Nashville Housing Project

April 10th, 2023

By Craig Schlachter

Step 1: Ask

In this step, we define the problem and objectives of our case study and its desired outcome.

1.1 Background

Nashville, Tennessee is one of the fastest growing metro-areas in the USA. With many people looking for increased job opportunities and affordable housing, Nashville presents itself as a fitting destination.

This project's aim is to perform a descriptive analysis of the Nashville housing market and the key statistical indicators. The analysis will provide key housing statistics for anyone looking to settle down in the Music City.

1.2 Business Task

Analyze a housing dataset to understand how variables affect housing prices and **discover trends** of the Nashville housing market.

1.3 Business Objectives

- What are the trends in sale price?
- What are the trends in housing inventory and total sales volume?
- How do different variables affect price?

1.4 Deliverables

- A clear summary of the business task
- A description of all data sources used
- Documentation of any cleaning or manipulation of data
- A summary of analysis
- Supporting visualizations and findings
- High-level recommendations based on the analysis

1.5 Key Stakeholders

- The key stakeholders of this project are any individuals who may be interested in moving to Nashville, Tennessee and want to understand the key summary statistics of the housing market and how that could affect their particular situation.

. . .

Step 2: Prepare

In the Prepare phase, we identify the data being used and its limitations.

2.1 Information on Data Source

1. The data is publicly available on Kaggle: Nashville Housing Data and stored in one csv file.
2. This dataset was scrapped by TMTHYJAMES of Kaggle and contains housing data for the years 2013 - 2016.
3. This dataset comprises 56,000 + rows of housing data.
4. Data collected includes sale date, sale price, land use, city, etc.

2.2 Limitations of Data Set

- Data was last collected over 6 years ago in 2016. The sale prices, total properties, and total number of sales have certainly changed since then. This data may not be timely or relevant.
- The data was web-scrapped, so we are unable to establish its integrity or accuracy.
- 10 of the 31 columns in this dataset are composed of 50% or more NULL values. So, we are unable to determine how reliable this data can be.

2.3 Is Data ROCCC?

A good data source is ROCCC which stands for Reliable, Original, Comprehensive, Current, and Cited.

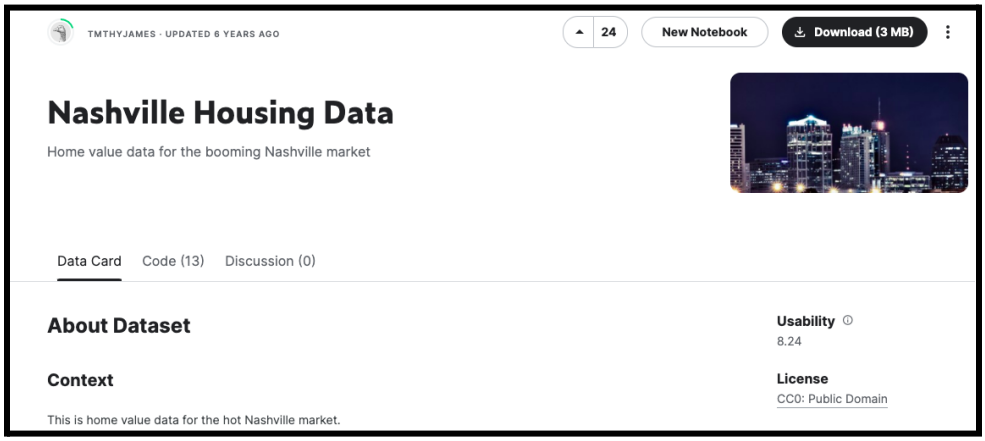
- Reliable - MED - Has over 56K+ rows, but a third of the columns contain 50% or more nulls
- Original - LOW - Third party provider TMTHYJAMES of Kaggle

- Comprehensive - MED - Parameters match most questions this project aims to answer
- Current - LOW - Data is over 6 years old and may not be relevant
- Cited - LOW - Data collected by third party, thus unknown

Overall, this dataset is considered bad quality and is not fit for making recommendations off the business insights.

2.4 Data Selection

The following file is downloaded and then imported into our created SQL table ‘nashville_housing’.



2.5 Tools

We are using SQL for data-wrangling and exploratory data analysis. Finally, we are using Tableau for visualizations.

