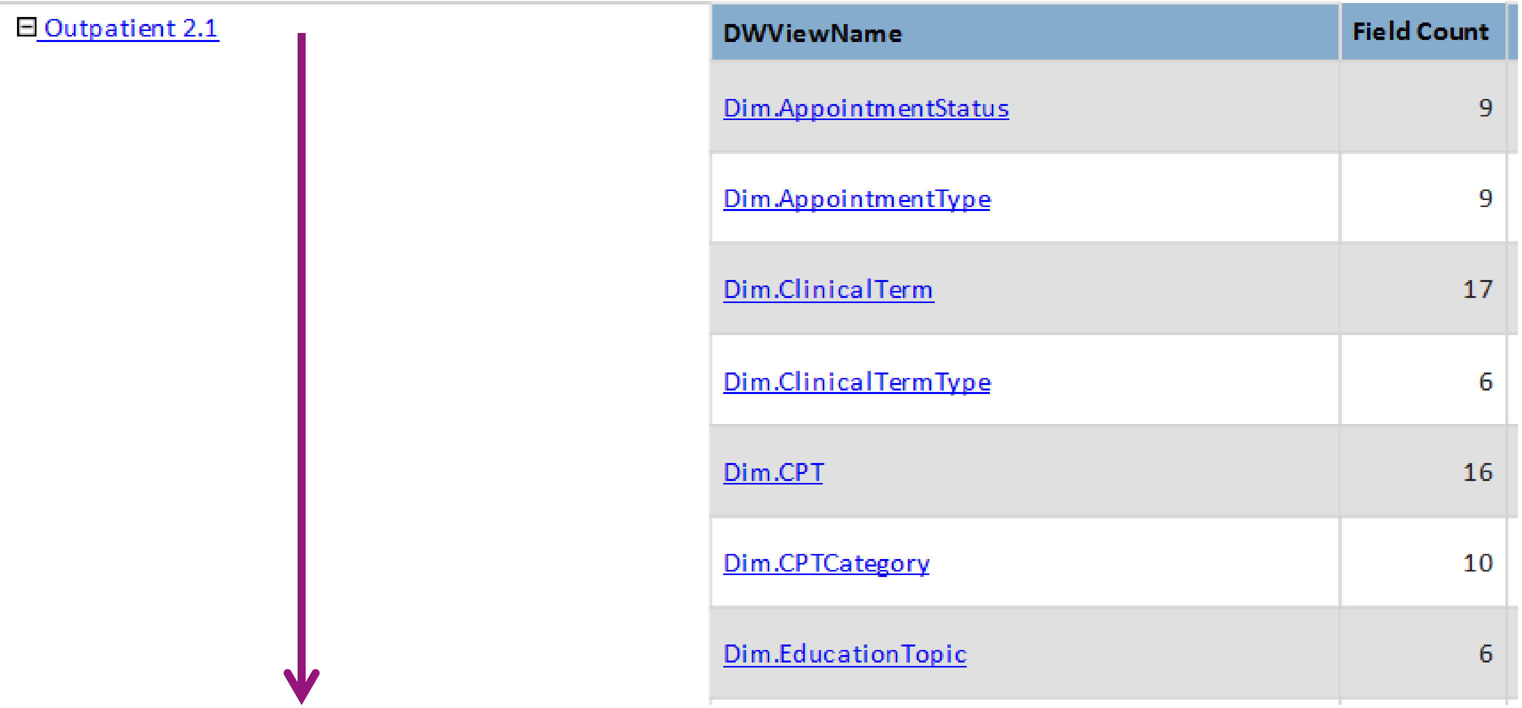
```sql
SELECT MaritalStatus , MaritalStatusRecode , COUNT (*)
FROM #MaritalRecode
GROUP BY MaritalStatus , MaritalStatusRecode ;
```

