Group 6 ETL Project

Brian, Kashish, Krystal, Zouheir

Extract: Resources

Sources:

1. Airbnb (csv file)

   https://www.kaggle.com/datasets/jinbonnie/chicago-airbnb-open-data?select=listings.csv

   https://www.kaggle.com/datasets/jinbonnie/chicago-airbnb-open-data?select=listings.csv, 21 March 2022.

2. Population and Demo (csv file)

   "Resources/chicago_population2013.csv"
   https://www.kaggle.com/datasets/sergejnuss/chicago-community-areas-demographics, 21 March 2022.

3. Community Area Numbers (csv file)

   "Resources/chicago-community-areas.csv"

Extract: Airbnb

In [2]:

```
# Import CSV with AirBnB data
csv_file = "Resources/listings.csv"
airbnb_data_df = pd.read_csv(csv_file)
airbnb_data_df.head()
```

Out[2]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	reviews
0	2384	Hyde Park - Walk to UChicago, 10 min to McCormick	2613	Rebecca	NaN	Hyde Park	41.78790	-87.58780	Private room	60	2	178	2019-12-15	
1	4505	394 Great Reviews. 127 y/o House. 40 yds to tr...	5775	Craig & Kathleen	NaN	South Lawndale	41.85495	-87.69696	Entire home/apt	105	2	395	2020-07-14	
2	7126	Tiny Studio Apartment 94 Walk Score	17928	Sarah	NaN	West Town	41.90289	-87.68182	Entire home/apt	60	2	384	2020-03-08	
3	9811	Barbara's Hideaway - Old Town	33004	At Home Inn	NaN	Lincoln Park	41.91769	-87.63788	Entire home/apt	65	4	49	2019-10-23	
4	10610	3 Comforts of Cooperative Living	2140	Lois	NaN	Hyde Park	41.79612	-87.59261	Private room	21	1	44	2020-02-14	

Extract: Population and Demo

In [30]:

```
# Import CSV with Chicago 2013 population numbers
csv_file = "Resources/chicago_population2013.csv"
population_df = pd.read_csv(csv_file, sep=';')
population_df.head()
```

Out[30]:

	Community Area	1	2	3	4	5	6	7	8	9	...	68	69	70	71	72	73	74	75	76
0	name	Rogers Park	West Ridge	Uptown	Lincoln Square	North Center	Lake View	Lincoln Park	Near North Side	Edison Park	...	Englewood	Greater Grand Crossing	Ashburn	Auburn Gresham	Beverly	Washington Heights	Mount Greenwood	Morgan Park	O'Hare
1	population	54991	71942	56362	39493	31867	94368	64116	80484	11187	...	30654	32602	41081	48743	20034	26493	19093	22544	12756
2	income	39482	47323	40324	57749	81524	70746	82707	76290	77678	...	19743	29663	62238	34767	83092	42053	80505	56886	49601
3	latinos	0.244	0.204	0.142	0.191	0.136	0.076	0.056	0.049	0.078	...	0.011	0.012	0.368	0.009	0.046	0.01	0.072	0.027	0.095
4	blacks	0.263	0.111	0.2	0.038	0.023	0.039	0.043	0.108	0.003	...	0.974	0.969	0.462	0.978	0.341	0.974	0.052	0.667	0.032

5 rows × 78 columns

Extract: Community Area Numbers

In [3]:

```
# Import CSV with community area numbers
csv_file = "Resources/chicago-community-areas.csv"
communities_df = pd.read_csv(csv_file)
communities_df.head()
```

Out[3]:

Community Area	1	2	3	4	5	6	7	8	9	...	68	69	70	71	72	73	74	75	76
0 name	Rogers Park	West Ridge	Uptown	Lincoln Square	North Center	Lake View	Lincoln Park	Near North Side	Edison Park	...	Englewood	Greater Grand Crossing	Ashburn	Auburn Gresham	Beverly	Washington Heights	Mount Greenwood	Morgan Park	O'Hare

1 rows × 78 columns

Transform: Airbnb Listing Data

Airbnb

1. Changed ID to index column
2. Dropped neighborhood group column
3. Dropped rows that had values of NaN
4. Separated the data frame into two for data normalization purposes
   a. First involving listing information named “listing_info_df”
   b. Second involving location information called “location_info_df”

