

# Data Analyst Cleaning Project

Facundo Cortez

# Introducción:

Aplicaremos Técnicas de Limpieza y Procesado de Datos

Nuestra base de datos se llama:  
**"NASHVILLE HOUSING DATA"**

Descripción:

UniqueID -> ID Único por Fila

ParcelID -> ID de la Parcela del Inmueble

LandUse -> Uso que se le da. Ej: FAMILIAR

PropertyAddress -> Dirección del Inmueble

SaleDate -> Fecha en la que se generó la venta del Inmueble

SalePrice -> Precio al cual se vendió el Inmueble

LegalReference -> Referencia Legal

SoldAsVacant -> Vendido como vacante

OwnerName -> Nombre del Propietario

OwnerAddress -> Dirección del Propietario

Acreage -> Valor de Acres

TaxDistrict -> Impuesto del Distrito

LandValue -> Valor de la tierra

BuildingValue -> Valor de la construcción

TotalValue -> Valor Total

YearBuilt -> Año de construcción

Bedrooms -> Habitaciones

Fullbath -> Baño completo

Halfbath -> Medio baño

# Limpieza de Datos en SQL

```
SELECT *
FROM [dbo].[NashvilleHousing]
```

|    | UniqueID | ParcelID        | LandUse         | PropertyAddress             | SaleDate                | SalePrice | LegalReference   | SoldAs/acant | OwnerName                                    | OwnerAddress                    | Acreage | TaxDistrict             | Land/sal |
|----|----------|-----------------|-----------------|-----------------------------|-------------------------|-----------|------------------|--------------|--|---------------------------------|---------|-------------------------|----------|
| 1  | 24022    | 082 03 0 158.00 | SINGLE FAMILY   | 213 CLEVELAND ST, NASHVILLE | 2014-11-07 00:00:00.000 | 177000    | 20141110-0103612 | No           | DANIEL CLINTON RYAN                          | 213 CLEVELAND ST, NASHVILLE, TN | 0.22    | URBAN SERVICES DISTRICT | 27000    |
| 2  | 11770    | 082 03 0 160.00 | VACANT RES LAND | 908 STOCKELL ST, NASHVILLE  | 2014-01-30 00:00:00.000 | 292000    | 20140204-0009738 | No           | LANCASTER, MARK A. & ANGELA D.               | 908 STOCKELL ST, NASHVILLE, TN  | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 3  | 5947     | 082 03 0 163.00 | SINGLE FAMILY   | 914 STOCKELL ST, NASHVILLE  | 2013-07-10 00:00:00.000 | 82000     | 20130711-0071726 | No           | CHEATHAM, HERBERT L.                         | 914 STOCKELL ST, NASHVILLE, TN  | 0.21    | URBAN SERVICES DISTRICT | 27000    |
| 4  | 11771    | 082 03 0 170.00 | SINGLE FAMILY   | 1010 STOCKELL ST, NASHVILLE | 2014-01-28 00:00:00.000 | 227500    | 20140130-0008515 | No           | DEFGIA, JOSEPH PAUL                          | 1010 STOCKELL ST, NASHVILLE, TN | 0.2     | URBAN SERVICES DISTRICT | 27000    |
| 5  | 7153     | 082 03 0 177.00 | SINGLE FAMILY   | 1104 STOCKELL ST, NASHVILLE | 2013-08-30 00:00:00.000 | 234500    | 20130909-0094860 | No           | COLLISAN, JASON                              | 1104 STOCKELL ST, NASHVILLE, TN | 0.2     | URBAN SERVICES DISTRICT | 27000    |
| 6  | 4678     | 082 03 0 182.00 | SINGLE FAMILY   | 1114 STOCKELL ST, NASHVILLE | 2013-06-14 00:00:00.000 | 215000    | 20130617-0061553 | No           | JEFFERSON, JONATHAN RAYMOND                  | 1114 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 7  | 4679     | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2013-06-17 00:00:00.000 | 69000     | 20130621-0063866 | No           | COOK, CHRISTOPHER J. & IZMAYLOVA, EUGENIA A. | 1118 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 8  | 18553    | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2014-07-31 00:00:00.000 | 330000    | 20140812-0072633 | No           | COOK, CHRISTOPHER J. & IZMAYLOVA, EUGENIA A. | 1118 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 9  | 24023    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2014-11-19 00:00:00.000 | 140000    | 20141121-0107304 | No           | CHADWICK, WILLIAM & PUMPHREY, TONI           | 1120 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 10 | 35003    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2015-07-31 00:00:00.000 | 375000    | 20150804-0077449 | No           | CHADWICK, WILLIAM & PUMPHREY, TONI           | 1120 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 11 | 28315    | 082 03 0 191.00 | SINGLE FAMILY   | 1117 STOCKELL ST, NASHVILLE | 2015-03-26 00:00:00.000 | 205000    | 20150327-0026523 | No           | URBAN JAMES P. & SARA E.                     | 1117 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 12 | 11772    | 082 03 0 194.00 | SINGLE FAMILY   | 1111 STOCKELL ST, NASHVILLE | 2014-01-14 00:00:00.000 | 82000     | 20140117-0004556 | No           | NOLAND, KEVIN                                | 1111 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 13 | 56224    | 082 03 0 199.00 | SINGLE FAMILY   | 1101 STOCKELL ST, NASHVILLE | 2016-10-06 00:00:00.000 | 100000    | 20161007-0106563 | No           | CLIFFVIEW PROPERTIES, LLC                    | 1101 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 14 | 50833    | 082 03 0 200.00 | SINGLE FAMILY   | 1019 STOCKELL ST, NASHVILLE | 2016-06-06 00:00:00.000 | 192500    | 20160607-0057477 | No           | MOORE DEVELOPMENT, GP                        | 1019 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 15 | 9021     | 082 03 0 203.00 | SINGLE FAMILY   | 1013 STOCKELL ST, NASHVILLE | 2013-10-15 00:00:00.000 | 57000     | 20131018-0109194 | No           | DEVRIES, ANDREW & RACHEL                     | 1013 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 16 | 33251    | 082 03 0 203.00 | SINGLE FAMILY   | 1013 STOCKELL ST, NASHVILLE | 2015-06-23 00:00:00.000 | 354050    | 20150625-0060991 | No           | DEVRIES, ANDREW & RACHEL                     | 1013 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 17 | 52133    | 082 03 0 203.00 | SINGLE FAMILY   | 1013 STOCKELL ST, NASHVILLE | 2016-07-15 00:00:00.000 | 354050    | 20160715-0060991 | No           | DEVRIES, ANDREW & RACHEL                     | 1013 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 18 | 8077     | 082 03 0 205.00 | SINGLE FAMILY   | 1009 STOCKELL ST, NASHVILLE | 2013-09-19 00:00:00.000 | 50000     | 20130927-0101683 | No           | SCHAFER, ALLISON J.                          | 1009 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 19 | 12497    | 082 03 0 205.00 | SINGLE FAMILY   | 1009 STOCKELL ST, NASHVILLE | 2014-02-13 00:00:00.000 | 253980    | 20140218-0013652 | No           | SCHAFER, ALLISON J.                          | 1009 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 20 | 11773    | 082 03 0 206.00 | SINGLE FAMILY   | 1007 STOCKELL ST, NASHVILLE | 2014-01-17 00:00:00.000 | 160000    | 20140122-0006150 | No           | LALACONA, JASON & REEBLE, MINDY              | 1007 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 21 | 38273    | 082 03 0 206.00 | SINGLE FAMILY   | 1007 STOCKELL ST, NASHVILLE | 2015-09-01 00:00:00.000 | 212000    | 20150903-0099949 | No           | LALACONA, JASON & REEBLE, MINDY              | 1007 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |
| 22 | 17062    | 082 03 0 208.00 | SINGLE FAMILY   | 1003 STOCKELL ST, NASHVILLE | 2014-06-20 00:00:00.000 | 80000     | 20140626-0056124 | No           | FRANKLIN, ROBERT J., JR.                     | 1003 STOCKELL ST, NASHVILLE, TN | 0.19    | URBAN SERVICES DISTRICT | 27000    |

-----  
-----

```
/*
-- Estandarizacion de formato de Fechas
*/
```

```
SELECT *
FROM dbo.NashvilleHousing
```

-- Podremos apreciar que tenemos un formato que nos muestra la HORA, lo cual no nos interesa en este caso.

```
SELECT SaleDate, CONVERT(Date,SaleDate) -- Vamos a utilizar CONVERT
para convertir nuestro dato a solo Fecha.
FROM dbo.NashvilleHousing
```

-- Ahora podemos apreciar nuestros 2 campos (SaleDate y la Nueva Columna Convertida)

SQL\_DirtyDataset.s...VLDC293\Facu (59)) SQLQuery2.sql - D...VLDC293\Facu (62))

```
*/
SELECT *
FROM dbo.NashvilleHousing
-- Podemos apreciar que tenemos un formato que nos muestra la HORA, lo cual no nos interesa en este caso.
SELECT SaleDate, CONVERT(Date,SaleDate) -- Vamos a utilizar CONVERT para convertir nuestro dato a solo Fecha.
FROM dbo.NashvilleHousing
-- Ahora podemos apreciar nuestros 2 campos (SaleDate y la Nueva Columna Convertida)
```

