CIS 5200 Term Project Tutorial

Authors: Jeremy Contreras, Kanchon Bishnu, Yu Wang

Instructor: Jongwook Woo

Date: 05/09/2025

Lab Tutorial

Jeremy Contreras (jcontr185@calstatela.edu)

Kanchon Bishnu(kbishnu@calstatela.edu)

Yu Wang(ywang289@calstatela.edu)

UK Housing Prices Paid Analysis

Objectives

In this hands-on lab, you will learn how to:

- Download Data using Kaggle API & upload to Hadoop distributed file system (hdfs)
- Create External tables using Hive
- Query UK Housing Prices Paid dataset using Apache Hive commands
- Export data & Perform data visualization using Microsoft Excel

Platform Spec

- CPU Speed: 2.45 GHz
- # of CPU cores: 3
- # of nodes: 5 nodes, 2 master, 3 worker
- Total Memory Size: 155 GB

Step 1: Download Data using Kaggle API and Upload to HDFS

Perform the steps to download data and upload to hdfs

1. Create Kaggle API key at https://www.kaggle.com/settings . Select create new token under API to download Kaggle.json .

2. **Transfer kaggle.json to the remote machine** (your Hadoop cluster node). Replace: "**C:\Users\Jeremy Contreras\Downloads\kaggle.json"** with your own directory to kaggle.json and ssh login and cluster ip address. Replace jcontr185 with your account name.

   ```
   scp "C:\Users\Jeremy Contreras\Downloads\kaggle.json" jcontr185@144.24.13.0:~
   ```

3. Open another terminal and SSH into cluster:

   ssh jcontr185**@**144.24.13.0

   Note: Change jcontr185 to your own username. Change with own ip address for cluster: 144.24.13.0.

4. Install Kaggle API:

   ```
   pip install kaggle
   ```

Note: Install pip if not already installed and ssh into cluster:

#. On the cluster, install pip for Python 3.6

 # Download the Python-3.6 compatible get-pip installer

```
curl -sS https://bootstrap.pypa.io/pip/3.6/get-pip.py -o get-pip.py
```

 # Run it under python3 on the cluster

```
python3 get-pip.py --user
```

5. **Add your local bin dir into your PATH:**

```
echo 'export PATH="$HOME/.local/bin:$PATH"' >> ~/.bashrc
```

```
source ~/.bashrc
```

6. **Verify pip & install Kaggle CLI**

```
pip3 --version
```

```
pip3 install --user kaggle
```

```
kaggle --version
```

7. **Download the dataset from Kaggle:**

```
kaggle datasets download -d hm-land-registry/uk-housing-prices-paid -f price_paid_records.csv
```

8. **List files to verify the download**: ls

9. Unzip the file:

```
unzip price_paid_records.csv.zip
```

10. Create file directory. Note: replace jcontr185 with your account name:

```
hdfs dfs -mkdir -p /user/jcontr185/hive_data/price_paid_data
```

11. Upload to hdfs. Note replace jcontr185 with your account name:

```
hdfs dfs -put price_paid_records.csv /user/jcontr185/hive_data/price_paid_data/
```

12. Verify. Note replace jcontr185 with your account name:

```
hdfs dfs -ls -h /user/jcontr185/hive_data/price_paid_data/
```

```
-bash-4.2$ hdfs dfs -ls -h /user/jcontr185/hive_data/price_paid_data/
Found 1 items
-rw-r--r--   3 jcontr185 hdfs      2.2 G 2025-03-27 23:17 /user/jcontr185/hive_data/price_paid_data/price_paid_records_copy_1.csv
-bash-4.2$
```

Step 2: Create External Table Using Hive

‘

This step is to create external tables using Hive.

1. Enter: beeline
2. Use your database: USE jcontr185;

3. Create external table that stores summarized UK housing transaction data based on the Prices Paid dataset. Note: Replace jcontr185 with your account name.

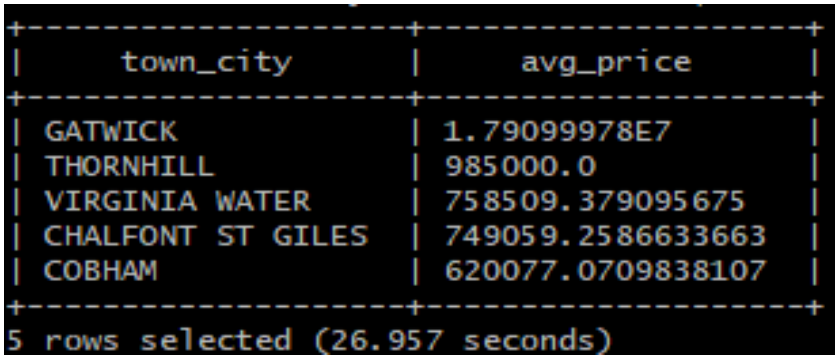
```
CREATE EXTERNAL TABLE IF NOT EXISTS uk_housing_summary (

    transaction_unique_id STRING,

    housing_prices BIGINT,

    transfer_date STRING,

    property_type STRING,

    old_new STRING,

    duration STRING,

    town_city STRING,

    district STRING,

    county STRING,

    ppd_category STRING

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

LOCATION '/user/jcontr185/hive_data/price_paid_data'

TBLPROPERTIES ('skip.header.line.count'='1');
```

Step 3: Query UK Housing Prices Paid dataset using Hive commands

This step is to query data using Hive for analysis.

1. Find the top 5 cities with highest average housing price from uk_housing_summary table.

```
SELECT town_city, AVG(housing_prices) AS avg_price
FROM uk_housing_summary
GROUP BY town_city
ORDER BY avg_price DESC
LIMIT 5;
```



town_city	avg_price
GATWICK	1.79099978E7
THORNHILL	985000.0
VIRGINIA WATER	758509.379095675
CHALFONT ST GILES	749059.2586633663
COBHAM	620077.0709838107

5 rows selected (26.957 seconds)

2. Query for transaction_unique_id, housing_prices, town_city columns from uk_housing_summary table.

```
SELECT transaction_unique_id, housing_prices, town_city
FROM uk_housing_summary
LIMIT 10;
```

transaction_unique_id	housing_prices	town_city
{81B82214-7FBC-4129-9F6B-4956B4A663AD}	25000	OLDHAM
{8046EC72-1466-42D6-A753-4956BF7CD8A2}	42500	GRAYS
{278D581A-5BF3-4FCE-AF62-4956D87691E6}	45000	HIGHBRIDGE
{1D861C06-A416-4865-973C-4956DB12CD12}	43150	BEDFORD
{DD8645FD-A815-43A6-A7BA-4956E58F1874}	18899	WAKEFIELD
{895E4E63-203F-476A-9AA9-42389DD0AE5C}	81750	SALISBURY
{FB195C27-E790-45FD-847A-4238BC94546A}	56000	WITNEY
{1D6B01EC-DC33-4147-8A21-4238BEB2D4C1}	31000	ST. AUSTELL
{B8D0F817-4553-448D-A2C1-4238BF81C6FA}	82000	GREENFORD
{6DD27423-CC39-4B31-A848-4238D58268D4}	10000	FERNDALE

10 rows selected (0.254 seconds)