In [5]:
```
airbnb_data_df = airbnb_data_df.drop(["neighbourhood_group"], axis=1)
```

In [6]:
```
airbnb_data_df.set_index("id", inplace=True)
```

In [7]:
```
airbnb_data_df.dropna(how = "any", inplace=True)
```

In [8]:
```
listing_info_df = airbnb_data_df.iloc[:, [1, 2, 6, 7, 8, 9, 10, 11, 12, 13]]

location_info_df = airbnb_data_df.iloc[:, [0, 3, 4, 5]]
```

In [9]:
```
location_info_df = location_info_df.rename(columns={"name": "address", "neighbourhood": "neighborhood"})
```

In [10]:
```
listing_info_df.index = listing_info_df.index.astype(int)
```

In [11]:
```
location_info_df["neighborhood"] = location_info_df["neighborhood"].str.lower()
location_info_df["neighborhood"] = location_info_df["neighborhood"].str.replace("\'", "")
```

In [49]:
```
location_info_df
listing_info_df
```

Out[49]:

id	host_id	host_name	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count	availability_365
2384	2613	Rebecca	Private room	60	2	178	2019-12-15	2.56	1	353
4505	5775	Craig & Kathleen	Entire home/apt	105	2	395	2020-07-14	2.81	1	155
7126	17928	Sarah	Entire home/apt	60	2	384	2020-03-08	2.81	1	321
9811	33004	At Home Inn	Entire home/apt	65	4	49	2019-10-23	0.63	9	300
10610	2140	Lois	Private room	21	1	44	2020-02-14	0.61	5	168
...	...	...	...	...	...	...	...	...	...	...
45351578	77382816	Edmund	Entire home/apt	67	1	2	2020-09-20	2.00	1	307
45368527	128265803	Victor	Private room	24	1	2	2020-09-20	2.00	4	14
45386114	324740940	Andrea	Entire home/apt	97	1	2	2020-09-19	2.00	4	81
45433310	111872967	Julie	Private room	54	1	1	2020-09-18	1.00	4	359
45465696	8803468	Ashley	Private room	92	1	1	2020-09-19	1.00	4	330

5265 rows × 10 columns

airbnb_data_df

Out[2]:

	id	name	host_id	host_name	neighbourhood_group	neighbourhood	latitude	longitude	room_type	price	minimum_nights	number_of_reviews	last_review	revi
0	2384	Hyde Park - Walk to UChicago, 10 min to McCormick	2613	Rebecca	NaN	Hyde Park	41.78790	-87.58780	Private room	60	2	178	2019-12-15	
1	4505	394 Great Reviews. 127 y/o House. 40 yds to tr...	5775	Craig & Kathleen	NaN	South Lawndale	41.85495	-87.69696	Entire home/apt	105	2	395	2020-07-14	
2	7126	Tiny Studio Apartment 94 Walk Score	17928	Sarah	NaN	West Town	41.90289	-87.68182	Entire home/apt	60	2	384	2020-03-08	
3	9811	Barbara's Hideaway - Old Town	33004	At Home Inn	NaN	Lincoln Park	41.91769	-87.63788	Entire home/apt	65	4	49	2019-10-23	
4	10610	3 Comforts of Cooperative Living	2140	Lois	NaN	Hyde Park	41.79612	-87.59261	Private room	21	1	44	2020-02-14	

listing_info_df

Out[49]:

	host_id	host_name	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count	availability_365
id										
2384	2613	Rebecca	Private room	60	2	178	2019-12-15	2.56	1	353
4505	5775	Craig & Kathleen	Entire home/apt	105	2	395	2020-07-14	2.81	1	155
7126	17928	Sarah	Entire home/apt	60	2	384	2020-03-08	2.81	1	321
9811	33004	At Home Inn	Entire home/apt	65	4	49	2019-10-23	0.63	9	300
10610	2140	Lois	Private room	21	1	44	2020-02-14	0.61	5	168
...	...	...	...	...	...	...	...	...	...	...
45351578	77382816	Edmund	Entire home/apt	67	1	2	2020-09-20	2.00	1	307
45368527	128265803	Victor	Private room	24	1	2	2020-09-20	2.00	4	14
45386114	324740940	Andrea	Entire home/apt	97	1	2	2020-09-19	2.00	4	81
45433310	111872967	Julie	Private room	54	1	1	2020-09-18	1.00	4	359
45465696	8803468	Ashley	Private room	92	1	1	2020-09-19	1.00	4	330