100 %

Results Messages

|    | UniquelD | ParcelID        | LandUse         | PropertyAddress             | SaleDate                | SalePrice | LegalReference   | SoldAsVacant | OwnerName    |
|----|----------|-----------------|-----------------|-----------------------------|-------------------------|-----------|------------------|--------------|--------------|
| 1  | 24022    | 082 03 0 158.00 | SINGLE FAMILY   | 213 CLEVELAND ST, NASHVILLE | 2014-11-07 00:00:00.000 | 177000    | 20141110-0103612 | No           | DANIEL, CLIF |
| 2  | 11770    | 082 03 0 160.00 | VACANT RES LAND | 908 STOCKELL ST, NASHVILLE  | 2014-01-30 00:00:00.000 | 292000    | 20140204-0009738 | No           | LANCASTER    |
| 3  | 5947     | 082 03 0 163.00 | SINGLE FAMILY   | 914 STOCKELL ST, NASHVILLE  | 2013-07-10 00:00:00.000 | 82000     | 20130711-0071726 | No           | CHEATHAM,    |
| 4  | 11771    | 082 03 0 170.00 | SINGLE FAMILY   | 1010 STOCKELL ST, NASHVILLE | 2014-01-28 00:00:00.000 | 227500    | 20140130-0008515 | No           | DEFIGLIA, JC |
| 5  | 7153     | 082 03 0 177.00 | SINGLE FAMILY   | 1104 STOCKELL ST, NASHVILLE | 2013-08-30 00:00:00.000 | 224500    | 20130909-0094860 | No           | COLLIGAN, J  |
| 6  | 4678     | 082 03 0 182.00 | SINGLE FAMILY   | 1114 STOCKELL ST, NASHVILLE | 2013-06-14 00:00:00.000 | 215000    | 20130617-0061553 | No           | JEFFERSON    |
| 7  | 4679     | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2013-06-17 00:00:00.000 | 69000     | 20130621-0063866 | No           | COOK, CHRI   |
| 8  | 18553    | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2014-07-31 00:00:00.000 | 330000    | 20140812-0072633 | No           | COOK, CHRI   |
| 9  | 24023    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2014-11-19 00:00:00.000 | 140000    | 20141121-0107304 | No           | CHADWICK,    |
| 10 | 35003    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2015-07-31 00:00:00.000 | 375000    | 20150804-0077449 | No           | CHADWICK,    |
| 11 | 28315    | 082 03 0 191.00 | SINGLE FAMILY   | 1117 STOCKELL ST, NASHVILLE | 2015-03-26 00:00:00.000 | 205000    | 20150327-0026523 | No           | URBAN JAM    |

| SaleDate                | (No column name) |
|-------------------------|------------------|
| 2014-11-07 00:00:00.000 | 2014-11-07       |
| 2014-01-30 00:00:00.000 | 2014-01-30       |
| 2013-07-10 00:00:00.000 | 2013-07-10       |
| 2014-01-28 00:00:00.000 | 2014-01-28       |
| 2013-08-30 00:00:00.000 | 2013-08-30       |
| 2013-06-14 00:00:00.000 | 2013-06-14       |
| 2013-06-17 00:00:00.000 | 2013-06-17       |
| 2014-07-31 00:00:00.000 | 2014-07-31       |

```
-- Adherimos una nueva Columna llamada SaleDate2

ALTER TABLE NashvilleHousing
ADD SaleDate2 Date

-- Y a esa misma columna le proporcionamos los Datos de SaleDate
pero convertidos

UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET SaleDate2 = CONVERT(Date,SaleDate)

-- Visualizamos nuestros campos
```

SQL\_DirtyDataset.s...VLDC293\Facu (59)) SQLQuery2.sql - D...VLDC293\Facu (62))

```
-- Ahora podemos apreciar nuestros 2 campos (SaleDate y la Nueva Columna Convertida)
-- Adherimos una nueva Columna llamada SaleDate2
ALTER TABLE NashvilleHousing
ADD SaleDate2 Date
-- Y a esa misma columna le proporcionamos los Datos de SaleDate pero convertidos
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo campo
SET SaleDate2 = CONVERT(Date,SaleDate)
-- Visualizamos nuestros campos
SELECT SaleDate , SaleDate2
FROM dbo.NashvilleHousing
```

100 %

Results Messages

|    | SaleDate                | SaleDate2  |
|----|-------------------------|------------|
| 1  | 2014-11-07 00:00:00.000 | 2014-11-07 |
| 2  | 2014-01-30 00:00:00.000 | 2014-01-30 |
| 3  | 2013-07-10 00:00:00.000 | 2013-07-10 |
| 4  | 2014-01-28 00:00:00.000 | 2014-01-28 |
| 5  | 2013-08-30 00:00:00.000 | 2013-08-30 |
| 6  | 2013-06-14 00:00:00.000 | 2013-06-14 |
| 7  | 2013-06-17 00:00:00.000 | 2013-06-17 |
| 8  | 2014-07-31 00:00:00.000 | 2014-07-31 |
| 9  | 2014-11-19 00:00:00.000 | 2014-11-19 |
| 10 | 2015-07-31 00:00:00.000 | 2015-07-31 |
| 11 | 2015-03-26 00:00:00.000 | 2015-03-26 |
| 12 | 2014-01-14 00:00:00.000 | 2014-01-14 |
| 13 | 2016-10-06 00:00:00.000 | 2016-10-06 |
| 14 | 2016-06-06 00:00:00.000 | 2016-06-06 |
| 15 | 2013-10-15 00:00:00.000 | 2013-10-15 |
| 16 | 2015-06-23 00:00:00.000 | 2015-06-23 |

```
SELECT SaleDate , SaleDate2
FROM dbo.NashvilleHousing
```

-- DROP COLUMN

/\*  
\*/

-----  
-----

-- Actualizamos el nombre apropiado a las Direcciones de las  
Propiedades

-- Buscaremos valores NULOS

```
SELECT PropertyAddress  
FROM dbo.NashvilleHousing  
WHERE PropertyAddress is null
```

The screenshot shows a SQL Server Enterprise Manager interface. At the top, there are two tabs: 'SQL\_DirtyDataset.s...VLDC293\Facu (59))' and 'SQLQuery2.sql - D...VLDC293\Facu (62))'. The active tab is 'SQLQuery2.sql'. The query editor displays the following SQL code:

```
-- DROP COLUMN  
  
/*  
*/  
  
-----  
-----  
  
-- Actualizamos el nombre apropiado a las Direcciones de las Propiedades  
  
-- Buscaremos valores NULOS  
  
SELECT PropertyAddress  
FROM dbo.NashvilleHousing  
WHERE PropertyAddress is null  
  
-- Vamos a checkear el ID de la Parcela "ParcelID"  
  
SELECT *
```

Below the query editor, there is a 'Results' tab and a 'Messages' tab. The 'Results' tab is active, showing a table with the following data:

|    | PropertyAddress |
|----|-----------------|
| 12 | NULL            |
| 13 | NULL            |
| 14 | NULL            |
| 15 | NULL            |
| 16 | NULL            |
| 17 | NULL            |
| 18 | NULL            |
| 19 | NULL            |
| 20 | NULL            |
| 21 | NULL            |
| 22 | NULL            |
| 23 | NULL            |
| 24 | NULL            |
| 25 | NULL            |
| 26 | NULL            |
| 27 | NULL            |

-- Vamos a checkear el ID de la Parcela "ParcelID"

```
SELECT *
FROM dbo.NashvilleHousing
ORDER BY ParcelID
```

-- Encontramos que tenemos valores repetidos y tienen el mismo Property/Address  
-- Previamente sabíamos que nos íbamos a encontrar con estos datos porque habíamos hecho el Analisis Exploratorio en Google Colab  
-- Con Pandas Profiling

-- Vamos a checkear el ID de la Parcela "ParcelID"

SELECT \*

FROM dbo.NashvilleHousing

ORDER BY ParcelID

-- Encontramos que tenemos valores repetidos y tienen el mismo Property/Address

-- Previamente sabíamos que nos íbamos a encontrar con estos datos porque habíamos hecho el Analisis Exploratorio en Google Colab

