BUAN 6346 - BIG DATA

Sacramento Real Estate Transactions Analysis Project

Team 12
Pinghui Guo
Sushant Wavhal
Jiyun Yao
Long Zhao

CONTENTS

STREET	CITY	ZIPCODE	BEDROOMS	BATHS	AREA_SQ_FT	PRICE	PRICE_SQ_FT	PROPERTY_A	LOCALITY_RA	TYPE	SALE_DAY	SALE_DATE	LATTITUDE	LONGITUDE
3526 HIGH S	SACRAMENT	95838	2	1	836	59222	70.84	17	8	Residential	Wednesday	21-May	38.631913	-121.43488
51 OMAHA C	SACRAMENT	95823	3	1	1167	68212	58.45	10	8	Residential	Wednesday	21-May	38.478902	-121.43103
2796 BRANC	SACRAMENT	95815	2	1	796	68880	86.53	9	7	Residential	Wednesday	21-May	38.618305	-121.44384
2805 JANETT	SACRAMENT	95815	2	1	852	69307	81.35	25	3	Residential	Wednesday	21-May	38.616835	-121.43915
6001 MCMAI	SACRAMENT	95824	2	1	797	81900	102.76	15	5	Residential	Wednesday	21-May	38.51947	-121.43577
5828 PEPPER	SACRAMENT	95841	3	1	1122	89921	80.14	15	2	Condo	Wednesday	21-May	38.662595	-121.32781

/01

Dataset

The Sacramento real estate transactions file is a list of 985 real estate transactions in the Sacramento area reported over a five-day period, as reported by the Sacramento Bee.

Why this Dataset

- This dataset contains 15 columns and 986 rows, and the initial dataset is clear to analyze utilizing the knowledge we got in big data class to solve the real estate dilemmas in real world.
- Based on exploration, we figured out the dominant property type and explored other horizons for the real estate transactions in the Sacramento area, for instance, which city has the expensive/cheap houses, we were able to establish the relationships between the various aspects of a particular property.
- Therefore, we can give recommendations and predictions, helping the realtors design their realtoring strategy.

Clean Dataset

- Dropped useless columns which were not holding credibility in the analysis
- Divided particular columns for better analytic results.
- Cleaned the null values.

Load Data

- **Created Hive table from csv file**

```
zhaolong@cluster-7b8b-m:~/project$ hive -e "select * from transactions limit 5;"

Logging initialized using configuration in jar:file:/usr/lib/hive/lib/hive-common-2.1.1.jar!/hive-log4j2.properties Async: true
OK
3526 HIGH ST    SACRAMENTO      95838   2       1       836     59222   70.84   17      8       Residential     Wednesday       21-May  38.631912       -121.434875
51 OMAHA CT     SACRAMENTO      95823   3       1       1167    68212   58.45   10      8       Residential     Wednesday       21-May  38.4789 -121.43103
2796 BRANCH ST  SACRAMENTO      95815   2       1       796     68880   86.53   9       7       Residential     Wednesday       21-May  38.618305       -121.44384
2805 JANETTE WAY        SACRAMENTO      95815   2       1       852     69307   81.35   25      3       Residential     Wednesday       21-May  38.616837       -121.43915
6001 MCMAHON DR SACRAMENTO      95824   2       1       797     81900   102.76  15      5       Residential     Wednesday       21-May  38.51947        -121.43577
Time taken: 2.169 seconds, Fetched: 5 row(s)
```

- **Created spark dataframe from Hive table**

```
scala> df.show(5)
+---------------+----------+-------+--------+-----+----------+-----+----------+-----------+--------------+----------+---------+---------+---------+-----------+
|         street|      city|zipcode|bedrooms|baths|area_sq_ft|price|price_sq_ft|property_age|locality_rating|      type| sale_day|sale_date| lattitude|  longitude|
+---------------+----------+-------+--------+-----+----------+-----+----------+-----------+--------------+----------+---------+---------+---------+-----------+
|    3526 HIGH ST|SACRAMENTO|  95838|       2|    1|       836|59222|     70.84|         17|             8|Residential|Wednesday|   21-May|38.631912|-121.434875|
|     51 OMAHA CT|SACRAMENTO|  95823|       3|    1|      1167|68212|     58.45|         10|             8|Residential|Wednesday|   21-May|  38.4789| -121.43103|
|  2796 BRANCH ST|SACRAMENTO|  95815|       2|    1|       796|68880|     86.53|          9|             7|Residential|Wednesday|   21-May|38.618305| -121.44384|
|2805 JANETTE WAY|SACRAMENTO|  95815|       2|    1|       852|69307|     81.35|         25|             3|Residential|Wednesday|   21-May|38.616837| -121.43915|
| 6001 MCMAHON DR|SACRAMENTO|  95824|       2|    1|       797|81900|    102.76|         15|             5|Residential|Wednesday|   21-May| 38.51947| -121.43577|
+---------------+----------+-------+--------+-----+----------+-----+----------+-----------+--------------+----------+---------+---------+---------+-----------+
only showing top 5 rows
```