3. Query to find property type ordered by count in descending order.

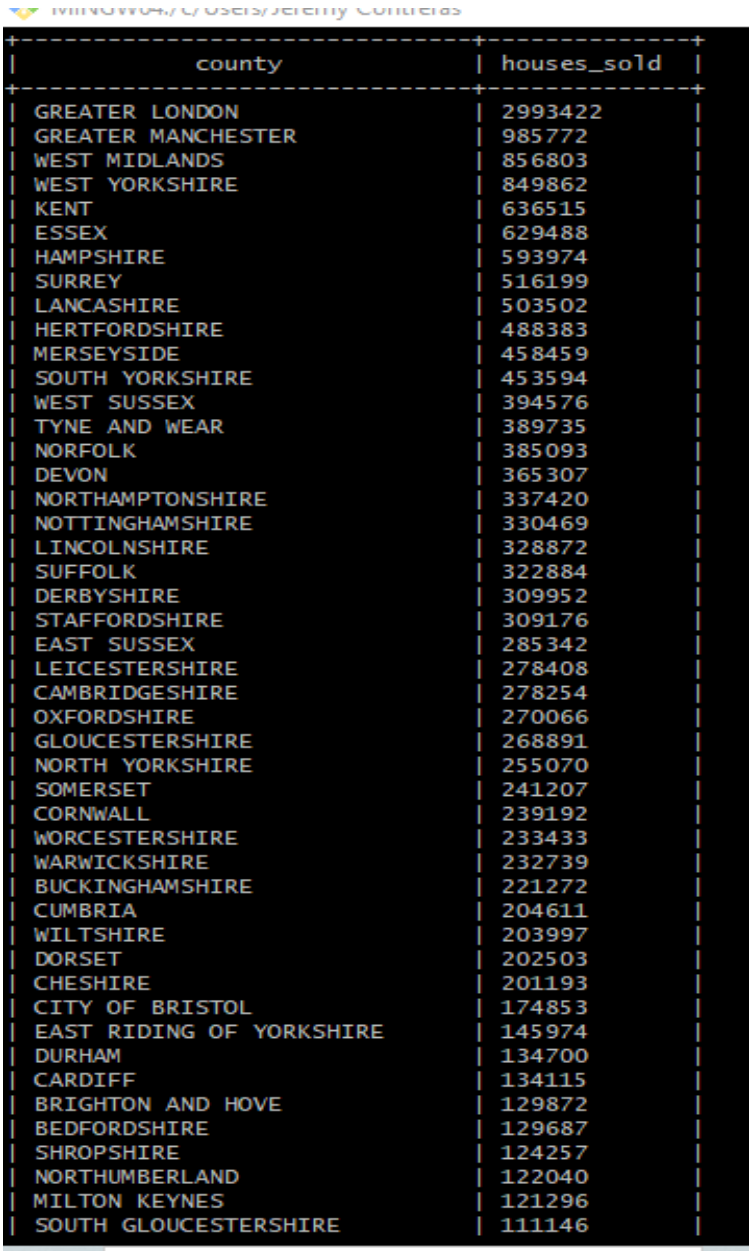
```
SELECT property_type, COUNT(*) AS count

FROM uk_housing_summary

GROUP BY property_type

ORDER BY count DESC;
```

property_type	count
T	6918811
S	6216218
D	5170327
F	4083424
O	100568

5 rows selected (51.188 seconds)

4. Query to find count of houses sold by county.

```
SELECT county, COUNT(*) AS houses_sold FROM uk_housing_summary GROUP BY county   ORDER BY

houses_sold DESC;
```



county	houses_sold
GREATER LONDON	2993422
GREATER MANCHESTER	985772
WEST MIDLANDS	856803
WEST YORKSHIRE	849862
KENT	636515
ESSEX	629488
HAMPSHIRE	593974
SURREY	516199
LANCASHIRE	503502
HERTFORDSHIRE	488383
MERSEYSIDE	458459
SOUTH YORKSHIRE	453594
WEST SUSSEX	394576
TYNE AND WEAR	389735
NORFOLK	385093
DEVON	365307
NORTHAMPTONSHIRE	337420
NOTTINGHAMSHIRE	330469
LINCOLNSHIRE	328872
SUFFOLK	322884
DERBYSHIRE	309952
STAFFORDSHIRE	309176
EAST SUSSEX	285342
LEICESTERSHIRE	278408
CAMBRIDGESHIRE	278254
OXFORDSHIRE	270066
GLOUCESTERSHIRE	268891
NORTH YORKSHIRE	255070
SOMERSET	241207
CORNWALL	239192
WORCESTERSHIRE	233433
WARWICKSHIRE	232739
BUCKINGHAMSHIRE	221272
CUMBRIA	204611
WILTSHIRE	203997
DORSET	202503
CHESHIRE	201193
CITY OF BRISTOL	174853
EAST RIDING OF YORKSHIRE	145974
DURHAM	134700
CARDIFF	134115
BRIGHTON AND HOVE	129872
BEDFORDSHIRE	129687
SHROPSHIRE	124257
NORTHUMBERLAND	122040
MILTON KEYNES	121296
SOUTH GLOUCESTERSHIRE	111146

5. Query to find the average price per property type.

```
SELECT property_type, AVG(housing_prices) AS avg_price From uk_housing_summary GROUP BY

property_type ORDER BY avg_price DESC;
```

property_type	avg_price
O	1295773.7555683716
D	250097.46717451332
F	174982.8965765005
S	148967.52373050625
T	136534.02159879782

6. Query to count transactions by year.

```
SELECT SUBSTR(transfer_date, 1, 4) AS year, COUNT(*) AS total_transactions FROM

uk_housing_summary GROUP BY SUBSTR(transfer_date, 1, 4) ORDER BY year;
```

year	total_transactions
1995	796777
1996	964695
1997	1093669
1998	1049739
1999	1194064
2000	1128742
2001	1245112
2002	1351256
2003	1257140
2004	1270409
2005	1061686
2006	1326161
2007	1272921
2008	650492
2009	625662
2010	663342
2011	661055
2012	668295
2013	810111
2014	982943
2015	1007421
2016	1032558
2017	375098

7. Query to find average price by town and property type.

```
SELECT town_city, property_type, AVG(housing_prices) AS avg_price FROM uk_housing_summary

GROUP BY town_city, property_type ORDER BY avg_price DESC LIMIT 10;
```

town_city	property_type	avg_price
GATWICK	O	1.79099978E7
HAYES	O	5940261.019607843
VIRGINIA WATER	O	5730000.0
OXTED	O	4957997.0625
SWANLEY	O	4679558.944444444
WEST DRAYTON	O	4677269.2
CATTERICK GARRISON	O	4663867.75
WATLINGTON	O	4588114.285714285
UXBRIDGE	O	4445714.954248366
WEYBRIDGE	O	4198242.584905661

8. Query to find top 5 towns with the most property transactions.

```
SELECT town_city, COUNT(*) AS total_transactions

FROM uk_housing_summary

GROUP BY town_city

ORDER BY total_transactions DESC

LIMIT 5;
```

town_city	total_transactions
LONDON	1784194
MANCHESTER	366133
BRISTOL	344839
BIRMINGHAM	330358
NOTTINGHAM	292532

5 rows selected (23.54 seconds)

9. Query to find top 10 housing prices over £5 million .

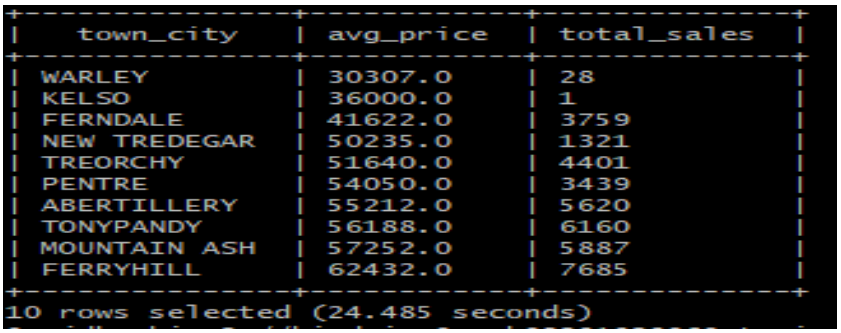
```
SELECT  town_city, district, county, housing_prices, transfer_date,  property_type

FROM uk_housing_summary

WHERE housing_prices > 5000000

ORDER BY housing_prices DESC

LIMIT 10;
```

town_city	district	county	housing_prices	transfer_date	property_type
LONDON	CITY OF WESTMINSTER	GREATER LONDON	98900000	2016-11-24 00:00	O
BIRMINGHAM	BIRMINGHAM	WEST MIDLANDS	98765391	2017-02-09 00:00	O
LONDON	CAMDEN	GREATER LONDON	98446300	2017-04-06 00:00	O
READING	WOKINGHAM	WOKINGHAM	98250000	2015-10-21 00:00	O
LONDON	CITY OF WESTMINSTER	GREATER LONDON	97630000	2017-05-25 00:00	O
LONDON	CAMDEN	GREATER LONDON	96840522	2016-06-15 00:00	O
LONDON	CITY OF WESTMINSTER	GREATER LONDON	96652091	2015-07-20 00:00	O
LONDON	CAMDEN	GREATER LONDON	96350000	2016-12-12 00:00	O
LONDON	CITY OF LONDON	GREATER LONDON	96264933	2014-12-31 00:00	O
LONDON	CITY OF LONDON	GREATER LONDON	96000000	2014-04-30 00:00	O