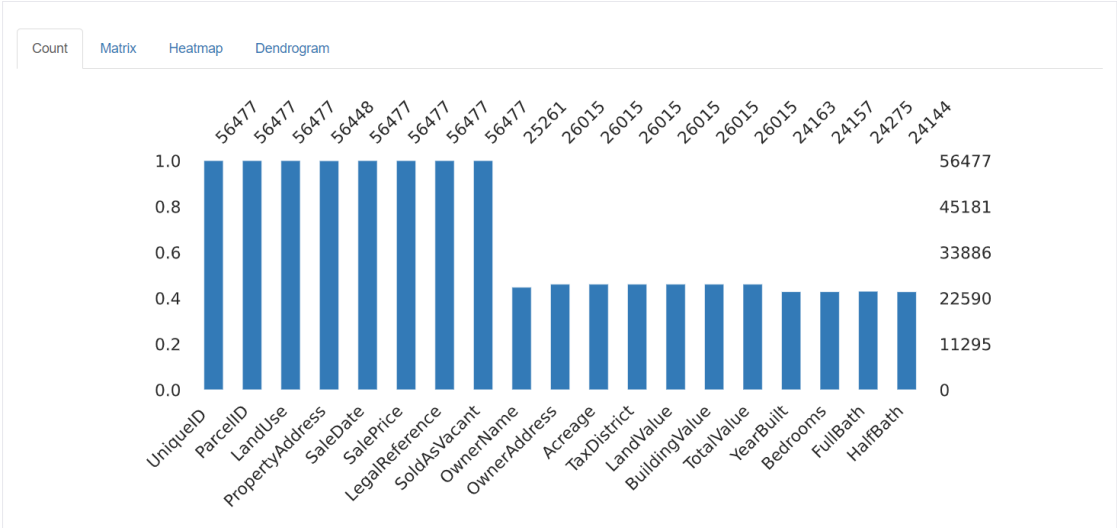
-- Con Pandas Profiling

|    | UniqueID | ParcelID        | LandUse       | PropertyAddress                   | SaleDate                | SalePrice | LegalReference   | SoldAs/Vacant | OwnerName                         |
|----|----------|-----------------|---------------|-----------------------------------|-------------------------|-----------|------------------|---------------|-----------------------------------|
| 34 | 15542    | 012 03 0 012.00 | SINGLE FAMILY | 600 SUNNYSLOPE CT. GOODLETTSVILLE | 2014-05-14 00:00:00.000 | 205900    | 20140529-0045974 | No            | SKAGGS, BROCK A. & DENISE         |
| 35 | 54584    | 012 03 0 013.00 | SINGLE FAMILY | 604 SUNNYSLOPE CT. GOODLETTSVILLE | 2016-09-23 00:00:00.000 | 228000    | 20160927-0101636 | No            | RED THUNDER, LLC                  |
| 36 | 1207     | 012 03 0 018.00 | SINGLE FAMILY | 609 SUNNYSLOPE CT. GOODLETTSVILLE | 2013-03-15 00:00:00.000 | 160000    | 20130319-0026986 | No            | O'CONNELL, JAMES L. & VANESSA R.  |
| 37 | 667      | 012 03 0 023.00 | SINGLE FAMILY | 621 GAYLEMORE DR. GOODLETTSVILLE  | 2013-02-08 00:00:00.000 | 129900    | 20130215-0015687 | No            | TUDERS, KEVIN                     |
| 38 | 34752    | 012 03 0 025.00 | SINGLE FAMILY | 613 GAYLEMORE DR. GOODLETTSVILLE  | 2015-07-17 00:00:00.000 | 230000    | 20150721-0071254 | No            | BOISE, LINDSEY J. & BURDEN, GINGE |
| 39 | 39426    | 015 14 0 011.00 | SINGLE FAMILY | 3128 UNION HILL RD. JOELTON       | 2015-10-01 00:00:00.000 | 90000     | 20151006-0101964 | No            | IBARRA-ROMERO, GREGORIO & REG     |
| 40 | 48717    | 015 14 0 045.00 | SINGLE FAMILY | 7515 GARY RD. JOELTON             | 2016-05-27 00:00:00.000 | 160000    | 20160606-0056507 | No            | PORTER, BRIAN KELLY               |
| 41 | 53117    | 015 14 0 053.00 | SINGLE FAMILY | 7620 GERALD DR. JOELTON           | 2016-08-08 00:00:00.000 | 195900    | 20160810-0083381 | No            | TEMPLE, LORENE                    |
| 42 | 39427    | 015 14 0 054.00 | SINGLE FAMILY | 7624 GERALD DR. JOELTON           | 2015-10-09 00:00:00.000 | 122431    | 20151013-0103951 | No            | STEWART, DONNA D.                 |
| 43 | 5903     | 015 14 0 055.00 | SINGLE FAMILY | 7628 GERALD DR. JOELTON           | 2013-07-12 00:00:00.000 | 119900    | 20130718-0074201 | No            | PROULX, JENNIFER L. & JESSICA     |
| 44 | 6993     | 015 14 0 060.00 | SINGLE FAMILY | 3113 MILLIKEN DR. JOELTON         | 2013-08-20 00:00:00.000 | 145000    | 20130821-0088382 | No            | SMITH, HOLLI NICHOLE              |
| 45 | 43071    | 015 14 0 060.00 | SINGLE FAMILY | 3113 MILLIKEN DR. JOELTON         | 2016-01-15 00:00:00.000 | 160000    | 20160119-0004923 | No            | SMITH, HOLLI NICHOLE              |
| 46 | 45286    | 015 14 0 061.00 | SINGLE FAMILY | 3117 MILLIKEN DR. JOELTON         | 2016-03-18 00:00:00.000 | 169900    | 20160413-0035454 | No            | DISMANG, JEREMY                   |
| 47 | 3267     | 015 14 0 071.00 | SINGLE FAMILY | 3107 MARGIE DR. JOELTON           | 2013-05-10 00:00:00.000 | 103500    | 20130513-0047997 | No            | DOWNES, JESSICA                   |
| 48 | 24753    | 015 14 0 075.00 | SINGLE FAMILY | 3124 MARGIE DR. JOELTON           | 2015-07-31 00:00:00.000 | 164000    | 20150803-0076737 | No            | MIDBY, SHAMI                      |

|                  |              |         |
|------------------|--------------|---------|
| PropertyAddress  | Distinct     | 45068   |
| Categorical      | Distinct (%) | 79.8%   |
| HIGH CARDINALITY | Missing      | 29      |
|                  | Missing (%)  | 0.1%    |
|                  | Memory size  | 4.6 MiB |

1

# Missing values





```
-- Entonces aplicando un JOIN en la misma Tabla
-- Estamos trayendo los mismos campos pero con la distincion del
UniqueID
```

```
SELECT A.ParcelID , A.PropertyAddress , B.ParcelID,
B.PropertyAddress
FROM dbo.NashvilleHousing A
JOIN dbo.NashvilleHousing B
  ON A.ParcelID = B.ParcelID
  AND A.UniqueID <> B.UniqueID
WHERE A.PropertyAddress is null
```

```
-- Encontramos que tenemos en nuestra columna de
"Property/Address" los valores NULL
```

```
-- Entonces aplicando un JOIN en la misma Tabla
-- Estamos trayendo los mismos campos pero con la distincion del UniqueID

SELECT A.ParcelID , A.PropertyAddress , B.ParcelID, B.PropertyAddress
FROM dbo.NashvilleHousing A
JOIN dbo.NashvilleHousing B
  ON A.ParcelID = B.ParcelID
  AND A.UniqueID <> B.UniqueID
WHERE A.PropertyAddress is null

-- Encontramos que tenemos en nuestra columna de "Property/Address" los valores NULL

-- Vamos a agregar una nueva columna en la que vamos a sustituir ese de la Tabla A , con los valores de la Tabla B (Property/Address)
```

100 %

Results Messages

|    | ParcelID         | PropertyAddress | ParcelID         | PropertyAddress                   |
|----|------------------|-----------------|------------------|-----------------------------------|
| 1  | 025 07 0 031.00  | NULL            | 025 07 0 031.00  | 410 ROSEHILL CT, GOODLETTSVILLE   |
| 2  | 026 01 0 069.00  | NULL            | 026 01 0 069.00  | 141 TWO MILE PIKE, GOODLETTSVILLE |
| 3  | 026 05 0 017.00  | NULL            | 026 05 0 017.00  | 208 EAST AVE, GOODLETTSVILLE      |
| 4  | 026 06 0A 038.00 | NULL            | 026 06 0A 038.00 | 109 CANTON CT, GOODLETTSVILLE     |
| 5  | 033 06 0 041.00  | NULL            | 033 06 0 041.00  | 1129 CAMPBELL RD, GOODLETTSVILLE  |
| 6  | 033 06 0A 002.00 | NULL            | 033 06 0A 002.00 | 1116 CAMPBELL RD, GOODLETTSVILLE  |
| 7  | 033 15 0 123.00  | NULL            | 033 15 0 123.00  | 438 W CAMPBELL RD, GOODLETTSVILLE |
| 8  | 042 13 0 075.00  | NULL            | 042 13 0 075.00  | 222 FOXBORO DR, MADISON           |
| 9  | 043 04 0 014.00  | NULL            | 043 04 0 014.00  | 112 HILLER DR, OLD HICKORY        |
| 10 | 043 09 0 074.00  | NULL            | 043 09 0 074.00  | 213 B LOVELL ST, MADISON          |
| 11 | 043 13 0 308.00  | NULL            | 043 13 0 308.00  | 224 HICKORY ST, MADISON           |
| 12 | 034 03 0 059.00  | NULL            | 034 03 0 059.00  | 2117 PAULA DR, MADISON            |
| 13 | 034 03 0 059.00  | NULL            | 034 03 0 059.00  | 2117 PAULA DR, MADISON            |
| 14 | 034 07 0B 015.00 | NULL            | 034 07 0B 015.00 | 2524 VAL MARIE DR, MADISON        |
| 15 | 034 07 0B 015.00 | NULL            | 034 07 0B 015.00 | 2524 VAL MARIE DR, MADISON        |
| 16 | 034 07 0B 015.00 | NULL            | 034 07 0B 015.00 | 2524 VAL MARIE DR, MADISON        |

-- Vamos a agregar una nueva columna en la que vamos a sustituir ese de la Tabla A , con los valores de la Tabla B (Property/Address)

```
-- CODIGO -> "PropertyAddress is NULL"
SELECT A.ParcelID , A.PropertyAddress , B.ParcelID,
B.PropertyAddress, ISNULL (A.PropertyAddress , B.PropertyAddress)
FROM dbo.NashvilleHousing A
JOIN dbo.NashvilleHousing B
ON A.ParcelID = B.ParcelID
AND A.UniqueID <> B.UniqueID
WHERE A.PropertyAddress is null
```

-- Vamos a agregar una nueva columna en la que vamos a sustituir ese de la Tabla A , con los valores de la Tabla B (Property/Address)

-- CODIGO -> "PropertyAddress is NULL"

SELECT A.ParcelID , A.PropertyAddress , B.ParcelID, B.PropertyAddress, ISNULL (A.PropertyAddress , B.PropertyAddress)