Step 3: Process

Here, we will process the data to ensure it is clean, correct, relevant, complete and free of errors and outliers by performing:

- Explore and observe the dataset
- Standardize the sale date format
- Populating property address data
- Break out the property and owner address into individual columns (address, city, state)
- Change ‘Y’ and ‘N’ to ‘Yes’ and ‘No’ in Sold as Vacant field
- Remove Duplicates
- Delete unused columns (Only doing this for demonstration purposes)
- Save cleaned data to a new file

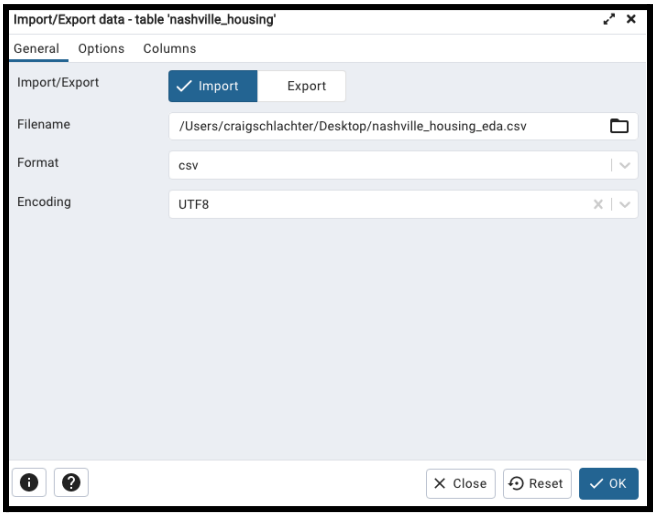
3.1 Preparing the Environment

The SQL table is created, columns are named, and data types are set.

```sql
-- Table: public.nashville_housing

-- DROP TABLE IF EXISTS public.nashville_housing;

CREATE TABLE IF NOT EXISTS public.nashville_housing
(
    uniqueid integer,
    parcelid text COLLATE pg_catalog."default",
    landuse text COLLATE pg_catalog."default",
    saleprice integer,
    legalreference text COLLATE pg_catalog."default",
    soldasvacant text COLLATE pg_catalog."default",
    ownername text COLLATE pg_catalog."default",
    acreage numeric,
    landvalue integer,
    buildingvalue integer,
    totalvalue integer,
    yearbuilt integer,
    bedroom integer,
    fullbath integer,
    halfbath integer,
    saledateconverted date,
    propertysplitaddress text COLLATE pg_catalog."default",
    propertysplitcity text COLLATE pg_catalog."default",
    ownersplitaddress text COLLATE pg_catalog."default",
    ownersplitcity text COLLATE pg_catalog."default",
    ownersplitstate text COLLATE pg_catalog."default"
)

TABLESPACE pg_default;

ALTER TABLE IF EXISTS public.nashville_housing
    OWNER to postgres;
```

3.2 Importing Dataset

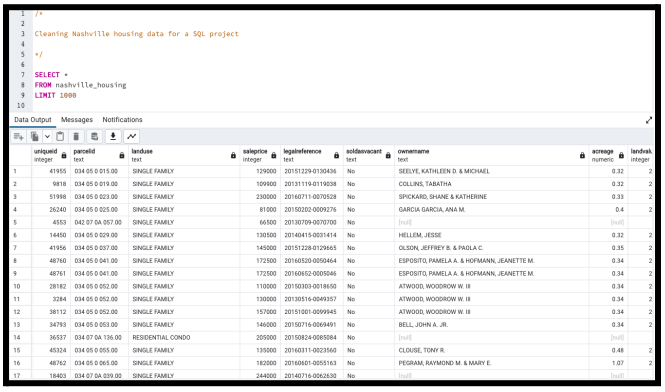
Reading in the selected file.



3.3 Data Cleaning and Manipulation

1. Observe and familiarize with data
2. Standardize the sale date field
3. Check for nulls or missing values
4. Perform validation checks of data

Previewing the first 100 rows to familiarize with the data.



Our dataset revealed several noteworthy features: **‘saledate’** was datetime and had all times set to '00:00:00', so we'll reset it to 'YYYY-MM-DD'. We'll also **populate nulls in ‘property address’**. Finally, we'll separate address, city, and state in **‘property address’ and ‘owner address’** for better data organization.

Now, I’m standardizing the **‘saledate’** column for better data structuring.

```sql
-- Standardizing the saledate format
-- Tried casting and updating, didn't work
-- So, I altered my table and this wasy did the trick.
SELECT saledate, CAST(saledate AS DATE)
FROM nashville_housing

UPDATE nashville_housing
SET saledate = CAST(saledate AS DATE)

ALTER TABLE nashville_housing
ADD saledateconverted DATE;

UPDATE nashville_housing
SET saledateconverted = CAST(saledate AS DATE)

SELECT saledateconverted
FROM nashville_housing
```

