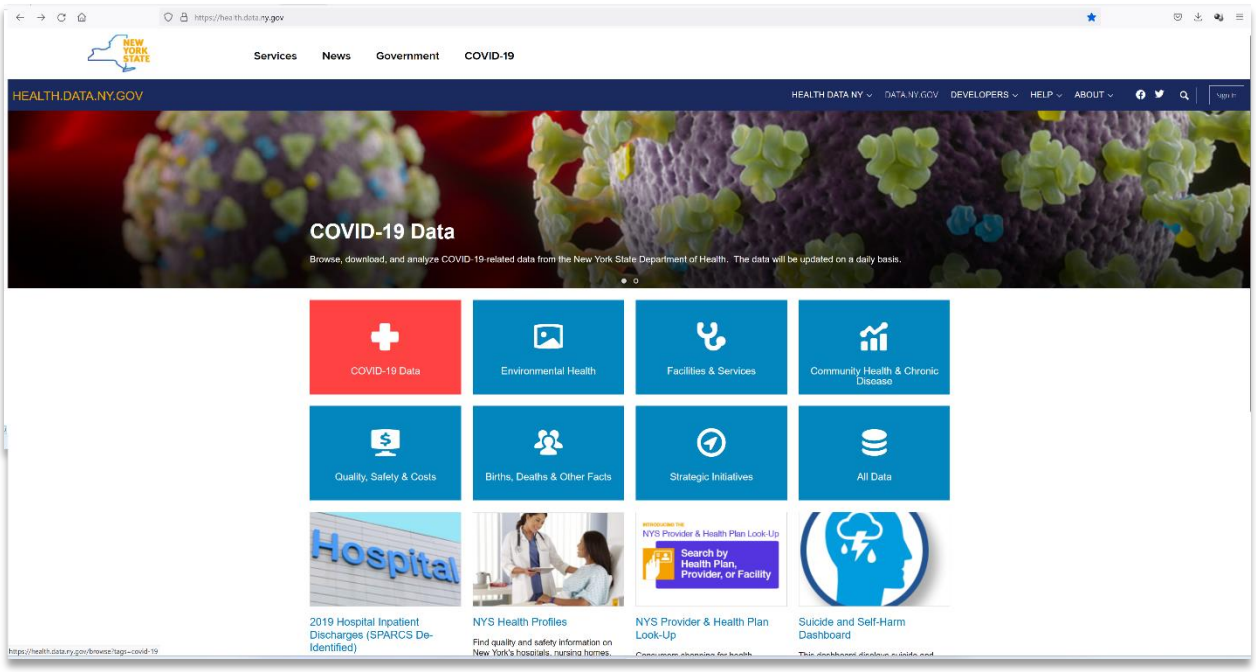
STEP 1: PREPERATION

1.1 COLLECTING THE REQUIRED DATA

The data set chosen for the project is a real data set from [The 2019 Statewide Planning and Research Cooperative System (SPARCS) Inpatient De-identified File](#).



It's a one stop shop for all we need. SPARCS is a comprehensive all payer data reporting system established in cooperation between the healthcare industry and New York state government that collects patient level demographic details, diagnoses, treatments and charges from each hospital, totaling 2,339,462 discharges. It is the next best thing to direct access to primary source electronic medical records, and offers the following advantages:

- No time wasted requisitioning data, because its already publicly accessible
- Eliminates legal risk, and time wasted redacting info, because its already de-identified
- Information is reliable due to being directly from the primary source facilities
- Large sample size from which to infer statistically meaningful insights

1.2 LIMITATIONS OF DATA SET

Due to de-identification on certain measures like age and dates of service, some granularity is lost for use cases such as mapping events to specific age groups, and time periods. Furthermore, the race category uses an aggregated "Other Race" element that hinders studies into racial sub groups such as Asian and Native American populations. Lack of nuanced gender and sexual orientation identifiers are also limiting factors preventing analyses taking into account these specific populations. Note: Actual state employees have access to the identified data set which has no such limitations.

1.3 IMPORTING DATA

Our Objectives:

a. **Create a new table**
b. **Create a View to receive our data**
c. **Create Bulk Insert SQL script to quickly import data**
d. **Check if the data were successfully inserted**

1.3.a Creating a new table

Using a SQL script, we're creating a table called SPARC2019.

When creating the new table, we first have to decide which column data types will meet our need for a single table database schema.

A wise sage said one time that *"life is like a box of chocolates; you never know what you're gonna get."* Data can sometimes be that way as well. Fortunately, the New York Department of Health provides a Data Dictionary to guide us through setting up our table in SQL Server!

On import, we want pristine data to avoid loading errors and truncation so we're sticking as close as reasonably possible with the recommended data types.

See dictionary files below:

Field Name	Definition
Health Service Area	Type is Char. Length is 15. A description of the Health Service Area (HSA) which the hospital is located. Blank for enhanced de-identification record Capital/Adirondack, Central NY, Finger Lakes, Hudson Valley, Long Island, York City, Southern Tier, Western NY.
Hospital County	Type is Char. Length is 11. A description of the county in which the hospit located. Blank for records with enhanced de-identification.
Operating Certificate Number	Type is Char. Length is 12. The facility Operating Certificate Number asassigned by NYS Department of Health. Blank for records with enhanced de-identification.
Permanent Facility Id	Type is Num. Length is 6. Permanent Facility Identifier. Blank for records v enhanced de-identification.
Facility Name	Type is Char. Length is 112. The name of the facility where services were performed based on the Permanent Facility Identifier (PFI), as maintained the NYSDOH Division of Health Facility Planning. For records with enhanc de-identification, 'Redacted for Confidentiality' appears.
Age Group	Type is Char. Length is 11. Age in years at time of discharge. Grouped into the following age groups: 0 to 17, 18 to 29, 30 to 49, 50 to and 70 or Older.
Zip Code - 3 digits	Type is Char. Length is 3. The first three digits of the patient's zip code. Blank for: - population size less than 20,000 - enhanced de-identification records, or - cell size less than 10 on population classification strata. "OOS" are Out of State zip codes.
Gender	Type is Char. Length is 1. Patient gender: (M) Male, (F) Female, (U) Unkno
Race	Type is Char. Length is 32. Patient race. Black/African American, Multi-racial, Other Race, White. Other Race includes Native Americans and Asian/Pacific Islander.
Ethnicity	Type is Char. Length is 20. Patient ethnicity. The ethnicity of the patient: Spanish/Hispanic Origin, Not of Spanish/Hisp Origin, Multi-ethnic Unknown.
Length of Stay	Type is Char. Length is 5. The total number of patient days at an acute lev and/or other than acute care level (excluding leave of absence days) (Discharge Date - Admission Date) + 1. Length of Stay greater than or equal to 120 days has been aggregated to days.
Type of Admission	Type is Char. Length is 15. A description of the manner in which the patie was admitted to the health care facility: Elective, Emergency, Newborn, Not Available, Trauma, Urgent.
Patient Disposition	Type is Char. Length is 37. The patient's destination or status upon discha
Discharge Year	Type is Char. Length is 4. The year (CCYY) of discharge.
CCSR Diagnosis Code	Type is Char. Length is 10. AHRQ Clinical Classification Software Refined (CCSR) Diagnosis Category Code. More information on the CCSR system may be found at the direct link: https://www.hcup-us.ahrq.gov/toolssoftware/ccsr/ccs_refined.jsp
CCSR Diagnosis Description	Type is Char. Length is 250. AHRQ Clinical Classification Software Refined (CCSR) Diagnosis Category Description. More information on the CCSR system may be found at the direct link: https://www.hcup-us.ahrq.gov/toolssoftware/ccsr/ccs_refined.jsp

Field Name	Definition
CCSR Procedure Code	Type is Char. Length is 10. AHRQ Clinical Classification Software Refined (CCSR) ICD-10 Procedure Category Code. More information on the CCSR system may be found at the direct link: https://www.hcup-us.ahrq.gov/toolssoftware/ccsr/ccs_refined.jsp
CCSR Procedure Description	Type is Char. Length is 250. AHRQ Clinical Classification Software Refined (CCSR) ICD-10 Procedure Category Description. More information on the system may be found at the direct link: https://www.hcup-us.ahrq.gov/toolssoftware/ccsr/ccs_refined.jsp
APR DRG Code	Type is Char. Length is 3. The All Patients Refined Diagnosis Related Grou (APR-DRG) Classification Code
APR DRG Description	Type is Char. Length is 500. The APR-DRG Classification Code Description in Calendar Year 2019, Version 36 of the APR- DRG Grouper. http://www.health.ny.gov/statistics/sparcs/sysdoc/appy.htm
APR MDC Code	Type is Char. Length is 2. All Patient Refined Major Diagnostic Category (MDC) Code. APR-DRG Codes 001-006 and 950-956 may group to more th one MDC Code. All other APR DRGs group to one MDC category.
APR MDC Description	Type is Char. Length is 500. All Patient Refined Major Diagnostic Categor (APR MDC) Description.
APR Severity of Illness Code	Type is Char. Length is 1. The APR-DRG Severity of Illness Code: 0, 1, 2, 3,
APR Severity of Illness Description	Type is Char. Length is 8. All Patient Refined Severity of Illness (APR SOI) Description: Undetermined (0), Minor (1), Moderate (2), Major (3), Extreme (4).
APR Risk of Mortality	Type is Char. Length is 8. All Patient Refined Risk of Mortality (APR ROM) Description: Undetermined (0), Minor (1), Moderate (2), Major (3), Extreme (4).
APR Medical Surgical Description	Type is Char. Length is 14. The APR-DRG specific classification of Medical Surgical or Not Applicable.
Payment Typology 1	Type is Char. Length is 25. A description of the type of payment for this occurrence.
Payment Typology 2	Type is Char. Length is 25. A description of the type of payment for this occurrence.
Payment Typology 3	Type is Char. Length is 25. A description of the type of payment for this occurrence.
Birth Weight	Type is Char. Length is 5. The neonate birth weight in grams; rounded to nearest 100 g.
Emergency Department Indicator	Type is Char. Length is 1. The Emergency Department Indicator is set bas on the submitted revenue codes. If the record contained an Emergency Department revenue code of 045X, the indicator is set to "Y", otherwise be "N".
Total Charges	Type is Char. Length is 8. Total charges for the discharge.
Total Costs	Type is Char. Length is 8. Total estimated cost for the discharge.

The data dictionary recommends Char and Numeric data types, but because we're using SQL server, we will change all data types to Varchar except to assign Int to column "Permanent Facility Id" and Decimal type to the "Total Charges" and "Total Costs" columns since we may have a later use for these in numerical calculations.

One thing we notice right away in the dictionary is that there is no unique Identifier column.

Houston, we have a Normalization problem!

To get our future table in compliance with the first rule of normalization, we need to create a primary key that makes each record unique.

We'll add an identity column (ID) with data type INT, and a primary key constraint (PK_ID) which will make each row have unique values and put them in sequential order.

Now it's time to run our SQL script and create the table!

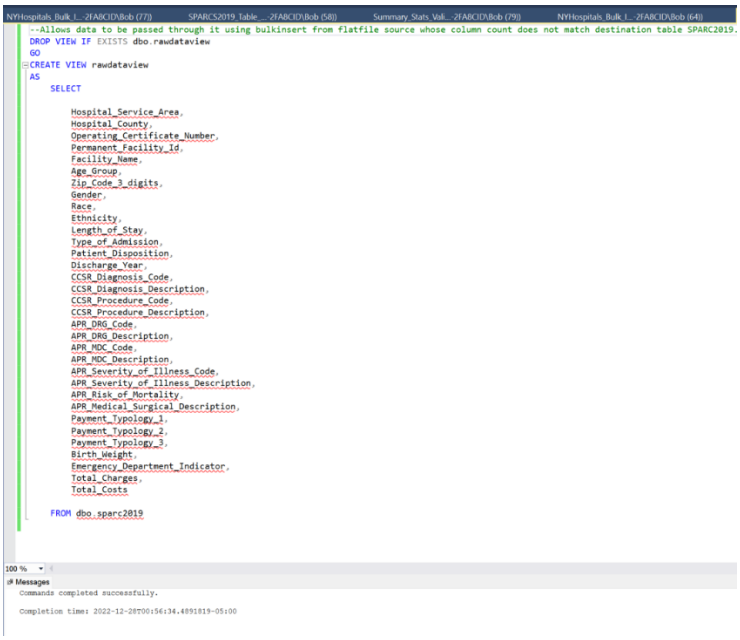
Here's the final script:

Success! Now it's time to create a view through which we'll insert our data into the table.

1.3.b Creating a View

We're using SQL to create a view called "rawdataview."

Here's our script.

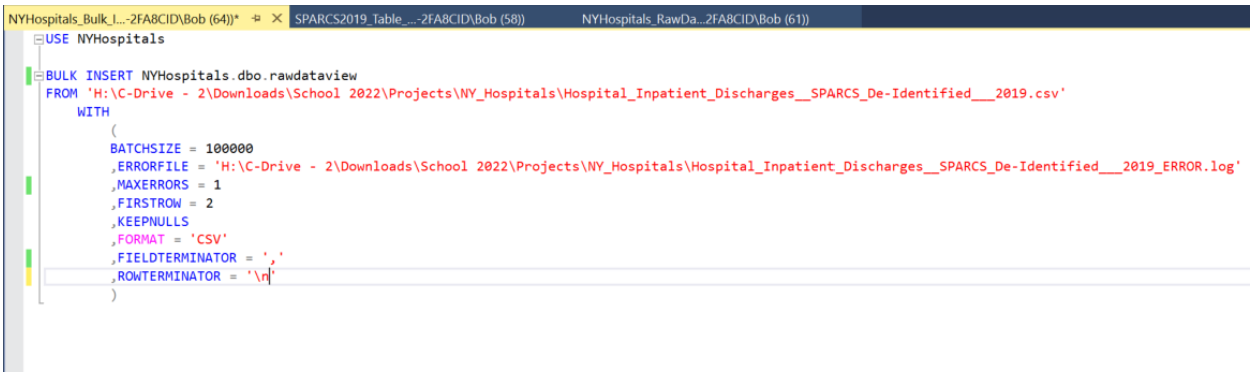


Creating our view is very important. Because we added an additional column to our table that the data set doesn't have, the view allows us an easy way to pass the fields in the flat file into the table without running into loading errors due to column count mismatch.

Now it's time to create the bulk insert statement to do the data import heavy lifting.

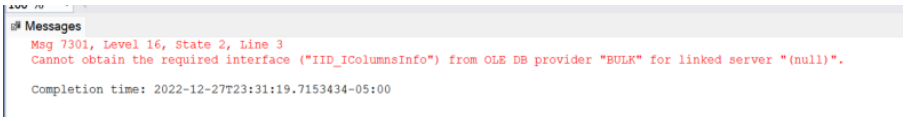
1.3.c Creating a Bulk Insert Statement

The bulk insert command is extremely convenient to load large amounts of data quickly, and with a measure of control over the inputs.

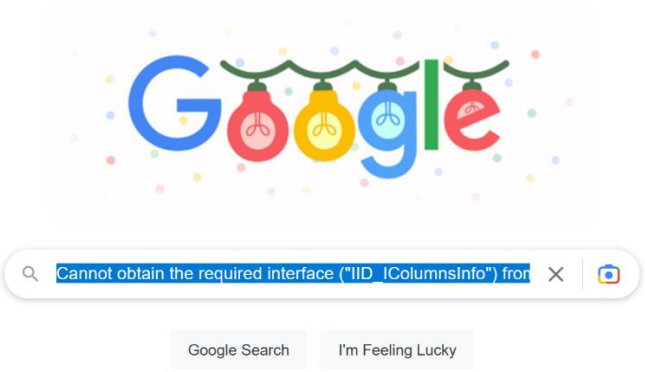


Let's run it.