FROM dbo.NashvilleHousing A

JOIN dbo.NashvilleHousing B

ON A.ParcelID = B.ParcelID

AND A.UniqueID <> B.UniqueID

WHERE A.PropertyAddress is null

-- Perfecto ahora podremos crear un UPDATE

100 %

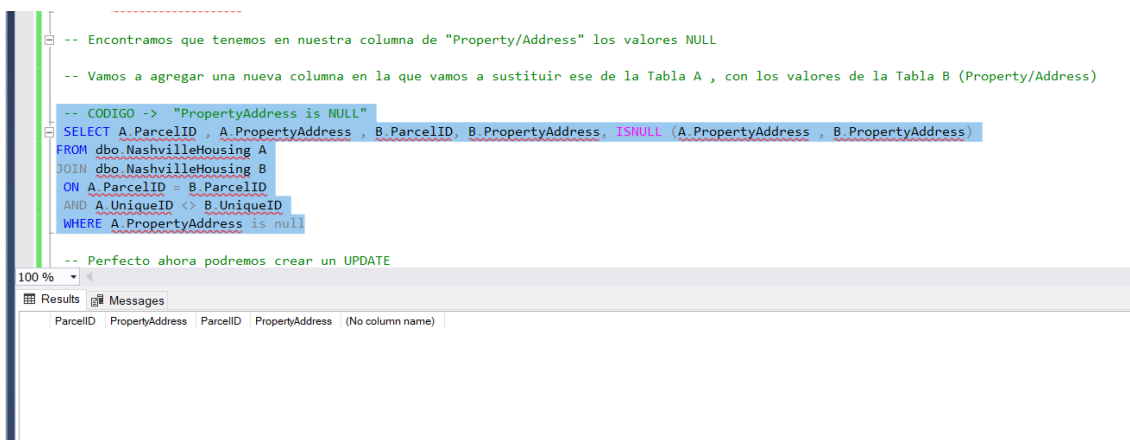
ResultsMessages

|    | ParcelID         | PropertyAddress | ParcelID         | PropertyAddress                   | (No column name)                  |
|----|------------------|-----------------|------------------|-----------------------------------|-----------------------------------|
| 1  | 025 07 0 031.00  | NULL            | 025 07 0 031.00  | 410 ROSEHILL CT, GOODLETTSVILLE   | 410 ROSEHILL CT, GOODLETTSVILLE   |
| 2  | 026 01 0 069.00  | NULL            | 026 01 0 069.00  | 141 TWO MILE PIKE, GOODLETTSVILLE | 141 TWO MILE PIKE, GOODLETTSVILLE |
| 3  | 026 05 0 017.00  | NULL            | 026 05 0 017.00  | 208 EAST AVE, GOODLETTSVILLE      | 208 EAST AVE, GOODLETTSVILLE      |
| 4  | 026 06 0A 038.00 | NULL            | 026 06 0A 038.00 | 109 CANTON CT, GOODLETTSVILLE     | 109 CANTON CT, GOODLETTSVILLE     |
| 5  | 033 06 0 041.00  | NULL            | 033 06 0 041.00  | 1129 CAMPBELL RD, GOODLETTSVILLE  | 1129 CAMPBELL RD, GOODLETTSVILLE  |
| 6  | 033 06 0A 002.00 | NULL            | 033 06 0A 002.00 | 1116 CAMPBELL RD, GOODLETTSVILLE  | 1116 CAMPBELL RD, GOODLETTSVILLE  |
| 7  | 033 15 0 123.00  | NULL            | 033 15 0 123.00  | 438 W CAMPBELL RD, GOODLETTSVILLE | 438 W CAMPBELL RD, GOODLETTSVILLE |
| 8  | 042 13 0 075.00  | NULL            | 042 13 0 075.00  | 222 FOXBORO DR, MADISON           | 222 FOXBORO DR, MADISON           |
| 9  | 043 04 0 014.00  | NULL            | 043 04 0 014.00  | 112 HILLER DR, OLD HICKORY        | 112 HILLER DR, OLD HICKORY        |
| 10 | 043 09 0 074.00  | NULL            | 043 09 0 074.00  | 213 B LOVELL ST, MADISON          | 213 B LOVELL ST, MADISON          |
| 11 | 043 13 0 308.00  | NULL            | 043 13 0 308.00  | 224 HICKORY ST, MADISON           | 224 HICKORY ST, MADISON           |
| 12 | 034 03 0 059.00  | NULL            | 034 03 0 059.00  | 2117 PAULA DR, MADISON            | 2117 PAULA DR, MADISON            |
| 13 | 034 03 0 059.00  | NULL            | 034 03 0 059.00  | 2117 PAULA DR, MADISON            | 2117 PAULA DR, MADISON            |
| 14 | 034 07 0B 015.00 | NULL            | 034 07 0B 015.00 | 2524 VAL MARIE DR, MADISON        | 2524 VAL MARIE DR, MADISON        |

-- Perfecto ahora podremos crear un UPDATE

```
UPDATE A
SET PropertyAddress = ISNULL (A.PropertyAddress ,
B.PropertyAddress)
FROM dbo.NashvilleHousing A
JOIN dbo.NashvilleHousing B
ON A.ParcelID = B.ParcelID
AND A.UniqueID <> B.UniqueID
WHERE A.PropertyAddress is null
```

-- Si volvemos a correr el CODIGO -> "PropertyAddress is NULL" .  
Veremos que no nos aparece ningun Valor Null ( Tabla vacía)



The screenshot shows a SQL Server Enterprise Manager interface. The top pane displays a SQL query with several comments in Spanish. The query is a SELECT statement that joins two instances of the 'NashvilleHousing' table (A and B) based on 'ParcelID' and 'UniqueID', filtering for rows where 'PropertyAddress' is null in table A. The query uses the 'ISNULL' function to replace null values with values from table B. The bottom pane shows the 'Results' tab, which is currently empty, indicating that no rows were returned by the query. The 'Messages' tab is also visible but empty.

```
-- Encontramos que tenemos en nuestra columna de "Property/Address" los valores NULL
-- Vamos a agregar una nueva columna en la que vamos a sustituir ese de la Tabla A , con los valores de la Tabla B (Property/Address)
-- CODIGO -> "PropertyAddress is NULL"
SELECT A.ParcelID , A.PropertyAddress , B.ParcelID, B.PropertyAddress, ISNULL (A.PropertyAddress , B.PropertyAddress)
FROM dbo.NashvilleHousing A
JOIN dbo.NashvilleHousing B
ON A.ParcelID = B.ParcelID
AND A.UniqueID <> B.UniqueID
WHERE A.PropertyAddress is null
-- Perfecto ahora podremos crear un UPDATE
```

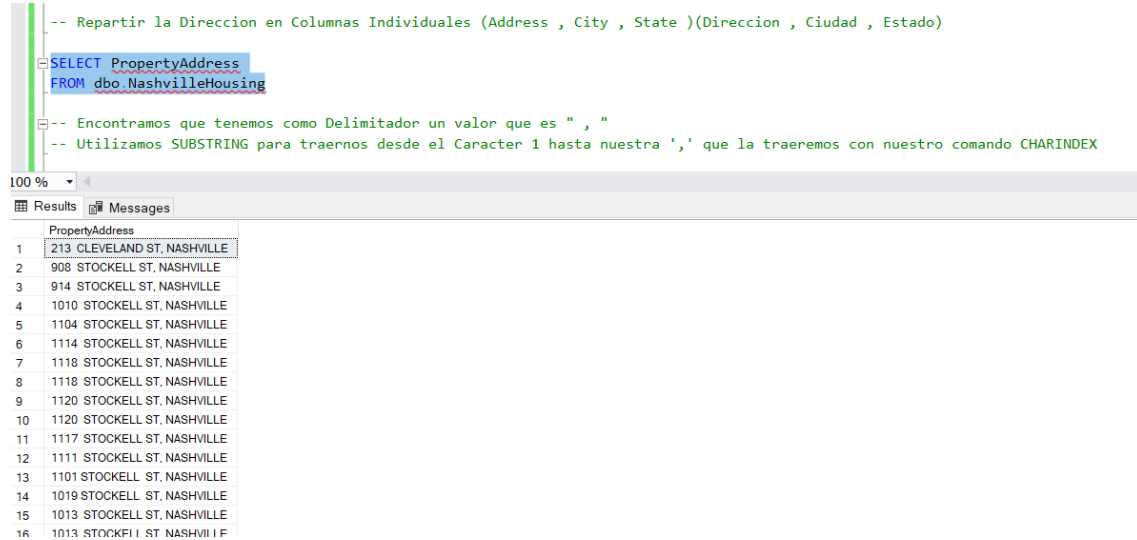
100 %

Results Messages

| ParcelID | PropertyAddress | ParcelID | PropertyAddress | (No column name) |
|----------|-----------------|----------|-----------------|------------------|
|----------|-----------------|----------|-----------------|------------------|

```
-- Repartir la Direccion en Columnas Individuales (Address , City , State )(Direccion , Ciudad , Estado)
```

```
SELECT PropertyAddress  
FROM dbo.NashvilleHousing
```



The screenshot shows a SQL query window with the following text:

```
-- Repartir la Direccion en Columnas Individuales (Address , City , State )(Direccion , Ciudad , Estado)  
  
SELECT PropertyAddress  
FROM dbo.NashvilleHousing  
  
-- Encontramos que tenemos como Delimitador un valor que es " , "  
-- Utilizamos SUBSTRING para traernos desde el Caracter 1 hasta nuestra ',' que la traeremos con nuestro comando CHARINDEX
```

Below the query window, the 'Results' tab is active, displaying a table with the following data:

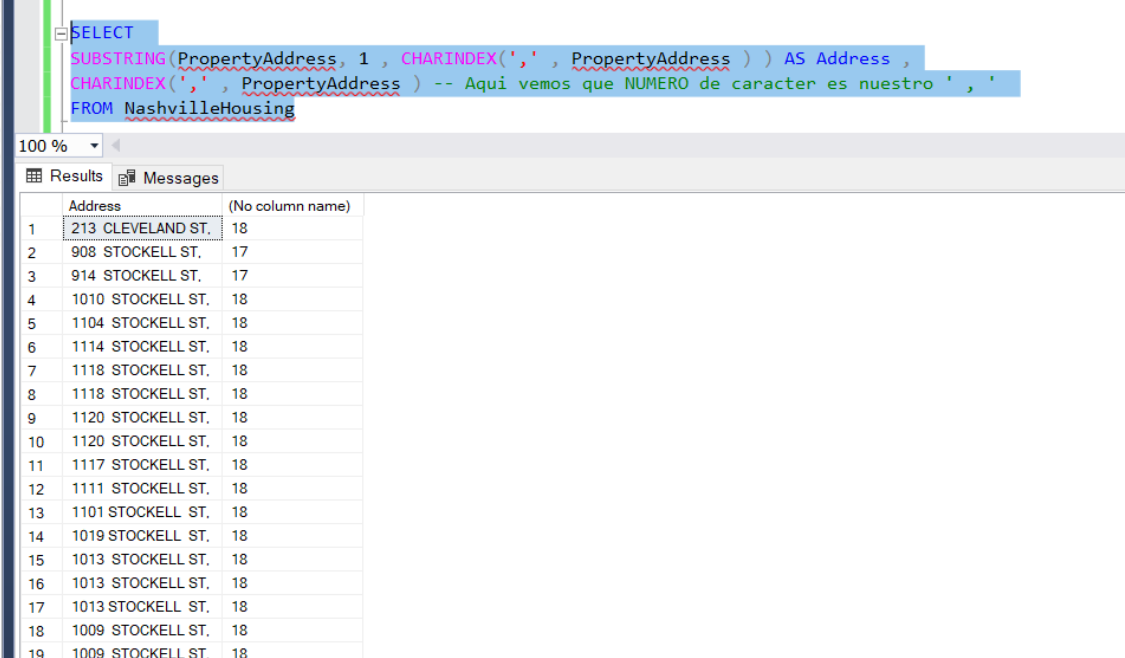
|    | PropertyAddress             |
|----|-----------------------------|
| 1  | 213 CLEVELAND ST, NASHVILLE |
| 2  | 908 STOCKELL ST, NASHVILLE  |
| 3  | 914 STOCKELL ST, NASHVILLE  |
| 4  | 1010 STOCKELL ST, NASHVILLE |
| 5  | 1104 STOCKELL ST, NASHVILLE |
| 6  | 1114 STOCKELL ST, NASHVILLE |
| 7  | 1118 STOCKELL ST, NASHVILLE |
| 8  | 1118 STOCKELL ST, NASHVILLE |
| 9  | 1120 STOCKELL ST, NASHVILLE |
| 10 | 1120 STOCKELL ST, NASHVILLE |
| 11 | 1117 STOCKELL ST, NASHVILLE |
| 12 | 1111 STOCKELL ST, NASHVILLE |
| 13 | 1101 STOCKELL ST, NASHVILLE |
| 14 | 1019 STOCKELL ST, NASHVILLE |
| 15 | 1013 STOCKELL ST, NASHVILLE |
| 16 | 1013 STOCKELL ST, NASHVILLE |

```
-- Encontramos que tenemos como Delimitador un valor que es " , "  
-- Utilizamos SUBSTRING para traernos desde el Caracter 1 hasta  
nuestra ',' que la traeremos con nuestro comando CHARINDEX
```

```

SELECT
SUBSTRING(PropertyAddress, 1 , CHARINDEX(',', PropertyAddress) )
AS Address ,
CHARINDEX(',', PropertyAddress) -- Aqui vemos que NUMERO de
caracter es nuestro ' , '
FROM NashvilleHousing