	MaritalStatus	MaritalStatusRecode	(No column name)
1	NEVER MARRIED	Never Married	356
2	SEPARATED	Separated	78
3	WIDOWED	Widowed	1810
4	UNKNOWN	Missing	775
5	*Missing*	Missing	2918
6	DIVORCED	Divorced	196
7	MARRIED	Married	3867

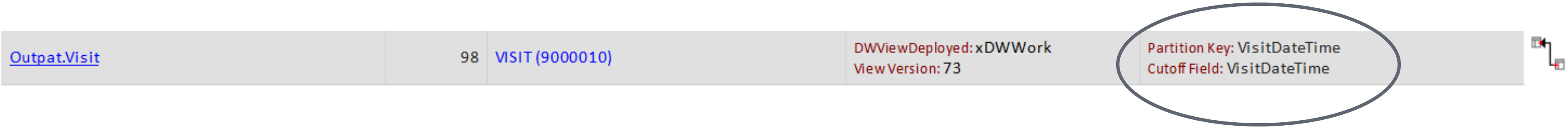
Practice Problems…

1. Create a temporary table of patients on which to practice your SQL technique
2. Recode marital status
3. **Find the most recent visit for a cohort of patients**

Outpatient Documentation

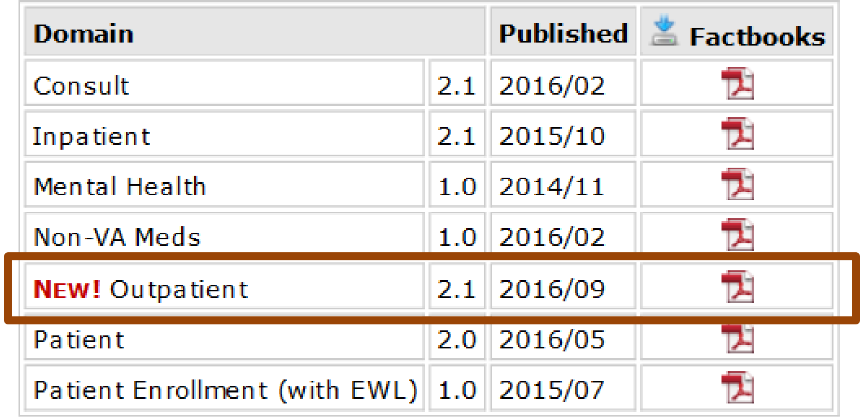


DWViewName	Field Count
Dim.AppointmentStatus	9
Dim.AppointmentType	9
Dim.ClinicalTerm	17
Dim.ClinicalTermType	6
Dim.CPT	16
Dim.CPTCategory	10
Dim.EducationTopic	6

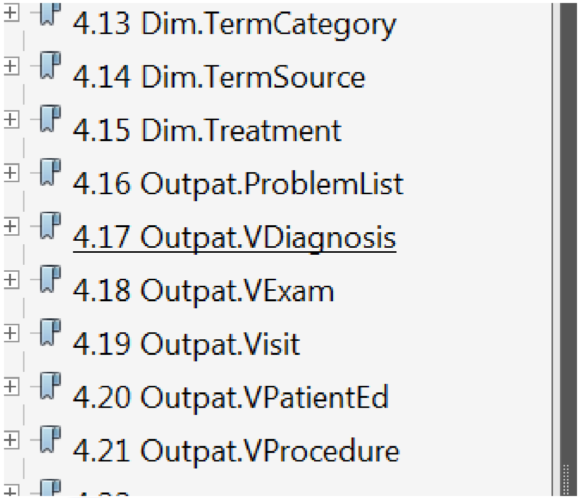


Outpat.Visit	98	VISIT (9000010)	DWViewDeployed: xDWWork View Version: 73	Partition Key: VisitDateTime Cutoff Field: VisitDateTime