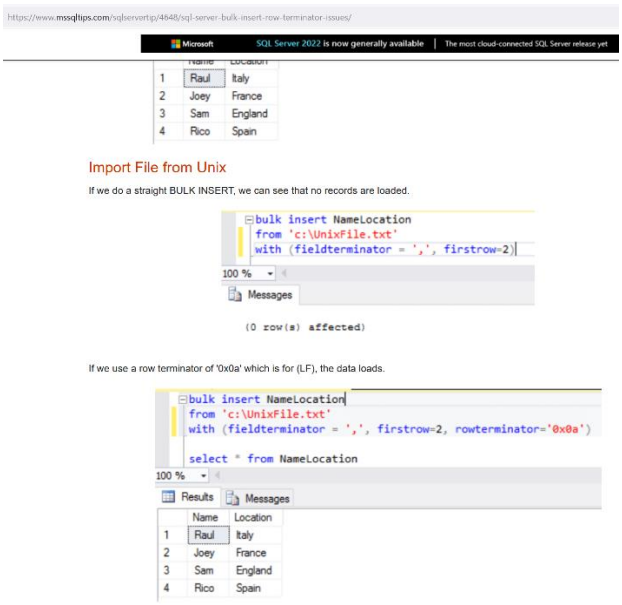
UGH OH!



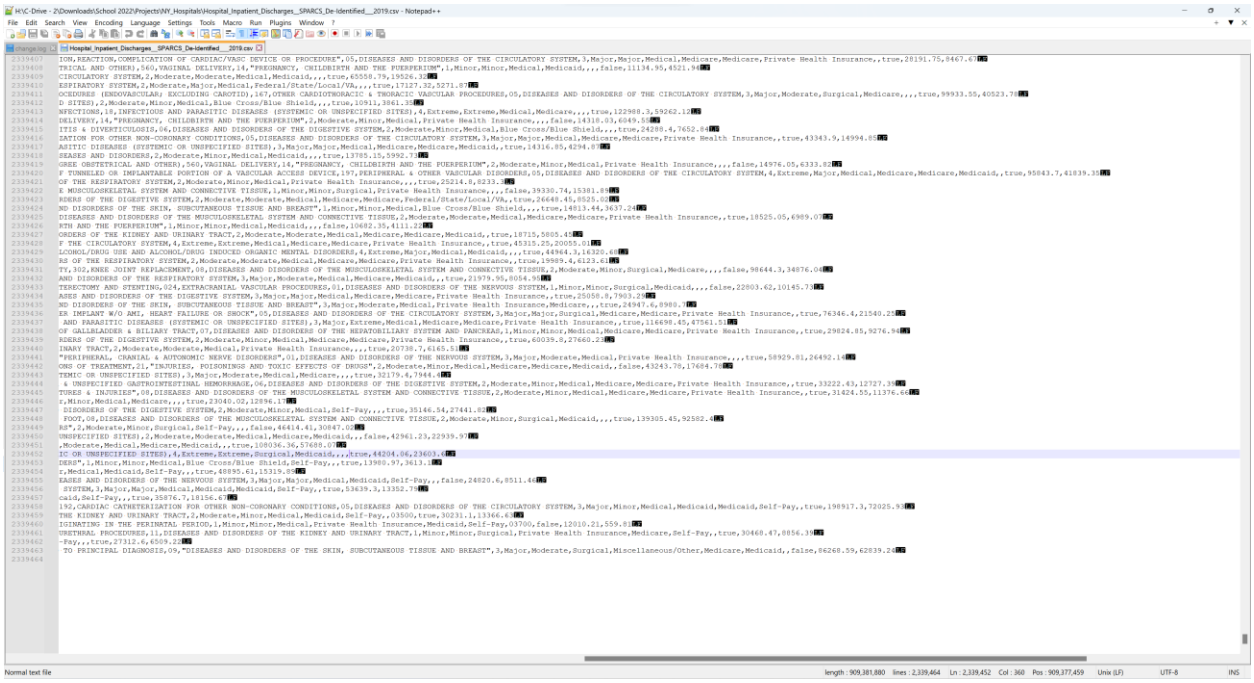
What's this error all about? Let's use Google to find out!



Here's one of our results, taken from the very helpful site at [MSSQLTIPS]



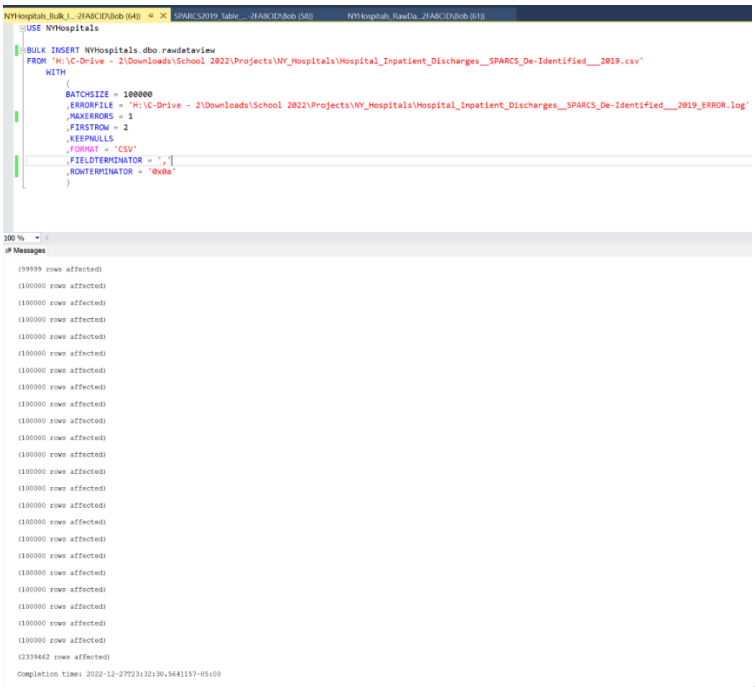
Ahh! We need to check which row terminator our dataset file is using. Let's do that now in Notepad++



`false,86268.59,62839.24LF`

We found a culprit!

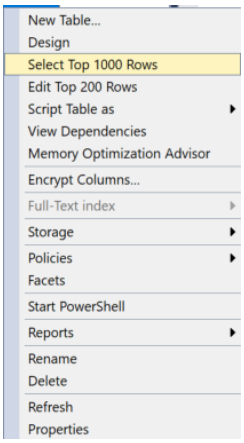
It seems the file was created in a Linux/Unix environment with Line Feed terminators. Let's update and rerun the script.

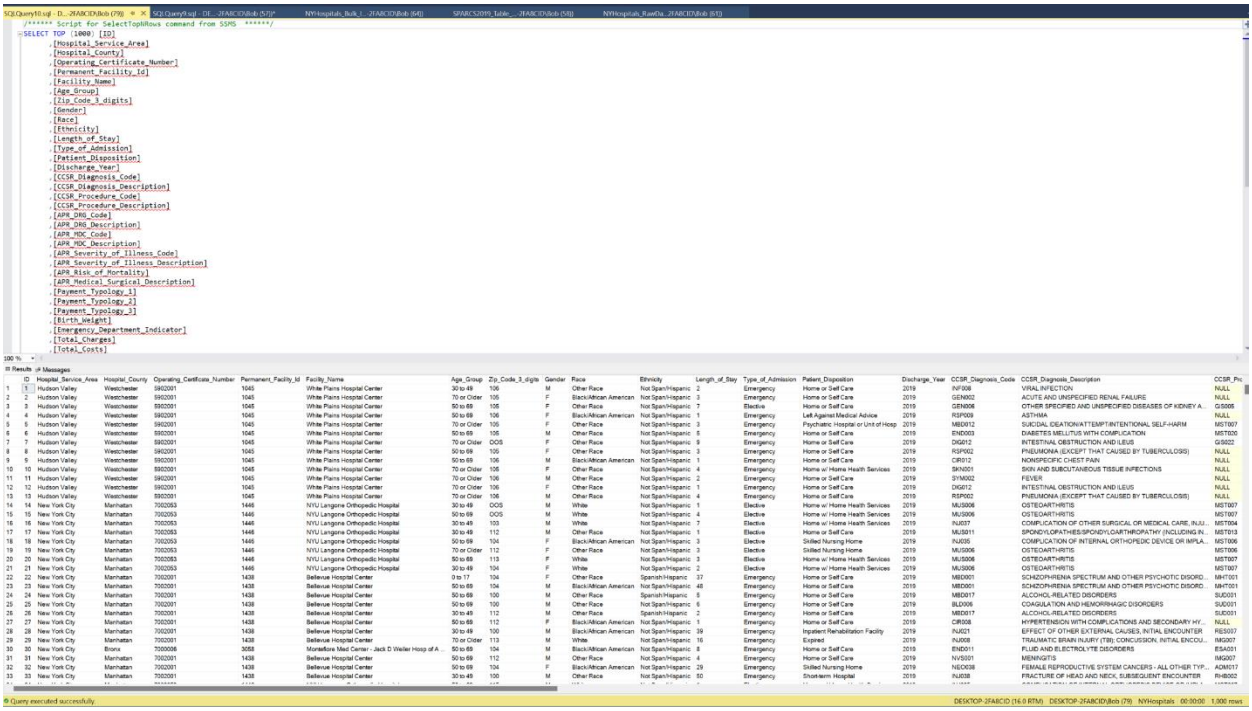


This time it worked!

1.3.d Checking our data import

Let's check if our dataset was imported into the SPARC2019 table by right clicking on the SPARC2019 table and choosing the "Select Top 1000 Rows" option.





The data are imported, very good! We also see the ID identity column correctly numbering our records.

It's time to start the data quality review and cleaning process.

1.4 ASSESSING AND CLEANING THE DATA

Solving the business task at hand requires having quality data from which to infer correct insights, and the steps taken now will help ensure that we can rely upon the data we're going to use for our analysis. We will begin exploring the data set to understand what we are working with, make changes as needed, and note where to begin forming our research questions.

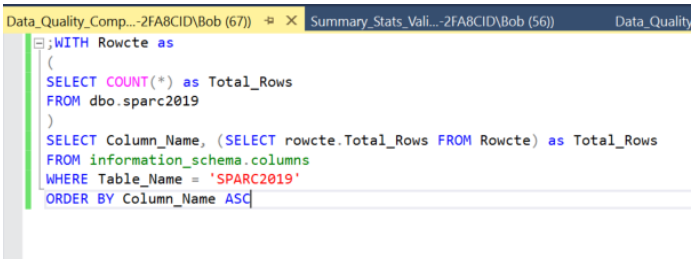
Our Objectives:

a. **Ensure data are complete**
b. **Ensure data are unique**
c. **Ensure data are consistent**
d. **Ensure data are accurate**

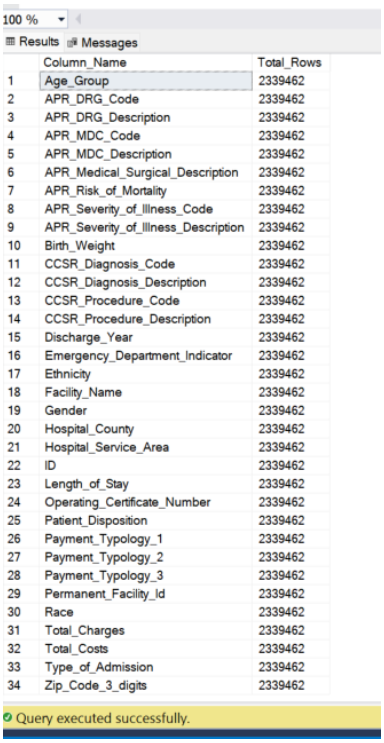
1.4.a Ensuring Data are Complete

1). Checking for missing data

The first thing we will do following uploading this data set is to check if all the data has transferred in or if there are missing elements that need to be noted, and taken into account. We can easily compare our imported records and column total with that of the original source by executing a count query.



The query calls the SQL Server information_schema and gives us the total number of columns, including our custom identity column, and rows.



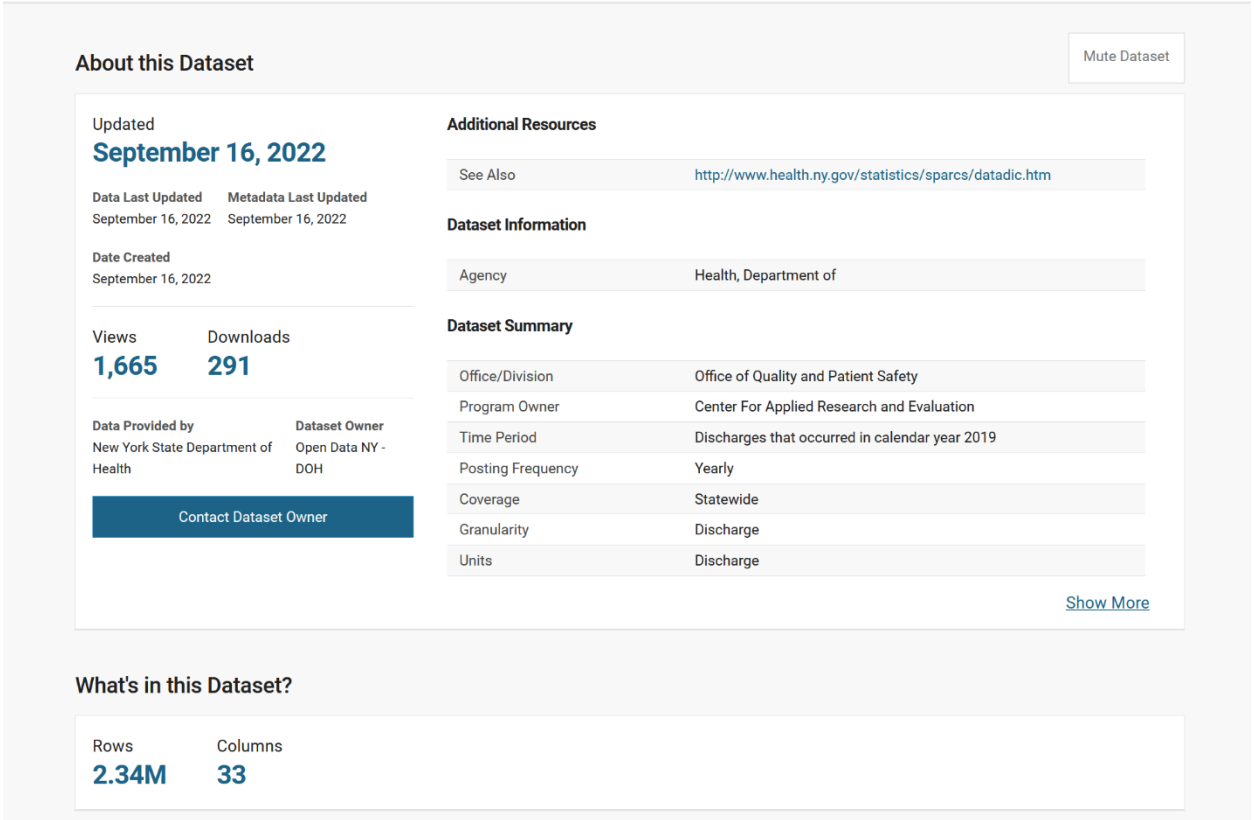
100 %

Results | Messages

	Column_Name	Total_Rows
1	Age_Group	2339462
2	APR_DRG_Code	2339462
3	APR_DRG_Description	2339462
4	APR_MDC_Code	2339462
5	APR_MDC_Description	2339462
6	APR_Medical_Surgical_Description	2339462
7	APR_Risk_of_Mortality	2339462
8	APR_Severity_of_Illness_Code	2339462
9	APR_Severity_of_Illness_Description	2339462
10	Birth_Weight	2339462
11	CCSR_Diagnosis_Code	2339462
12	CCSR_Diagnosis_Description	2339462
13	CCSR_Procedure_Code	2339462
14	CCSR_Procedure_Description	2339462
15	Discharge_Year	2339462
16	Emergency_Department_Indicator	2339462
17	Ethnicity	2339462
18	Facility_Name	2339462
19	Gender	2339462
20	Hospital_County	2339462
21	Hospital_Service_Area	2339462
22	ID	2339462
23	Length_of_Stay	2339462
24	Operating_Certificate_Number	2339462
25	Patient_Disposition	2339462
26	Payment_Typology_1	2339462
27	Payment_Typology_2	2339462
28	Payment_Typology_3	2339462
29	Permanent_Facility_Id	2339462
30	Race	2339462
31	Total_Charges	2339462
32	Total_Costs	2339462
33	Type_of_Admission	2339462
34	Zip_Code_3_digits	2339462

Query executed successfully.