5265 rows × 10 columns

Transform: Population and Demographic Data

Population and Demo

1. Separated data by “;” in order to make data usable
2. Set index as community area
3. Transpose the table to make data joinable with Airbnb data
4. Renamed column headers to more easily load into postgres

In [31]:
```
# Make the column names lowercase
population_df.columns = population_df.columns.str.lower()

# Replace spaces in column names with underscores
population_df.columns = population_df.columns.str.replace(' ', '_')
```

In [32]:
```
population_df.set_index("community_area", inplace=True)
population_df = population_df.transpose()
```

In [33]:
```
population_df = population_df.rename(columns={"name": "neighborhood"})
```

In [37]:
```
population_df["neighborhood"] = population_df["neighborhood"].str.lower()

# Replace spaces in column names with underscores
population_df["neighborhood"] = population_df["neighborhood"].str.replace("\'", "")
```

In [43]:
```
population_df.index = population_df.index.astype(int)
```

In [44]:
```
population_df
```

Out[44]:

community_area	neighborhood	population	income	latinos	blacks	white	asian	other
1	rogers park	54991	39482	0.244	0.263	0.393	0.064	0.036
2	west ridge	71942	47323	0.204	0.111	0.427	0.225	0.032
3	uptown	56362	40324	0.142	0.2	0.516	0.114	0.028
4	lincoln square	39493	57749	0.191	0.038	0.631	0.111	0.029
5	north center	31867	81524	0.136	0.023	0.773	0.045	0.022
...	...	...	...	...	...	...	...	...
73	washington heights	26493	42053	0.01	0.974	0.005	0	0.012
74	mount greenwood	19093	80505	0.072	0.052	0.86	0.007	0.01
75	morgan park	22544	56886	0.027	0.667	0.287	0.004	0.014
76	ohare	12756	49601	0.095	0.032	0.772	0.083	0.019
77	edgewater	56521	43331	0.165	0.143	0.547	0.116	0.029

77 rows × 8 columns

Transform: Community Area Data

Community Area Numbers

1. Made all “neighborhoods” lowercase and removed apostrophes.
2. Set index as community area
3. Transposed table to make it joinable in postgres

In [13]:
```
communities_df.columns = communities_df.columns.str.lower()

# Replace spaces in column names with underscores
communities_df.columns = communities_df.columns.str.replace(' ', '_')

communities_df.set_index("community_area", inplace=True)
```

In [14]:
```
communities_df = communities_df.transpose()
```

In [15]:
```
communities_df = communities_df.rename(columns={"name": "neighborhood"})
```