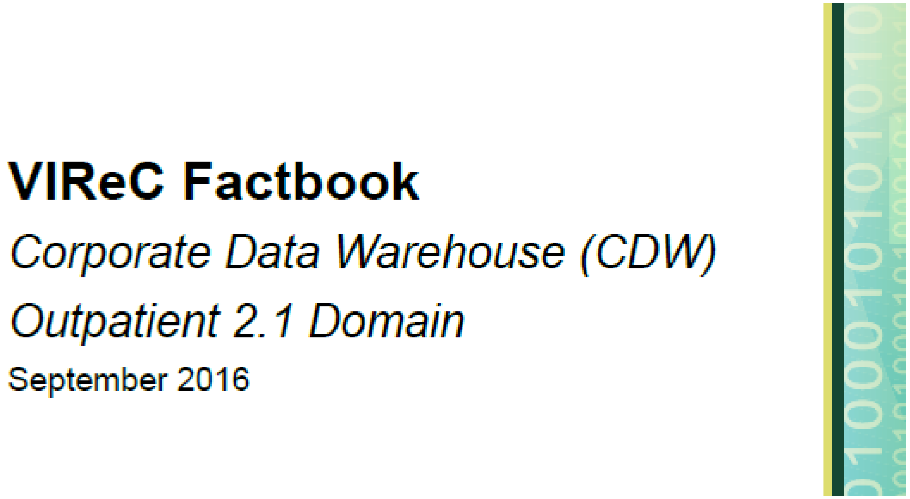
And, a VIReC *Outpatient Domain Factbook*



Domain		Published	Factbooks
Consult	2.1	2016/02	📄
Inpatient	2.1	2015/10	📄
Mental Health	1.0	2014/11	📄
Non-VA Meds	1.0	2016/02	📄
NEW! Outpatient	2.1	2016/09	📄
Patient	2.0	2016/05	📄
Patient Enrollment (with EWL)	1.0	2015/07	📄



- 4.13 Dim.TermCategory
- 4.14 Dim.TermSource
- 4.15 Dim.Treatment
- 4.16 Outpat.ProblemList
- 4.17 Outpat.VDiagnosis
- 4.18 Outpat.VExam
- 4.19 Outpat.Visit
- 4.20 Outpat.VPatientEd
- 4.21 Outpat.VProcedure



VIReC Factbook

Corporate Data Warehouse (CDW)

Outpatient 2.1 Domain

September 2016

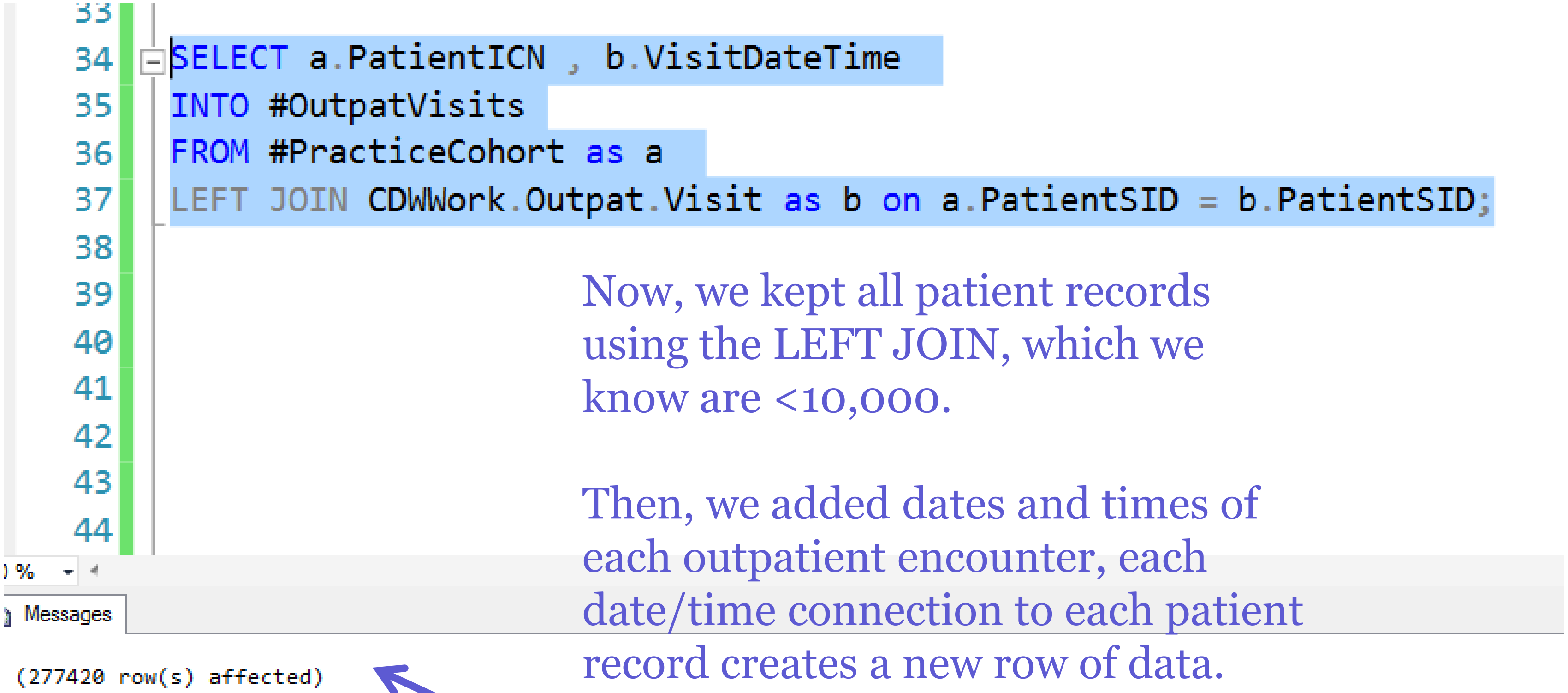
http://vaww.virec.research.va.gov/CDW/Documentation.htm