Dataset Exploration

Data types of columns

Table description

```
scala> spark.sql("describe transactions").show()
+--------------+---------+-------+
|      col_name|data_type|comment|
+--------------+---------+-------+
|        street|   string|   null|
|          city|   string|   null|
|       zipcode|   string|   null|
|      bedrooms|      int|   null|
|         baths|      int|   null|
|    area_sq_ft|      int|   null|
|         price|      int|   null|
|   price_sq_ft|    float|   null|
|  property_age|      int|   null|
|locality_rating|      int|   null|
|          type|   string|   null|
|      sale_day|   string|   null|
|     sale_date|   string|   null|
|      lattitude|    float|   null|
|     longitude|    float|   null|
+--------------+---------+-------+
```

```
scala> df.describe().show()
+-------+--------------------+--------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------+--------+--------+------------------+------------------+
|summary|              street|    city|           zipcode|          bedrooms|             baths|       area_sq_ft|             price|       price_sq_ft|      property_age|    locality_rating|  type| sale_day|sale_date|         lattitude|         longitude|
+-------+--------------------+--------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------+--------+--------+------------------+------------------+
|  count|                 985|     985|               985|               985|               985|               985|               985|               985|               985|               985|   985|     985|     985|               985|               985|
|   mean|                null|    null|  95750.69746192894| 3.369543147208122| 2.151269035532995|1667.0842639593912|260728.62538071067|156.12536056441098|15.012182741116751| 5.373604060913705|  null|    null|    null|38.607732069915926|-121.35598227408937|
| stddev|                null|    null|  85.17607208451431|0.9125849411497298|0.8229809697253422| 686.0933607022346|140514.23502884913|59.135808244367794| 6.061494976380461|2.3200508355763|  null|    null|    null|0.1454329068361815|0.13827799318782233|
|    min|   1 KENNELFORD CIR|ANTELOPE|             95603|                 1|                 1|               484|             40000|             37.67|                 5|                 2| Condo|  Friday|  15-May|          38.241512|         -121.551704|
|    max|9970 STATE HIGHWA...|  WILTON|             95864|                 8|                 5|              4822|            949000|             619.67|                25|                 9|Unkown|Wednesday|  21-May|          39.02081|         -120.597595|
+-------+--------------------+--------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------------------+------+--------+--------+------------------+------------------+
```

Exploration

/02

City column

- **Distinct values for city column**

```
scala> df.select("city").distinct().count()
res6: Long = 39
```

- **Null and missing values for city column**

```
scala> df.filter(isnan(df("city")) || isnull(df("city")) || df("city") === " " || df("city") === "").count()
res7: Long = 0
```

- **Top 10 cities with their percent ratio in the whole dataset**

```
+--------------+-----+------------------+
|          city|count|            ratio%|
+--------------+-----+------------------+
|    SACRAMENTO|  439|44.568527918781726|
|     ELK GROVE|  114|11.573604060913706|
|       LINCOLN|   72| 7.309644670050762|
|     ROSEVILLE|   48| 4.873096446700507|
|CITRUS HEIGHTS|   35|3.5532994923857872|
|      ANTELOPE|   33|3.3502538071065993|
|RANCHO CORDOVA|   28|2.8426395939086295|
|EL DORADO HILLS|   23|  2.33502538071066|
|          GALT|   21| 2.131979695431472|
|NORTH HIGHLANDS|   21| 2.131979695431472|
+--------------+-----+------------------+
only showing top 10 rows
```

Zipcode column

- **Distinct values for zipcode column**

```
scala> df.select("zipcode").distinct().count()
res6: Long = 68
```

- **Null and missing values for zipcode column**

```
scala> df.filter(isnan(df("zipcode")) || isnull(df("zipcode")) || df("zipcode") === " " || df("zipcode") === "").count()
res7: Long = 0
```

- **Top 10 zipcodes having the highest count with their percent ratios.**

zipcode	count	ratio%
95648	72	7.309644670050762
95823	61	6.192893401015229
95828	45	4.568527918781726
95758	44	4.467005076142132
95838	37	3.7563451776649748
95835	37	3.7563451776649748
95757	36	3.654822335025381
95624	34	3.451776649746193
95843	33	3.3502538071065993
95621	28	2.8426395939086295