In [16]:
```
communities_df.index = communities_df.index.astype(int)
```

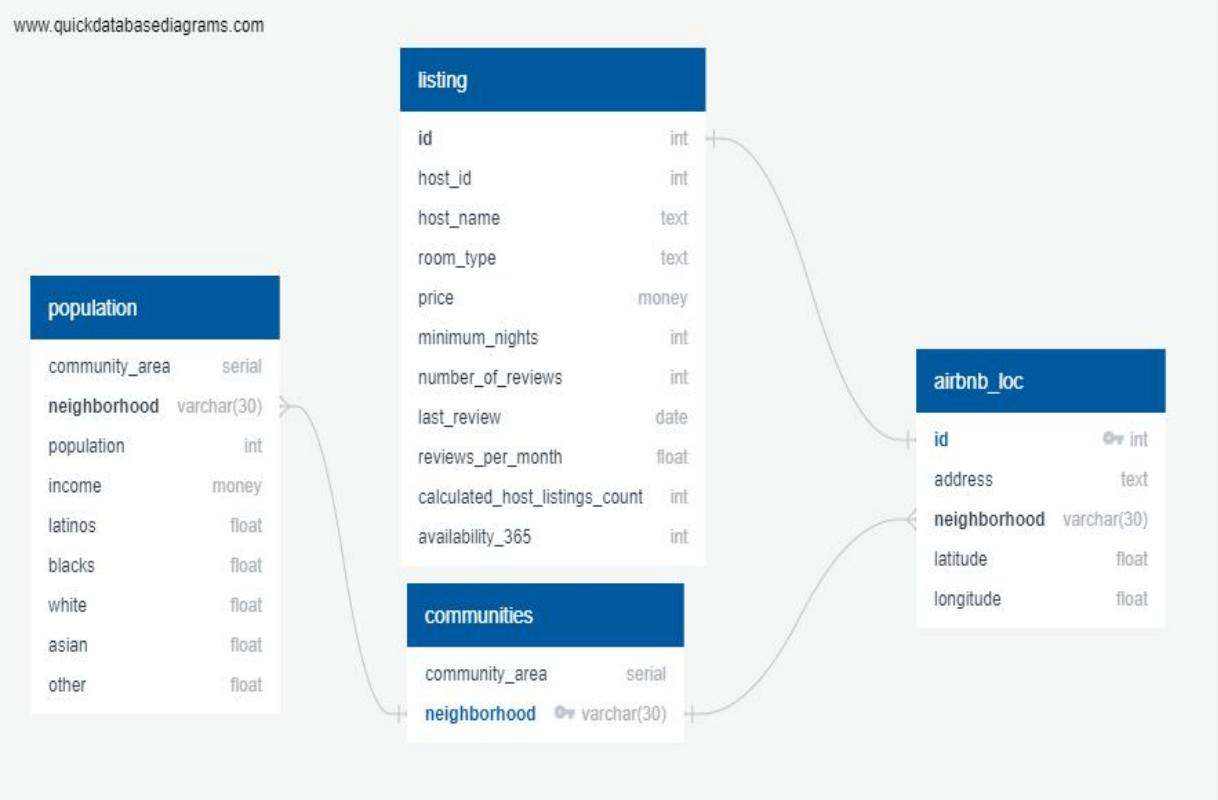
In [17]:
```
communities_df["neighborhood"] = communities_df["neighborhood"].str.lower()
communities_df["neighborhood"] = communities_df["neighborhood"].str.replace("\'", "")
communities_df
```

Out[17]:

community_area	neighborhood
1	rogers park
2	west ridge
3	uptown
4	lincoln square
5	north center
...	...
73	washington heights
74	mount greenwood
75	morgan park
76	ohare
77	edgewater

77 rows × 1 columns

Load: Table Relationship Diagram



Load: Postgres Table Formatting

```sql
DROP TABLE IF EXISTS communities CASCADE;
DROP TABLE IF EXISTS listing CASCADE;
DROP TABLE IF EXISTS airbnb_loc CASCADE;
DROP TABLE IF EXISTS population CASCADE;

CREATE TABLE communities (
    community_area serial NOT NULL,
    neighborhood varchar(30) NOT NULL,
        PRIMARY KEY(neighborhood)
);

CREATE TABLE listing (
    "id" int PRIMARY KEY NOT NULL,
    host_id int NOT NULL,
    host_name text NOT NULL,
    room_type text NOT NULL,
    price money   NOT NULL,
    minimum_nights int   NOT NULL,
    number_of_reviews int   NOT NULL,
    last_review date   NOT NULL,
    reviews_per_month float   NOT NULL,
    calculated_host_listings_count int   NOT NULL,
    availability_365 int   NOT NULL
);
```

```sql
CREATE TABLE airbnb_loc (
    "id" int PRIMARY KEY NOT NULL,
    address text   NOT NULL,
    neighborhood varchar(30)  NOT NULL,
    latitude float   NOT NULL,
    longitude float   NOT NULL
);

CREATE TABLE population (
    community_area serial PRIMARY KEY NOT NULL,
    neighborhood varchar(30)  NOT NULL,
    population int   NOT NULL,
        income money NOT NULL,
    latinos float   NOT NULL,
    blacks float   NOT NULL,
        white float   NOT NULL,
        asian float   NOT NULL,
        other float   NOT NULL,
                FOREIGN KEY (neighborhood) REFERENCES communities(neighborhood)
);

ALTER TABLE "airbnb_loc" ADD CONSTRAINT "fk_airbnb_loc_id" FOREIGN KEY("id")
REFERENCES "listing" ("id");

ALTER TABLE "airbnb_loc" ADD CONSTRAINT "fk_airbnb_loc_neighborhood" FOREIGN KEY("neighborhood")
REFERENCES "communities"
```