10 rows selected (16.078 seconds)

10. Query to find the top 10 towns with the lowest average housing prices.

```
SELECT town_city, ROUND(AVG(housing_prices)) AS avg_price,

COUNT(*) AS total_sales

FROM uk_housing_summary

GROUP BY town_city

ORDER BY avg_price ASC

LIMIT 10;
```



town_city	avg_price	total_sales
WARLEY	30307.0	28
KELSO	36000.0	1
FERNDALE	41622.0	3759
NEW TREDEGAR	50235.0	1321
TREORCHY	51640.0	4401
PENTRE	54050.0	3439
ABERTILLERY	55212.0	5620
TONYPANDY	56188.0	6160
MOUNTAIN ASH	57252.0	5887
FERRYHILL	62432.0	7685

10 rows selected (24.485 seconds)

11. Query to categorize property sales in the town of Liverpool into different price ranges and count how many properties fall into each price range.

```
SELECT
  price_range,
  COUNT(*) AS sales_count
FROM (
  SELECT
    CASE
      WHEN housing_prices < 100000 THEN '< £100k'
      WHEN housing_prices BETWEEN 100000 AND 199999 THEN '£100k - £199k'
      WHEN housing_prices BETWEEN 200000 AND 299999 THEN '£200k - £299k'
      WHEN housing_prices BETWEEN 300000 AND 499999 THEN '£300k - £499k'
      ELSE '£500k+'
    END AS price_range
  FROM uk_housing_summary
  WHERE LOWER(town_city) = 'liverpool'
```

```
    AND housing_prices IS NOT NULL
) AS subquery
GROUP BY price_range
ORDER BY sales_count DESC;
```

```
INFO  : Concurrency mode is disab
+----------------+--------------+
|  price_range   | sales_count  |
+----------------+--------------+
| < £100k        | 130587       |
| £100k - £199k  | 72556        |
| £200k - £299k  | 13895        |
| £300k - £499k  | 4971         |
| £500k+         | 1315         |
+----------------+--------------+
5 rows selected (27.594 seconds)
```

12. Query to calculate average monthly housing prices across the dataset.

```
SELECT
  year(TO_DATE(transfer_date)) AS year,
  month(TO_DATE(transfer_date)) AS month,
  ROUND(AVG(housing_prices), 2) AS avg_price
FROM
  uk_housing_summary
WHERE
  transfer_date IS NOT NULL
GROUP BY
  year(TO_DATE(transfer_date)),
  month(TO_DATE(transfer_date))
ORDER BY
  year, month;
```

year	month	avg_price
1995	1	68355.43
1995	2	65628.61
1995	3	65747.46
1995	4	67779.58
1995	5	67026.34
1995	6	67959.87
1995	7	70283.56
1995	8	70553.13
1995	9	68304.95
1995	10	67534.92
1995	11	67015.43
1995	12	67988.56
1996	1	68234.51
1996	2	66382.23
1996	3	66226.91
1996	4	69662.57
1996	5	69431.61
1996	6	70990.87
1996	7	73691.03
1996	8	74671.18
1996	9	73697.04
1996	10	72749.78
1996	11	72630.77
1996	12	73917.04
1997	1	74606.19
1997	2	73447.09
1997	3	73286.78
1997	4	76592.53
1997	5	77164.34
1997	6	79221.83
1997	7	81525.8
1997	8	80658.62
1997	9	81448.84
1997	10	79536.44
1997	11	79518.92
1997	12	81026.89
1998	1	82233.14
1998	2	80272.98
1998	3	82320.94
1998	4	84377.0
1998	5	84096.21
1998	6	86174.89
1998	7	88043.55
1998	8	89191.69
1998	9	88483.07
1998	10	85371.14
1998	11	84739.66

13. Query to find the top 10 housing prices in London.

```
SELECT

 transaction_unique_id,

 housing_prices,

 transfer_date,

 property_type,
```

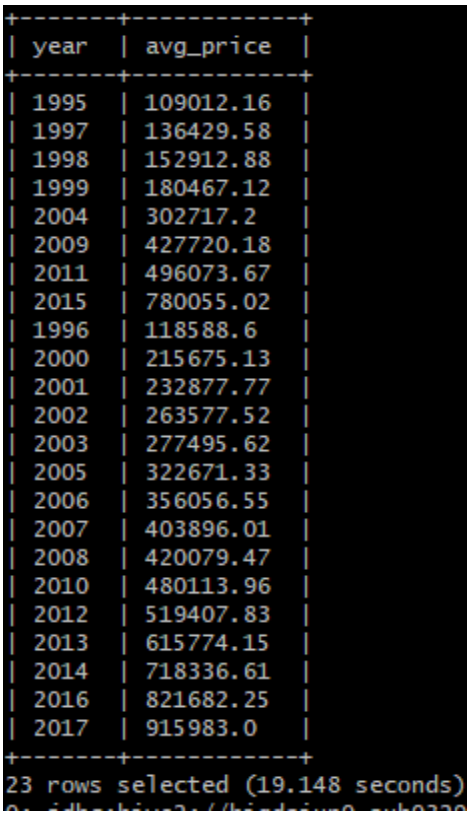
```
  town_city,

  county

FROM

  uk_housing_summary

WHERE

  town_city = 'LONDON'

ORDER BY

  housing_prices DESC

LIMIT 10;
```

transaction_unique_id	housing_prices	transfer_date	property_type	town_city	county
{50F18103-E682-9FD5-E050-A8C063054923}	98900000	2016-11-24 00:00	0	LONDON	GREATER LONDON
{4E95D758-283E-EDA1-E050-A8C0630539E2}	98446300	2017-04-06 00:00	0	LONDON	GREATER LONDON
{5376B386-560C-34C1-E053-6B04A8C09FF6}	97630000	2017-05-25 00:00	0	LONDON	GREATER LONDON
{3914047A-8399-3206-E050-A8C063057647}	96840522	2016-06-15 00:00	0	LONDON	GREATER LONDON
{21E5FEB7-4B56-2439-E050-A8C06205342E}	96652091	2015-07-20 00:00	0	LONDON	GREATER LONDON
{453D27A3-E10E-EF91-E050-A8C0630574D7}	96350000	2016-12-12 00:00	0	LONDON	GREATER LONDON
{21E5FEB7-01C2-2439-E050-A8C06205342E}	96264933	2014-12-31 00:00	0	LONDON	GREATER LONDON
{21E5FEB6-B2C8-2439-E050-A8C06205342E}	96000000	2014-04-30 00:00	0	LONDON	GREATER LONDON
{288DCE2A-135E-E510-E050-A8C0620580E}	95900000	2015-12-02 00:00	0	LONDON	GREATER LONDON
{21E5FEB7-0C64-2439-E050-A8C06205342E}	94390560	2013-09-12 00:00	0	LONDON	GREATER LONDON