Bedrooms column

- **Distinct values in the bedrooms column**

```
scala> df.select("bedrooms").distinct().count()
res9: Long = 7
```

- **Null and missing values in the bedrooms column**

```
scala> df.filter(isnan(df("bedrooms")) || isnull(df("bedrooms")) || df("bedrooms") === " " || df("bedrooms") === "").count()
res11: Long = 0
```

- **Bedrooms summary**

summary	bedrooms
count	985
mean	3.369543147208122
stddev	0.9125849411497298
min	1
max	8

- **Top 10 bedrooms count with their percent ratio**

bedrooms	count	ratio%
3	431	43.756345177664976
4	298	30.253807106598984
2	138	14.010152284263960
5	102	10.355329949238579
1	10	1.015228426395939
6	5	0.5076142131979695
8	1	0.10152284263959391

Baths column

- **Distinct values for the Baths column**

```
scala> df.select("bedrooms").distinct().count()
res9: Long = 7
```

- **Null and missing values for the Baths column**

```
scala> df.filter(isnan(df("bedrooms")) || isnull(df("bedrooms")) || df("bedrooms") === " " || df("bedrooms") === "").count()
res11: Long = 0
```

- **Baths summary**

```
+-------+------------------+
|summary|          bedrooms|
+-------+------------------+
|  count|               985|
|   mean| 3.369543147208122|
| stddev|0.9125849411497298|
|    min|                 1|
|    max|                 8|
+-------+------------------+
```

- **Top 10 Baths count with their percent ratio**

```
+--------+-----+------------------+
|bedrooms|count|            ratio%|
+--------+-----+------------------+
|       3|  431|43.756345177664976|
|       4|  298|30.253807106598984|
|       2|  138| 14.01015228426396|
|       5|  102|10.355329949238579|
|       1|   10| 1.015228426395939|
|       6|    5|0.5076142131979695|
|       8|    1|0.10152284263959391|
+--------+-----+------------------+
```