```



100 %

Results Messages

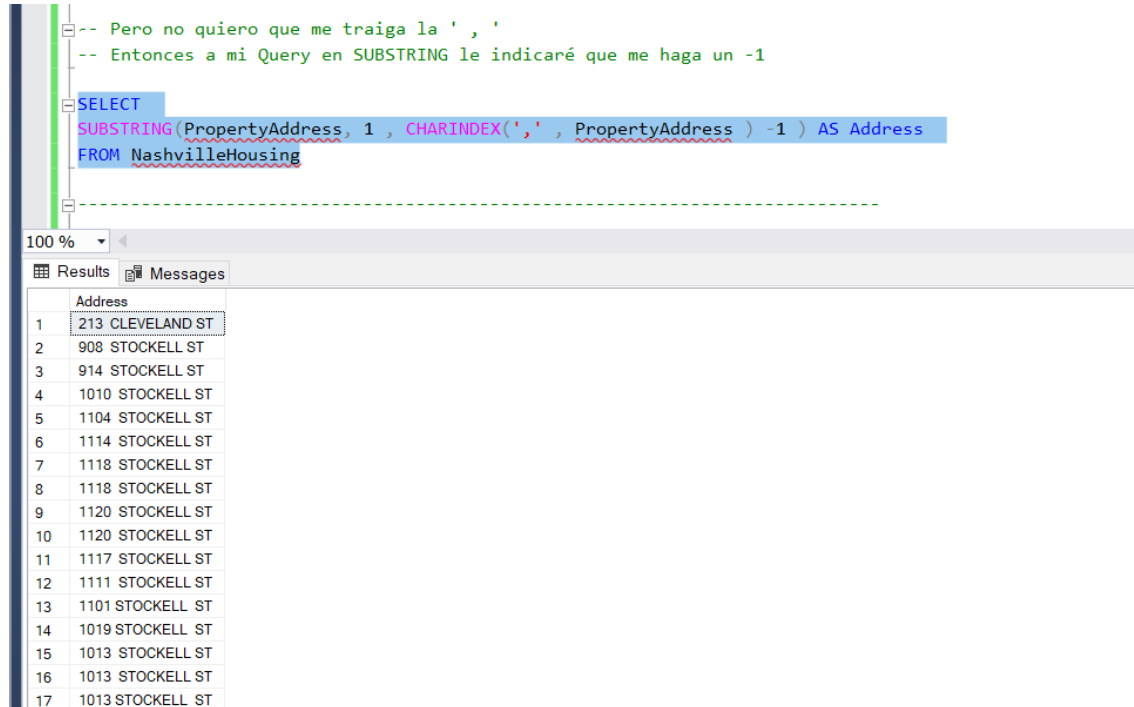
|    | Address           | (No column name) |
|----|-------------------|------------------|
| 1  | 213 CLEVELAND ST. | 18               |
| 2  | 908 STOCKELL ST.  | 17               |
| 3  | 914 STOCKELL ST.  | 17               |
| 4  | 1010 STOCKELL ST. | 18               |
| 5  | 1104 STOCKELL ST. | 18               |
| 6  | 1114 STOCKELL ST. | 18               |
| 7  | 1118 STOCKELL ST. | 18               |
| 8  | 1118 STOCKELL ST. | 18               |
| 9  | 1120 STOCKELL ST. | 18               |
| 10 | 1120 STOCKELL ST. | 18               |
| 11 | 1117 STOCKELL ST. | 18               |
| 12 | 1111 STOCKELL ST. | 18               |
| 13 | 1101 STOCKELL ST. | 18               |
| 14 | 1019 STOCKELL ST. | 18               |
| 15 | 1013 STOCKELL ST. | 18               |
| 16 | 1013 STOCKELL ST. | 18               |
| 17 | 1013 STOCKELL ST. | 18               |
| 18 | 1009 STOCKELL ST. | 18               |
| 19 | 1009 STOCKELL ST. | 18               |

```

-- Pero no quiero que me traiga la ' , '
-- Entonces a mi Query en SUBSTRING le indicaré que me haga un -1

```

```
SELECT
SUBSTRING(PropertyAddress, 1 , CHARINDEX(',', PropertyAddress ) -1
) AS Address
FROM NashvilleHousing
```



```
-- Pero no quiero que me traiga la ', '
-- Entonces a mi Query en SUBSTRING le indicaré que me haga un -1

SELECT
SUBSTRING(PropertyAddress, 1 , CHARINDEX(',', PropertyAddress ) -1 ) AS Address
FROM NashvilleHousing
```

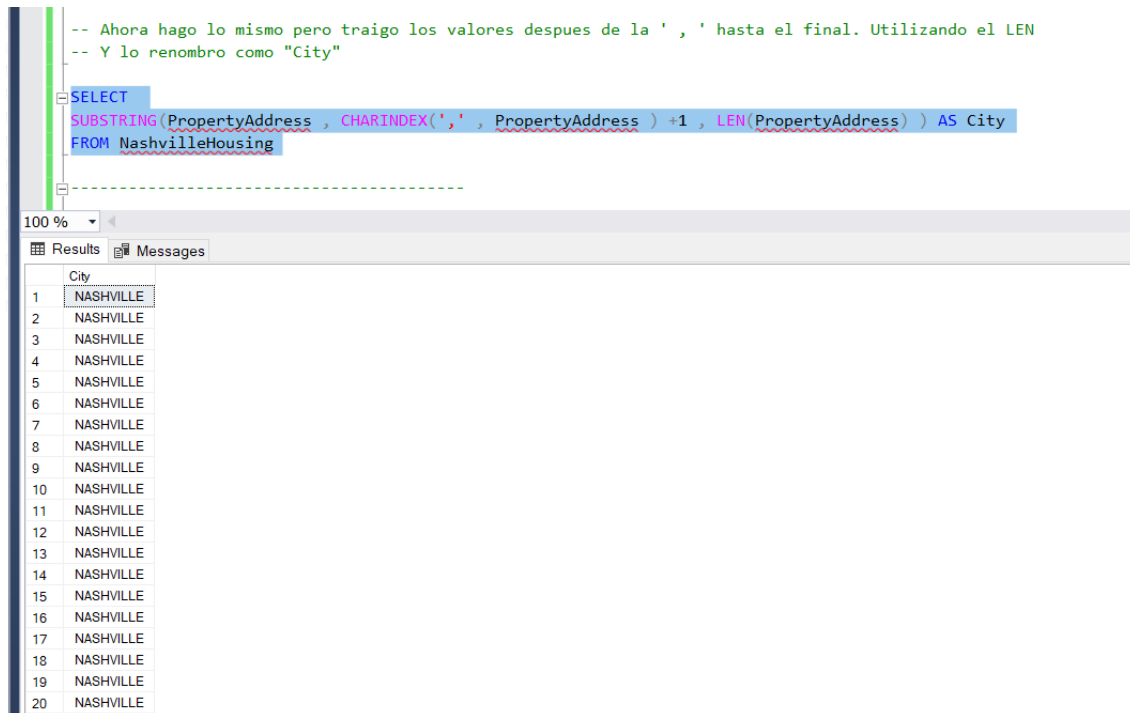
100 %

Results Messages

|    | Address          |
|----|------------------|
| 1  | 213 CLEVELAND ST |
| 2  | 908 STOCKELL ST  |
| 3  | 914 STOCKELL ST  |
| 4  | 1010 STOCKELL ST |
| 5  | 1104 STOCKELL ST |
| 6  | 1114 STOCKELL ST |
| 7  | 1118 STOCKELL ST |
| 8  | 1118 STOCKELL ST |
| 9  | 1120 STOCKELL ST |
| 10 | 1120 STOCKELL ST |
| 11 | 1117 STOCKELL ST |
| 12 | 1111 STOCKELL ST |
| 13 | 1101 STOCKELL ST |
| 14 | 1019 STOCKELL ST |
| 15 | 1013 STOCKELL ST |
| 16 | 1013 STOCKELL ST |
| 17 | 1013 STOCKELL ST |

```
-- Ahora hago lo mismo pero traigo los valores despues de la ' , '
hasta el final. Utilizando el LEN
-- Y lo renombro como "City"
```

```
SELECT
SUBSTRING(PropertyAddress , CHARINDEX(',' , PropertyAddress ) +1 ,
LEN(PropertyAddress) ) AS City
FROM NashvilleHousing
```



The screenshot shows a SQL query window with the following text:

```
-- Ahora hago lo mismo pero traigo los valores despues de la ' , ' hasta el final. Utilizando el LEN
-- Y lo renombro como "City"

SELECT
SUBSTRING(PropertyAddress , CHARINDEX(',' , PropertyAddress ) +1 , LEN(PropertyAddress) ) AS City
FROM NashvilleHousing
```

Below the query window, the 'Results' tab is active, displaying a table with 20 rows and one column named 'City'. All rows contain the value 'NASHVILLE'.

|    | City      |
|----|-----------|
| 1  | NASHVILLE |
| 2  | NASHVILLE |
| 3  | NASHVILLE |
| 4  | NASHVILLE |
| 5  | NASHVILLE |
| 6  | NASHVILLE |
| 7  | NASHVILLE |
| 8  | NASHVILLE |
| 9  | NASHVILLE |
| 10 | NASHVILLE |
| 11 | NASHVILLE |
| 12 | NASHVILLE |
| 13 | NASHVILLE |
| 14 | NASHVILLE |
| 15 | NASHVILLE |
| 16 | NASHVILLE |
| 17 | NASHVILLE |
| 18 | NASHVILLE |
| 19 | NASHVILLE |
| 20 | NASHVILLE |

-----

```
-- Voy a crear nuevas columnas
```

```
ALTER TABLE NashvilleHousing
ADD PropertyAdrees2 nvarchar(255)
```

```
ALTER TABLE NashvilleHousing
ADD City nvarchar (255)
```

-----

-- Actualizo Address

```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET PropertyAdrees2 = SUBSTRING(PropertyAddress, 1 , CHARINDEX(',',
, PropertyAddress ) -1 )
```