Connecting Outpatient to Patient

- In order to find most recent outpatient activity for each record in my practice cohort…
- I will connect to Outpat.Visit and retrieve the column that indicates the date and time of that episode of care, VisitDateTime

```
SELECT a.PatientICN ,
       b.VisitDateTime
INTO #OutpatVisits
FROM #PracticeCohort as a
LEFT JOIN CDWWork.Outpat.Visit as b ON a.PatientSID = b.PatientSID ;
```

One patient record → Many outpatient records



```
SELECT a.PatientICN , b.VisitDateTime
INTO #OutpatVisits
FROM #PracticeCohort as a
LEFT JOIN CDWWork.Outpat.Visit as b on a.PatientSID = b.PatientSID;
```

```
(277420 row(s) affected)
```

Now, we kept all patient records using the LEFT JOIN, which we know are <10,000.

Then, we added dates and times of each outpatient encounter, each date/time connection to each patient record creates a new row of data.

All of this is stored in the temporary table called #OutpatVisits

To find the most recent

- We need to create a subset (partition) of all visits for each patient
- We need for that list to be ordered by date and time of visit
- We need to keep track of that order so we can systematically find the most recent activity

```
SELECT PatientICN , VisitDateTime,
    ROW_NUMBER () OVER
    (PARTITION BY PatientICN ORDER BY VisitDateTime DESC)
    AS RowNumber
INTO #OrderedVisits
FROM #OutpatVisits;
```

Note: the over clause is a part of windows function that has many variation; see text referenced at the end of the cyberseminar for more information about these options.

Let’s break it down, OVER clause

```
SELECT PatientICN , VisitDateTime,
    ROW_NUMBER () OVER
    (PARTITION BY PatientICN ORDER BY VisitDateTime DESC)
    AS RowNumber
INTO #OrderedVisits
FROM #OutpatVisits;
```

- PARTITION BY clause tells SSMS to break #OutpatVisits into separate segments for each PatientICN
- ORDER BY clause tells SSMS to organize each row of data in each PatientICN segment by the date/time of the visit *
- DESC tells SSMS to put the most recent date on the first line *
- ROW_NUMBER creates a row number for each row of data in each segment *
- AS is used to store the row number in a new column called RowNumber

* These represent one set of options available for use in the OVER clause.