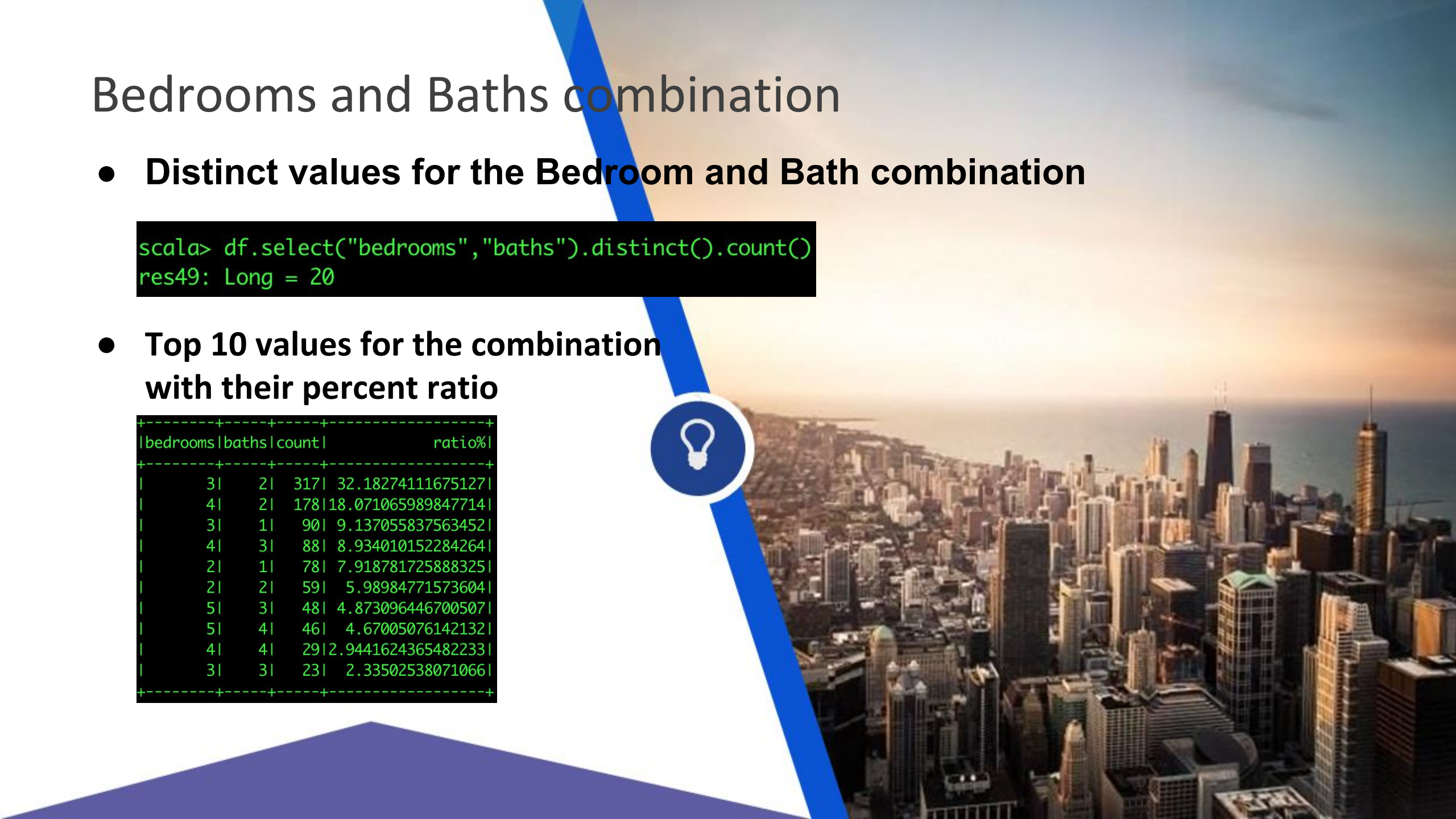
Bedrooms and Baths combination

- **Distinct values for the Bedroom and Bath combination**

```
scala> df.select("bedrooms","baths").distinct().count()
res49: Long = 20
```

- **Top 10 values for the combination with their percent ratio**

bedrooms	baths	count	ratio%
3	2	317	32.18274111675127
4	2	178	18.071065989847714
3	1	90	9.137055837563452
4	3	88	8.934010152284264
2	1	78	7.918781725888325
2	2	59	5.98984771573604
5	3	48	4.873096446700507
5	4	46	4.67005076142132
4	4	29	2.9441624365482233
3	3	23	2.33502538071066



Area column

- **Distinct values for the area column**

```
scala> df.select("area_sq_ft").distinct().count()
res20: Long = 681
```

- **Null and missing values in the area column**

```
scala> df.filter(isnan(df("area_sq_ft")) || isnull(df("area_sq_ft")) || df("area_sq_ft") === " " || df("area_sq_ft") === "").count()
res23: Long = 0
```

- **Area summary**

summary	area_sq_ft
count	985
mean	1667.084263959391
stddev	686.0933607022346
min	484
max	4822

- **Top 10 areas with their percent ratio**

area_sq_ft	count	ratio%
1955	50	5.0761421319796955
1120	8	0.8121827411167513
795	6	0.6091370558375634
1410	5	0.5076142131979695
1499	5	0.5076142131979695
1039	5	0.5076142131979695
1080	5	0.5076142131979695
1302	4	0.40609137055837563
1144	4	0.40609137055837563
1475	4	0.40609137055837563

Price column

- **Distinct values in price column**

```
scala> df.select("price").distinct().count()
res26: Long = 604
```

- **Null and missing values in price column**

```
scala> df.filter(isnan(df("price")) || isnull(df("price")) || df("price") === " " || df("price") === "").count()
res28: Long = 0
```

- **Price summary**

summary	price
count	985
mean	260728.62538071067
stddev	140514.23502884913
min	40000
max	949000

- **Top 10 prices with their percent ratio**

price	count	ratio%
489700	50	5.0761421319796955
220000	10	1.015228426395939
150000	9	0.9137055837563453
250000	9	0.9137055837563453
170000	9	0.9137055837563453
300000	8	0.8121827411167513
205000	8	0.8121827411167513
145000	7	0.7106598984771574
180000	7	0.7106598984771574
215000	7	0.7106598984771574

Property Age column

- **Distinct values in Property age column**

```
scala> df.select("property_age").distinct().count()
res36: Long = 21
```

- **Null and missing values**

```
scala> df.filter(isnan(df("property_age")) || isnull(df("property_age")) || df("property_age") === " " || df("property_age") === "").count(
res38: Long = 0
```

- **Property age summary**

summary	property_age
count	985
mean	15.012182741116751
stddev	6.0614949763804615
min	5
max	25

- **Top 10 property ages with their percent ratio**

property_age	count	ratio%
20	54	5.4822335025380715
22	53	5.380710659898477
13	52	5.279187817258883
11	52	5.279187817258883
12	50	5.0761421319796955
5	50	5.0761421319796955
8	50	5.0761421319796955
7	50	5.0761421319796955
24	50	5.0761421319796955
14	49	4.974619289340102