We can compare our 33 columns (+1 custom column) and 2,339,462 records to the source. It's a solid match.



About this Dataset

Mute Dataset

Updated
September 16, 2022

Data Last Updated
September 16, 2022

Metadata Last Updated
September 16, 2022

Date Created
September 16, 2022

Views
1,665

Downloads
291

Data Provided by
New York State Department of Health

Dataset Owner
Open Data NY - DOH

Contact Dataset Owner

Additional Resources

See Also	http://www.health.ny.gov/statistics/sparcs/datadic.htm

Dataset Information

Agency	Health, Department of

Dataset Summary

Office/Division	Office of Quality and Patient Safety
Program Owner	Center For Applied Research and Evaluation
Time Period	Discharges that occurred in calendar year 2019
Posting Frequency	Yearly
Coverage	Statewide
Granularity	Discharge
Units	Discharge

Show More

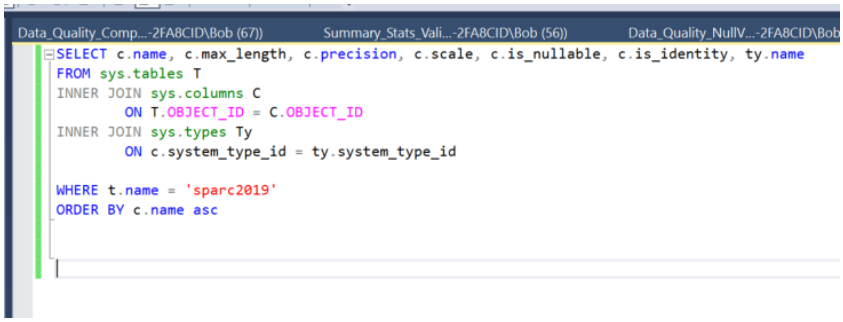
What's in this Dataset?

Rows
2.34M

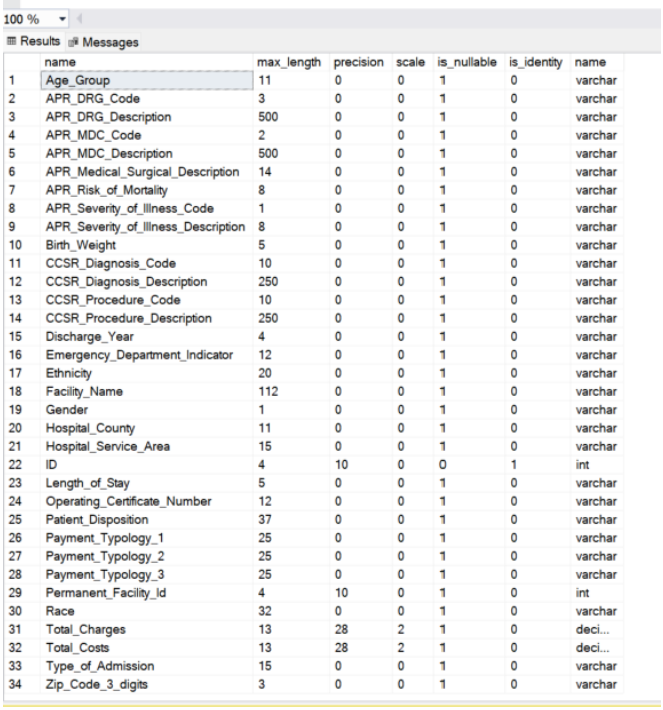
Columns
33

2). Checking Data Structure

Now that we're confident nothing was lost during the import, it's time to check our table's data structures to make sure everything is assigned appropriately. To do this we will run a SQL script that calls and joins the sys.tables, sys.types and sys.columns tables.



The query shows our table structure with the values we assigned at creation, including the max size for the columns fields, the data type, including precision and scale for the numeric data types. We also see our ID column is marked as an Identity column, along with the nullable statuses. It's a very handy script!


100 %

Results Messages

	name	max_length	precision	scale	is_nullable	is_identity	name
1	Age_Group	11	0	0	1	0	varchar
2	APR_DRG_Code	3	0	0	1	0	varchar
3	APR_DRG_Description	500	0	0	1	0	varchar
4	APR_MDC_Code	2	0	0	1	0	varchar
5	APR_MDC_Description	500	0	0	1	0	varchar
6	APR_Medical_Surgical_Description	14	0	0	1	0	varchar
7	APR_Risk_of_Mortality	8	0	0	1	0	varchar
8	APR_Severity_of_Illness_Code	1	0	0	1	0	varchar
9	APR_Severity_of_Illness_Description	8	0	0	1	0	varchar
10	Birth_Weight	5	0	0	1	0	varchar
11	CCSR_Diagnosis_Code	10	0	0	1	0	varchar
12	CCSR_Diagnosis_Description	250	0	0	1	0	varchar
13	CCSR_Procedure_Code	10	0	0	1	0	varchar
14	CCSR_Procedure_Description	250	0	0	1	0	varchar
15	Discharge_Year	4	0	0	1	0	varchar
16	Emergency_Department_Indicator	12	0	0	1	0	varchar
17	Ethnicity	20	0	0	1	0	varchar
18	Facility_Name	112	0	0	1	0	varchar
19	Gender	1	0	0	1	0	varchar
20	Hospital_County	11	0	0	1	0	varchar
21	Hospital_Service_Area	15	0	0	1	0	varchar
22	ID	4	10	0	0	1	int
23	Length_of_Stay	5	0	0	1	0	varchar
24	Operating_Certificate_Number	12	0	0	1	0	varchar
25	Patient_Disposition	37	0	0	1	0	varchar
26	Payment_Typology_1	25	0	0	1	0	varchar
27	Payment_Typology_2	25	0	0	1	0	varchar
28	Payment_Typology_3	25	0	0	1	0	varchar
29	Permanent_Facility_Id	4	10	0	1	0	int
30	Race	32	0	0	1	0	varchar
31	Total_Charges	13	28	2	1	0	deci...
32	Total_Costs	13	28	2	1	0	deci...
33	Type_of_Admission	15	0	0	1	0	varchar
34	Zip_Code_3_digits	3	0	0	1	0	varchar

The values match what we initially set up, so everything looks in order here.

3). Checking for Irrelevant Data

Our next task is to assess the presence of irrelevant data such as data that doesn't serve our business need, unknown, and null values. We will delete those affecting our data, and take note the locations of any others.

Null Values

The general approach we will take with nulls is:

1. Ignore them if they do not impact calculations
2. Impute values if applicable
3. Remove or replace them with another value if they impact calculations

That said, nulls are viewed within the context of the data set they appear in. They can be an indicator that other things are amiss with the data, but in a different context, there may be perfectly legitimate reasons for their presence. For example, null values under secondary or tertiary payment typology likely indicate that the patient did not have additional insurance coverage or opted not to select a payment option. However, a null under payment_typology_1 could imply a data entry error where the patient's status wasn't confirmed, or entered appropriately, which may warrant further review. Let's continue.

To count null values in our columns, we can use a variety of methods such as COUNT or a SUM function with a Case Statement.

```
-- Counts the total null values within each specified column

SELECT 'Hospital_Service_Area' as ColumnName,SUM(CASE WHEN Hospital_Service_Area IS NULL THEN 1 ELSE 0 END) as Total_Nulls FROM dbo.SPARC2019 UNION ALL
SELECT 'Hospital_County',SUM(CASE WHEN Hospital_County IS NULL THEN 1 ELSE 0 END) as col2 FROM dbo.SPARC2019 UNION ALL
SELECT 'Operating_Certificate_Number',SUM(CASE WHEN Operating_Certificate_Number IS NULL THEN 1 ELSE 0 END) as col3 FROM dbo.SPARC2019 UNION ALL
SELECT 'Permanent_Facility_Id',SUM(CASE WHEN Permanent_Facility_Id IS NULL THEN 1 ELSE 0 END) as col4 FROM dbo.SPARC2019 UNION ALL
SELECT 'Zip_Code_3_digits',SUM(CASE WHEN Zip_Code_3_digits IS NULL THEN 1 ELSE 0 END) as col5 FROM dbo.SPARC2019 UNION ALL
SELECT 'CCSR_Diagnosis_Code',SUM(CASE WHEN CCSR_Diagnosis_Code IS NULL THEN 1 ELSE 0 END) as col6 FROM dbo.SPARC2019  UNION ALL
SELECT 'CCSR_Diagnosis_Description',SUM(CASE WHEN CCSR_Diagnosis_Description IS NULL THEN 1 ELSE 0 END) as col7 FROM dbo.SPARC2019 UNION ALL
SELECT 'CCSR_Procedure_Code',SUM(CASE WHEN CCSR_Procedure_Code IS NULL THEN 1 ELSE 0 END) as col8 FROM dbo.SPARC2019  UNION ALL
SELECT 'CCSR_Procedure_Description',SUM(CASE WHEN CCSR_Procedure_Description IS NULL THEN 1 ELSE 0 END) as col9 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_DRG_Code',SUM(CASE WHEN APR_DRG_Code IS NULL THEN 1 ELSE 0 END) as col10 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_DRG_Description',SUM(CASE WHEN APR_DRG_Description IS NULL THEN 1 ELSE 0 END) as col11 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_MDC_Code',SUM(CASE WHEN APR_MDC_Code IS NULL THEN 1 ELSE 0 END) as col12 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_MDC_Description',SUM(CASE WHEN APR_MDC_Description IS NULL THEN 1 ELSE 0 END) as col13 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_Severity_of_Illness_Code',SUM(CASE WHEN APR_Severity_of_Illness_Code IS NULL THEN 1 ELSE 0 END) as col14 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_Severity_of_Illness_Description',SUM(CASE WHEN APR_Severity_of_Illness_Description IS NULL THEN 1 ELSE 0 END) as col15 FROM dbo.SPARC2019 UNION ALL
SELECT 'APR_Risk_of_Mortality',SUM(CASE WHEN APR_Risk_of_Mortality IS NULL THEN 1 ELSE 0 END) as col16 FROM dbo.SPARC2019 UNION ALL
SELECT 'Payment_Typology_2',SUM(CASE WHEN Payment_Typology_2 IS NULL THEN 1 ELSE 0 END) as col17 FROM dbo.SPARC2019 UNION ALL
SELECT 'Payment_Typology_3',SUM(CASE WHEN Payment_Typology_3 IS NULL THEN 1 ELSE 0 END) as col18 FROM dbo.SPARC2019 UNION ALL
SELECT 'Birth_Weight',SUM(CASE WHEN Birth_Weight IS NULL THEN 1 ELSE 0 END) as col19 FROM dbo.SPARC2019 ORDER BY 1
```

Here are the results of the case statement query:

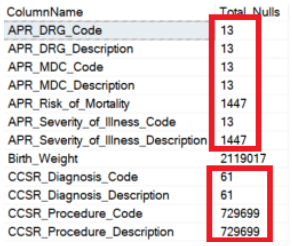
100 %

Results | Messages

	ColumnName	Total_Nulls
1	APR_DRG_Code	13
2	APR_DRG_Description	13
3	APR_MDC_Code	13
4	APR_MDC_Description	13
5	APR_Risk_of_Mortality	1447
6	APR_Severity_of_Illness_Code	13
7	APR_Severity_of_Illness_Description	1447
8	Birth_Weight	2119017
9	CCSR_Diagnosis_Code	61
10	CCSR_Diagnosis_Description	61
11	CCSR_Procedure_Code	729699
12	CCSR_Procedure_Description	729699
13	Hospital_County	9505
14	Hospital_Service_Area	9505
15	Operating_Certificate_Number	9505
16	Payment_Typology_2	968787
17	Payment_Typology_3	1867656
18	Permanent_Facility_Id	9071
19	Zip_Code_3_digits	42930

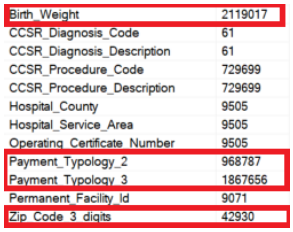
That's a lot of nulls! We need to plan our approach to find out if they interfere with our investigation.

One thing that immediately stands out in the summary list are the nulls clustered into certain groupings, APR and CCSR in particular. Perhaps there are common factors within these clusters that we can uncover. We'll start off exploring the APR clusters, and then CCSRs.



ColumnName	Total_Nulls
APR_DRG_Code	13
APR_DRG_Description	13
APR_MDC_Code	13
APR_MDC_Description	13
APR_Risk_of_Mortality	1447
APR_Severity_of_Illness_Code	13
APR_Severity_of_Illness_Description	1447
Birth_Weight	2119017
CCSR_Diagnosis_Code	61
CCSR_Diagnosis_Description	61
CCSR_Procedure_Code	729699
CCSR_Procedure_Description	729699

Other things that stand out are a few very large null clusters, perhaps indicating some particular data is not acquired on a significant portion of patients.



Birth_Weight	2119017
CCSR_Diagnosis_Code	61
CCSR_Diagnosis_Description	61
CCSR_Procedure_Code	729699
CCSR_Procedure_Description	729699
Hospital_County	9505
Hospital_Service_Area	9505
Operating_Certificate_Number	9505
Payment_Typology_2	968787
Payment_Typology_3	1867656
Permanent_Facility_Id	9071
Zip_Code_3_digits	42930

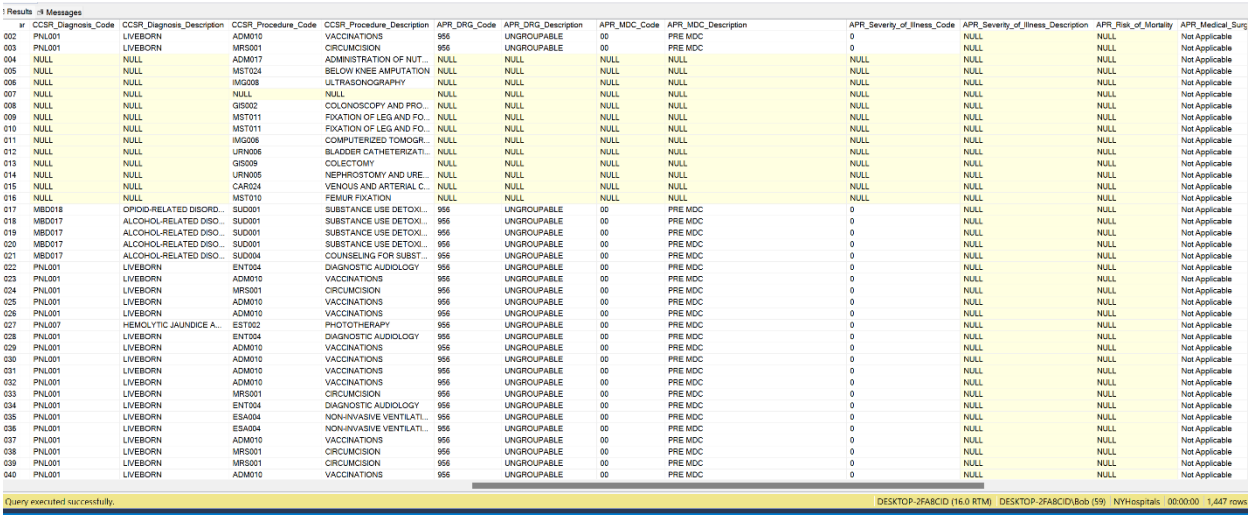
Many null entries are anticipated, and can be safely ignored because they're either irrelevant to the business task or there are parts of a patient's hospitalization where data are not expected to be collected. For example, Birth Weight wouldn't be a measurement taken on someone who isn't a newborn!