Next, I’m populating ‘property address’ by performing a self join to check for matching ‘parcelid’ but different ‘uniqueid’.

```sql
-- Populating property address data


SELECT *
FROM nashville_housing
--WHERE propertyaddress IS NULL
ORDER BY parcelid


SELECT nha.parcelid,
       nha.propertyaddress,
       nhb.parcelid,
       nhb.propertyaddress
FROM nashville_housing nha
JOIN nashville_housing nhb ON (nha.parcelid=nhb.parcelid) AND nha.uniqueid <> nhb.uniqueid
WHERE nha.propertyaddress IS NULL


UPDATE nashville_housing nha
SET propertyaddress = nhb.propertyaddress
FROM nashville_housing nhb
WHERE nha.parcelid = nhb.parcelid
AND nha.uniqueid <> nhb.uniqueid
AND nha.propertyaddress IS NULL
```

Now, we are breaking out the ‘property and owner address’ into individual columns (address, city, state).

```sql
-- Breaking out address into individual columns (address, city, state)


SELECT propertyaddress
FROM nashville_housing
--WHERE propertyaddress IS NULL
--ORDER BY parcelid

SELECT propertyaddress,
       SUBSTRING(propertyaddress,1,STRPOS(propertyaddress,',')-1) AS address,
       SUBSTRING(propertyaddress, STRPOS(propertyaddress,',')+1 , LENGTH(propertyaddress)) AS city
FROM nashville_housing


ALTER TABLE nashville_housing
ADD propertysplitaddress text;

UPDATE nashville_housing
SET propertysplitaddress = SUBSTRING(propertyaddress,1,STRPOS(propertyaddress,',')-1)


ALTER TABLE nashville_housing
ADD propertysplitcity text;

UPDATE nashville_housing
SET propertysplitcity = SUBSTRING(propertyaddress, STRPOS(propertyaddress,',')+1 , LENGTH(propertyaddress))
```

```sql
SELECT owneraddress
FROM nashville_housing



SELECT SPLIT_PART(owneraddress,',',1),
       SPLIT_PART(owneraddress,',',2),
       SPLIT_PART(owneraddress,',',3)
FROM nashville_housing



ALTER TABLE nashville_housing
ADD ownersplitaddress text;

UPDATE nashville_housing
SET ownersplitaddress = SPLIT_PART(owneraddress,',',1)



ALTER TABLE nashville_housing
ADD ownersplitcity text;

UPDATE nashville_housing
SET ownersplitcity = SPLIT_PART(owneraddress,',',2)



ALTER TABLE nashville_housing
ADD ownersplitstate text;

UPDATE nashville_housing
SET ownersplitstate = SPLIT_PART(owneraddress,',',3)
```

Moving on, we'll change 'Y' and 'N' to 'Yes' and 'No' in Sold as Vacant field

```sql
-- Changing Y and N to Yes and No in 'Sold as Vacant' field


SELECT DISTINCT(soldasvacant),
       COUNT(*)
FROM nashville_housing
GROUP BY 1
ORDER BY 2 DESC


SELECT soldasvacant,
       CASE WHEN soldasvacant = 'Y' THEN 'Yes'
       WHEN soldasvacant = 'N' THEN 'No'
       ELSE soldasvacant
       END

FROM nashville_housing



UPDATE nashville_housing
SET soldasvacant =  CASE WHEN soldasvacant = 'Y' THEN 'Yes'
       WHEN soldasvacant = 'N' THEN 'No'
       ELSE soldasvacant
       END
```

Next, we are removing duplicates **based on** if they share the **same parcel id**, **property address**, **sale price**, **sale date**, and **legal reference**.

```sql
-- Remove Duplicates


WITH row_num_cte AS (
                     SELECT *,
                            ROW_NUMBER() OVER (PARTITION BY parcelid,
                                               propertyaddress,
                                               saleprice,
                                               saledate,
                                               legalreference
                                              ORDER BY uniqueid) AS row_num
                     FROM nashville_housing
                     --ORDER BY parcelid
                   ),
         dups AS (                -- capturing only the duplicates
                     SELECT *
                     FROM row_num_cte
                     WHERE row_num > 1
                 )

DELETE FROM nashville_housing -- Using uniqueid instead of parcelid
WHERE uniqueid IN (SELECT uniqueid FROM dups)
```

From here we will delete unused columns **(Only doing this for demonstration purposes)**

```
175  -- Delete unused columns (Only doing this for demonstration purposes)
176
177
178
179  SELECT *
180  FROM nashville_housing
181
182
183  ALTER TABLE nashville_housing
184  DROP COLUMN owneraddress,
185  DROP COLUMN taxdistrict,
186  DROP COLUMN propertyaddress,
187  DROP COLUMN saledate
188  ;
```

Finally, we save the cleaned data to a new csv file.

```
192  -- Saving cleaned data to a new file
193
194
195
196  SELECT *
197  FROM nashville_housing
198
199
200
201  COPY (SELECT * FROM nashville_housing) TO '/Users/craigschlachter/Desktop/nashville_housing_eda.csv' WITH CSV HEADER;
```

. . .