Locality Rating column

- **Distinct values in Property rating column**

```
scala> df.select("LOCALITY_RATING").distinct().count()
res41: Long = 8
```

- **Null and missing values**

```
scala> df.filter(isnan(df("LOCALITY_RATING")) || isnull(df("LOCALITY_RATING")) || df("LOCALITY_RATING") === " " || df("LOCALITY_RATING") === "").count()
res43: Long = 0
```

- **Locality rating summary**

summary	LOCALITY_RATING
count	985
mean	5.3736040609137055
stddev	2.3200508355763
min	2
max	9

- **Top 10 property ratings with their percent ratio**

LOCALITY_RATING	count	ratio%
2	141	14.31472081218274
3	135	13.705583756345177
8	122	12.385786802030458
7	122	12.385786802030458
5	121	12.284263959390863
4	116	11.776649746192893
6	115	11.6751269035533
9	113	11.472081218274113

Type column

- **Distinct values in Type column**

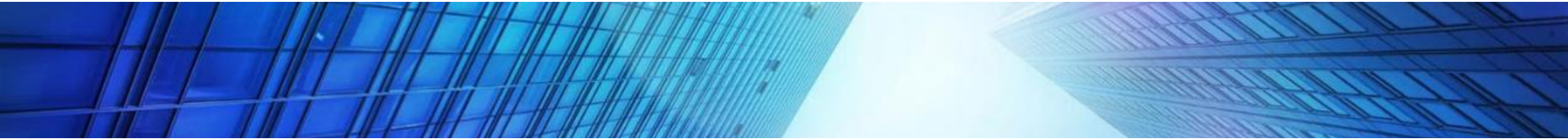
```
scala> df.select("type").distinct().count()
res46: Long = 4
```

- **Null and missing values**

```
scala> df.filter(isnan(df("type")) || isnull(df("type")) || df("type") === " " || df("type") === "").count()
res47: Long = 0
```

- **Percent ratio of each property type**

```
+------------+-----+------------------+
|        type|count|            ratio%|
+------------+-----+------------------+
| Residential|  917|  93.09644670050761|
|       Condo|   54| 5.482233502538071|
|Multi-Family|   13| 1.319796954314721|
|      Unkown|    1|0.10152284263959391|
+------------+-----+------------------+
```



Analysis

Tools: Hive and Spark

/03

Analysis I

❖ City based analysis

1.1 Average price for each city

```
// Average  of  Price
df.groupBy("city").avg("price").sort(avg("price").desc).show()
+---------------+------------------+
|           city|        avg(price)|
+---------------+------------------+
|    SLOUGHHOUSE|          949000.0|
|    GRANITE BAY| 678733.3333333334|
|         WILTON|          617508.4|
|         LOOMIS|          567000.0|
|         PENRYN|          506688.0|
|EL DORADO HILLS|491698.95652173914|
|  GARDEN VALLEY|          490000.0|
|        LINCOLN|436940.59722222225|
|         FOLSOM|414960.17647058825|
|         AUBURN|          405890.8|
|      GREENWOOD|          395000.0|
|        ROCKLIN|381835.82352941175|
|   WALNUT GROVE|          380000.0|
|    PLACERVILLE|          363863.4|
|     GOLD RIVER|          358000.0|
|      ROSEVILLE|         324528.25|
|      FAIR OAKS| 303500.6666666667|
|           COOL|          300000.0|
| RANCHO MURIETA|          297750.0|
|     CARMICHAEL|         295684.75|
+---------------+------------------+
only showing top 20 rows
```

Analysis 1

1.2

Average square footage for each city

```
// Average  of square footage
df.groupBy("city").avg("area_sq_ft").sort(avg("area_sq_ft").desc).show()
+---------------+------------------+
|           city|   avg(area_sq_ft)|
+---------------+------------------+
|    SLOUGHHOUSE|            4822.0|
|         WILTON|            3487.4|
|    GRANITE BAY|            2974.0|
|      GREENWOOD|            2846.0|
|EL DORADO HILLS| 2556.695652173913|
|  GARDEN VALLEY|            2475.0|
|        ROCKLIN|2231.6470588235293|
| RANCHO MURIETA|            2225.0|
|         FOLSOM| 2200.764705882353|
|         AUBURN|            2191.4|
|         MATHER|            2093.0|
|     GOLD RIVER|           1993.25|
| RANCHO CORDOVA| 1955.642857142857|
|      ELK GROVE| 1950.517543859649|
|        LINCOLN|1915.0833333333333|
|      ROSEVILLE|1867.4166666666667|
|   WALNUT GROVE|            1727.0|
|    PLACERVILLE|            1717.9|
|       ANTELOPE|1716.6363636363637|
|      FAIR OAKS|1659.7777777777778|
+---------------+------------------+
only showing top 20 rows
```