Since the null count for CCSR_Procedure_Code and CSR_Procedure_Description matches one another, we will ignore them because this is not out of the ordinary. It's to be expected that not all patients will have a procedure performed on visit. Zipcode nulls will be ignored as the patient's home location isn't directly relevant to the scope of the business task. Hospital_County, Hospital_Service_Area and Operating_Certificate_Number will be ignored because their values don't impact our business task. We will also ignore nulls generated on the payment typology columns as we have no way to collect missing payment information or verify that the absence of patient payment method entries are actually invalid values.

We'll filter by column APR_Severity_of_Illness_Description since this contains the most nulls in the group, and see what we get.

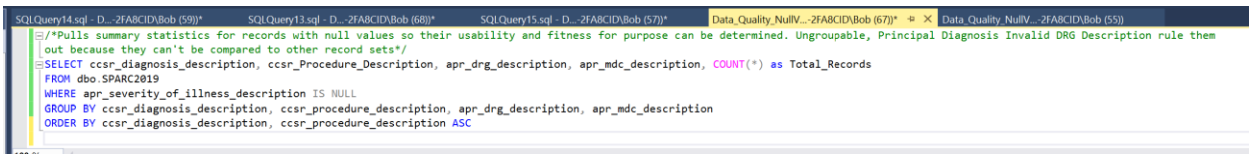
```sql
-- Pulls selected nulled records that are split into a group of 13 and 1447 records.
SELECT *
FROM dbo.SPARC2019
WHERE APR_Severity_of_Illness_Description IS NULL
ORDER BY ID  ASC
```

Interesting. It looks like the 13 records from the summary statistics are part of the larger group of 1447. It also seems that most, if not all, relevant clinical information is nulled out.



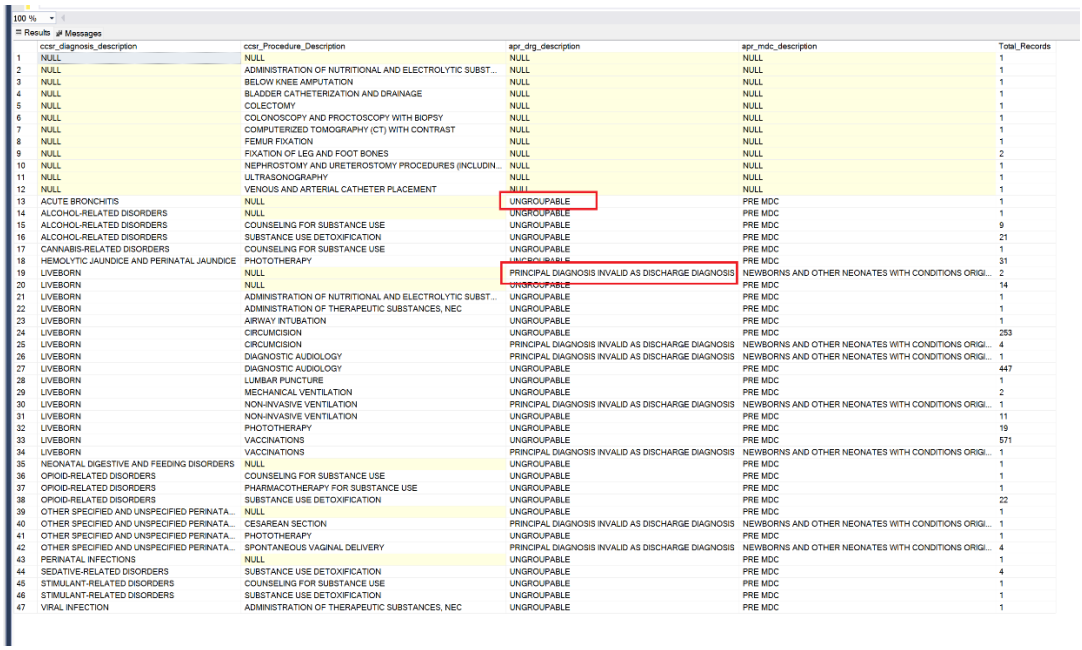
	CCSR_Diagnosis_Code	CCSR_Diagnosis_Description	CCSR_Procedure_Code	CCSR_Procedure_Description	APR_DRG_Code	APR_DRG_Description	APR_MDC_Code	APR_MDC_Description	APR_Severity_of_Illness_Code	APR_Severity_of_Illness_Description	APR_Risk_of_Mortality	APR_Medical_Surg
002	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
003	PNL001	LIVEBORN	MRS001	CIRCUMCISION	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
004	NULL	NULL	ADM017	ADMINISTRATION OF NUT...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
005	NULL	NULL	MST024	BELOW KNEE AMPUTATION	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
006	NULL	NULL	IMG008	ULTRASONOGRAPHY	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
007	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
008	NULL	NULL	GIS002	COLONOSCOPY AND PRO...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
009	NULL	NULL	MST011	FIXATION OF LEG AND FO...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
010	NULL	NULL	MST011	FIXATION OF LEG AND FO...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
011	NULL	NULL	IMG006	COMPUTERIZED TOMOGR...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
012	NULL	NULL	URN006	BLADDER CATHETERIZATI...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
013	NULL	NULL	GIS009	COLECTOMY	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
014	NULL	NULL	URN005	NEPHROSTOMY AND URE...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
015	NULL	NULL	CAR024	VENOUS AND ARTERIAL C...	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
016	NULL	NULL	MST010	FEMUR FIXATION	NULL	NULL	NULL	NULL	NULL	NULL	NULL	Not Applicable
017	MBD018	OPIOID-RELATED DISORD...	SUD001	SUBSTANCE USE DETOXI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
018	MBD017	ALCOHOL-RELATED DISO...	SUD001	SUBSTANCE USE DETOXI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
019	MBD017	ALCOHOL-RELATED DISO...	SUD001	SUBSTANCE USE DETOXI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
020	MBD017	ALCOHOL-RELATED DISO...	SUD001	SUBSTANCE USE DETOXI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
021	MBD017	ALCOHOL-RELATED DISO...	SUD004	COUNSELING FOR SUBST...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
022	PNL001	LIVEBORN	ENT004	DIAGNOSTIC AUDIOLOGY	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
023	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
024	PNL001	LIVEBORN	MRS001	CIRCUMCISION	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
025	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
026	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
027	PNL007	HEMOLYTIC JAUNDICE A...	EST002	PHOTOTHERAPY	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
028	PNL001	LIVEBORN	ENT004	DIAGNOSTIC AUDIOLOGY	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
029	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
030	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
031	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
032	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
033	PNL001	LIVEBORN	MRS001	CIRCUMCISION	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
034	PNL001	LIVEBORN	ENT004	DIAGNOSTIC AUDIOLOGY	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
035	PNL001	LIVEBORN	ESA004	NON-INVASIVE VENTILATI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
036	PNL001	LIVEBORN	ESA004	NON-INVASIVE VENTILATI...	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
037	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
038	PNL001	LIVEBORN	MRS001	CIRCUMCISION	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
039	PNL001	LIVEBORN	MRS001	CIRCUMCISION	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable
040	PNL001	LIVEBORN	ADM010	VACCINATIONS	956	UNGROUPABLE	00	PRE MDC	0	NULL	NULL	Not Applicable

Although insightful, we should take a bird's eye view of the entire record set through another summary statistics query. Let's run one now.



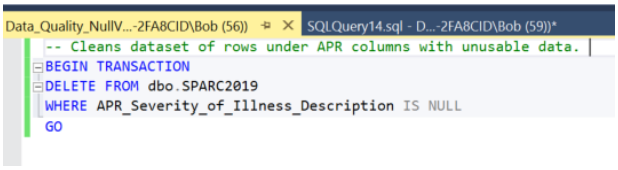
```sql
/*Pulls summary statistics for records with null values so their usability and fitness for purpose can be determined. Ungroupable, Principal Diagnosis Invalid DRG Description rule them
out because they can't be compared to other record sets*/
SELECT ccsr_diagnosis_description, ccsr_Procedure_Description, apr_drg_description, apr_mdc_description, COUNT(*) as Total_Records
FROM dbo.SPARC2019
WHERE apr_severity_of_illness_description IS NULL
GROUP BY ccsr_diagnosis_description, ccsr_procedure_description, apr_drg_description, apr_mdc_description
ORDER BY ccsr_diagnosis_description, ccsr_procedure_description ASC
```

We see that the records are categorized in the APR classification system as "Ungroupable" or "Principal Diagnosis Invalid as Discharge Diagnosis." This probably means they are not usable for our purpose as we likely need to be able to classify patients according to their problems and illness severity.



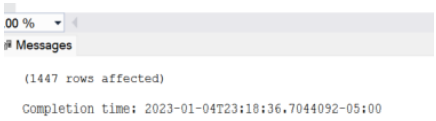
	ccsr_diagnosis_description	ccsr_Procedure_Description	apr_drg_description	apr_mdc_description	Total_Records
1	NULL	NULL	NULL	NULL	1
2	NULL	ADMINISTRATION OF NUTRITIONAL AND ELECTROLYTIC SUBST...	NULL	NULL	1
3	NULL	BELOW KNEE AMPUTATION	NULL	NULL	1
4	NULL	BLADDER CATHETERIZATION AND DRAINAGE	NULL	NULL	1
5	NULL	COLECTOMY	NULL	NULL	1
6	NULL	COLONOSCOPY AND PROCTOSCOPY WITH BIOPSY	NULL	NULL	1
7	NULL	COMPUTERIZED TOMOGRAPHY (CT) WITH CONTRAST	NULL	NULL	1
8	NULL	FEMUR FIXATION	NULL	NULL	1
9	NULL	FIXATION OF LEG AND FOOT BONES	NULL	NULL	2
10	NULL	NEPHROSTOMY AND URETEROSTOMY PROCEDURES (INCLUDIN...	NULL	NULL	1
11	NULL	ULTRASONOGRAPHY	NULL	NULL	1
12	NULL	VENOUS AND ARTERIAL CATHETER PLACEMENT	NULL	NULL	1
13	ACUTE BRONCHITIS	NULL	UNGROUPABLE	PRE MDC	1
14	ALCOHOL-RELATED DISORDERS	NULL	UNGROUPABLE	PRE MDC	1
15	ALCOHOL-RELATED DISORDERS	COUNSELING FOR SUBSTANCE USE	UNGROUPABLE	PRE MDC	9
16	ALCOHOL-RELATED DISORDERS	SUBSTANCE USE DETOXIFICATION	UNGROUPABLE	PRE MDC	21
17	CANNABIS-RELATED DISORDERS	COUNSELING FOR SUBSTANCE USE	UNGROUPABLE	PRE MDC	1
18	HEMOLYTIC JAUNDICE AND PERINATAL JAUNDICE	PHOTOTHERAPY	UNGROUPABLE	PRE MDC	31
19	LIVEBORN	NULL	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	2
20	LIVEBORN	NULL	UNGROUPABLE	PRE MDC	14
21	LIVEBORN	ADMINISTRATION OF NUTRITIONAL AND ELECTROLYTIC SUBST...	UNGROUPABLE	PRE MDC	1
22	LIVEBORN	ADMINISTRATION OF THERAPEUTIC SUBSTANCES, NEC	UNGROUPABLE	PRE MDC	1
23	LIVEBORN	AIRWAY INTUBATION	UNGROUPABLE	PRE MDC	1
24	LIVEBORN	CIRCUMCISION	UNGROUPABLE	PRE MDC	253
25	LIVEBORN	CIRCUMCISION	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	4
26	LIVEBORN	DIAGNOSTIC AUDIOLOGY	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	1
27	LIVEBORN	DIAGNOSTIC AUDIOLOGY	UNGROUPABLE	PRE MDC	447
28	LIVEBORN	LUMBAR PUNCTURE	UNGROUPABLE	PRE MDC	1
29	LIVEBORN	MECHANICAL VENTILATION	UNGROUPABLE	PRE MDC	2
30	LIVEBORN	NON-INVASIVE VENTILATION	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	1
31	LIVEBORN	NON-INVASIVE VENTILATION	UNGROUPABLE	PRE MDC	11
32	LIVEBORN	PHOTOTHERAPY	UNGROUPABLE	PRE MDC	19
33	LIVEBORN	VACCINATIONS	UNGROUPABLE	PRE MDC	571
34	LIVEBORN	VACCINATIONS	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	1
35	NEONATAL DIGESTIVE AND FEEDING DISORDERS	NULL	UNGROUPABLE	PRE MDC	1
36	OPIOID-RELATED DISORDERS	COUNSELING FOR SUBSTANCE USE	UNGROUPABLE	PRE MDC	1
37	OPIOID-RELATED DISORDERS	PHARMACOTHERAPY FOR SUBSTANCE USE	UNGROUPABLE	PRE MDC	1
38	OPIOID-RELATED DISORDERS	SUBSTANCE USE DETOXIFICATION	UNGROUPABLE	PRE MDC	22
39	OTHER SPECIFIED AND UNSPECIFIED PERINATA...	NULL	UNGROUPABLE	PRE MDC	1
40	OTHER SPECIFIED AND UNSPECIFIED PERINATA...	CESAREAN SECTION	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	1
41	OTHER SPECIFIED AND UNSPECIFIED PERINATA...	PHOTOTHERAPY	UNGROUPABLE	PRE MDC	1
42	OTHER SPECIFIED AND UNSPECIFIED PERINATA...	SPONTANEOUS VAGINAL DELIVERY	PRINCIPAL DIAGNOSIS INVALID AS DISCHARGE DIAGNOSIS	NEWBORNS AND OTHER NEONATES WITH CONDITIONS ORIGI...	4
43	PERINATAL INFECTIONS	NULL	UNGROUPABLE	PRE MDC	1
44	SEDATIVE-RELATED DISORDERS	SUBSTANCE USE DETOXIFICATION	UNGROUPABLE	PRE MDC	4
45	STIMULANT-RELATED DISORDERS	COUNSELING FOR SUBSTANCE USE	UNGROUPABLE	PRE MDC	1
46	STIMULANT-RELATED DISORDERS	SUBSTANCE USE DETOXIFICATION	UNGROUPABLE	PRE MDC	1
47	VIRAL INFECTION	ADMINISTRATION OF THERAPEUTIC SUBSTANCES, NEC	UNGROUPABLE	PRE MDC	1

We'll need to remove these to clean up our database and have only relevant records. We can test deleting records in all APR categories with the following safe delete query:



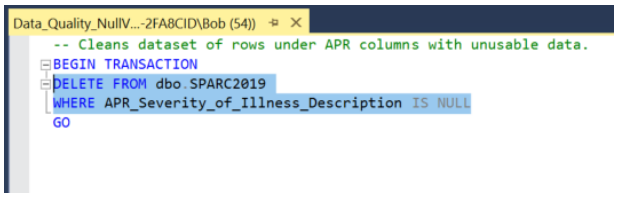
```
Data_Quality_NullV...-2FA8CID\Bob (56))   SQLQuery14.sql - D...-2FA8CID\Bob (59))*
-- Cleans dataset of rows under APR columns with unusable data.
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE APR_Severity_of_Illness_Description IS NULL
GO
```

1,447 Records deleted, which matches the earlier summary count.



