**Staging Franchise Grandpad:**

-- Author: Abhijeet Das

-- Create date: 05/11/2020

-- Modified date: 05/12/2020

-- Description: Move data from raw GrandPad to Staging franchise\_Grandpad. Pulls data from

-- raw.GP\_grandmas\_hisc\_v4\_ADLS and Staging.Franchise.

-- Pulls Franchise\_Number, org\_name and major\_org.

-- Joins data with Staging.Franchise using Left OUTER Join

-- ==================================================================

CREATE PROCEDURE [Staging].[usp\_GP\_Populating\_FranchiseGP\_ADLS]

(@SessionTag uniqueidentifier= NULL)

AS

BEGIN

SET NOCOUNT ON;

Declare @time\_stmp datetime = SYSDATETIME()

DECLARE @StoredProcedureName VARCHAR(100) = '[Staging].[GP\_Populating\_FranchiseGP\_ADLS]';

DECLARE @ErrorLog varchar(max) = ''

IF (@SessionTag IS NULL)

SET @SessionTag = newid();

truncate table Staging.Franchise\_GrandPad;

BEGIN TRY

MERGE Staging.Franchise\_GrandPad as Target

USING

(

select distinct Source.Franchise\_number [Franchise\_Number],

Source.org\_name [Org\_Name],

Source.major\_org [Major\_Org],

'raw.GP\_grandmas\_hisc\_v4\_ADLS & Staging.Franchise' [Source\_string],

@SessionTag [Tag],

@time\_stmp [Date\_time],

F.Franchise\_Key [FK\_Franchise\_key]

From [raw].[GP\_grandmas\_hisc\_v4\_ADLS] source

LEFT OUTER JOIN Staging.Franchise F

ON F.Franchise\_Number =source.Franchise\_Number

WHERE source.modifier= 'real'

AND LEN(source.franchise\_number)=3

) as Source

(

[Franchise\_Number],

[Org\_Name],

[Major\_Org],

[Source\_string],

[Tag],

[Date\_time],

[FK\_Franchise\_key]

)

ON (Source.[Franchise\_Number] = Target.Franchise\_Number)

WHEN NOT MATCHED BY TARGET

THEN INSERT (

Franchise\_Number,

org\_name,

major\_org,

ETL\_Sources,

ETL\_Tag,

ETL\_Datetime,

ETL\_Datetime\_Modified,

ETL\_Tag\_Modified,

cc\_franchise\_key

)

VALUES

([Franchise\_Number],

[Org\_Name],

[Major\_Org],

[Source\_string],

[Tag],

[Date\_time],

[Date\_time],

[Tag],

[FK\_Franchise\_key]

);

END TRY

BEGIN CATCH

--Grab all of the error info, and return as a string using the dbo.fn\_ReturnErrorInfo() function

SELECT @ErrorLog = dbo.fn\_ReturnErrorInfo()

EXEC [dbo].[usp\_Insert\_Log\_Entry]

@StoredProcedureName,

'ERROR populating the Staging.Franchise\_GrandPad table from the [raw].[GP\_grandmas\_hisc\_v4\_ADLS] and table Staging.Franchise',

@@ROWCOUNT,

@SessionTag,

@@SERVERNAME,

'[HISC\_EDW]',

'ERROR',

@ErrorLog;

END CATCH

END

**Staging Client Grandpad:**

-- Author: Abhijeet Das

-- Create date: 05/16/2020

-- Modified date: 05/16/2020

-- Description: Move data from raw GrandPad to Staging Client\_GrandPad. Pulls data from

-- raw.GP\_grandmas\_hisc\_v4\_ADLS and Staging.Client.

-- Pulls the data Client\_alt\_key, Client\_key, middle name and date of birth .

-- Staging.Client using Inner Join

-- also adding the filter Is\_deactivated to keep the data optimum

-- ==================================================================

Create PROCEDURE [Staging].[usp\_GP\_Populating\_ClientGP\_ADLS]

(@SessionTag uniqueidentifier= NULL)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @StoredProcedureName VARCHAR(100) = '[Staging].[usp\_GP\_Populating\_ClientGP\_ADLS]';

DECLARE @ErrorLog varchar(max) = ''

Declare @time\_stmp datetime = SYSDATETIME()

IF (@SessionTag IS NULL)

SET @SessionTag = newid();

truncate table Staging.Client\_GrandPad;

BEGIN TRY

MERGE Staging.Client\_GrandPad as Target

USING

(

select distinct

Source.first\_name [First\_Name],

C.Middle\_Name [Middle\_Name ],

Source.last\_name [Last\_Name ],

C.Date\_of\_Birth [Date\_of\_Birth],

Source.modifier [modifier],

Source.status [user\_status],

Source.birthday [birthday],

Source.city [city],

Source.state [state],

Source.created [Created\_date],

source.user\_id\_public [user\_id\_public],

Source.franchise\_Number [franchise\_Number],

C.Client\_Key [staging\_client\_key],

C.Patient\_ID\_Alt [staging\_patient\_id],

'raw.GP\_grandmas\_hisc\_v4 & Staging.Client' [Source\_string],

@SessionTag [Tag],

@time\_stmp [Date\_time]

From [raw].[GP\_grandmas\_hisc\_v4\_ADLS] Source

Inner JOIN Staging.Client C

ON C.First\_Name= Source.first\_name

And C.Last\_Name= Source.last\_name

WHERE source.modifier = 'real'

AND LEN(source.franchise\_number)=3

AND C.Is\_deactivated = 'false'

) as Source

(

[First\_Name ],

[Middle\_Name ],

[Last\_Name],

[Date\_of\_Birth],

[modifier ],

[user\_status ],

[birthday],

[city ],

[state ],

[Created\_date ],

[user\_id\_public ],

[franchise\_Number],

[staging\_client\_key ],

[staging\_patient\_id ],

[Source\_string],

[Tag],

[Date\_time]

)

ON (Source.[Franchise\_Number] = Target.Franchise\_Number)

WHEN NOT MATCHED BY TARGET

THEN INSERT (

First\_Name ,

Middle\_Name ,

Last\_Name ,

Date\_of\_Birth,

modifier ,

user\_status ,

birthday,

city ,

state ,

Created\_date ,

user\_id\_public ,

franchise\_Number,

staging\_client\_key ,

staging\_patient\_id ,

ETL\_Sources,

ETL\_Tag,

ETL\_DateTime,

ETL\_Datetime\_Modified ,

ETL\_Tag\_Modified

)

VALUES

([First\_Name ],

[Middle\_Name ],

[Last\_Name],

[Date\_of\_Birth],

[modifier ],

[user\_status ],

[birthday],

[city ],

[state ],

[Created\_date ],

[user\_id\_public ],

[franchise\_Number],

[staging\_client\_key ],

[staging\_patient\_id ],

[Source\_string],

[Tag],

[Date\_time],

[Date\_time],

[Tag]);

END TRY

BEGIN CATCH

--Grab all of the error info, and return as a string using the dbo.fn\_ReturnErrorInfo() function

SELECT @ErrorLog = dbo.fn\_ReturnErrorInfo()

EXEC [dbo].[usp\_Insert\_Log\_Entry]

@StoredProcedureName,

'ERROR populating the Staging.Client\_GrandPad table from the [raw].[GP\_grandmas\_hisc\_v4\_ADLS] and Staging.Client table',

@@ROWCOUNT,

@SessionTag,

@@SERVERNAME,

'[HISC\_Staging]',

'ERROR',

@ErrorLog;

END CATCH

END