-- Actualizo City

```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET City = SUBSTRING(PropertyAddress , CHARINDEX(',',' ,
PropertyAddress ) +1 , LEN(PropertyAddress) )
```

-- Vemos que hemos podido adherir nuestros 2 nuevos campos al Final de la Tabla

```
SELECT *
FROM NashvilleHousing
```

| SaleDate2  | PropertyAdrees2  | City      |
|------------|------------------|-----------|
| 2014-11-07 | 213 CLEVELAND ST | NASHVILLE |
| 2014-01-30 | 908 STOCKELL ST  | NASHVILLE |
| 2013-07-10 | 914 STOCKELL ST  | NASHVILLE |
| 2014-01-28 | 1010 STOCKELL ST | NASHVILLE |
| 2013-08-30 | 1104 STOCKELL ST | NASHVILLE |
| 2013-06-14 | 1114 STOCKELL ST | NASHVILLE |
| 2013-06-17 | 1118 STOCKELL ST | NASHVILLE |
| 2014-07-31 | 1118 STOCKELL ST | NASHVILLE |
| 2014-11-19 | 1120 STOCKELL ST | NASHVILLE |
| 2015-07-31 | 1120 STOCKELL ST | NASHVILLE |
| 2015-03-26 | 1117 STOCKELL ST | NASHVILLE |
| 2014-01-14 | 1111 STOCKELL ST | NASHVILLE |
| 2016-10-06 | 1101 STOCKELL ST | NASHVILLE |
| 2016-06-06 | 1019 STOCKELL ST | NASHVILLE |
| 2013-10-15 | 1013 STOCKELL ST | NASHVILLE |
| 2015-06-23 | 1013 STOCKELL ST | NASHVILLE |

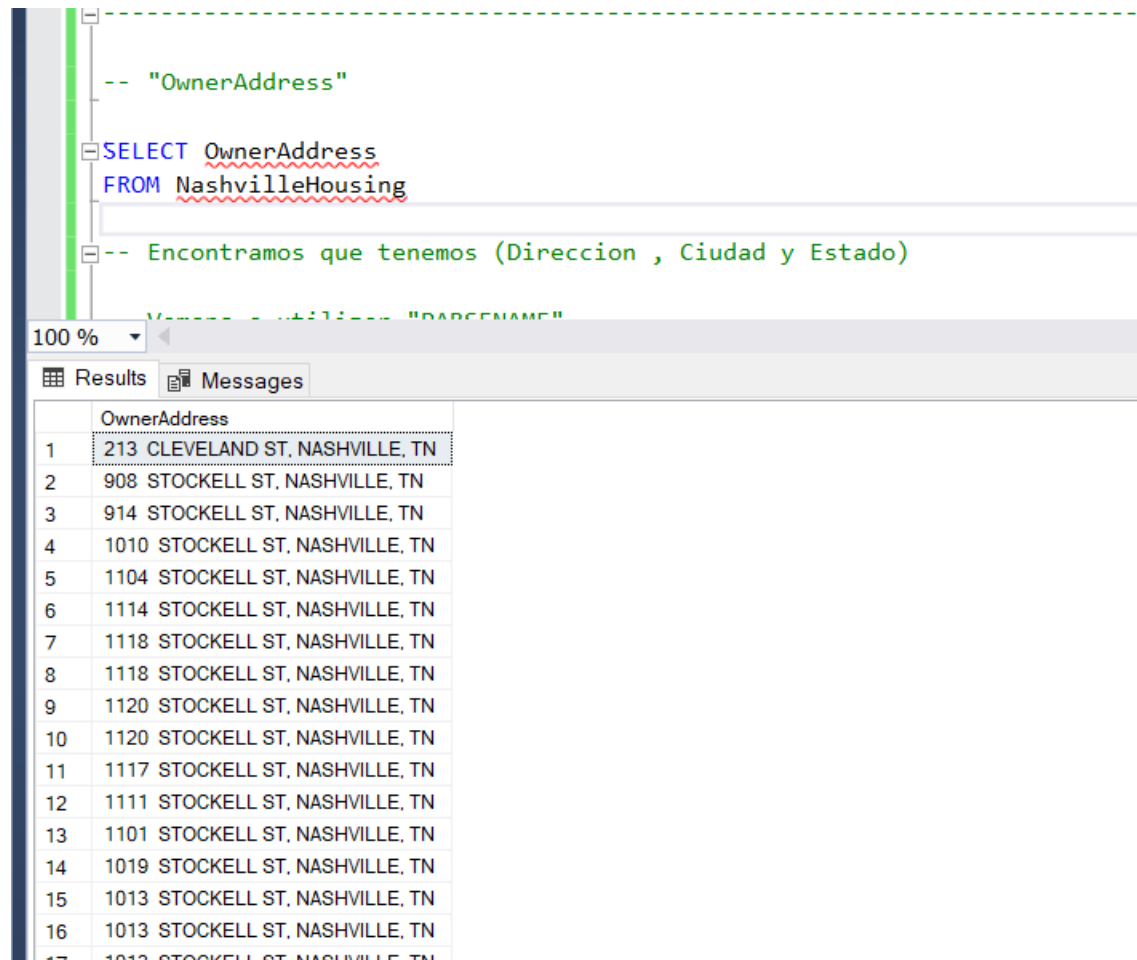
-----  
-----



```
-- "OwnerAddress"
```

```
SELECT OwnerAddress  
FROM NashvilleHousing
```

```
-- Encontramos que tenemos (Direccion , Ciudad y Estado)
```



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

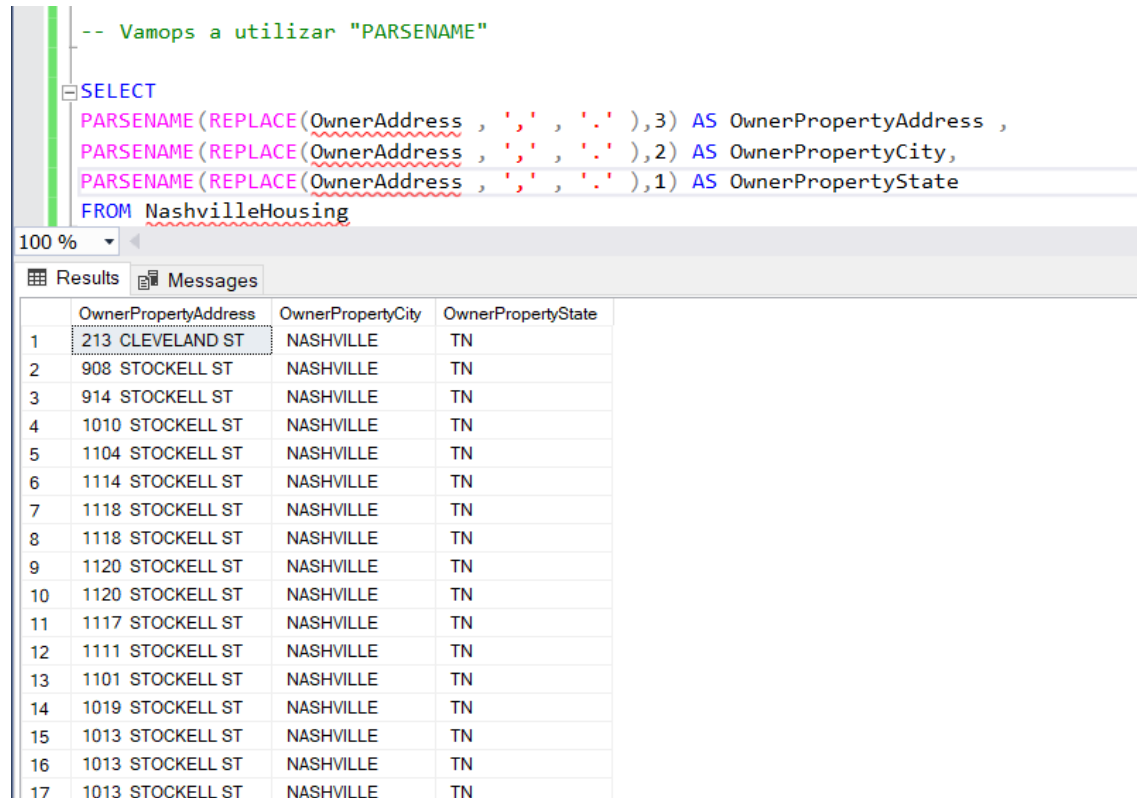
```
-- "OwnerAddress"  
  
SELECT OwnerAddress  
FROM NashvilleHousing  
  
-- Encontramos que tenemos (Direccion , Ciudad y Estado)  
  
-- Vamos a utilizar "DABCFNAME"
```

The results window shows the output of the query, which is a list of addresses. The results are displayed in a table with two columns: a row number and the address string.

|    | OwnerAddress                    |
|----|---------------------------------|
| 1  | 213 CLEVELAND ST, NASHVILLE, TN |
| 2  | 908 STOCKELL ST, NASHVILLE, TN  |
| 3  | 914 STOCKELL ST, NASHVILLE, TN  |
| 4  | 1010 STOCKELL ST, NASHVILLE, TN |
| 5  | 1104 STOCKELL ST, NASHVILLE, TN |
| 6  | 1114 STOCKELL ST, NASHVILLE, TN |
| 7  | 1118 STOCKELL ST, NASHVILLE, TN |
| 8  | 1118 STOCKELL ST, NASHVILLE, TN |
| 9  | 1120 STOCKELL ST, NASHVILLE, TN |
| 10 | 1120 STOCKELL ST, NASHVILLE, TN |
| 11 | 1117 STOCKELL ST, NASHVILLE, TN |
| 12 | 1111 STOCKELL ST, NASHVILLE, TN |
| 13 | 1101 STOCKELL ST, NASHVILLE, TN |
| 14 | 1019 STOCKELL ST, NASHVILLE, TN |
| 15 | 1013 STOCKELL ST, NASHVILLE, TN |
| 16 | 1013 STOCKELL ST, NASHVILLE, TN |
| 17 | 1013 STOCKELL ST, NASHVILLE, TN |

```
-- Vamos a utilizar "PARSENAME"
```

```
SELECT  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),3) AS  
OwnerPropertyAddress ,  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),2) AS  
OwnerPropertyCity,  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),1) AS  
OwnerPropertyState  
FROM NashvilleHousing
```



The screenshot shows a SQL query window with the following text:

```
-- Vamos a utilizar "PARSENAME"  
  
SELECT  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),3) AS OwnerPropertyAddress ,  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),2) AS OwnerPropertyCity,  
PARSENAME(REPLACE(OwnerAddress , ',' , '.' ),1) AS OwnerPropertyState  
FROM NashvilleHousing
```

Below the query window, the 'Results' tab is active, displaying a table with 17 rows and 3 columns: OwnerPropertyAddress, OwnerPropertyCity, and OwnerPropertyState. The first row is highlighted.