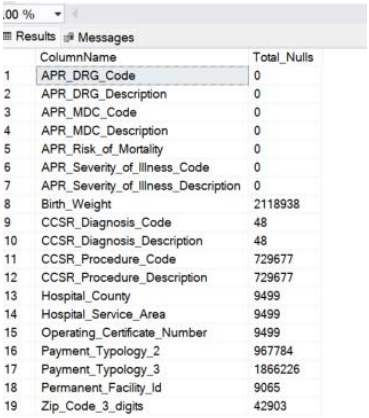
```
00 %
Messages

(1447 rows affected)

Completion time: 2023-01-04T23:18:36.7044092-05:00
```

Now running the actual delete statement.



```
Data_Quality_NullV...-2FA8CID\Bob (54))
-- Cleans dataset of rows under APR columns with unusable data.
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE APR_Severity_of_Illness_Description IS NULL
GO
```

Let's check our results with summary query:



	ColumnName	Total_Nulls
1	APR_DRG_Code	0
2	APR_DRG_Description	0
3	APR_MDC_Code	0
4	APR_MDC_Description	0
5	APR_Risk_of_Mortality	0
6	APR_Severity_of_Illness_Code	0
7	APR_Severity_of_Illness_Description	0
8	Birth_Weight	2118938
9	CCSR_Diagnosis_Code	48
10	CCSR_Diagnosis_Description	48
11	CCSR_Procedure_Code	729677
12	CCSR_Procedure_Description	729677
13	Hospital_County	9499
14	Hospital_Service_Area	9499
15	Operating_Certificate_Number	9499
16	Payment_Typology_2	967784
17	Payment_Typology_3	1866226
18	Permanent_Facility_Id	9065
19	Zip_Code_3_digits	42903

The null APR records are gone!

Time to look at the CCSR groupings.

The diagnosis codes are missing, and since we need it to make meaningful inferences about the patient's stay, we will go ahead and delete these records.

```
-- Cleans dataset of rows under CCSR columns with unusable data.
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE CCSR_Diagnosis_Code IS NULL
GO
```

```
(48 rows affected)

Completion time: 2023-01-05T00:10:59.8614378-05:00
```

48 Records deleted. Let's rerun the summary count.

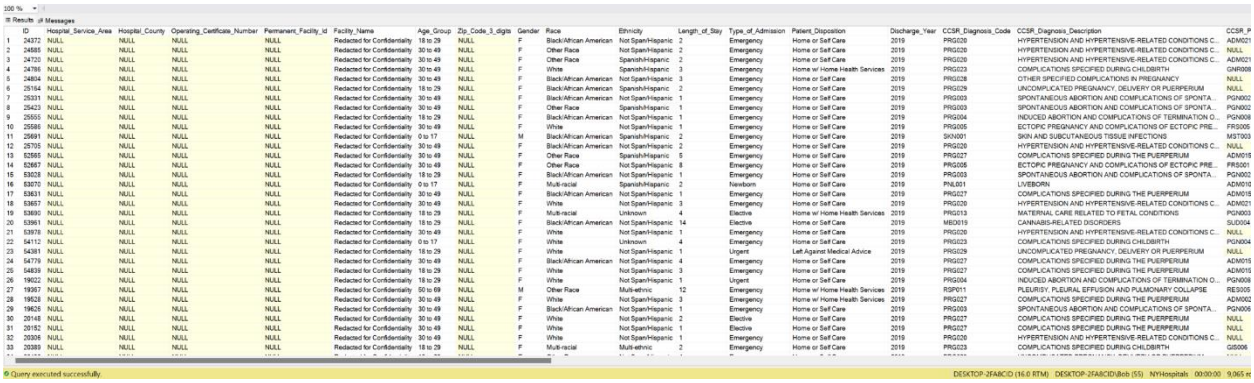
	ColumnName	Total_Nulls
1	APR_DRG_Code	0
2	APR_DRG_Description	0
3	APR_MDC_Code	0
4	APR_MDC_Description	0
5	APR_Risk_of_Mortality	0
6	APR_Severity_of_Illness_Code	0
7	APR_Severity_of_Illness_Description	0
8	Birth_Weight	2118890
9	CCSR_Diagnosis_Code	0
10	CCSR_Diagnosis_Description	0
11	CCSR_Procedure_Code	729673
12	CCSR_Procedure_Description	729673
13	Hospital_County	9499
14	Hospital_Service_Area	9499
15	Operating_Certificate_Number	9499
16	Payment_Typology_2	967763
17	Payment_Typology_3	1866191
18	Permanent_Facility_Id	9065
19	Zip_Code_3_digits	42902

The summary count verifies the deletions.

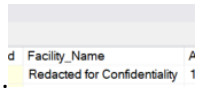
Let’s check the Permanent_facility_ID column.

```
-- Identifies all 9065 nulled records in Second Approach step for Permanent_facility_id column

SELECT *
FROM dbo.SPARC2019
WHERE Permanent_Facility_Id is null
```

Here’s our result table.



We found something interesting in the Facility Name column.



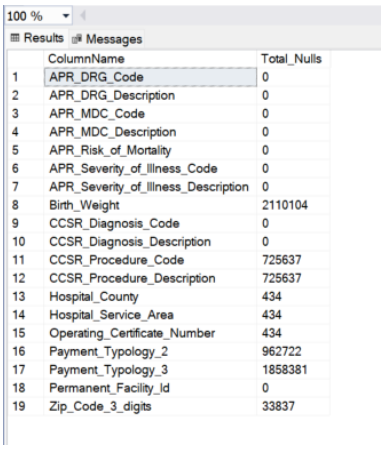
d	Facility_Name	A
	Redacted for Confidentiality	1

Looks like a complete subset of patients had a special redaction placed to ensure they could not be reidentified due to the sensitive nature of their care. Unfortunately, the redaction removed the facility name which we need to make inferences about the care provided. We will have to delete these.

```
-- Deletes all 9065 nulled records in Second Approach step for Permanent_facility_id column
BEGIN TRAN
DELETE FROM dbo.SPARC2019
WHERE Permanent_Facility_Id is null
GO
```

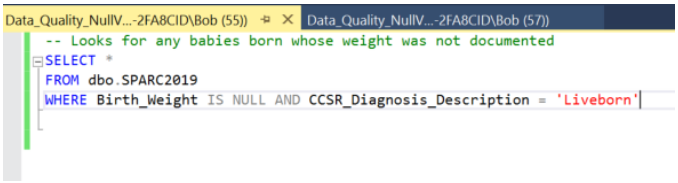
```
100 %
Messages

(9065 rows affected)

Completion time: 2023-01-05T01:12:21.1861221-05:00
```

Deletion verified.



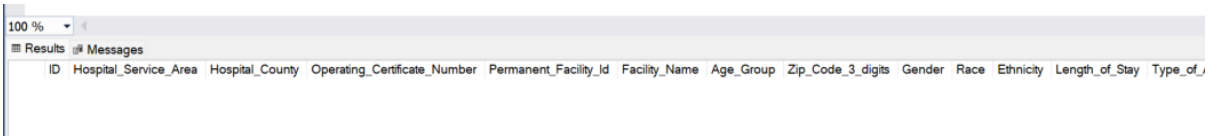
	ColumnName	Total_Nulls
1	APR_DRG_Code	0
2	APR_DRG_Description	0
3	APR_MDC_Code	0
4	APR_MDC_Description	0
5	APR_Risk_of_Mortality	0
6	APR_Severity_of_Illness_Code	0
7	APR_Severity_of_Illness_Description	0
8	Birth_Weight	2110104
9	CCSR_Diagnosis_Code	0
10	CCSR_Diagnosis_Description	0
11	CCSR_Procedure_Code	725637
12	CCSR_Procedure_Description	725637
13	Hospital_County	434
14	Hospital_Service_Area	434
15	Operating_Certificate_Number	434
16	Payment_Typology_2	962722
17	Payment_Typology_3	1858381
18	Permanent_Facility_Id	0
19	Zip_Code_3_digits	33837

Next, let's check Birth Weight to look for any babies born whose weight may not have been logged.



```
-- Looks for any babies born whose weight was not documented
SELECT *
FROM dbo.SPARC2019
WHERE Birth_Weight IS NULL AND CCSR_Diagnosis_Description = 'Liveborn'
```

Here are our results.



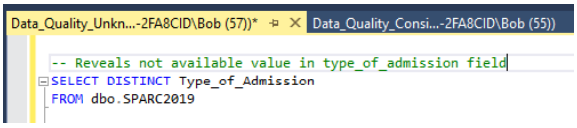
ID	Hospital_Service_Area	Hospital_County	Operating_Certificate_Number	Permanent_Facility_Id	Facility_Name	Age_Group	Zip_Code_3_digits	Gender	Race	Ethnicity	Length_of_Stay	Type_of_

Looks like there aren't any babies that didn't get weighted, and with that we are finished working with our null information!

Unknown or Not Available Values

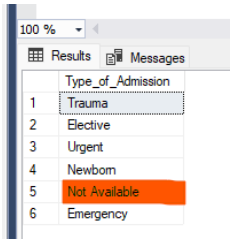
Not all data fields will contain values. These can show up in the data set as "unknowns" or "unavailable." We can find these values sometimes by performing a SELECT Distinct query on the field to get the range of possibly input values. Unknowns may be useful if there are other data that allow inferences to be made without that value being present, however they need to be removed if they may affect calculations or offer no insight. In our use case, hospital admission type is a value we intend to use to find out how patients are entering the facilities, therefore it must contain a usable value.

Let's explore this.



```
-- Reveals not available value in type_of_admission field
SELECT DISTINCT Type_of_Admission
FROM dbo.SPARC2019
```

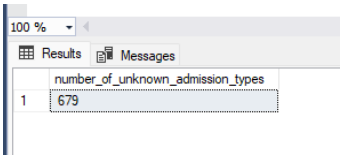
In our Type_of_Admission column, we have a value labeled "Not Available."



	Type_of_Admission
1	Trauma
2	Elective
3	Urgent
4	Newborn
5	Not Available
6	Emergency

The query counts the total amount of “Not Available” admission types.

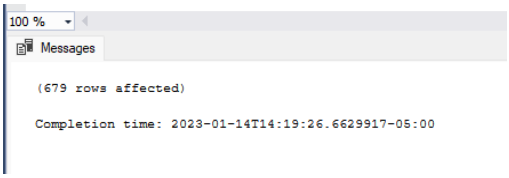
```
-- counts total number of admission types listed as "Not Available"

SELECT COUNT(*) as number_of_unknown_admission_types
FROM dbo.SPARC2019
WHERE Type_of_Admission = 'Not Available'
```



	number_of_unknown_admission_types
1	679

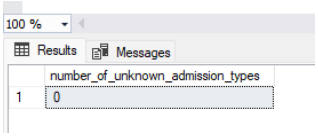
We should remove these using the following query.

```
-- Removes records listed as "Not Available"

BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE Type_of_Admission = 'Not Available'
ROLLBACK TRANSACTION
```



```
(679 rows affected)

Completion time: 2023-01-14T14:19:26.6629917-05:00
```

We confirm the records are deleted.



	number_of_unknown_admission_types
1	0

1.4.b Ensuring Data Are Unique

A second critical component of quality data is ensuring that the data are unique, and not redundant or duplicated. It is important that we can trust that any calculations are not skewed from an over representative sample coming from any duplicate information.

Disclosure: This example is for demonstrative purposes only. Some data being removed may not actually be duplicate records. Due to confidentiality of the original data set, I lack access to the actual identifiers such as a contact service number, or hospital account record number that could link the encounter to a specific patient and thus verify the uniqueness of the record(s). The process is the same, otherwise. In

any event, in absence of completely unique identifiers, the data highlighted here warrant additional scrutiny and consideration for removal from the dataset prior to analysis.

We will scan the dataset to check for potential duplicates.

```
-- Obtains the total number of unique records in the dataset that have duplicates
Select COUNT(group_count) as 'Unique_Records_With_Duplicates'
FROM
    (
        SELECT
        [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
        [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
        [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs], COUNT(*) as group_count

        FROM dbo.SPARC2019
        GROUP BY
        [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
        [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
        [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs]

        HAVING COUNT(*) > 1
    ) Unique_Record_Count
```

Let's check the results.

100 %

Results | Messages

	Unique_Records_With_Duplicates
1	5298

Looks like we have 5,298 unique records that contain potential duplicates.

Let's find the total potential duplicates.

```
-- Obtains the total number of duplicate records
Select SUM(group_count) - COUNT(group_count) as 'Total_Duplicate_Records'
FROM
    (
        SELECT
        [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
        [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
        [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs], COUNT(*) as group_count

        FROM dbo.SPARC2019
        GROUP BY
        [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
        [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
        [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs]

        HAVING COUNT(*) > 1
    ) Duplicate_Record_Count
```

There are a total of 6,956 potential duplicate records in our database.

100 %

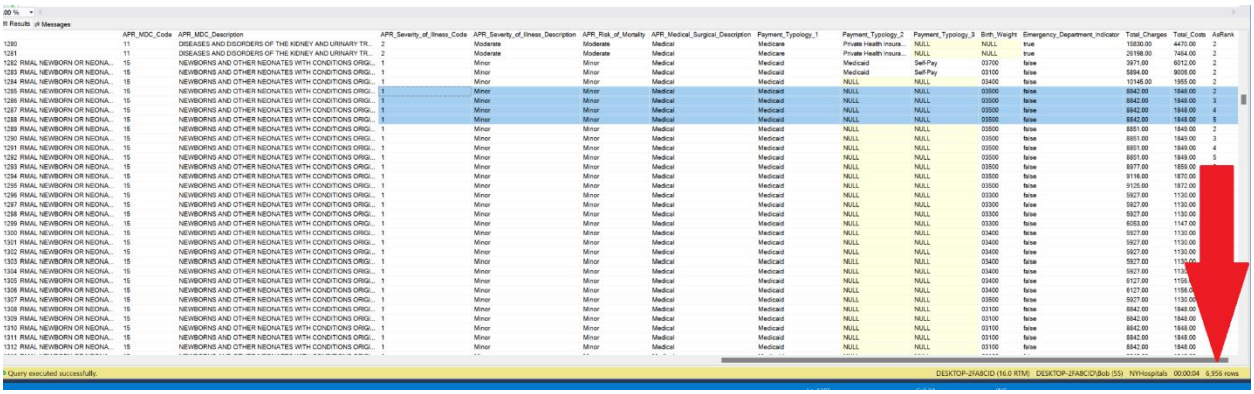
Results | Messages

	Total_Duplicate_Records
1	6956

We should retrieve a sample of the potential duplicates to visually confirm the data.

```sql
-- Retrieves the duplicate records
SELECT *
FROM
    (
        SELECT *,
        DENSE_RANK() OVER(PARTITION BY
            [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
            [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
            [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs]

        ORDER BY ID) as AsRank
        FROM DBO.SPARC2019
    ) RankCheck
WHERE RankCheck.AsRank > 1
```

Visually confirming the data verifies the initial assessment of 6,956 potential duplicate records and provides examples of the potential duplicate information.



We see records for newborns that appear to contain similar information.

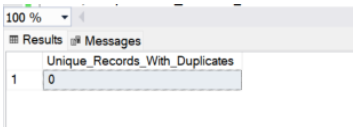
It's time to delete the data.