10 rows selected (21.053 seconds)

14. Query to find the average price of housing prices by year in London.

```
SELECT

  YEAR(TO_DATE(transfer_date)) AS year,

  ROUND(AVG(housing_prices), 2) AS avg_price

FROM

  uk_housing_summary

WHERE

  town_city = 'LONDON'

  AND transfer_date IS NOT NULL

GROUP BY

  YEAR(TO_DATE(transfer_date))

SORT BY

  year ASC;
```



year	avg_price
1995	109012.16
1997	136429.58
1998	152912.88
1999	180467.12
2004	302717.2
2009	427720.18
2011	496073.67
2015	780055.02
1996	118588.6
2000	215675.13
2001	232877.77
2002	263577.52
2003	277495.62
2005	322671.33
2006	356056.55
2007	403896.01
2008	420079.47
2010	480113.96
2012	519407.83
2013	615774.15
2014	718336.61
2016	821682.25
2017	915983.0

23 rows selected (19.148 seconds)

Step 4

This step is to export data & perform data visualization using Microsoft Excel.

Install 3D map feature:

If the 3-D Map button is missing

Step 1 – re-enable the COM add-in

File ▶ Options ▶ Add-ins.

At the bottom, set Manage ▶ COM Add-ins ▶ Go.

Tick Microsoft 3-D Maps for Excel (or Microsoft Power Map for Excel), OK, then restart Excel

Create 3D Map

Export data:

Note: Replace jcontr185 with your own username

1. **In beeline run the following command:**

```
INSERT OVERWRITE DIRECTORY '/user/jcontr185/hive_data/monthly_avg_price_geo_full'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
SELECT
 YEAR(TO_DATE(transfer_date)) AS year,
 MONTH(TO_DATE(transfer_date)) AS month,
 town_city,
 district,
 county,
 ROUND(AVG(housing_prices), 2) AS avg_price
FROM
 uk_housing_summary
WHERE
 transfer_date IS NOT NULL
 AND town_city IS NOT NULL
 AND district IS NOT NULL
 AND county IS NOT NULL
GROUP BY
 YEAR(TO_DATE(transfer_date)),
```

```
 MONTH(TO_DATE(transfer_date)),
 town_city,
 district,
 county
ORDER BY
 year, month, county, district, town_city;
```

2. Download Data from HDFS:

```
hdfs dfs -get /user/jcontr185/hive_data/monthly_avg_price_geo_full/000000_0
monthly_avg_price_geo_full.csv
```

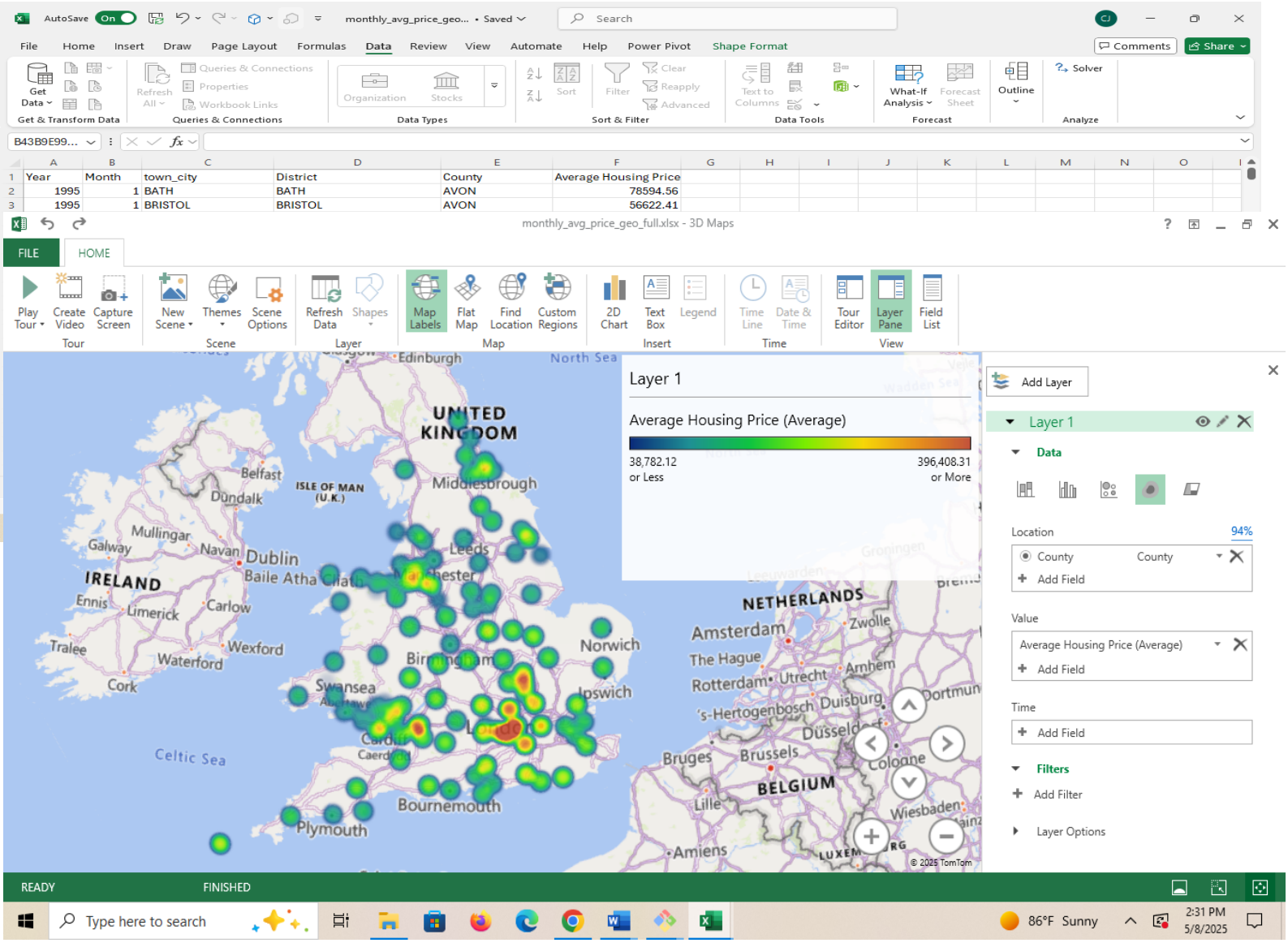
Note: Replace jcontr185 with your own username

3. Transfer to local machine

```
scp jcontr185@144.24.13.0:~/monthly_avg_price_geo_full.csv "C:/Users/Jeremy
Contreras/Downloads/"
```

Note: Replace with your username, ip address, and path to your directory

4. **Open Excel. Click insert at top of data and right click to insert entire row. Name the columns the following: Year, Month, town_city, district, county, average price.**



5. **Go to Data tab and select Data Model and 3D map under Data Tools . In the layer pane for location add county and select county under select drop down. For value field add average price and make it average. Also select map labels.**

Create Visualization part 2 for yearly and monthly average prices.