Analysis 1

1.3

Average age of the property for every city

```
// Average  of Age of the House
df.groupBy("city").avg("property_age").sort(avg("property_age").desc).show()
+---------------+------------------+
|           city| avg(property_age)|
+---------------+------------------+
|      GREENWOOD|              25.0|
|     FORESTHILL|              24.0|
|         PENRYN|              23.0|
|   MEADOW VISTA|              23.0|
|    GRANITE BAY|22.333333333333332|
|         AUBURN|              19.0|
|WEST SACRAMENTO|              19.0|
|DIAMOND SPRINGS|              19.0|
|     GOLD RIVER|             18.25|
|           COOL|              18.0|
|         LOOMIS|              17.5|
|         MATHER|              17.0|
|         FOLSOM|              16.0|
| CITRUS HEIGHTS|15.857142857142858|
|        ELVERTA|             15.75|
|   CAMERON PARK|15.555555555555555|
|      FAIR OAKS|15.333333333333334|
|     SACRAMENTO|15.287015945330296|
|    PLACERVILLE|              15.2|
|       ANTELOPE|15.030303030303031|
+---------------+------------------+
only showing top 20 rows
```

Analysis 1

1.4

Total number of transactions in each city

```
// Total number of transactions in each city
df.groupBy("city").count().orderBy(desc("count")).show()
+---------------+-----+
|           city|count|
+---------------+-----+
|     SACRAMENTO|  439|
|      ELK GROVE|  114|
|        LINCOLN|   72|
|      ROSEVILLE|   48|
| CITRUS HEIGHTS|   35|
|       ANTELOPE|   33|
| RANCHO CORDOVA|   28|
|EL DORADO HILLS|   23|
|NORTH HIGHLANDS|   21|
|           GALT|   21|
|     CARMICHAEL|   20|
|        ROCKLIN|   17|
|         FOLSOM|   17|
|      RIO LINDA|   13|
|     ORANGEVALE|   11|
|    PLACERVILLE|   10|
|      FAIR OAKS|    9|
|   CAMERON PARK|    9|
|         AUBURN|    5|
|         WILTON|    5|
+---------------+-----+
only showing top 20 rows
```

Analysis 1

1.5

Prime property Type in each city

```
// Prime property  Type
df.groupBy("city","type").count().orderBy(desc("count")).show()
+--------------+------------+-----+
|          city|        type|count|
+--------------+------------+-----+
|    SACRAMENTO| Residential|  402|
|     ELK GROVE| Residential|  108|
|       LINCOLN| Residential|   70|
|     ROSEVILLE| Residential|   41|
|      ANTELOPE| Residential|   32|
|CITRUS HEIGHTS| Residential|   32|
|    SACRAMENTO|       Condo|   27|
|RANCHO CORDOVA| Residential|   26|
|EL DORADO HILLS| Residential|   23|
|NORTH HIGHLANDS| Residential|   21|
|          GALT| Residential|   21|
|    CARMICHAEL| Residential|   17|
|       ROCKLIN| Residential|   17|
|        FOLSOM| Residential|   16|
|     RIO LINDA| Residential|   13|
|    ORANGEVALE| Residential|   11|
|    SACRAMENTO|Multi-Family|   10|
|   PLACERVILLE| Residential|   10|
|  CAMERON PARK| Residential|    8|
|     FAIR OAKS| Residential|    8|
+--------------+------------+-----+
only showing top 20 rows
```

Analysis I

1.6 Distinct Zip-codes for each city (39 cities and 69 zip-codes)

```
// Distinct Zip-codes  for each city
df.rollup("city","zipcode").count().orderBy("city").show(150)
```