Load: VS Code

Load

In [24]:
```
# Create database connection
rds_connection_string = f"{user}:{password}@localhost:5432/{db_name}"
engine = create_engine(f'postgresql://{rds_connection_string}')
```

In [48]:
```
# Check for tables
inspector = inspect(engine)
print(inspector.get_table_names())
```

In [66]:
```
# Load CSV converted DataFrame into database
communities_df.to_sql(name='communities', con=engine, if_exists='append', index=False)
```

In [67]:
```
# Load CSV converted DataFrame into database
listing_info_df.to_sql(name='listing', con=engine, if_exists='append', index=True)
#resultDf.to_sql('table_name', engine, schema="schema_name", if_exists="append", index=False)
```

In [68]:
```
# Load CSV converted DataFrame into database
location_info_df.to_sql(name='airbnb_loc', con=engine, if_exists='append', index=True)
```

In [46]:
```
population_df.to_sql(name='population', con=engine, if_exists='append', index=False)
```

In [47]:
```
# Confirm data has been added
pd.read_sql_query('select * from listing', con=engine).head()
```

Out[47]:

	id	host_id	host_name	room_type	price	minimum_nights	number_of_reviews	last_review	reviews_per_month	calculated_host_listings_count	availability_365
0	2384	2613	Rebecca	Private room	$60.00	2	178	2019-12-15	2.56	1	353
1	4505	5775	Craig & Kathleen	Entire home/apt	$105.00	2	395	2020-07-14	2.81	1	155
2	7126	17928	Sarah	Entire home/apt	$60.00	2	384	2020-03-08	2.81	1	321
3	9811	33004	At Home Inn	Entire home/apt	$65.00	4	49	2019-10-23	0.63	9	300
4	10610	2140	Lois	Private room	$21.00	1	44	2020-02-14	0.61	5	168

In [ ]:
```
# Load CSV converted DataFrame into database
df.to_sql(name='table_name', con=engine, if_exists='append', index=False)
```

Load: Relationship Queries and Tables

QUERY 1

```
--Number of listings per neighborhood
select neighborhood, count(neighborhood) as listing_count
from airbnb_loc
group by neighborhood
order by count(neighborhood) DESC;
```

	neighborhood character varying (30)	listing_count bigint
1	west town	638
2	lake view	476
3	near north side	439
4	logan square	364
5	lincoln park	269
6	loop	228
7	near west side	218
8	lower west side	173
9	uptown	152
10	edgewater	142
11	irving park	139
12	avondale	135
13	north center	126
14	rogers park	113
15	bridgeport	104
16	near south side	103
17	grand boulevard	91

Tables Used
1. Airbnb_loc

Functions Used
1. Group by
2. Order by
3. Desc

Load: Relationship Queries and Tables

QUERY 2

```
--Simple join for population, neighborhood, and listing address
select population.population, airbnb_loc.neighborhood, airbnb_loc.address
from airbnb_loc
inner join population
        on airbnb_loc.neighborhood = population.neighborhood;
```

	population integer	neighborhood character varying (30)	address text
1	25681	hyde park	Hyde Park - Walk to UChicago, 10 min to McCormick
2	79288	south lawndale	394 Great Reviews. 127 y/o House. 40 yds to train.
3	82236	west town	Tiny Studio Apartment 94 Walk Score
4	64116	lincoln park	Barbara's Hideaway - Old Town
5	25681	hyde park	3 Comforts of Cooperative Living
6	64116	lincoln park	The Biddle House (#1)
7	80484	near north side	Chicago GOLD COAST 1 Bedroom Condo
8	64116	lincoln park	Lincoln Park Guest House
9	82236	west town	*** Luxury in Chicago! 2BR/ 2Ba / Parking / BBQ **
10	64116	lincoln park	Private Apt 1 Block to Fullerton L Red Line - Deck
11	64116	lincoln park	Top 2/1 Block to Fullerton L Red Line Deck & Yard
12	72791	logan square	Sanitized, Huge, Quirky Bucktown Loft + Parking
13	56362	uptown	Andersonville - Perfect location!
14	80484	near north side	Central guestroom! Walk everywhere!
15	64116	lincoln park	2 Bed MCM 1 Block to Fullerton Red line L & Garage
16	82236	west town	Rest, Relax and Explore
17	56362	uptown	5★Cubs/Riviera/AragonTRAIN2Bdrms/Ba