First, we will run our safe delete SQL script to test the deletion set before fully committing.

```sql
-- Deletes duplicate records from dataset

BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE ID IN
    (
        SELECT ID
        FROM
            (
                SELECT *,
                DENSE_RANK() OVER(PARTITION BY
                    [Hospital_Service_Area], [Hospital_County], [Operating_Certificate_Number], [Permanent_Facility_Id], [Facility_Name], [Age_Group], [Zip_Code_3_digits], [Gender], [Race], [Ethnicity],
                    [Length_of_Stay], [Type_of_Admission], [Patient_Disposition], [CCSR_Diagnosis_Code],[CCSR_Procedure_Code], [APR_DRG_Code], [APR_MDC_Code], [APR_Severity_of_Illness_Code], [APR_Risk_of_Mortality],
                    [Payment_Typology_1], [Payment_Typology_2], [Payment_Typology_3], [Birth_Weight], [Emergency_Department_Indicator], [Total_Charges], [Total_Costs]

                ORDER BY ID) as AsRank
                FROM DBO.SPARC2019
            ) RankCheck
        WHERE RankCheck.AsRank > 1

    )
ROLLBACK TRANSACTION
```

Here are the results.

Looks like we were successful. Let's reset the transaction, execute the deletion script, and rerun our unique records with duplicates query.



Our data set is cleaned of duplicates. Mission accomplished!

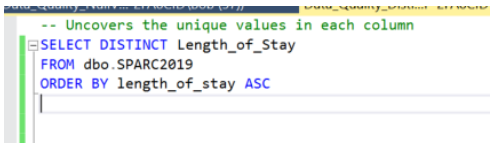
1.4.c Ensuring Data Consistency

Data elements should have the same context and values across locations for our analysis to be trustworthy, and avoid loss of information. They should not conflict, and should not change value, or meaning, when used in different contexts. We will check our dataset for any inconsistencies between values stored within the fields.

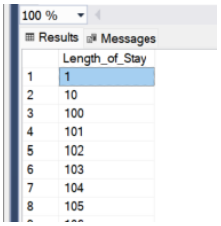
1). Extracting and Filtering Distinct Values

A relatively simple way to perform consistency verification is querying columns to extract each distinct value. This helps to not only check data consistency, but also provides important context through becoming familiar with the values stored in the table.

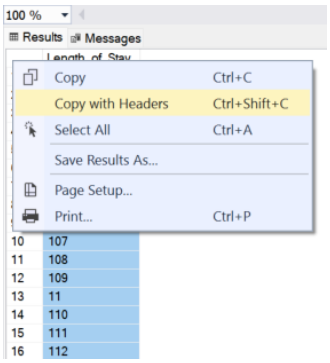
Let's begin by executing the following query, adjusting the column name for each specific column.



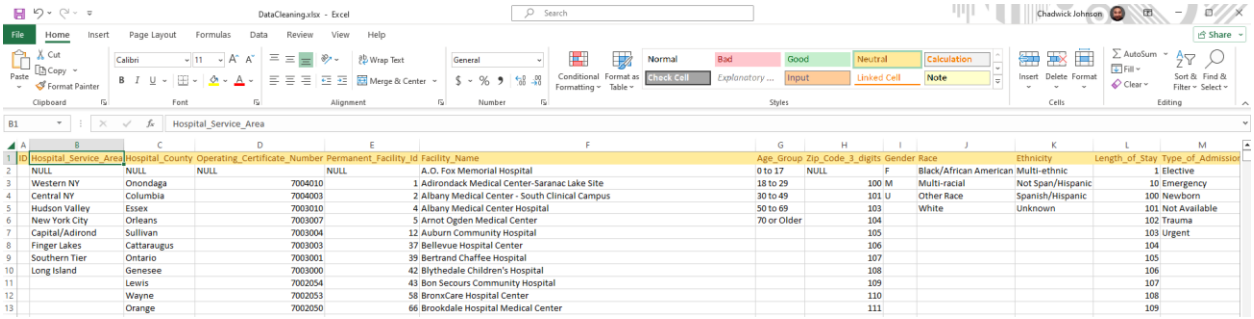
This gives us the following:



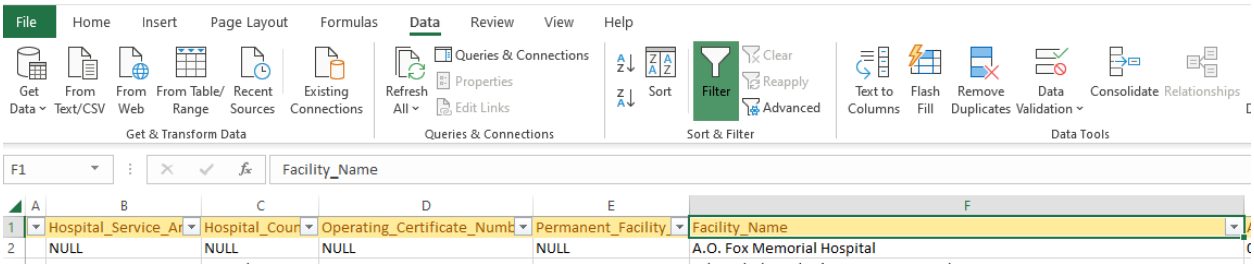
In SSMS we left click to highlight the column, right click on the same area, and click **Copy with Headers**.



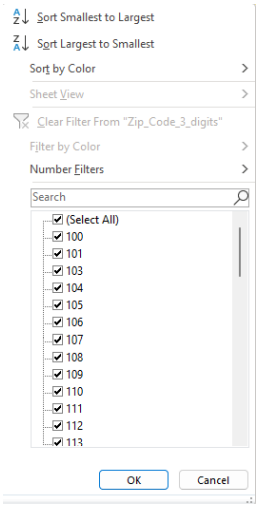
We create a new Excel spreadsheet and paste the copied rows until we have a complete column list.



The spreadsheet preparation is complete when we enable Filters on the columns.

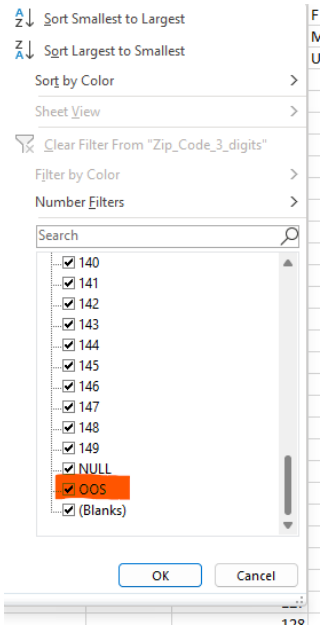


Filtering allows quick browsing through the data rows so we can look for anomalies such as wrong format, data type mismatch, inappropriate characters, and ambiguous values. Let's check our filters.



2). Examining Zip Code Column

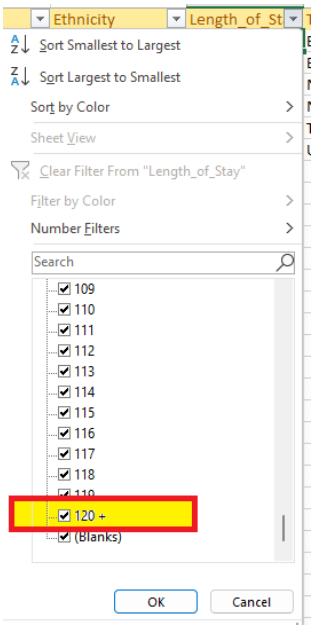
We notice something odd in the Zip_code_3_digits column. Sometimes it's useful to note values that, while not incorrect, they may later need to be taken into account when analyzing. The "OOS" string value is out of place with the ordinal zip code values within the column.



Conferring with the data dictionary we see that OOS stands for "Out of State". We will need to take this into account if we run queries on this column during analysis. We may have missed this nuance without checking the data dictionary, or if we didn't perform this consistency check. Zip codes are ordinal data, and do not require us to make any transformations to the field.

3). Examining Length_of_Stay Column

In the Length_of_Stay filter we come across a non-numerical value "120 +":



The data dictionary is consulted.

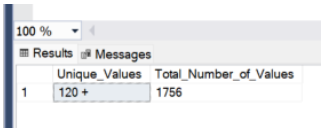
Length of Stay	Type is Char. Length is 5. The total number of patient days at an acute level and/or other than acute care level (excluding leave of absence days) (Discharge Date - Admission Date) + 1. Length of Stay greater than or equal to 120 days has been aggregated to 120+ days.

120 + is an aggregation cut off point, and therefore not the actual value pertaining to the individual encounter. We know the unit of measurement is in days, and it is a field we will likely perform calculations on therefore we need a way to make it quantifiable by removing any non-numerical characters from the dataset.

Let's check how many instances there are of non-numerical characters in the column.

```
-- Checks the column for any value containing non-numerical characters and counts the total number
SELECT DISTINCT length_of_stay as Unique_Values, count(*) as Total_Number_of_Values
FROM dbo.sparc2019
WHERE Length_of_Stay LIKE '%[^0-9]%'
Group By Length_of_Stay
```

The only non-numerical character to show up was 120 + and there are 1,756 instances in the data.



Now we can use the Replace function to simulate removing the non-numerical data.

```sql
-- Lists all occurances of LOS 120 + days, removes the non numerical characters, and counts the length of the string to verify that only the numerical characters remain.
SELECT length_of_stay, REPLACE('120 +', ' +','') as Formatted_Text, LEN(REPLACE('120 +', ' +','')) as Character_Length
FROM dbo.SPARC2019
WHERE Length_of_Stay = '120 +'
```

The results table show the newly Formatted Text as numerical, and the length function returns three, indicating the column now has the intended number of numerical characters.

100 %

Results | Messages

	length_of_stay	Formatted_Text	Character_Length
1	120 +	120	3
2	120 +	120	3
3	120 +	120	3
4	120 +	120	3
5	120 +	120	3
6	120 +	120	3
7	120 +	120	3
8	120 +	120	3
9	120 +	120	3
10	120 +	120	3
11	120 +	120	3

Let's replace the characters in the dataset using a test update query.

```sql
-- Removes non numerical characters from value '120 +' in LOS column
BEGIN TRANSACTION
UPDATE dbo.SPARC2019
SET Length_of_Stay = REPLACE(length_of_stay,'120 +','120')
FROM dbo.SPARC2019
WHERE Length_of_Stay = '120 +'
ROLLBACK TRANSCATION
```

The 1,756 updated rows match the count of rows from the initial non-numerical characters check.

100 %

Messages

```
(1756 rows affected)

Completion time: 2023 01 06T23:54:03.2604225 05:00
```

To verify the column character length, and presence of non-numerical characters we run

Query 1

```sql
-- Checks the length of the columns to verify that only numerical digits remain
SELECT DISTINCT length_of_stay, LEN(length_of_stay) as Length_of_Characters
from dbo.SPARC2019
```

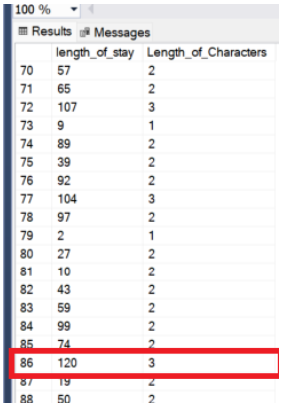
and

Query 2

```sql
-- Checks the column for any value containing non-numerical characters and counts the total number
SELECT DISTINCT length_of_stay as Unique_Values, count(*) as Total_Number_of_Values
FROM dbo.sparc2019
WHERE Length_of_Stay LIKE '%[^0-9]%'
Group By Length_of_Stay
```

Here are the results:

Query 1



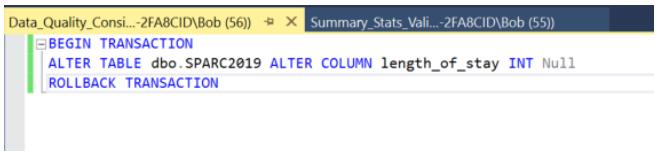
	length_of_stay	Length_of_Characters
70	57	2
71	65	2
72	107	3
73	9	1
74	89	2
75	39	2
76	92	2
77	104	3
78	97	2
79	2	1
80	27	2
81	10	2
82	43	2
83	59	2
84	99	2
85	74	2
86	120	3
87	19	2
88	50	2

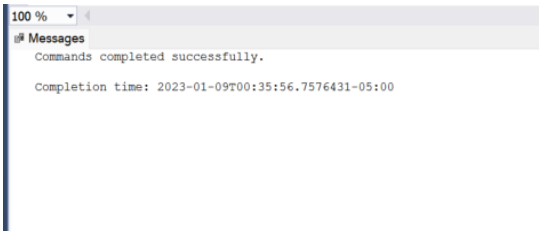
Query 2

Unique_Values	Total_Number_of_Values

The Length of Stay column has now been brought back into consistency, and there are no longer any non-numerical characters present. We will run the query again, without the test parameters to permanently update ethe dataset.

The last change we need to make to the Length_of_Stay column is to change the column data type to INT.





The column now has an INT data type. Let's verify using the sys.columns and sys.types query.

```
SELECT c.name, c.max_length, c.precision, c.scale, c.is_nullable, c.is_identity, ty.name
FROM sys.tables T
INNER JOIN sys.columns C
        ON T.OBJECT_ID = C.OBJECT_ID
INNER JOIN sys.types Ty
        ON c.system_type_id = ty.system_type_id
WHERE t.name = 'sparc2019'
ORDER BY c.name asc


select *
from dbo.SPARC2019
```

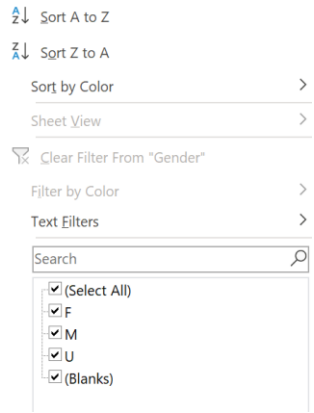
.00 %

Results | Messages

	name	max_length	precision	scale	is_nullable	is_identity	name
1	Age_Group	11	0	0	1	0	varchar
2	APR_DRG_Code	3	0	0	1	0	varchar
3	APR_DRG_Description	500	0	0	1	0	varchar
4	APR_MDC_Code	2	0	0	1	0	varchar
5	APR_MDC_Description	500	0	0	1	0	varchar
6	APR_Medical_Surgical_Description	14	0	0	1	0	varchar
7	APR_Risk_of_Mortality	8	0	0	1	0	varchar
8	APR_Severity_of_Illness_Code	1	0	0	1	0	varchar
9	APR_Severity_of_Illness_Description	8	0	0	1	0	varchar
10	Birth_Weight	5	0	0	1	0	varchar
11	CCSR_Diagnosis_Code	10	0	0	1	0	varchar
12	CCSR_Diagnosis_Description	250	0	0	1	0	varchar
13	CCSR_Procedure_Code	10	0	0	1	0	varchar
14	CCSR_Procedure_Description	250	0	0	1	0	varchar
15	Discharge_Year	4	0	0	1	0	varchar
16	Emergency_Department_Indicator	12	0	0	1	0	varchar
17	Ethnicity	20	0	0	1	0	varchar
18	Facility_Name	112	0	0	1	0	varchar
19	Gender	1	0	0	1	0	varchar
20	Hospital_County	11	0	0	1	0	varchar
21	Hospital_Service_Area	15	0	0	1	0	varchar
22	ID	4	10	0	0	1	int
23	Length_of_Stay	4	10	0	1	0	int
24	Operating_Certificate_Number	12	0	0	1	0	varchar
25	Patient_Disposition	37	0	0	1	0	varchar

4). Examining Gender Column

Our next challenge is found in the Gender Column



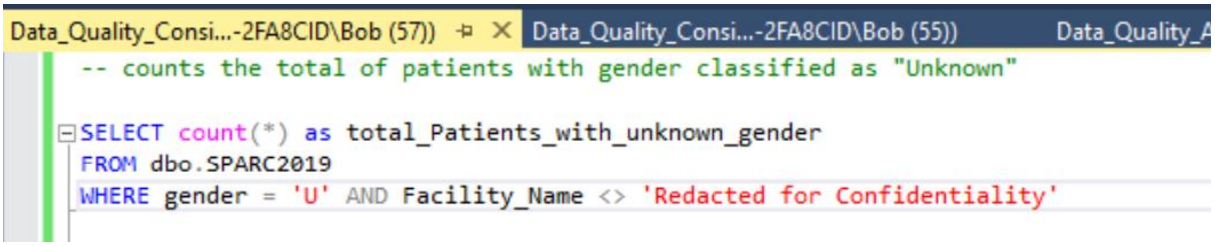
After conferring with the Data Dictionary,

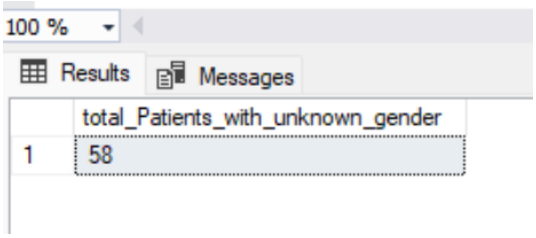
Gender	Type is Char. Length is 1. Patient gender: (M) Male, (F) Female, (U) Unknown.

our options are, F (female), M (male) and U (unknown).