Note: Replace jcontr185 with your own username

1. In beeline Enter:

```
INSERT OVERWRITE DIRECTORY '/user/jcontr185/hive_data/monthly_avg_price_summary'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
SELECT
 YEAR(TO_DATE(transfer_date)) AS year,
 MONTH(TO_DATE(transfer_date)) AS month,
 ROUND(AVG(housing_prices), 2) AS avg_price
FROM
 uk_housing_summary
WHERE
 transfer_date IS NOT NULL
GROUP BY
 YEAR(TO_DATE(transfer_date)),
 MONTH(TO_DATE(transfer_date))
ORDER BY
 year, month;
```

2. **SSH into cluster and run:**

```
hdfs dfs -get /user/jcontr185/hive_data/monthly_avg_price_summary/000000_0

monthly_avg_price_summary.csv
```

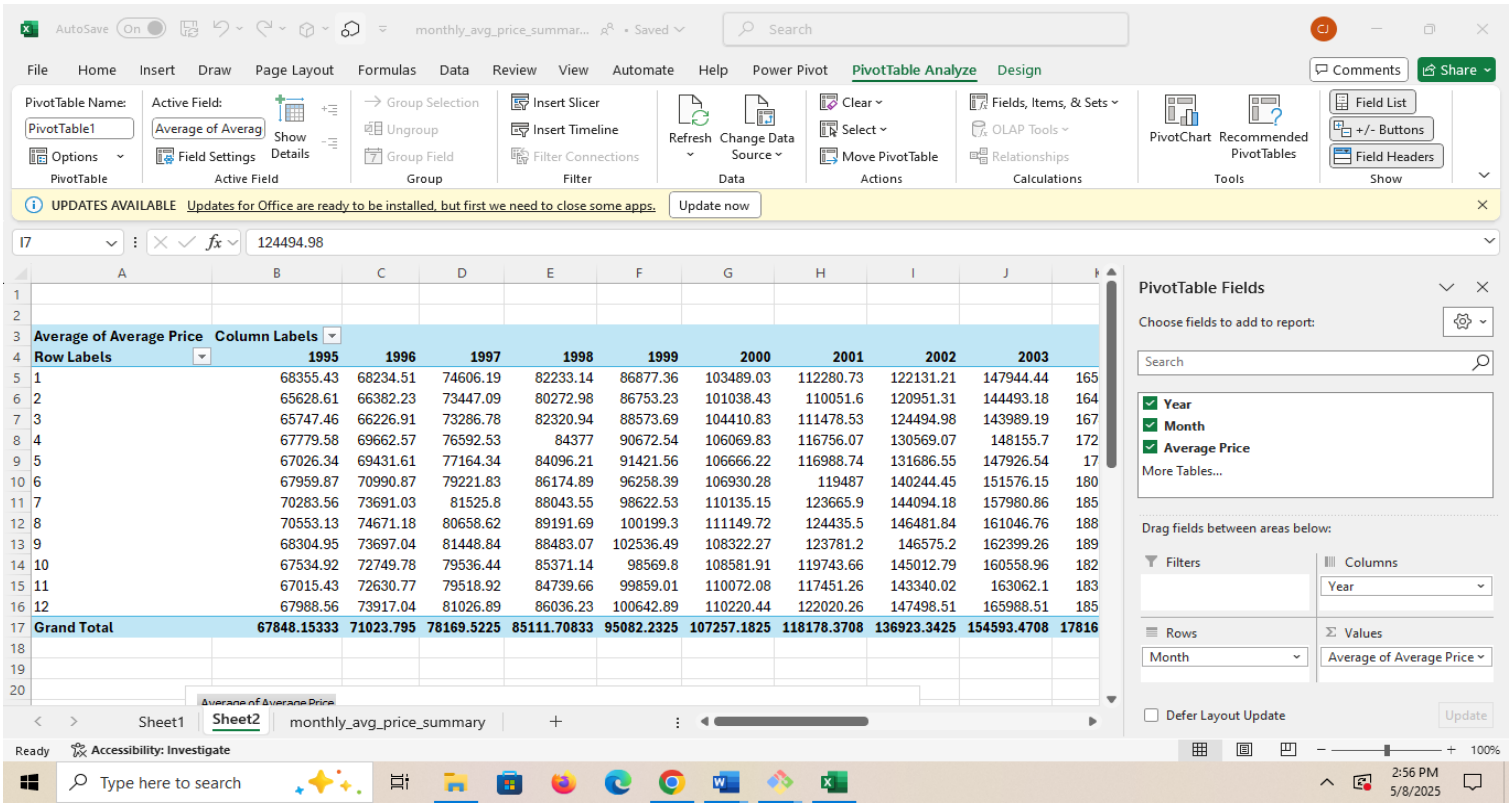
3. Use SCP to copy the file to local machine:

```
scp jcontr185@144.24.13.0:~/monthly_avg_price_summary.csv "C:/Users/Jeremy Contreras/"
```

Note: Replace with your username, ip address, and path to your directory

4. Open excel file: At top of data right click and select insert entire row. Name the columns: Year, Month, Average Price.
5. Click any cell in the data. Press CTRL + A to select all data. Go to insert pivot table and select PivotTable. Choose new worksheet and click ok.

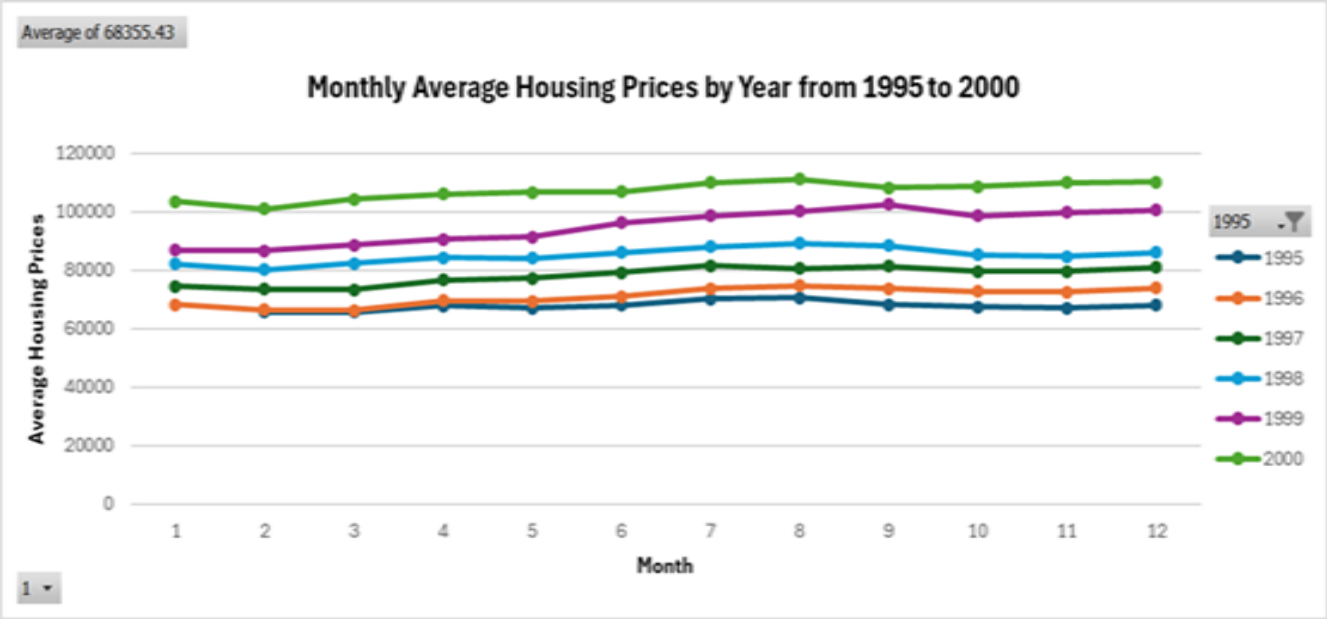
6. In the pivot table fields panel: drag month to rows area. Drag Year to columns. Drag Average Price to Values. Make sure it is set to average by clicking dropdown in Values selecting Value Field Settings and and choose Average.