Step 4: Analyze

4.1 Performing Calculations

Pulling statistics for analysis:

1. Reviewing and selecting the data for our exploratory data analysis.
2. Finding the distribution of Nashville housing prices.
3. Calculating the daily average sale prices over time.
4. Tracking the total number of housing sales in each city over time.
5. Establishing the average price for each property type.
6. Calculating the average sale price for each distinct bedroom and bathroom count.
7. Determining the distribution of properties across the Nashville metro area.
8. Finding the distribution of sales per sale date.
9. Determining the count of each unique property type and their distribution across the Nashville metro area.

[Step 1's Results]

```
10
11
12
13  SELECT *
14  FROM nash_housing_eda
15  ORDER BY saledateconverted ASC
16
17
18
19  -- Selecting data we are going to be using
20  -- Seeing null in bedroom, fullbath, halfbath is common
21
22
23  SELECT propertysplitcity,
24         saledateconverted,
25         saleprice,
26         landuse,
27         bedroom,
28         fullbath,
29         halfbath
30  FROM nash_housing_eda
31  WHERE LOWER(propertysplitcity) LIKE '%nash%'
32  ORDER BY 1,2
```

[Step 1's Results]

	propertysplitcity text	saledateconverted date	saleprice integer	landuse text	bedroom integer	fullbath integer	halfbath integer
1	NASHVILLE	2013-01-02	252000	RESIDENTIAL CONDO	[null]	[null]	[null]
2	NASHVILLE	2013-01-02	135000	RESIDENTIAL CONDO	[null]	[null]	[null]
3	NASHVILLE	2013-01-02	152000	SINGLE FAMILY	4	3	0
4	NASHVILLE	2013-01-02	50000	SINGLE FAMILY	3	1	0
5	NASHVILLE	2013-01-02	255000	RESIDENTIAL CONDO	[null]	[null]	[null]
6	NASHVILLE	2013-01-02	72500	VACANT RESIDENTIAL LAND	5	3	1
7	NASHVILLE	2013-01-02	225000	SINGLE FAMILY	2	1	0
8	NASHVILLE	2013-01-03	98000	SINGLE FAMILY	2	1	0
9	NASHVILLE	2013-01-03	55900	SINGLE FAMILY	[null]	[null]	[null]
10	NASHVILLE	2013-01-04	120000	SINGLE FAMILY	7	2	0
11	NASHVILLE	2013-01-04	275000	RESIDENTIAL CONDO	[null]	[null]	[null]
12	NASHVILLE	2013-01-04	135790	SINGLE FAMILY	[null]	[null]	[null]
13	NASHVILLE	2013-01-04	83000	SINGLE FAMILY	4	3	0
14	NASHVILLE	2013-01-04	20000	SINGLE FAMILY	3	1	0
15	NASHVILLE	2013-01-04	215044	SINGLE FAMILY	3	2	1
16	NASHVILLE	2013-01-04	262500	SINGLE FAMILY	2	2	0
17	NASHVILLE	2013-01-04	195000	SINGLE FAMILY	3	2	0
18	NASHVILLE	2013-01-04	194000	VACANT RESIDENTIAL LAND	[null]	[null]	[null]
19	NASHVILLE	2013-01-04	100	SINGLE FAMILY	2	1	0
20	NASHVILLE	2013-01-04	267300	SINGLE FAMILY	4	2	0
21	NASHVILLE	2013-01-04	113000	RESIDENTIAL CONDO	[null]	[null]	[null]
22	NASHVILLE	2013-01-04	77500	SINGLE FAMILY	3	1	0
23	NASHVILLE	2013-01-04	93736	SINGLE FAMILY	2	1	0
24	NASHVILLE	2013-01-07	93000	ZERO LOT LINE	3	2	1

[Step 2's Results]

```sql
-- Looking at the distribution of sale prices(#1)
-- Shows the full distribution and the sale price you're likely to pay in Nashville

SELECT saleprice,
       COUNT(*) AS sale_count
FROM nash_housing_eda
GROUP BY 1
ORDER BY 2 DESC
```

Data Output | Messages | Notifications

	saleprice integer	sale_count bigint
1	150000	549
2	200000	439
3	160000	430
4	120000	409
5	125000	403
6	175000	398
7	130000	392
8	140000	384
9	165000	379
10	135000	371
11	250000	366
12	110000	360
13	115000	348
14	225000	334
15	170000	332
16	100000	329
17	155000	319