Example of a partition from #OrderedVisits

PatientICN	VisitDateTime	RowNumber
1	1/16/2009 0:01	1
1	10/15/2009 0:01	2
1	9/15/2009 0:01	3
1	8/15/2009 0:01	4
1	7/15/2009 0:01	5
1	1/11/2009 0:01	6
1	1/11/2009 0:02	7
1	12/4/2007 0:01	8

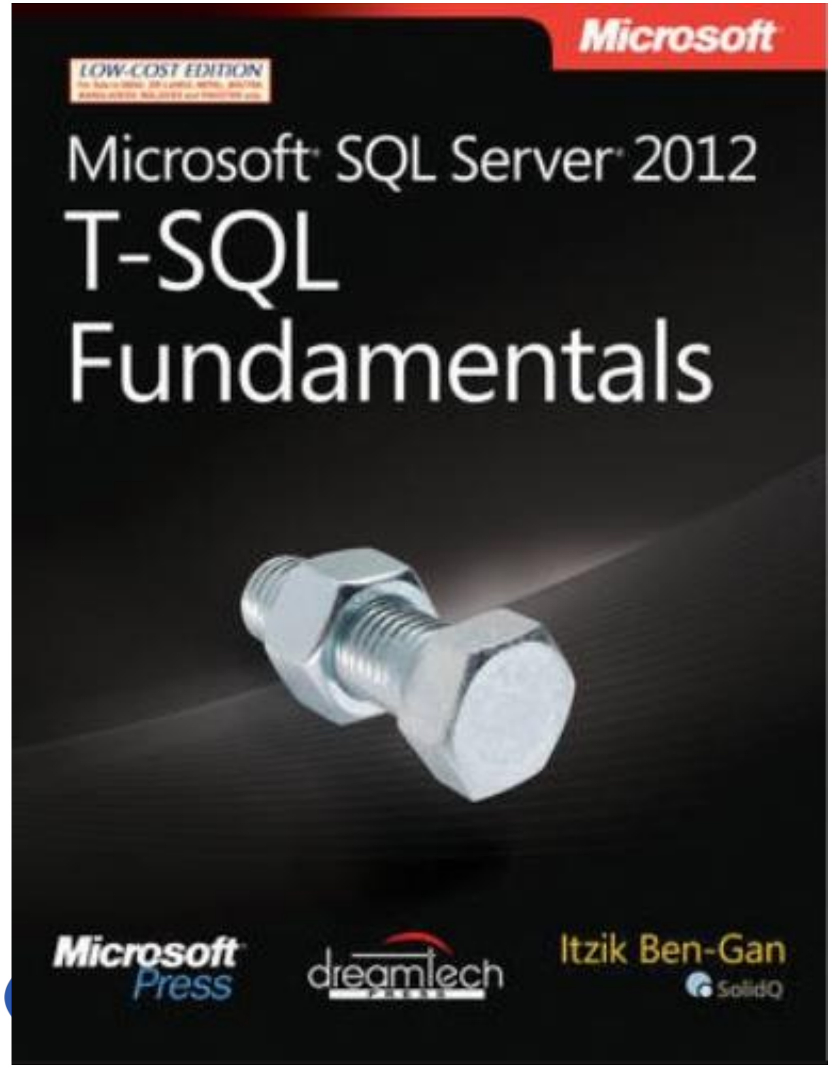
Now you can write a query to select the most recent visit per patient

```
SELECT PatientICN ,
       VisitDateTime
FROM #OrderedVisits
WHERE RowNumber = 1;
```

Summary/Conclusions

- So, now you can
  - use temporary tables, partition dates, converted functions.. to optimize your code
  - recode a column in SQL to meet your research needs
  - use “partitions” to select desired records when multiple records are present

For additional SQL tips…



Contact Information

Margaret Gonsoulin, PhD

VIReC@va.gov

708-202-2413

Questions?