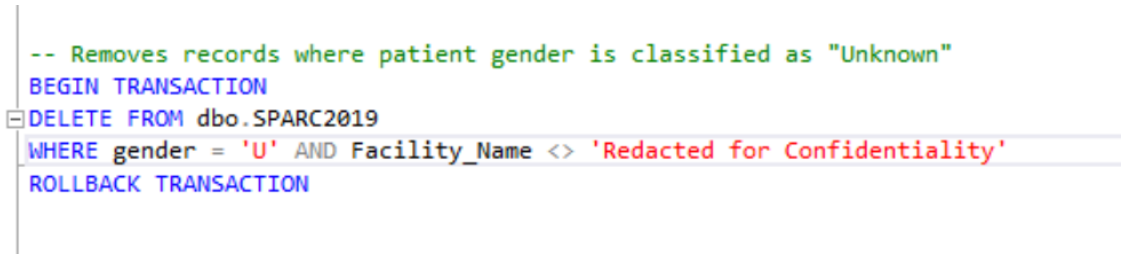
For the purposes of our study, we are limiting inferences to only patients whose have clearly identifiable gender as the data set does not contain the nuanced identifier information allowing us to distinguish a patient's gender non-conforming or LGBTQ+ status. In this context, the only valid values are Female and Male, therefore we must address "Unknown" values.

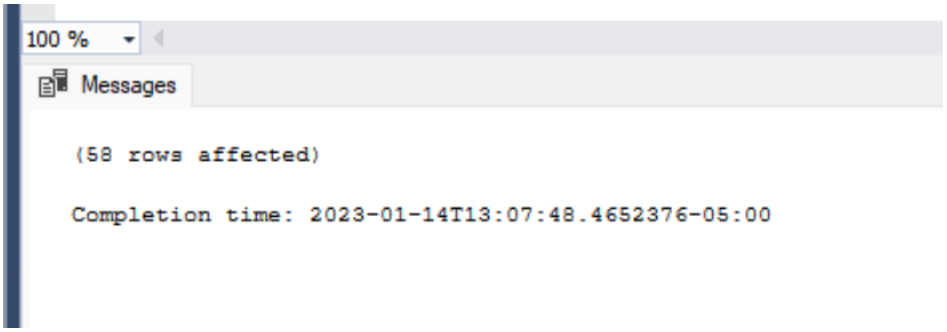
Let's check the total number of patients with an "Unknown" gender classification.



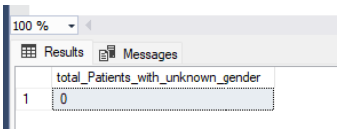


Ok, we will delete the records and run the count query again to verify deletion.





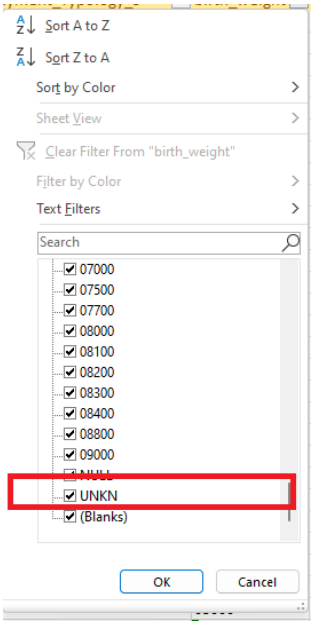
We confirm deletion:



All records with the “Unknown” gender classification are removed.

5). Examining Birth_Weight Column

We now move on to inspecting the Birth_Weight column. Here, we see what appear to be numerical data alongside a string value titled ‘UNKN,’ presumably meaning Unknown.



The data dictionary doesn’t explicitly identify the “Unkn” element; however, it does verify that the unit of measurement is in grams.

Birth Weight	Type is Char. Length is 5. The neonate birth weight in grams; rounded to nearest 100 g.

We will make the column consistent by first changing the unknown values to null since nulls are the logical equivalent to an “unknown” value. We will remove leading zeros, and finally change the column data type to INT.

Let’s review the total records that will be affected.

```sql
select count(*) as Total_Patients
from dbo.SPARC2019
where Birth_Weight = 'unkn'
```

```
Results  Messages
Total_Patients
280
```

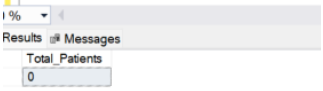
Let’s replace the UNKN values in the Birth_Weight column with Nulls.

```sql
-- Makes the Birth_Weight column consistant to only containing INT data type by removing 'unkn' strings through out and replacing all occuranes with NULL value.
BEGIN TRANSACTION
UPDATE dbo.SPARC2019
SET Birth_Weight = NULL
WHERE Birth_Weight = 'unkn'
ROLLBACK TRANSACTION
```

```
Messages

(280 rows affected)

Completion time: 2023-01-10T02:18:21.1735401-05:00
```

Now let’s confirm the change.

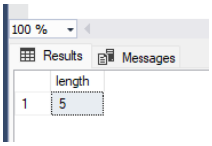
```
select count(*) as Total_Patients
from dbo.SPARC2019
where Birth_Weight = 'unkn'
```



Before we remove the leading zeros, lets check the column length to find any inconstant length values. The column length is an indicator of additional hidden character values that may not show up visually, like spaces. It can also be used to determine if the data in your column is too long or too short as compared to the intended outputs for values, and units of measurement.

```
-- Checks character length of values in birth_weight column to determine the presence of any inconsistant lengths or inappropriate character types in the column
SELECT LEN(birth_weight) as length
FROM dbo.SPARC2019
WHERE birth_weight IS NOT NULL
GROUP BY LEN(birth_weight)
```

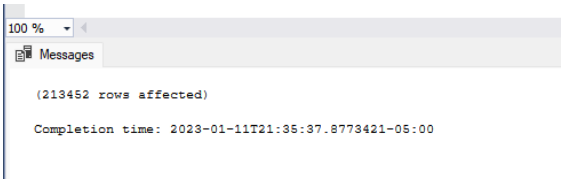
Our column goes up to five characters long at the widest.



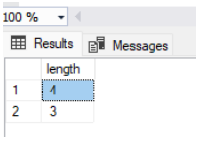
Now let's remove the leading zeros and update the column with the cleaned number set.

```
-- Removes leading zeros, extra spaces, and updates birth_weight column with cleaned number set
BEGIN TRANSACTION
UPDATE dbo.SPARC2019
SET Birth_Weight =
TRIM(SUBSTRING(
    birth_weight,
    PATINDEX('%[1-9]%',birth_weight),
    LEN(birth_weight) - PATINDEX('%[1-9]%',birth_weight) +1))
WHERE Birth_Weight IS NOT NULL
ROLLBACK TRANSACTION
```

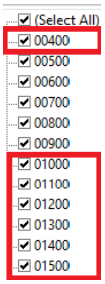
213,452 rows updated!



Let's check the column length again.

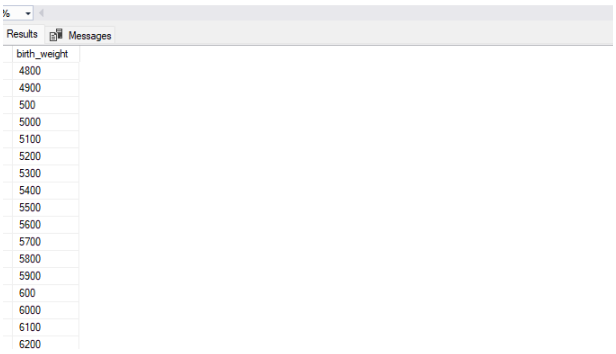


When checking birth_weight in our Excel sheet, we see the lengths (4 and 3) match the expected display length output for the gram unit of measurement.



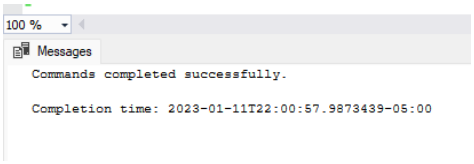
To be confident that the operation left only the correct values in the desired format, the data transformation is visually confirmed from the data set.

```
-- Pulls sample from birth_weight column and Confirms column update was appropriately changed
SELECT DISTINCT top(100) birth_weight
FROM dbo.SPARC2019
WHERE birth_weight is not null
ORDER BY birth_weight ASC
```

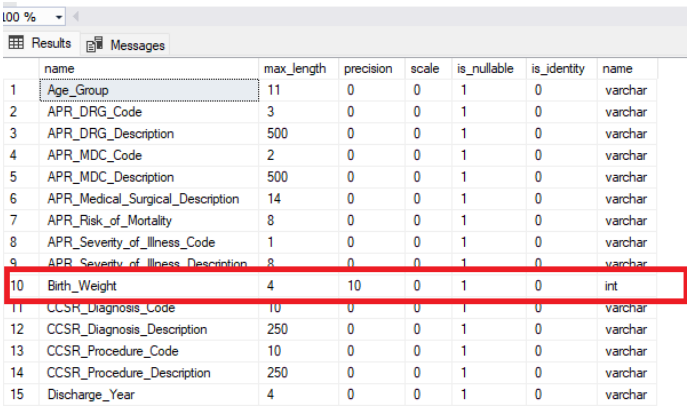


Time to wrap things up by changing the column data type.

```
-- Converts column to INT data type
BEGIN TRANSACTION
ALTER TABLE dbo.SPARC2019
ALTER COLUMN birth_weight INT NULL
ROLLBACK TRANSACTION
```



Let's verify the column data type change.



Results | Messages

	name	max_length	precision	scale	is_nullable	is_identity	name
1	Age_Group	11	0	0	1	0	varchar
2	APR_DRG_Code	3	0	0	1	0	varchar
3	APR_DRG_Description	500	0	0	1	0	varchar
4	APR_MDC_Code	2	0	0	1	0	varchar
5	APR_MDC_Description	500	0	0	1	0	varchar
6	APR_Medical_Surgical_Description	14	0	0	1	0	varchar
7	APR_Risk_of_Mortality	8	0	0	1	0	varchar
8	APR_Severity_of_Illness_Code	1	0	0	1	0	varchar
9	APR_Severity_of_Illness_Description	8	0	0	1	0	varchar
10	Birth_Weight	4	10	0	1	0	int
11	CCSR_Diagnosis_Code	10	0	0	1	0	varchar
12	CCSR_Diagnosis_Description	250	0	0	1	0	varchar
13	CCSR_Procedure_Code	10	0	0	1	0	varchar
14	CCSR_Procedure_Description	250	0	0	1	0	varchar
15	Discharge_Year	4	0	0	1	0	varchar

The column is now set to Integer data type. Mission accomplished!

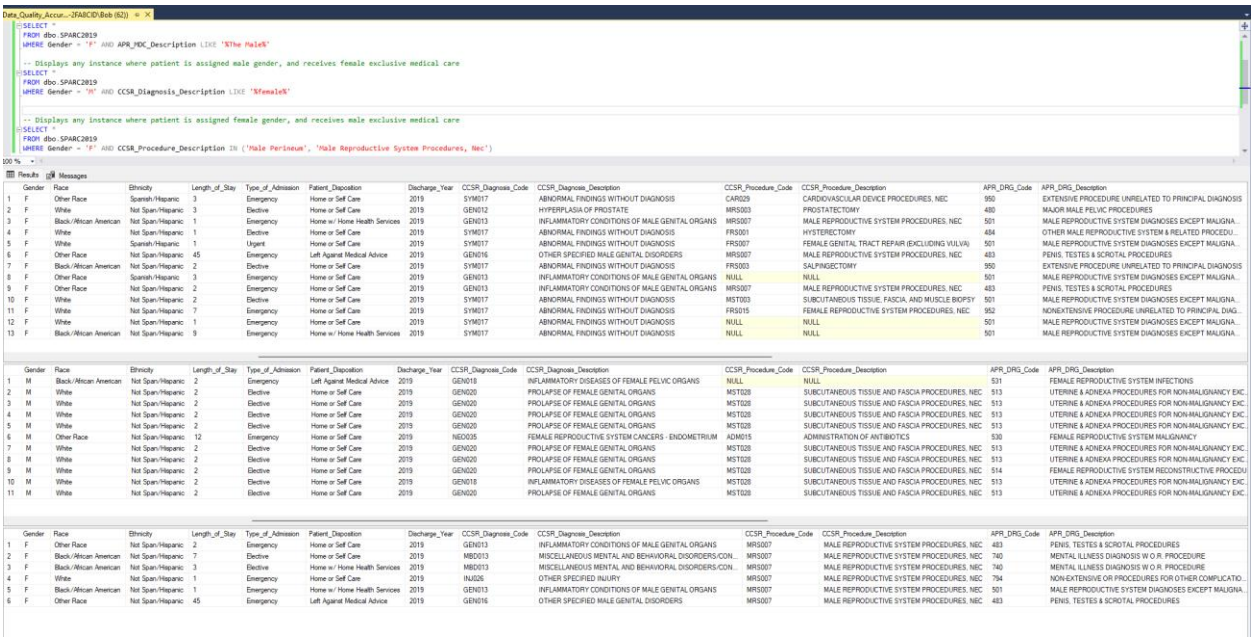
1.4.d Ensuring Data are Accurate

The final leg of our data cleaning process is to review the data for accuracy. For our data to be considered accurate, it must be precise, verifiably correct and useful. Data may still be wrong and unusable even if they pass other quality checks such as being the right type, have consistent values, or are unique. We will check if there are parts of the data set that don't meet the standards set by our business task.

Our checks will focus on services and procedures that should be mutual exclusive based on defined patient categories of gender, and admission type.