[Step 3's Results]

```sql
-- Looking at sale prices over time(#2)
-- Shows daily average sales prices


SELECT saledateconverted,
       ROUND(AVG(saleprice)) AS avg_sale_price
FROM nash_housing_eda
GROUP BY 1
ORDER BY 1
```

Data Output | Messages | Notifications

	saledateconverted date	avg_sale_price numeric
1	2013-01-02	284189
2	2013-01-03	99138
3	2013-01-04	137324
4	2013-01-07	210859
5	2013-01-08	102267

[Step 4's Results]

```sql
-- Looking at sales over time grouped by city and date(derivative of #2)


WITH city_sales AS (
                    SELECT propertysplitcity,
                           saledateconverted,
                           SUM(saleprice) AS total_sales
                    FROM nash_housing_eda
                    --WHERE propertysplitcity LIKE '%NASHVILLE%'
                    GROUP BY 1,2
                    ORDER BY 1,2
                    )

SELECT *,
       SUM(total_sales) OVER (PARTITION BY propertysplitcity
                              ORDER BY propertysplitcity, saledateconverted) AS running_total_sales
FROM city_sales
ORDER BY propertysplitcity, saledateconverted
```

Data Output | Messages | Notifications

	propertysplitcity text	saledateconverted date	total_sales bigint	running_total_sales numeric
1	ANTIOCH	2013-01-03	304000	304000
2	ANTIOCH	2013-01-04	831830	1135830
3	ANTIOCH	2013-01-07	349725	1485555
4	ANTIOCH	2013-01-08	139900	1625455
5	ANTIOCH	2013-01-09	190000	1815455

[Step 5's Results]

```sql
-- Looking at sale price by different land use(#9)
-- Cleaning landuse categories into a uniform structure


SELECT landuse,
       COUNT(*)
FROM nash_housing_eda
GROUP BY 1
ORDER BY 2 DESC


UPDATE nash_housing_eda
SET landuse = REPLACE(landuse,'VACANT RES LAND','VACANT RESIDENTIAL LAND')


UPDATE nash_housing_eda
SET landuse = REPLACE(landuse,'VACANT RESIENTIAL LAND','VACANT RESIDENTIAL LAND')


UPDATE nash_housing_eda
SET landuse = REPLACE(landuse,'RESIDENTIAL RESIDENTIAL CONDO','RESIDENTIAL CONDO')


UPDATE nash_housing_eda
SET landuse = REPLACE(landuse,'RESIDENTIAL CONDOMINIUM OFC  OR OTHER COM','RESIDENTIAL CONDO')
WHERE landuse LIKE '%RESIDENTIAL CONDOMINIUM OFC  OR OTHER COM%'
```

[Step 5's Results]

```sql
SELECT landuse,
       ROUND(AVG(saleprice)) AS avg_sale_price
FROM nash_housing_eda
GROUP BY 1
ORDER BY 2 DESC
```

Data Output | Messages | Notifications

	landuse text	avg_sale_price numeric
1	VACANT COMMERCIAL LAND	3235294
2	APARTMENT: LOW RISE (BUILT SINCE 1960)	2000000
3	DAY CARE CENTER	1577500
4	PARKING LOT	1225336
5	LIGHT MANUFACTURING	1200000
6	FOREST	1085330
7	CHURCH	840591

[Step 6's Results]

```
119  -- Looking at sale prices by bedroom and bathroom counts(#10)
120
121
122
123  SELECT bedroom,
124         fullbath,
125         halfbath,
126         COUNT(*) AS num_of_units,
127         ROUND(AVG(saleprice)) AS avg_sale_price
128  FROM nash_housing_eda
129  WHERE bedroom IS NOT NULL
130  AND fullbath IS NOT NULL
131  AND halfbath IS NOT NULL
132  GROUP BY 1,2,3
133  ORDER BY 4 DESC
134
```

Data Output | Messages | Notifications

	bedroom integer	fullbath integer	halfbath integer	num_of_units bigint	avg_sale_price numeric
1	3	2	0	4926	215119
2	2	1	0	3430	152899
3	3	1	0	2796	164496
4	3	1	1	2037	184741
5	4	2	0	1623	251828
6	3	2	1	1449	286543
7	3	3	0	1200	319701
8	4	3	0	1043	384018
9	2	2	0	925	216769
10	2	1	1	513	175627
11	4	2	1	487	362873
12	4	3	1	473	604079

[Step 7's Results]

```
141  SELECT propertysplitcity,
142         COUNT(*) AS num_of_properties
143  FROM nash_housing_eda
144  GROUP BY 1
145  ORDER BY 2 DESC
146
```