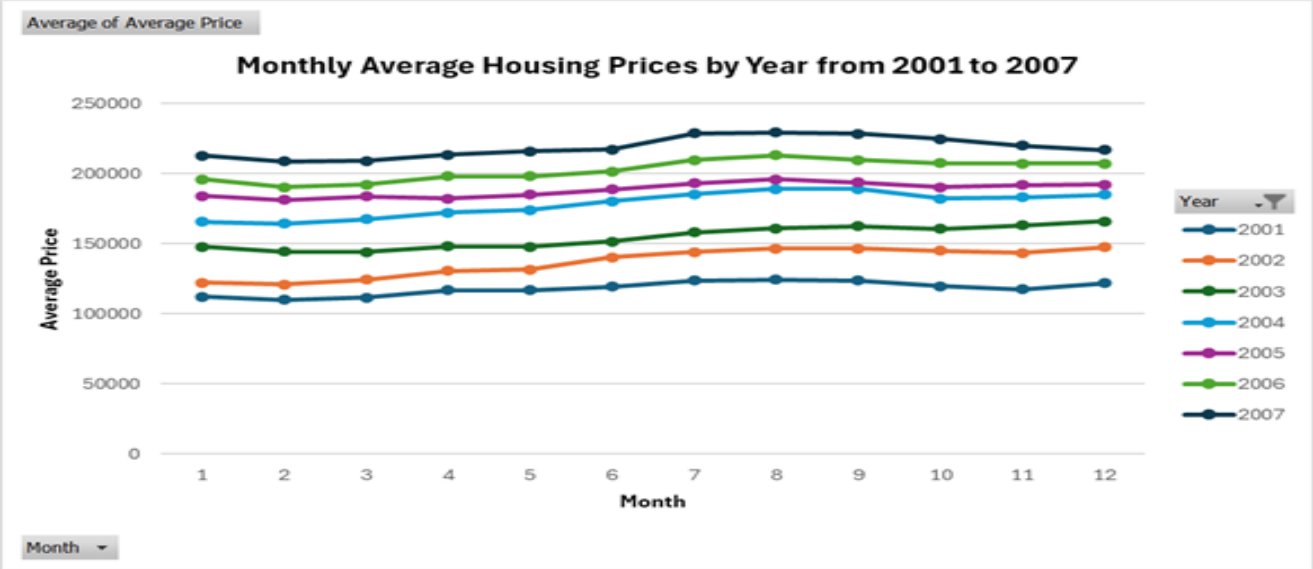
Average of Average Price	Column Labels									
Row Labels	1995	1996	1997	1998	1999	2000	2001	2002	2003	
1	68355.43	68234.51	74606.19	82233.14	86877.36	103489.03	112280.73	122131.21	147944.44	165
2	65628.61	66382.23	73447.09	80272.98	86753.23	101038.43	110051.6	120951.31	144493.18	164
3	65747.46	66226.91	73286.78	82320.94	88573.69	104410.83	111478.53	124494.98	143989.19	167
4	67779.58	69662.57	76592.53	84377	90672.54	106069.83	116756.07	130569.07	148155.7	172
5	67026.34	69431.61	77164.34	84096.21	91421.56	106666.22	116988.74	131686.55	147926.54	17
6	67959.87	70990.87	79221.83	86174.89	96258.39	106930.28	119487	140244.45	151576.15	180
7	70283.56	73691.03	81525.8	88043.55	98622.53	110135.15	123665.9	144094.18	157980.86	185
8	70553.13	74671.18	80658.62	89191.69	100199.3	111149.72	124435.5	146481.84	161046.76	188
9	68304.95	73697.04	81448.84	88483.07	102536.49	108322.27	123781.2	146575.2	162399.26	189
10	67534.92	72749.78	79536.44	85371.14	98569.8	108581.91	119743.66	145012.79	160558.96	182
11	67015.43	72630.77	79518.92	84739.66	99859.01	110072.08	117451.26	143340.02	163062.1	183
12	67988.56	73917.04	81026.89	86036.23	100642.89	110220.44	122020.26	147498.51	165988.51	185
Grand Total	67848.15333	71023.795	78169.5225	85111.70833	95082.2325	107257.1825	118178.3708	136923.3425	154593.4708	17816

7. Click anywhere in Pivot table. Go to insert tab click line chart under charts and choose line chart with markers. Format the chart by clicking plus sign to the right of chart and selecting axis and chart titles. For chart title enter: Monthly Average Housing Prices by Year. The x axis enter Month for title. For y-axis enter title of Average Price.

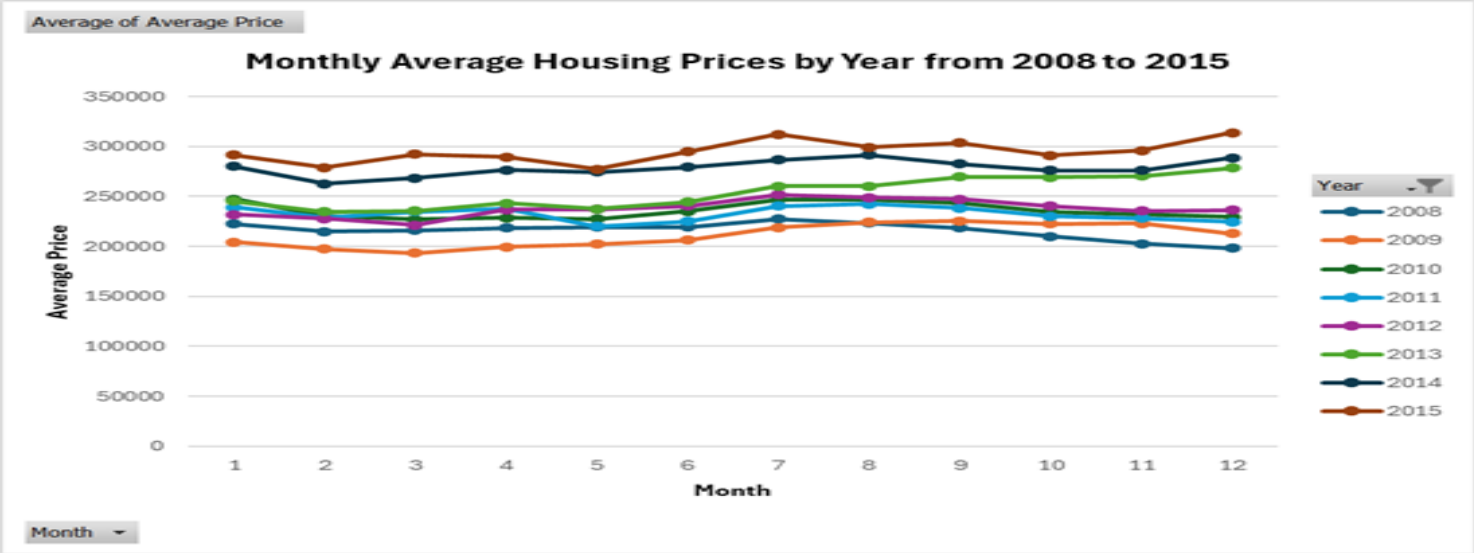
Filter based on years and select years 1995 to 2000. Filter is located on right side of chart.



Select filter again and filter based on years 2001 to 2007.



Select filter again and filter based on years from 2008 to 2015.



Create part 3 visualization for total transaction by year.