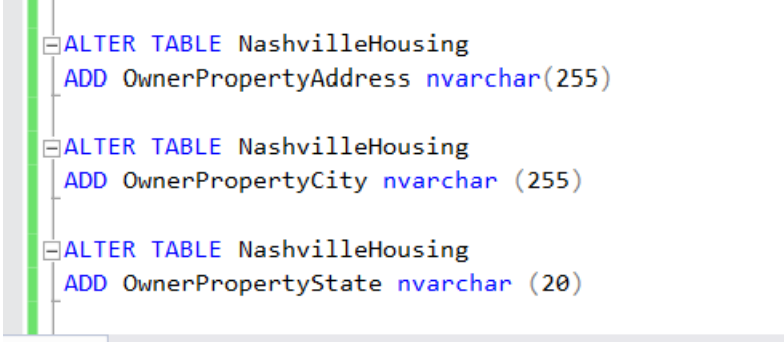
|    | OwnerPropertyAddress | OwnerPropertyCity | OwnerPropertyState |
|----|----------------------|-------------------|--------------------|
| 1  | 213 CLEVELAND ST     | NASHVILLE         | TN                 |
| 2  | 908 STOCKELL ST      | NASHVILLE         | TN                 |
| 3  | 914 STOCKELL ST      | NASHVILLE         | TN                 |
| 4  | 1010 STOCKELL ST     | NASHVILLE         | TN                 |
| 5  | 1104 STOCKELL ST     | NASHVILLE         | TN                 |
| 6  | 1114 STOCKELL ST     | NASHVILLE         | TN                 |
| 7  | 1118 STOCKELL ST     | NASHVILLE         | TN                 |
| 8  | 1118 STOCKELL ST     | NASHVILLE         | TN                 |
| 9  | 1120 STOCKELL ST     | NASHVILLE         | TN                 |
| 10 | 1120 STOCKELL ST     | NASHVILLE         | TN                 |
| 11 | 1117 STOCKELL ST     | NASHVILLE         | TN                 |
| 12 | 1111 STOCKELL ST     | NASHVILLE         | TN                 |
| 13 | 1101 STOCKELL ST     | NASHVILLE         | TN                 |
| 14 | 1019 STOCKELL ST     | NASHVILLE         | TN                 |
| 15 | 1013 STOCKELL ST     | NASHVILLE         | TN                 |
| 16 | 1013 STOCKELL ST     | NASHVILLE         | TN                 |
| 17 | 1013 STOCKELL ST     | NASHVILLE         | TN                 |

```
-- Creamos Nuevas Columnas

ALTER TABLE NashvilleHousing
ADD OwnerPropertyAddress nvarchar(255)

ALTER TABLE NashvilleHousing
ADD OwnerPropertyCity nvarchar (255)

ALTER TABLE NashvilleHousing
ADD OwnerPropertyState nvarchar (20)
```



```
ALTER TABLE NashvilleHousing
ADD OwnerPropertyAddress nvarchar(255)

ALTER TABLE NashvilleHousing
ADD OwnerPropertyCity nvarchar (255)

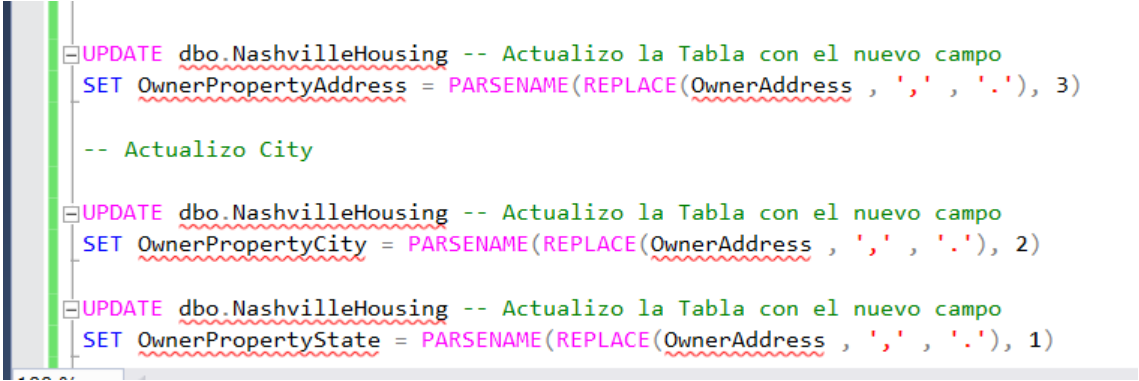
ALTER TABLE NashvilleHousing
ADD OwnerPropertyState nvarchar (20)
```

```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET OwnerPropertyAddress = PARSENAME(REPLACE(OwnerAddress , ',' ,
'.'), 3)
```

```
-- Actualizo City
```

```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET OwnerPropertyCity = PARSENAME(REPLACE(OwnerAddress , ',' ,
'.'), 2)
```

```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo
campo
SET OwnerPropertyState = PARSENAME(REPLACE(OwnerAddress , ',' ,
'.'), 1)
```



```
UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo campo
SET OwnerPropertyAddress = PARSENAME(REPLACE(OwnerAddress , ',' , '.'), 3)

-- Actualizo City

UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo campo
SET OwnerPropertyCity = PARSENAME(REPLACE(OwnerAddress , ',' , '.'), 2)

UPDATE dbo.NashvilleHousing -- Actualizo la Tabla con el nuevo campo
SET OwnerPropertyState = PARSENAME(REPLACE(OwnerAddress , ',' , '.'), 1)
```

-- Vemos que hemos podido adherir nuestros 2 nuevos campos al Final de la Tabla

```
SELECT *  
FROM NashvilleHousing
```

-- Chekeamos y hemos generado de forma adecuada el PARSEADO  
-- Y es mucho más útil

| OwnerPropertyAddress | OwnerPropertyCity | OwnerPropertyState |
|----------------------|-------------------|--------------------|
| 213 CLEVELAND ST     | NASHVILLE         | TN                 |
| 908 STOCKELL ST      | NASHVILLE         | TN                 |
| 914 STOCKELL ST      | NASHVILLE         | TN                 |
| 1010 STOCKELL ST     | NASHVILLE         | TN                 |
| 1104 STOCKELL ST     | NASHVILLE         | TN                 |
| 1114 STOCKELL ST     | NASHVILLE         | TN                 |
| 1118 STOCKELL ST     | NASHVILLE         | TN                 |
| 1118 STOCKELL ST     | NASHVILLE         | TN                 |
| 1120 STOCKELL ST     | NASHVILLE         | TN                 |
| 1120 STOCKELL ST     | NASHVILLE         | TN                 |
| 1117 STOCKELL ST     | NASHVILLE         | TN                 |
| 1111 STOCKELL ST     | NASHVILLE         | TN                 |
| 1101 STOCKELL ST     | NASHVILLE         | TN                 |
| 1019 STOCKELL ST     | NASHVILLE         | TN                 |
| 1013 STOCKELL ST     | NASHVILLE         | TN                 |
| 1013 STOCKELL ST     | NASHVILLE         | TN                 |

-----  
-----

```
-- Asignemos correctamente los Y & N a Yes & No en el campo
"SoldAsVacant"
```

```
SELECT DISTINCT(SoldAsVacant) , COUNT(SoldAsVacant)
FROM NashvilleHousing
GROUP BY SoldAsVacant
```

```
-- Asignemos correctamente los Y & N a Yes & No en el campo "SoldAsVacant"
SELECT DISTINCT(SoldAsVacant) , COUNT(SoldAsVacant)
FROM NashvilleHousing
GROUP BY SoldAsVacant
```

100 %

Results Messages

|   | SoldAsVacant | (No column name) |
|---|--------------|------------------|
| 1 | N            | 399              |
| 2 | Yes          | 4623             |
| 3 | Y            | 52               |
| 4 | No           | 51403            |

Utilizamos el CASE WHEN para poder Asignar en una Nueva Columna los valores Y & N  
Re-nombramos como NewSoldAsVacant

```
SELECT SoldAsVacant ,
CASE WHEN SoldAsVacant = 'Y' THEN 'Yes'
      WHEN SoldAsVacant = 'N' THEN 'No'
      ELSE SoldAsVacant
      END AS NewSoldAsVacant
FROM NashvilleHousing
```

[illegible]

--Actualizamos

```
UPDATE NashvilleHousing
SET SoldAsVacant =
CASE WHEN SoldAsVacant = 'Y' THEN 'Yes'
      WHEN SoldAsVacant = 'N' THEN 'No'
      ELSE SoldAsVacant
END
```

-- Probamos y veremos que nos funcionó

```
SELECT DISTINCT(SoldAsVacant) , COUNT(SoldAsVacant)
FROM NashvilleHousing
GROUP BY SoldAsVacant
```

The screenshot shows a SQL Server Enterprise Manager interface. At the top, there are two tabs: 'SQL\_DirtyDatasets...VLDC293\Facu (59))' and 'SQLQuery2.sql - D...VLDC293\Facu (62))'. The active tab displays the following SQL code:

```
UPDATE NashvilleHousing
SET SoldAsVacant =
CASE WHEN SoldAsVacant = 'Y' THEN 'Yes'
      WHEN SoldAsVacant = 'N' THEN 'No'
      ELSE SoldAsVacant
END

-- Probamos y veremos que nos funcionó

SELECT DISTINCT(SoldAsVacant) , COUNT(SoldAsVacant)
FROM NashvilleHousing
GROUP BY SoldAsVacant
```