Data Output | Messages | Notifications

	propertysplitcity text	num_of_properties bigint
1	NASHVILLE	40216
2	ANTIOCH	6286
3	HERMITAGE	3126
4	MADISON	2114
5	BRENTWOOD	1696
6	OLD HICKORY	1415
7	GOODLETTSVILLE	735
8	NOLENSVILLE	494

[Step 8's Results]

```
153  SELECT saledateconverted,
154         COUNT(*) AS num_of_sales
155  FROM nash_housing_eda
156  GROUP BY 1
157  ORDER BY 1,2 DESC
158
```

Data Output | Messages | Notifications

	saledateconverted date	num_of_sales bigint
1	2013-01-02	9
2	2013-01-03	6
3	2013-01-04	21
4	2013-01-07	25
5	2013-01-08	18
6	2013-01-09	10
7	2013-01-10	20

[Step 9's Results]

```
213  SELECT propertysplitcity,
214         landuse,
215         COUNT(*) AS num_of_units
216  FROM nash_housing_eda
217  GROUP BY 1,2
218  ORDER BY 3 DESC
219
```

Data Output | Messages | Notifications

	propertysplitcity text	landuse text	num_of_units bigint
1	NASHVILLE	SINGLE FAMILY	23317
2	NASHVILLE	RESIDENTIAL CONDO	11867
3	ANTIOCH	SINGLE FAMILY	4124
4	NASHVILLE	VACANT RESIDENTIAL LAND	2931
5	HERMITAGE	SINGLE FAMILY	2151
6	MADISON	SINGLE FAMILY	1314
7	OLD HICKORY	SINGLE FAMILY	1134
8	NASHVILLE	DUPLEX	1111
9	ANTIOCH	VACANT RESIDENTIAL LAND	1085
10	BRENTWOOD	SINGLE FAMILY	993
11	ANTIOCH	RESIDENTIAL CONDO	875
12	NASHVILLE	ZERO LOT LINE	663
13	HERMITAGE	RESIDENTIAL CONDO	567
14	GOODLETTSVILLE	SINGLE FAMILY	503
15	BRENTWOOD	RESIDENTIAL CONDO	487
16	NOLENSVILLE	SINGLE FAMILY	361
17	MADISON	RESIDENTIAL CONDO	333
18	MADISON	VACANT RESIDENTIAL LAND	277
19	HERMITAGE	VACANT RESIDENTIAL LAND	209

Interpreting statistical findings:

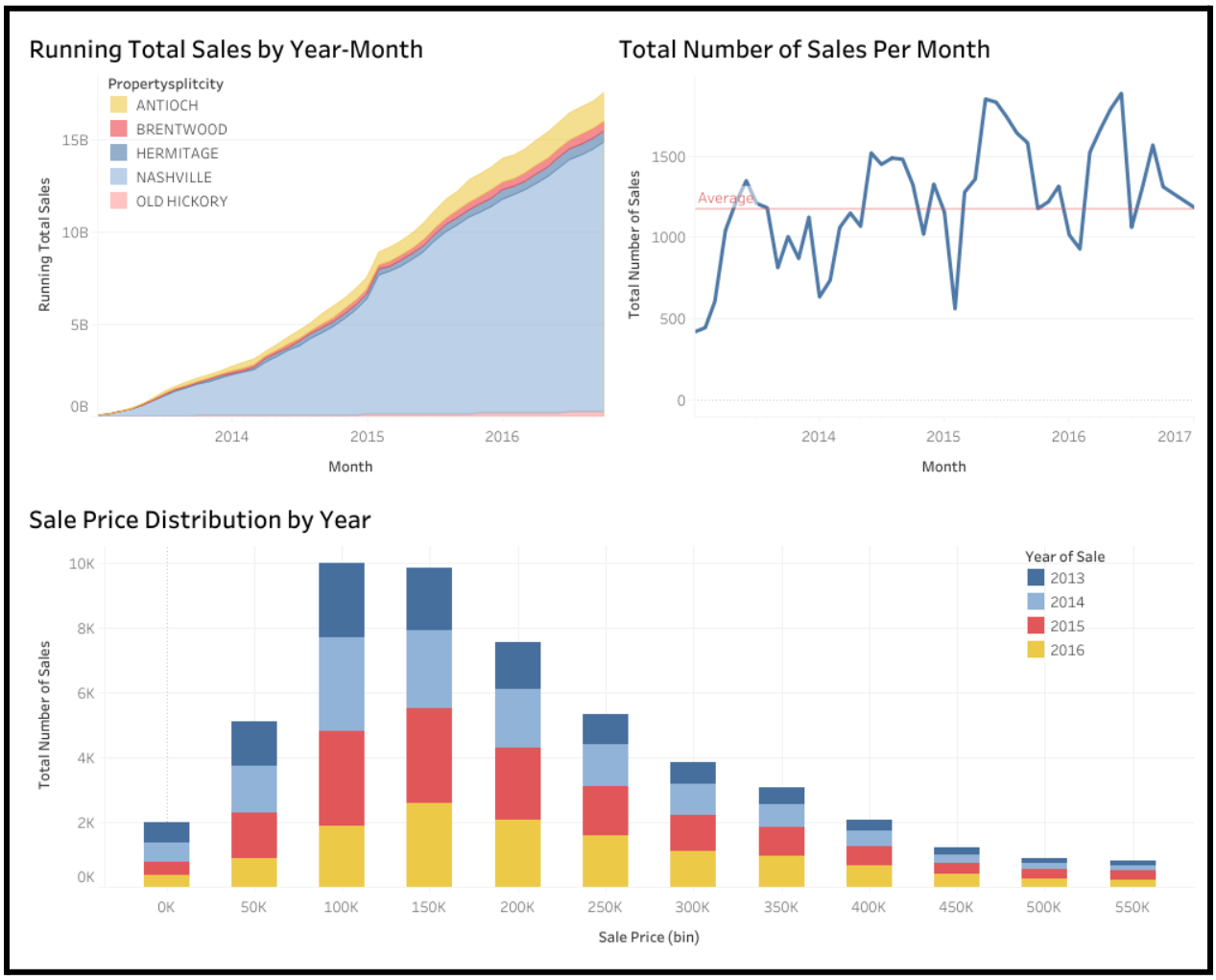
1. This project requires us to use the following **columns:** propertysplitycity, saledateconverted, saleprice, landuse, bedroom, fullbath, and halfbath.
2. Many nulls are present in the bedroom and bathroom columns so we'll filter these out when performing calculations for #6.
3. Looking at the distribution of sale prices in Nashville shows that most housing is priced in the range: **$100K - $250K**.
4. From January 2013 to January 2016, average sale prices have steadily grown by **11.29%** per year. Nashville is growing faster than the national average of **6.6%** during that time.
5. Nashville consists of six cities that produced approximately **$18.4 bln in housing sales** from 2013-2016. **Nashville** accounted for $14.5 bln(78.8%) of those sales.
6. The average sale prices for the most representative properties were: Condos $452K, Vacant. Res. Land $333K, Single Family $280K and Duplex $259K.
7. The average prices for distinct bed/bath pairings were: **3bed/2bath** $215k, **2bed/1bath** $152K, **3bed/1bath** $164k, **3bed/1.5bath** $184K, and **4bed/2bath** $251K.
8. Out of the approximately 56K properties in the Nashville metro area, roughly **40K(71.4%) properties** are located in Nashville.
9. Over the three years, there were **1,174 sales on average** each month. Average sales per month trend higher from May-July and lower in January and February.

. . .

Step 5: Share

In this step, we are creating visualizations and communicating our findings based on our analysis.