Note: Replace jcontr185 with your own username

1. In beeline:

```
INSERT OVERWRITE DIRECTORY '/user/jcontr185/hive_data/yearly_transaction_summary'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT

 SUBSTR(transfer_date, 1, 4) AS year,
```

```
  COUNT(*) AS total_transactions
FROM
  uk_housing_summary
GROUP BY
  SUBSTR(transfer_date, 1, 4)
ORDER BY
  year;
```

2. Download from HDFS:

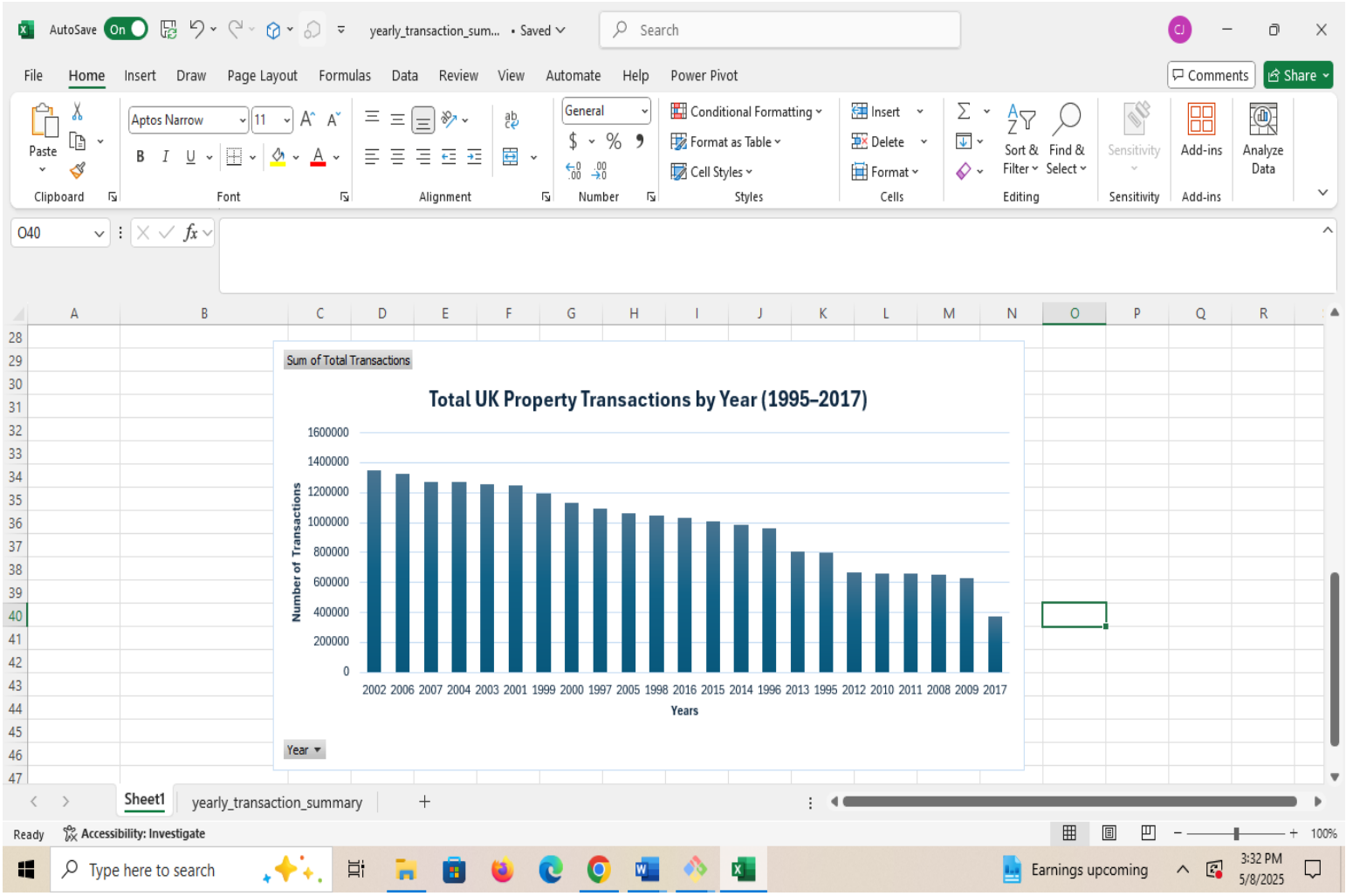
```
hdfs dfs -get /user/jcontr185/hive_data/yearly_transaction_summary/000000_0
yearly_transaction_summary.csv
```

3. Transfer file to local computer:

```
scp jcontr185@144.24.13.0:~/yearly_transaction_summary.csv "C:/Users/Jeremy
Contreras/Downloads/"
```

Note: Replace with your username, ip address, and path to your directory

4. Open Excel file. At top of data right click and insert entire row. Names the columns: Year, Total Transactions. Click any cell in dataset. Press CTRL + A to select all data. Go to Insert tab and click pivot table. In the dialog choose new worksheet and click ok. In the pivot table fields panel drag year to rows and total transactions to values. Click anywhere in pivot table and go to insert tab. Choose Column chart and select clustered column. Format the chart by clicking plus sign to right of chart and selecting chart title and axis title. For chart title: Total property transactions by year. For y-axis title enter: number of transactions. For x-axis enter: Years. To sort the chart right click on one of the bars and select sort by largest to smallest.



Create part 4 visualization of house sales count by county

Note: Replace jcontr185 with your own username

1. In beeline:

```
INSERT OVERWRITE DIRECTORY '/user/jcontr185/hive_data/county_sales_summary'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT

 county,

 COUNT(*) AS houses_sold

FROM

 uk_housing_summary

GROUP BY

 county

ORDER BY

 houses_sold DESC;
```

2. Retrieve file from HDFS:

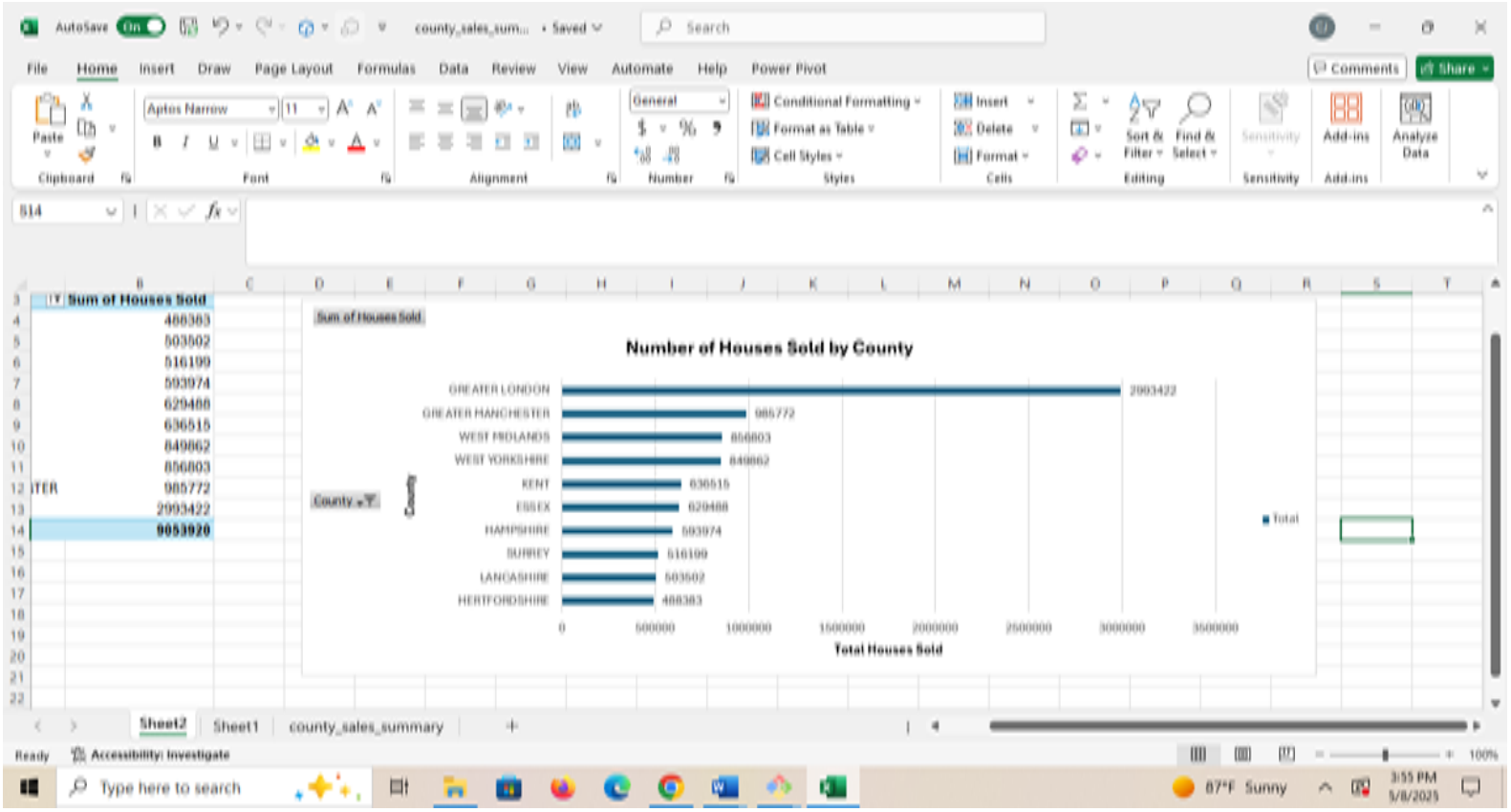
```
hdfs dfs -get /user/jcontr185/hive_data/county_sales_summary/000000_0 county_sales_summary.csv
```