Tables Used
1. Airbnb_loc
2. population

Functions Used
1. Inner Join

Load: Relationship Queries and Tables

QUERY 3

```
--Number of listings, population per neighborhood
select population.population, airbnb_loc.neighborhood
from airbnb_loc
inner join population
        on airbnb_loc.neighborhood = population.neighborhood
group by airbnb_loc.neighborhood, population.population;
```

Tables Used
1. Airbnb_loc
2. population

Functions Used
1. Group by
2. Inner Join

	population integer	neighborhood character varying (30)
1	11187	edison park
2	31028	chatham
3	44377	new city
4	35505	west englewood
5	53359	irving park
6	17931	north park
7	56323	humboldt park
8	23042	east side
9	45368	brighton park
10	55628	chicago lawn
11	32602	greater grand crossing
12	11717	washington park
13	30654	englewood
14	44619	roseland
15	2876	fuller park
16	13393	archer heights
17	98514	austin
18	31977	bridgeport
19	18109	west elsdon

Load: Relationship Queries and Tables

QUERY 4

```
--Number of "recently" reviewed listings per neighborhood
select airbnb_loc.neighborhood, count(airbnb_loc.neighborhood) as listing_count
from airbnb_loc
inner join listing
        on airbnb_loc."id" = listing."id"
where date_part('year', listing.last_review) >= 2020
group by neighborhood
order by count(neighborhood) DESC;
```

	neighborhood character varying (30)	listing_count bigint
1	west town	479
2	lake view	331
3	near north side	289
4	logan square	284
5	lincoln park	209
6	near west side	166
7	loop	164
8	lower west side	136
9	edgewater	114
10	north center	104
11	uptown	102
12	irving park	101
13	avondale	100
14	bridgeport	84
15	rogers park	82
16	near south side	80
17	east garfield park	69
18	lincoln square	61
19	hyde park	60
20	grand boulevard	57
21	south shore	57
22	armour square	53
23	portage park	52
24	woodlawn	52
25	humboldt park	47

Tables Used
1. airbnb_loc
2. listing

Functions Used
1. Group by
2. Order by
3. Desc
4. Where
5. Inner Join

Load: Relationship Queries and Tables

QUERY 5

```
--Number of frequently reviewed listings per neighborhood
select airbnb_loc.neighborhood, count(airbnb_loc.neighborhood) as listing_count
from airbnb_loc
inner join listing
        on airbnb_loc."id" = listing."id"
where listing.number_of_reviews >= 100
group by neighborhood
order by count(neighborhood) DESC;
```

Tables Used
1. airbnb_loc
2. listing

Functions Used
1. Group by
2. Order by
3. Desc
4. Inner join
5. where

	neighborhood character varying (30)	listing_count bigint
1	west town	149
2	logan square	91
3	lake view	83
4	lincoln park	53
5	near north side	45
6	lower west side	44
7	near west side	34
8	edgewater	28
9	north center	27
10	uptown	26
11	hyde park	23
12	avondale	23
13	east garfield park	21
14	rogers park	18
15	irving park	18
16	lincoln square	16
17	bridgeport	16
18	portage park	13
19	armour square	13

Load: Relationship Queries and Tables

QUERY 6

```
--Inexpensive room, frequently/recently reviewed
select airbnb_loc.neighborhood, count(airbnb_loc.neighborhood) as listing_count
from airbnb_loc
inner join listing
        on airbnb_loc."id" = listing."id"
inner join population
        on airbnb_loc.neighborhood = population.neighborhood
where CAST(listing.price as decimal) <= 100
        and listing.number_of_reviews >= 100
        and date_part('year', listing.last_review) >= 2020
        and population.population >=50000
        and CAST(population.income as decimal) >= 50000
group by airbnb_loc.neighborhood
order by count(airbnb_loc.neighborhood) DESC;
```

	neighborhood character varying (30)	listing_count bigint
1	west town	69
2	lake view	33
3	near north side	20
4	near west side	17
5	lincoln park	15
6	irving park	11
7	portage park	10

Tables Used
1. Airbnb_loc
2. Listing
3. population

Functions Used
1. Group by
2. Order by
3. Desc
4. Where
5. Inner join (2)

THANK YOU