city	zipcode	count
null	null	985
ANTELOPE	null	33
ANTELOPE	95843	33
AUBURN	95603	5
AUBURN	null	5
CAMERON PARK	null	9
CAMERON PARK	95682	9
CARMICHAEL	95608	20
CARMICHAEL	null	20
CITRUS HEIGHTS	null	35
CITRUS HEIGHTS	95621	28
CITRUS HEIGHTS	95610	7
COOL	95614	1
COOL	null	1
DIAMOND SPRINGS	null	1
DIAMOND SPRINGS	95619	1
EL DORADO	null	2
EL DORADO	95623	2
EL DORADO HILLS	95762	23
EL DORADO HILLS	null	23
ELK GROVE	95758	44
ELK GROVE	95624	34
ELK GROVE	95757	36
ELK GROVE	null	114
ELVERTA	null	4
ELVERTA	95626	4
FAIR OAKS	null	9
FAIR OAKS	95628	9
FOLSOM	95630	17
FOLSOM	null	17
FORESTHILL	95631	1
FORESTHILL	null	1
GALT	null	21
GALT	95632	21
GARDEN VALLEY	null	1
GARDEN VALLEY	95633	1
GOLD RIVER	95670	4
GOLD RIVER	null	4
GRANITE BAY	null	3
GRANITE BAY	95746	3
GREENWOOD	95635	1
GREENWOOD	null	1
LINCOLN	null	72
LINCOLN	95648	72
LOOMIS	95650	2
LOOMIS	null	2
MATHER	95655	1
MATHER	null	1
MEADOW VISTA	95722	1
MEADOW VISTA	null	1
NORTH HIGHLANDS	null	21
NORTH HIGHLANDS	95660	21
ORANGEVALE	null	11
ORANGEVALE	95662	11
PENRYN	null	1
PENRYN	95663	1
PLACERVILLE	95667	10
PLACERVILLE	null	10
POLLOCK PINES	null	3
POLLOCK PINES	95726	3
RANCHO CORDOVA	95670	17
RANCHO CORDOVA	null	28
RANCHO CORDOVA	95742	11
RANCHO MURIETA	95683	3
RANCHO MURIETA	null	3
RIO LINDA	95673	13
RIO LINDA	null	13
ROCKLIN	null	17
ROCKLIN	95765	11
ROCKLIN	95677	6
ROSEVILLE	95678	20
ROSEVILLE	95661	8
ROSEVILLE	null	48
ROSEVILLE	95747	20
SACRAMENTO	95822	24
SACRAMENTO	95826	18
SACRAMENTO	95819	4
SACRAMENTO	95841	7
SACRAMENTO	95833	20
SACRAMENTO	95824	12
SACRAMENTO	95864	5
SACRAMENTO	95817	7
SACRAMENTO	95831	10
SACRAMENTO	95838	37
SACRAMENTO	95827	9
SACRAMENTO	95825	13
SACRAMENTO	95821	6
SACRAMENTO	95829	11
SACRAMENTO	95832	12
SACRAMENTO	95835	37
SACRAMENTO	95820	23
SACRAMENTO	95823	61
SACRAMENTO	95834	22
SACRAMENTO	null	439
SACRAMENTO	95811	2
SACRAMENTO	95842	22
SACRAMENTO	95828	45
SACRAMENTO	95818	7
SACRAMENTO	95816	4
SACRAMENTO	95815	18
SACRAMENTO	95814	3
SHINGLE SPRINGS	95682	1
SHINGLE SPRINGS	null	1
SLOUGHHOUSE	95683	1
SLOUGHHOUSE	null	1
WALNUT GROVE	95690	1
WALNUT GROVE	null	1
WEST SACRAMENTO	null	3
WEST SACRAMENTO	95691	3
WILTON	95693	5
WILTON	null	5

// sum of this city

Analysis 2

Which type of property is selling well

2.1 Categorizing them based on Beds and Baths.

3B2B is the property that is selling well

```
// depending on beds and baths
df.groupBy("bedrooms","baths").count().orderBy(desc("count")).show()
+--------+-----+-----+
|bedrooms|baths|count|
+--------+-----+-----+
|       3|    2|  317|
|       4|    2|  178|
|       3|    1|   90|
|       4|    3|   88|
|       2|    1|   78|
|       2|    2|   59|
|       5|    3|   48|
|       5|    4|   46|
|       4|    4|   29|
|       3|    3|   23|
|       1|    1|   10|
|       5|    2|    6|
|       4|    1|    3|
|       6|    4|    3|
|       5|    5|    2|
|       2|    3|    1|
|       6|    5|    1|
|       3|    4|    1|
|       6|    3|    1|
|       8|    4|    1|
+--------+-----+-----+
```

Analysis 2

Which type of property is selling well

2.2 Categorizing them based on Area of the property (sq.ft)

Area_sqft_range = 1.0 i.e Property with area between 1000 & 2000 square footage is selling well

```
//depending on area(sqft) range

import org.apache.spark.ml.feature.Bucketizer

val splits = Range.Double(0,6000,1000).toArray

val bucketizer = new Bucketizer().setInputCol("area_sq_ft").setOutputCol("area_sqft_range").setSplits(splits)

val df2 = bucketizer.transform(df)

df2.groupBy("area_sqft_range").count().orderBy(desc("count")).show()

// 0.0: 0-1000; 1.0: 1000-2000; 2.0: 2000-3000; 3.0: 3000-4000; 4.0: 4000-5000
+---------------+-----+
|area_sqft_range|count|
+---------------+-----+
|            1.0|  667|
|            2.0|  147|
|            0.0|  109|
|            3.0|   57|
|            4.0|    5|
+---------------+-----+
```