3. Transfer to your local machine:

```
scp jcontr185@144.24.13.0:~/county_sales_summary.csv "C:/Users/Jeremy Contreras/Downloads/"
```

Note: Replace with your username, ip address, and path to your directory

4. Open Excel. At top of data right click and insert entire row. Name the columns: County, Houses Sold. Click any cell in dataset. Go to insert tab and select pivot table. In the dialog choose new worksheet and select ok. In the pivot table field panel drag county to rows and houses sold to values area. Click the dropdown next to row labels in pivot table and go to value filters and select top 10. Choose top 10 items by houses sold and click ok. Go to insert tab and under charts select bar chart option and choose clustered bar. To format chart click plus sign on right side of chart. Add data labels, chart title and axis title. For chart title enter: Number of houses sold by county. For y-axis title enter: county. For x-axis title enter: Total houses sold. Click on a bar in the chart and select sort smallest to largest.



Create part 5 visualization yearly average housing prices in London.

Note: Replace jcontr185 with your own username

1. In beeline:

```
INSERT OVERWRITE DIRECTORY '/user/jcontr185/hive_data/london_avg_price_by_year'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

SELECT
```

```
  YEAR(TO_DATE(transfer_date)) AS year,
  ROUND(AVG(housing_prices), 2) AS avg_price
FROM
  uk_housing_summary
WHERE
  town_city = 'LONDON'
  AND transfer_date IS NOT NULL
GROUP BY
  YEAR(TO_DATE(transfer_date))
ORDER BY
  year ASC;
```

2. Download the output from HDFS:

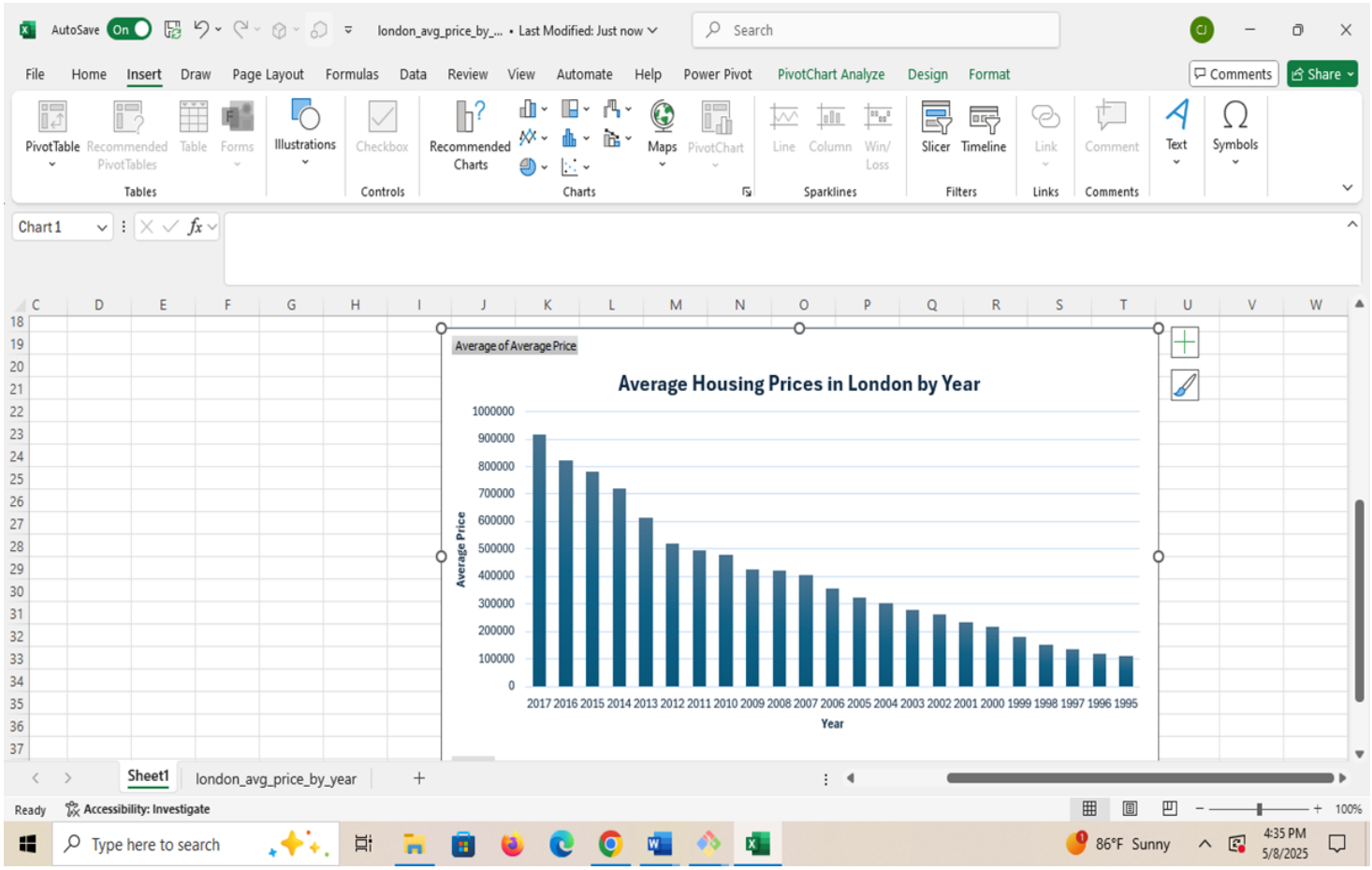
```
hdfs dfs -get /user/jcontr185/hive_data/london_avg_price_by_year/000000_0
london_avg_price_by_year.csv
```

3. Transfer file to local computer:

```
scp jcontr185@144.24.13.0:~/london_avg_price_by_year.csv "C:/Users/Jeremy
Contreras/Downloads/"
```

Note: Replace with your username, ip address, and path to your directory

4. Open Excel. At top of data right click and insert entire row. Name the columns: Year, Average Price. Click any cell in dataset. Go to insert tab and click Pivot Table. In the dialog box choose new worksheet and click ok. In the pivot table field panel: Drag Year to rows area and Average Price to the values area. Change it to Average by Clicking the dropdown and selecting value field settings and choose average. Click anywhere inside pivot table. Go to insert tab and click insert column or bar chart and select clustered column. Click the plus sign on the right side of chart and add axis title, chart title. For chart title enter: Average Housing Prices in London by Year. For y-axis title enter: Average Price. For x-axis title enter: Year.



References

a. URL OF Data Source: https://www.kaggle.com/datasets/hm-land-registry/uk-housing-prices-paid

b. URL of Github: https://github.com/Yuwang-maker/CIS-5200-Housing-Price-Project-