5.1 Data Visualizations and Findings

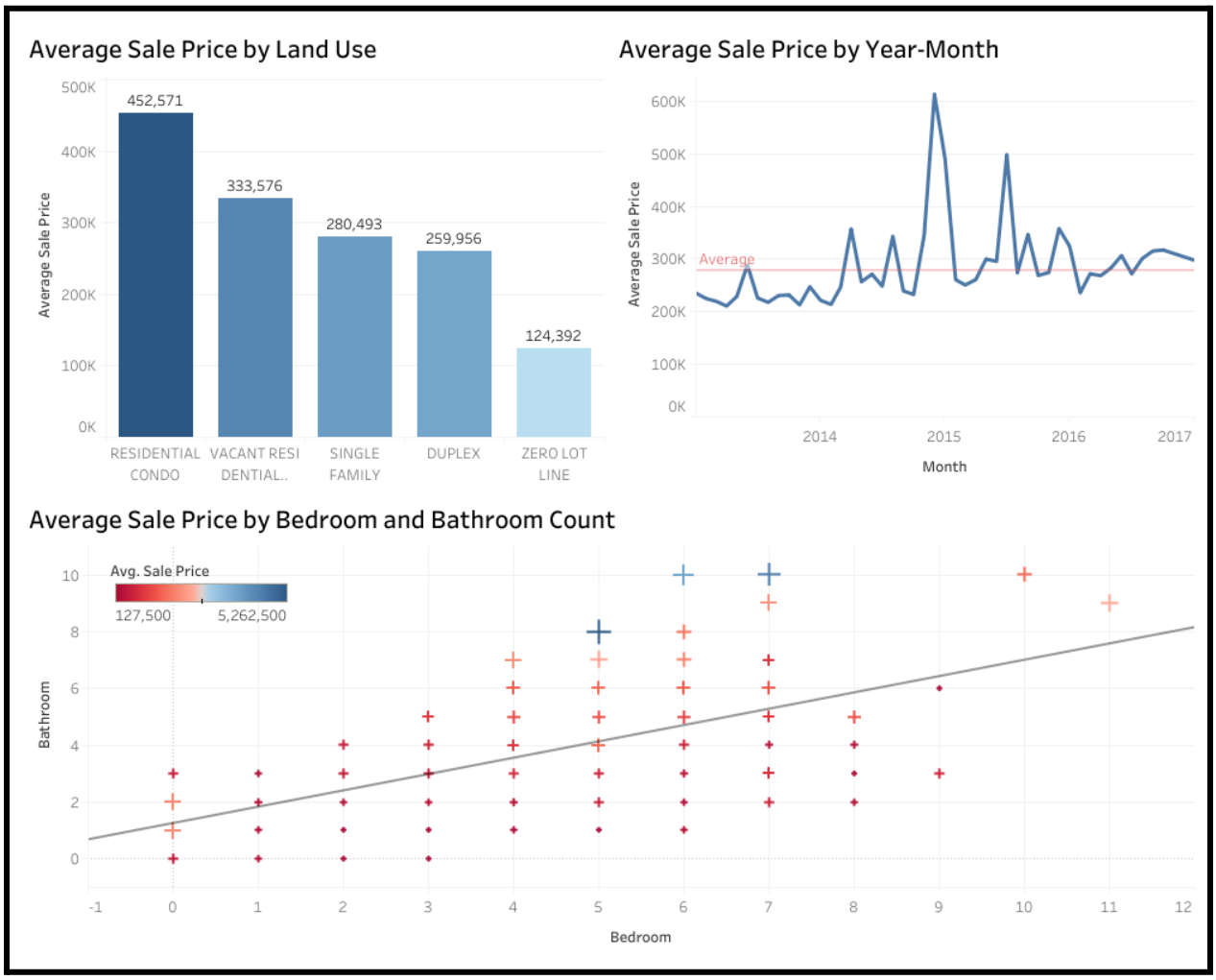


Housing Breakdown: Sales Prices and Volume Trends

This dashboard shows the overall sale price distribution, monthly sales volume compared to the average, and running total sales of properties across various municipalities in the Nashville metro area.

1. Between 2013 and 2016, the majority of Nashville's sale prices were concentrated between **$50K - $250K**. As we move **towards 2016**, the **percentage of higher sale prices** also increases.

2. There was a prominent increase in sales volume from 2013 to 2016. This trend is demonstrated by looking at the first two months of the year, where average sales rose from **432** (2013) to **973** (2016), with **686** (2014) and **859** (2015) in between.
3. Nashville had the highest running sales by a large margin compared to all other municipalities, accounting for **78.8% of total sales** during this period. Antioch was next highest, but **much lower at 8.2%**. Nashville's larger housing supply may be the cause of this.



Sale Price Trends: Land Use, Bed and Bath Count, and Year

This dashboard shows the pricing breakdown based on property land use, number of bedrooms and bathrooms, and year/month trends.

1. Condos have the highest price at **$452K**, followed in third by Single Family homes at **$280K** and Duplexes at **$259K**. This difference in affordability between Single Family homes and Condos may be due to the higher supply of Single Family homes (34K units) compared to Condos (14K units).
2. There was a significant increase in sales prices from 2013 to 2016. Looking at the first month of the year, the trend is obvious, sales prices rose from **$235K** (2013) to **$324K** (2016), with **$221K** (2014) and **$491K** (2015) positioned between.
3. The scatter plot shows that there is a **stronger trend** of increasing prices as the number of **bathrooms go up**, as opposed to when the number of bedrooms increases.

. . .

Step 6: Act

In the final step, we will be delivering our insights and providing recommendations based on our analysis.

Here, we revisit our business questions and share with you our high-level business recommendations.

1. **What are the trends in sale price?**
    - Nashville's home prices are rising and outpacing the national average. Despite the rising prices, there are still many affordable homes priced between $50K-$250K. As a result, we expect Nashville to remain a popular destination for relocation.

2. **What are the trends in housing inventory and total sales volume?**
    - Be mindful that the housing market in Nashville is growing rapidly, total sales revenue doubled each year while the total sales transactions increased 24% year over year. This may indicate that more higher-priced homes are being sold, while the uptick in sales

transactions may be driven by population growth, a strong local economy or appealing home prices.

3. **How do different variables affect price?**
    - Housing prices are influenced by its land use as well as how many bedrooms and bathrooms are present. Anyone relocating to Nashville will pay an average of $452K for condos, $280K for single family units, and $259K for a duplex. Also, consider that more bathrooms tend to correspond with higher property prices.