Analysis 3

- ***Positive/Negative relationship***

Very weak positive Relation between Rate(price/sq.ft) and Transaction(Price)

City wise referencing.

```
// III  Positive/Negative relationship
// between Rate  and Price
// very weak positive relationship
df.stat.corr("locality_rating", "price")
res88: Double = 0.046236652073434 53

df.stat.corr("locality_rating", "price_sq_ft")
res89: Double = 0.006328063486050017
```

Analysis 4

- ***Which property type falls into a particular group(Locality_rating, age group)***

4. 1 Groups of rating (0-2, 3-5, 6-10)

Number of properties persistent in each of this group

472 in 6-10, 372 in 3-5, 141 in 0-2

```
// 1. Groups of rating
import org.apache.spark.ml.feature.Bucketizer

val splits = Range.Double(0,10,3).toArray

val bucketizer = new Bucketizer().setInputCol("locality_rating").setOutputCol("rating_group").setSplits(splits)

val df2 = bucketizer.transform(df)

df2.groupBy("rating_group").count().orderBy(desc("count")).show()
// 0.0: 0-2; 1.0: 3-5; 2.0: 6-9
+------------+-----+
|rating_group|count|
+------------+-----+
|         2.0|  472|
|         1.0|  372|
|         0.0|  141|
+------------+-----+
```

Analysis 4

Which property type falls into a particular group(Locality_rating, age group)

4. 2. Age groups (5-10, 11-17, 18-25)

Number of properties persistent in these groups.

234 in 5-9, 245 in 10-14, 214 in 15-19, 292 in 20-25

```
// 2. Age groups
import org.apache.spark.ml.feature.Bucketizer

val splits = Range.Double(5,26,5).toArray

val bucketizer = new Bucketizer().setInputCol("property_age").setOutputCol("age_group").setSplits(splits)

val df2 = bucketizer.transform(df)

df2.groupBy("age_group").count().orderBy(desc("count")).show()
//0.0: 5-9; 1.0: 10-14; 2.0: 15-19; 3.0: 20-25
+---------+-----+
|age_group|count|
+---------+-----+
|      3.0|  292|
|      1.0|  245|
|      0.0|  234|
|      2.0|  214|
+---------+-----+
```

Analysis 5

- ***Which city has the most expensive houses (based on their categories-beds and baths)***

City: **Sloughhouse**

Type: **3B4B**

Price: **$949, 000**

```
// most expensive house, city, bedrooms, baths
df.groupBy("city","bedrooms","baths").max("price").orderBy(max("price").desc).show(10)
+---------------+--------+-----+----------+
|           city|bedrooms|baths|max(price)|
+---------------+--------+-----+----------+
|     SLOUGHHOUSE|       3|    4|    949000|             <---
|         WILTON|       4|    3|    884790|
|EL DORADO HILLS|       4|    3|    879000|
|         LOOMIS|       4|    4|    839000|
|EL DORADO HILLS|       6|    5|    830000|
|     GRANITE BAY|       5|    3|    760000|
|        LINCOLN|       3|    3|    755100|
|     SACRAMENTO|       5|    2|    699000|
|         WILTON|       5|    3|    691659|
|EL DORADO HILLS|       5|    5|    680000|
+---------------+--------+-----+----------+
```

Analysis 5

- ***5.2 Which city has the lowest expense houses***

  City: **North Highlands**

  Type: **2B1B**

  Price: **$63, 000**

```
// the cheapest house, city, bedrooms, baths
df.groupBy("city","bedrooms","baths").max("price").orderBy(max("price")).show(10)
+----------------+--------+-----+-----------+
|            city|bedrooms|baths|max(price)|
+----------------+--------+-----+-----------+
|NORTH HIGHLANDS|       2|    1|      63000|                <---
|       ELK GROVE|       2|    1|      71000|
| RANCHO MURIETA|       2|    2|      97750|
|       ELK GROVE|       1|    1|     100000|
| RANCHO CORDOVA|       2|    1|     115000|
| CITRUS HEIGHTS|       2|    1|     116250|
|         ELVERTA|       4|    2|     126714|
|NORTH HIGHLANDS|       2|    2|     131750|
|       RIO LINDA|       2|    1|     132000|
| RANCHO CORDOVA|       3|    1|     134000|
+----------------+--------+-----+-----------+
```

Thank you !