Below the code editor, there is a 'Results' tab which is currently selected. It displays the output of the query in a table format:

|   | SoldAsVacant | (No column name) |
|---|--------------|------------------|
| 1 | Yes          | 4675             |
| 2 | No           | 51802            |

```
-- Removamos Duplicados
-- Para identificarlos podemos utilizar RANKE, ROW_NUMBER (La forma
más simple)
```

```
-- Tambien Utilizaremos CTE (CommonTableExpression) WITH Querys
```

```
WITH NumeroFilaCTE AS (
SELECT * ,
    ROW_NUMBER() OVER (
        PARTITION BY ParcelID ,
            PropertyAddress,
                SalePrice,
                SaleDate,
                LegalReference
            ORDER BY UniqueID
        )
        numero_fila
FROM NashvilleHousing
)
SELECT *
FROM NumeroFilaCTE
WHERE numero_fila > 1 -- Si es MAYOR a 1 es Duplicado
ORDER BY PropertyAddress
```

```
-- Podemos ver todos esos valores duplicados
```

```
WITH NumeroFilaCTE AS (
SELECT * ,
    ROW_NUMBER() OVER (
        PARTITION BY ParcelID ,
            PropertyAddress,
                SalePrice,
                SaleDate,
                LegalReference
            ORDER BY UniqueID
        )
        numero_fila
FROM NashvilleHousing
)
SELECT *
FROM NumeroFilaCTE
WHERE numero_fila > 1 -- Si es MAYOR a 1 es Duplicado
ORDER BY PropertyAddress
```

-- Podemos ver todos esos valores duplicados

|     | Acreage | TaxDistrict             | LandValue | BuildingValue | TotalValue | YearBuilt | Bedrooms | FullBath | HalfBath | SaleDate2  | PropertyAddress2        | City      | OwnerPropertyAddress | OwnerPropertyCity | OwnerPropertyState | numero_fila |
|-----|---------|-------------------------|-----------|---------------|------------|-----------|----------|----------|----------|------------|-------------------------|-----------|----------------------|-------------------|--------------------|-------------|
| 91  |         |                         |           |               |            |           |          |          |          | 2015-02-19 | 6203 NEW YORK AVE       | NASHVILLE |                      |                   |                    | 2           |
| 92  |         |                         |           |               |            |           |          |          |          | 2015-02-20 | 6629 HICKORY RIM CT     | ANTIOCH   |                      |                   |                    | 2           |
| 93  |         |                         |           |               |            |           |          |          |          | 2015-02-17 | 713 44TH AVE N          | NASHVILLE |                      |                   |                    | 2           |
| 94  | 0.14    | URBAN SERVICES DISTRICT | 18000     | 95800         | 120800     | 1969      | 2        | 1        | 0        | 2015-02-13 | 743 CROLEY DR           | NASHVILLE | 743 CROLEY DR        | NASHVILLE         | TN                 | 2           |
| 95  |         |                         |           |               |            |           |          |          |          | 2015-02-13 | 813 POINT BREAK CR S    | ANTIOCH   |                      |                   |                    | 2           |
| 96  | 0.17    | URBAN SERVICES DISTRICT | 13000     | 49400         | 62400      | 1960      | 2        | 1        | 0        | 2015-02-20 | 815 31ST AVE N          | NASHVILLE | 815 31ST AVE N       | NASHVILLE         | TN                 | 2           |
| 97  | 0.23    | URBAN SERVICES DISTRICT | 27500     | 102000        | 129500     | 1967      | 3        | 2        | 0        | 2015-02-26 | 820 DRESDEN CT          | ANTIOCH   | 820 DRESDEN CT       | ANTIOCH           | TN                 | 2           |
| 98  |         |                         |           |               |            |           |          |          |          | 2015-02-13 | 8528 BEAUTIFUL VALLEY _ | NASHVILLE |                      |                   |                    | 2           |
| 99  |         |                         |           |               |            |           |          |          |          | 2015-02-19 | 8593 BEAUTIFUL VALLEY _ | NASHVILLE |                      |                   |                    | 2           |
| 100 |         |                         |           |               |            |           |          |          |          | 2014-02-03 | 904 BIRCHMILL PT S      | ANTIOCH   |                      |                   |                    | 2           |
| 101 | 0.17    | URBAN SERVICES DISTRICT | 21000     | 69400         | 95500      | 1954      | 2        | 1        | 0        | 2015-02-13 | 906 DELRAY DR           | NASHVILLE | 906 DELRAY DR        | NASHVILLE         | TN                 | 2           |
| 102 |         |                         |           |               |            |           |          |          |          | 2015-02-06 | 908 TOWNVIEW PL         | NASHVILLE |                      |                   |                    | 2           |
| 103 |         |                         |           |               |            |           |          |          |          | 2014-02-05 | 926 COARSEY DR          | NASHVILLE |                      |                   |                    | 2           |
| 104 | 0.66    | URBAN SERVICES DISTRICT | 14300     | 97300         | 111600     | 1958      | 2        | 1        | 0        | 2015-02-13 | 934 DRUMMOND DR         | NASHVILLE | 934 DRUMMOND DR      | NASHVILLE         | TN                 | 2           |



```

-- Borramos los duplicados

WITH NumeroFilaCTE AS (
SELECT * ,
    ROW_NUMBER() OVER (
        PARTITION BY ParcelID ,
                    PropertyAddress,
                    SalePrice,
                    SaleDate,
                    LegalReference
        ORDER BY
            UniqueID
        )
    numero_fila
FROM NashvilleHousing
)
DELETE -- Borramos
FROM NumeroFilaCTE
WHERE numero_fila > 1 -- Si es MAYOR a 1 es Duplicado

-- Pudimos borrar los Duplicados

```

SQL\_DirtyDataset.sql...u (59)) Executing... SQLQuery2.sql - D...VLDC293\Facu (62))

```

WITH NumeroFilaCTE AS (
SELECT * ,
    ROW_NUMBER() OVER (
        PARTITION BY ParcelID ,
                    PropertyAddress,
                    SalePrice,
                    SaleDate,
                    LegalReference
        ORDER BY
            UniqueID
        )
    numero_fila
FROM NashvilleHousing
)
DELETE -- Borramos
FROM NumeroFilaCTE
WHERE numero_fila > 1 -- Si es MAYOR a 1 es Duplicado

```

100 %

Results Messages

-- Si volvemos a mirar al Codigo del SELECT , veremos que ya no están ahí y nos devuelve una Tabla Vacía

```
WITH NumeroFilaCTE AS (
SELECT * ,
    ROW_NUMBER() OVER (
        PARTITION BY ParcelID ,
            PropertyAddress,
            SalePrice,
            SaleDate,
            LegalReference
        ORDER BY UniqueID
    )
    numero_fila
FROM NashvilleHousing
)
SELECT *
FROM NumeroFilaCTE
WHERE numero_fila > 1 -- Si es MAYOR a 1 es Duplicado
ORDER BY PropertyAddress

-- Podemos ver todos esos valores duplicados
```

100 %

Results Messages

| UniqueID | ParcelID | LandUse | PropertyAddress | SaleDate | SalePrice | LegalReference | SoldAsVacant | OwnerName | OwnerAddress | Acreage | TaxDistrict |
|----------|----------|---------|-----------------|----------|-----------|----------------|--------------|-----------|--------------|---------|-------------|
|----------|----------|---------|-----------------|----------|-----------|----------------|--------------|-----------|--------------|---------|-------------|

-----

-- Borraremos Columnas Inusuales y las que ya no nos sirven  
-- Lo estamos haciendo solo para poder aprovechar el Dataset y jugar :)

```
SELECT *
FROM NashvilleHousing

ALTER TABLE NashvilleHousing
DROP COLUMN OwnerAddress , PropertyAddress, SaleDate , TaxDistrict
```

```
-- Borraremos Columnas Inusuales y las que ya no nos sirven
-- No es una buena práctica pero lo estamos haciendo solo para poder aprovechar el Dataset y jugar :)

SELECT *
FROM NashvilleHousing

ALTER TABLE NashvilleHousing
DROP COLUMN OwnerAddress , PropertyAddress, SaleDate , TaxDistrict
```

100 %

Results Messages

|    | UniqueID | ParcelID        | LandUse         | PropertyAddress             | SaleDate                | SalePrice | LegalReference   | SoldAsVacant | OwnerName                   |
|----|----------|-----------------|-----------------|-----------------------------|-------------------------|-----------|------------------|--------------|-----------------------------|
| 1  | 24022    | 082 03 0 158.00 | SINGLE FAMILY   | 213 CLEVELAND ST, NASHVILLE | 2014-11-07 00:00:00.000 | 177000    | 20141110-0103612 | No           | DANIEL, CLINTON RYAN        |
| 2  | 11770    | 082 03 0 160.00 | VACANT RES LAND | 908 STOCKELL ST, NASHVILLE  | 2014-01-30 00:00:00.000 | 292000    | 20140204-0009738 | No           | LANCASTER, MARK A. & ANGELA |
| 3  | 5947     | 082 03 0 163.00 | SINGLE FAMILY   | 914 STOCKELL ST, NASHVILLE  | 2013-07-10 00:00:00.000 | 82000     | 20130711-0071726 | No           | CHEATHAM, HERBERT L.        |
| 4  | 11771    | 082 03 0 170.00 | SINGLE FAMILY   | 1010 STOCKELL ST, NASHVILLE | 2014-01-28 00:00:00.000 | 227500    | 20140130-0008515 | No           | DEFIGLIA, JOSEPH PAUL       |
| 5  | 7153     | 082 03 0 177.00 | SINGLE FAMILY   | 1104 STOCKELL ST, NASHVILLE | 2013-08-30 00:00:00.000 | 224500    | 20130909-0094860 | No           | COLLIGAN, JASON             |
| 6  | 4678     | 082 03 0 182.00 | SINGLE FAMILY   | 1114 STOCKELL ST, NASHVILLE | 2013-06-14 00:00:00.000 | 215000    | 20130617-0061553 | No           | JEFFERSON, JONATHAN RAYMC   |
| 7  | 4679     | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2013-06-17 00:00:00.000 | 69000     | 20130621-0063866 | No           | COOK, CHRISTOPHER J. & IZMA |
| 8  | 18553    | 082 03 0 184.00 | SINGLE FAMILY   | 1118 STOCKELL ST, NASHVILLE | 2014-07-31 00:00:00.000 | 330000    | 20140812-0072633 | No           | COOK, CHRISTOPHER J. & IZMA |
| 9  | 24023    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2014-11-19 00:00:00.000 | 140000    | 20141121-0107304 | No           | CHADWICK, WILLIAM & PUMPHF  |
| 10 | 35003    | 082 03 0 185.00 | SINGLE FAMILY   | 1120 STOCKELL ST, NASHVILLE | 2015-07-31 00:00:00.000 | 375000    | 20150804-0077449 | No           | CHADWICK, WILLIAM & PUMPHF  |