1). Gender Exclusive APR_MDC_Description and CCSR_Diagnosis_Description's

Let's run the following query:

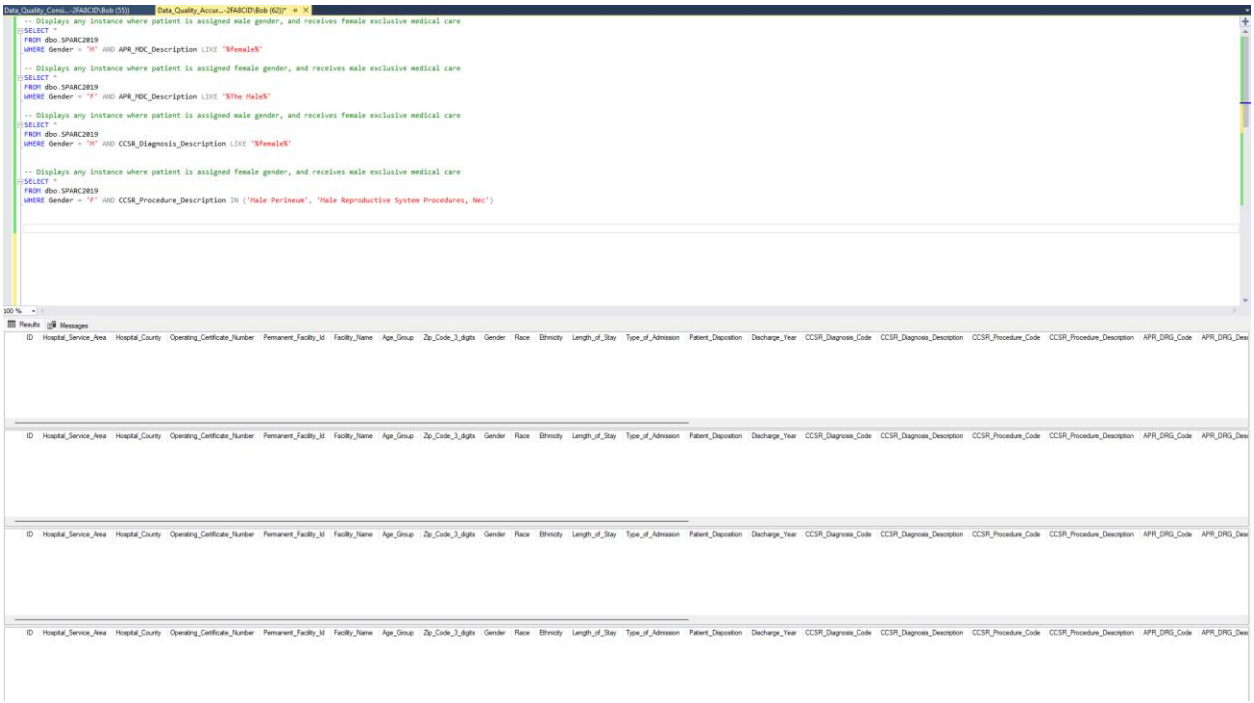


In the query results we see examples of female and male patients being assigned procedures for the opposite gender such as Males with APR_DRG_Descriptions of "Uterine & Adnexa" having Salpingectomies. We also see examples of the wrong gender being assigned to patients who are verifiably the opposite gender based on the medical treatment they received, such as in the CCSR procedure_description where a person classified as female is given a prostatectomy. While each of these are valid value options within their respective columns, they are inaccurate, because they do not represent appropriate responses on the patient record.

We'll use the following queries to update records, make corrections to some values, and delete records that have conflicting medical histories where it is not feasible to determine patient identity without having additional identifying information on the patient.

```
-- Assigns female gender to replace male gender
BEGIN TRANSACTION
UPDATE dbo.SPARC2019
SET Gender = 'F'
WHERE Gender = 'M' AND APR_MDC_Description LIKE '%female%'
ROLLBACK TRANSACTION

-- Assigns male gender to replace femle gender
BEGIN TRANSACTION
UPDATE dbo.SPARC2019
SET Gender = 'M'
WHERE ID IN ('305475','868791','1144608','1264757','1771897')
ROLLBACK TRANSACTION

-- Deletes records that have conflicting medical histories as these cannot be reconciled without consulting clinician or the patient record
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE ID IN ('905201','670610','1950761','26823','909126','233739','1085602','1034884')
ROLLBACK TRANSACTION

-- Deletes records that have excessively vague medical history where gender cannot be reasonably identified without consulting clinician or the patient record
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE ID IN ('2313318','364323','836567')
ROLLBACK TRANSACTION
```

Now we run the initial queries to verify that the data are removed.

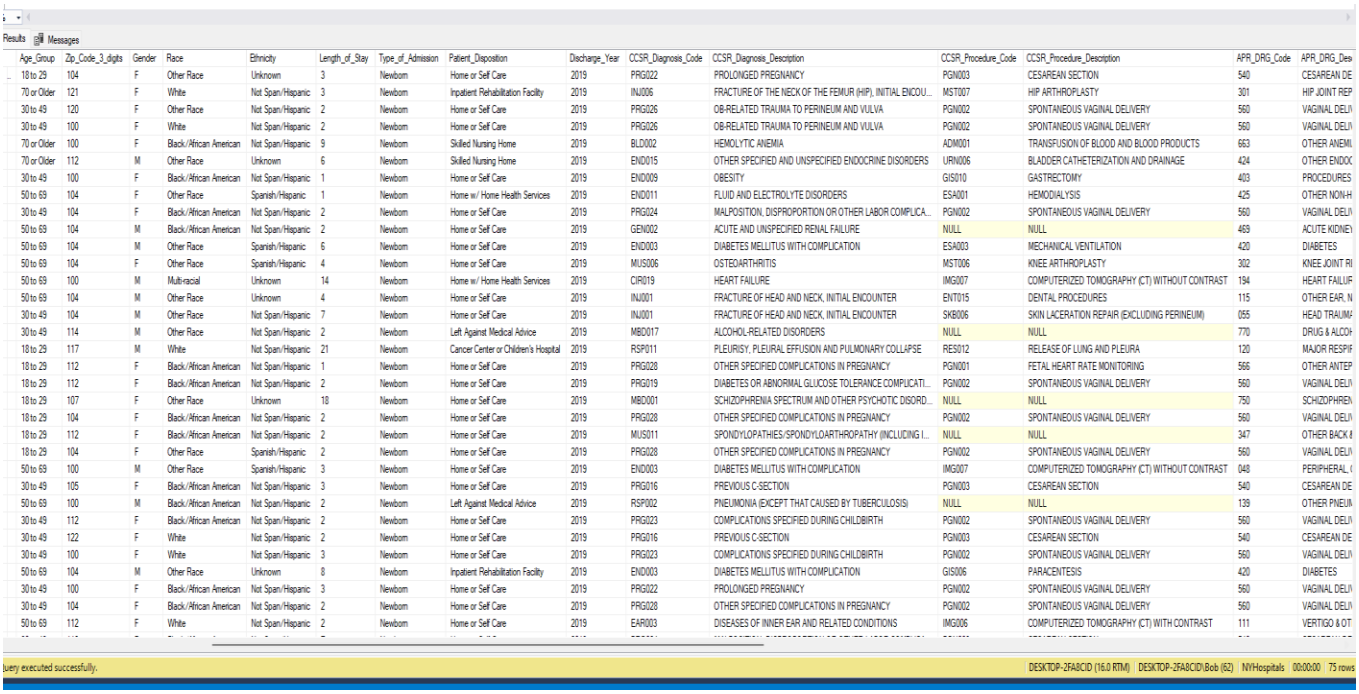


2). Inappropriate Admission Types

Next, we will check Newborn status in the Type_of_Admission field using the following query:

```
-- Checks for age groups that are not appropriate to be classified under Newborn admission type
SELECT *
FROM dbo.SPARC2019
WHERE Type_of_Admission  = 'newborn' AND Birth_Weight IS NULL AND Age_Group <> '0 to 17'
```

We immediately see inappropriate admission classifications in the results such as patients who are 70 years and older.



Age_Group	Zip_Code_3_digits	Gender	Race	Ethnicity	Length_of_Stay	Type_of_Admission	Patient_Disposition	Discharge_Year	CCSR_Diagnosis_Code	CCSR_Diagnosis_Description	CCSR_Procedure_Code	CCSR_Procedure_Description	APR_DRG_Code	APR_DRG_Des
18 to 29	104	F	Other Race	Unknown	3	Newborn	Home or Self Care	2019	PRG022	PROLONGED PREGNANCY	PGN003	CESAREAN SECTION	540	CESAREAN DE
70 or Older	121	F	White	Not Span/Hispanic	3	Newborn	Inpatient Rehabilitation Facility	2019	INJ006	FRACTURE OF THE NECK OF THE FEMUR (HIP), INITIAL ENCOU...	MST007	HIP ARTHROPLASTY	301	HIP JOINT REP
30 to 49	120	F	Other Race	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG026	OB-RELATED TRAUMA TO PERINEUM AND VULVA	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
30 to 49	100	F	White	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG026	OB-RELATED TRAUMA TO PERINEUM AND VULVA	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
70 or Older	100	F	Black/African American	Not Span/Hispanic	9	Newborn	Skilled Nursing Home	2019	BLD002	HEMOLYTIC ANEMIA	ADM001	TRANSFUSION OF BLOOD AND BLOOD PRODUCTS	663	OTHER ANEMI
70 or Older	112	M	Other Race	Unknown	6	Newborn	Skilled Nursing Home	2019	END015	OTHER SPECIFIED AND UNSPECIFIED ENDOCRINE DISORDERS	URN006	BLADDER CATHETERIZATION AND DRAINAGE	424	OTHER ENDOC
30 to 49	100	F	Black/African American	Not Span/Hispanic	1	Newborn	Home or Self Care	2019	END009	OBESITY	GIS010	GASTRECTOMY	403	PROCEDURES
50 to 69	104	F	Other Race	Spanish/Hispanic	1	Newborn	Home w/ Home Health Services	2019	END011	FLUID AND ELECTROLYTE DISORDERS	ESA001	HEMODIALYSIS	425	OTHER NON-H
30 to 49	104	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG024	MALPOSITION, DISPROPORTION OR OTHER LABOR COMPLICA...	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
50 to 69	104	M	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	GEN002	ACUTE AND UNSPECIFIED RENAL FAILURE	NULL	NULL	469	ACUTE KIDNEY
50 to 69	104	M	Other Race	Spanish/Hispanic	6	Newborn	Home or Self Care	2019	END003	DIABETES MELLITUS WITH COMPLICATION	ESA003	MECHANICAL VENTILATION	420	DIABETES
50 to 69	104	F	Other Race	Spanish/Hispanic	4	Newborn	Home or Self Care	2019	MUS006	OSTEOARTHRITIS	MST006	KNEE ARTHROPLASTY	302	KNEE JOINT RI
50 to 69	100	M	Multiracial	Unknown	14	Newborn	Home w/ Home Health Services	2019	CIR019	HEART FAILURE	IMG007	COMPUTERIZED TOMOGRAPHY (CT) WITHOUT CONTRAST	194	HEART FAILUF
50 to 69	104	M	Other Race	Unknown	4	Newborn	Home or Self Care	2019	INJ001	FRACTURE OF HEAD AND NECK, INITIAL ENCOUNTER	ENT015	DENTAL PROCEDURES	115	OTHER EAR, N
30 to 49	104	M	Other Race	Not Span/Hispanic	7	Newborn	Home or Self Care	2019	INJ001	FRACTURE OF HEAD AND NECK, INITIAL ENCOUNTER	SKB006	SKIN LACERATION REPAIR (EXCLUDING PERINEUM)	055	HEAD TRAUMA
30 to 49	114	M	Other Race	Not Span/Hispanic	2	Newborn	Left Against Medical Advice	2019	MBD017	ALCOHOL-RELATED DISORDERS	NULL	NULL	770	DRUG & ALCOH
18 to 29	117	M	White	Not Span/Hispanic	21	Newborn	Cancer Center or Children's Hospital	2019	RSP011	PLEURISY, PLEURAL EFFUSION AND PULMONARY COLLAPSE	RES012	RELEASE OF LUNG AND PLEURA	120	MAJOR RESPIF
18 to 29	112	F	Black/African American	Not Span/Hispanic	1	Newborn	Home or Self Care	2019	PRG028	OTHER SPECIFIED COMPLICATIONS IN PREGNANCY	PGN001	FETAL HEART RATE MONITORING	566	OTHER ANTEP
18 to 29	112	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG019	DIABETES OR ABNORMAL GLUCOSE TOLERANCE COMPLICATI...	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
18 to 29	107	F	Other Race	Unknown	18	Newborn	Home or Self Care	2019	MBD001	SCHIZOPHRENIA SPECTRUM AND OTHER PSYCHOTIC DISORD...	NULL	NULL	750	SCHIZOPHREN
18 to 29	104	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG028	OTHER SPECIFIED COMPLICATIONS IN PREGNANCY	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
18 to 29	112	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	MUS011	SPONDYLOPATHIES/SPONDYLOARTHROPATHY (INCLUDING I...	NULL	NULL	347	OTHER BACK &
18 to 29	104	F	Other Race	Spanish/Hispanic	2	Newborn	Home or Self Care	2019	PRG028	OTHER SPECIFIED COMPLICATIONS IN PREGNANCY	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
50 to 69	100	M	Other Race	Spanish/Hispanic	3	Newborn	Home or Self Care	2019	END003	DIABETES MELLITUS WITH COMPLICATION	IMG007	COMPUTERIZED TOMOGRAPHY (CT) WITHOUT CONTRAST	048	PERIPHERAL (
30 to 49	105	F	Black/African American	Not Span/Hispanic	3	Newborn	Home or Self Care	2019	PRG016	PREVIOUS C-SECTION	PGN003	CESAREAN SECTION	540	CESAREAN DE
50 to 69	100	M	Black/African American	Not Span/Hispanic	2	Newborn	Left Against Medical Advice	2019	RSP002	PNEUMONIA (EXCEPT THAT CAUSED BY TUBERCULOSIS)	NULL	NULL	139	OTHER PNEUM
30 to 49	112	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG023	COMPLICATIONS SPECIFIED DURING CHILDBIRTH	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
30 to 49	122	F	White	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG016	PREVIOUS C-SECTION	PGN003	CESAREAN SECTION	540	CESAREAN DE
30 to 49	100	F	White	Not Span/Hispanic	3	Newborn	Home or Self Care	2019	PRG023	COMPLICATIONS SPECIFIED DURING CHILDBIRTH	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
50 to 69	104	M	Other Race	Unknown	8	Newborn	Inpatient Rehabilitation Facility	2019	END003	DIABETES MELLITUS WITH COMPLICATION	GIS006	PARACENTESIS	420	DIABETES
30 to 49	100	F	Black/African American	Not Span/Hispanic	3	Newborn	Home or Self Care	2019	PRG022	PROLONGED PREGNANCY	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
30 to 49	104	F	Black/African American	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	PRG028	OTHER SPECIFIED COMPLICATIONS IN PREGNANCY	PGN002	SPONTANEOUS VAGINAL DELIVERY	560	VAGINAL DELI
50 to 69	112	F	White	Not Span/Hispanic	2	Newborn	Home or Self Care	2019	EAR003	DISEASES OF INNER EAR AND RELATED CONDITIONS	IMG006	COMPUTERIZED TOMOGRAPHY (CT) WITH CONTRAST	111	VERTIGO & OT

We will need to delete these records from the data set as we cannot confirm the type of admission the patient actually had.

```
-- Deletes records for age groups that are not appopriate to newborn admission type
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE Type_of_Admission  = 'newborn' AND Birth_Weight IS NULL AND Age_Group <> '0 to 17'
ROLLBACK TRANSACTION
```

```
100 %
Messages

(75 rows affected)

Completion time: 2023-01-12T23:46:47.0359111-05:00
```

Our last step is to rerun the previous select query and confirm deletion.

```
-- Checks for age groups that are not appropriate to be classified under Newborn admission type
SELECT *
FROM dbo.SPARC2019
WHERE Type_of_Admission  = 'newborn' AND Birth_Weight IS NULL AND Age_Group <> '0 to 17'

-- Deletes records for age groups that are not appopriate to newborn admission type
BEGIN TRANSACTION
DELETE FROM dbo.SPARC2019
WHERE Type_of_Admission  = 'newborn' AND Birth_Weight IS NULL AND Age_Group <> '0 to 17'
ROLLBACK TRANSACTION
```

Results Messages

ID	Hospital_Service_Area	Hospital_County	Operating_Certificate_Number	Permanent_Facility_Id	Facility_Name	Age_Group	Zip_Code_3_digits	Gender	Race	Ethnicity	Length_of_Stay	Type_of_Admission	Patient_Disposition	Discharge_Year	CCSR_Diagnosis_Code	CCSR_Diagnosis_Description	CCSR_Procedure_Code	CCSR_Procedure_Description	APR_DRG_Code	APR_DRG_Des

We have successfully transformed and cleaned our dataset!