San Francisco Eviction Database

Background and Data

Dataset Description

San Francisco's Open Data portal provides records of eviction notices filed with the San Francisco Rent Board (1997-Present). An eviction notice does not necessarily indicate that the tenant was eventually evicted. The data table is comprised of geographic and temporal records, as well as 19 different just causes for eviction. For specific details on how to acquire and load the data, please see Appendix 1.

Spatial Questions

This database can be used to conduct spatial and temporal analysis on eviction notices in San Francisco (1997-Present). The spatial questions explored within this project are by no means exhaustive and are meant to be built upon. The three questions included in this analysis explore the average number of eviction notices per tract under various constraints including having a significant Hispanic population, having a median income of less than $25,000 with reason_id constraints, and having a median income greater than $120,000.

Background and Future Questions

San Francisco is no stranger to housing shortages. The first major strain on the housing market was caused by the 1906 earthquake and fires. The "dot com era" of the 1990s placed increasing pressure on the housing market, making San Francisco the most expensive place to live in the United States, with a median selling price of $1.3 million and median rent of $2,600, down 25% since March of 2020. Over the past few decades tech companies have flocked to the city, bringing their wealthy engineers and developers into the housing market. In 2020, many companies in the Bay Area bought out their leases and allowed their employees to work remotely from anywhere in the world. It remains to be seen whether the shift to a remote workforce will present a long term pressure release on the city's rental and buyers market. This database can be used to answer a range of geospatial and temporal questions about eviction trends. In the coming years, new questions about the lasting impacts of a remote workforce could be incorporated into this analysis.

Additional Sources

For analytical purposes, tract level TIGER/Line Selected Demographic and Economic Data from the 2018 5-year American Community Survey will also be incorporated, however it will not be included in database normalization process detailed here. For your own research purposes, you should feel free to include any additional or alternative contextual data.

Normalization Process

For the purpose of minimizing redundancy and eliminating derived variables in the original dataset, several steps were taken to transform it into a normalized database. For your reference, the ETL script can be found in Appendix 2.

In its original form, the SF Evictions dataset contains several forms of location information as well as 19 reasons for just cause eviction. The first set of steps in the normalization process included preparing the lat/lon column to be spatialized, or cast to geometry. This was done using the TRIM() and SPLIT_PART() function, which split the single lat/lon column into two columns and rid them of their non-numeric characters. Also in the first step, any numeric values with improper characters or empty values were set as NULL. Once these steps were taken, the new 'clean_original_notice' table was ready to be formed into three new tables: reason_lookup, eviction_notice, and eviction_reason.

The 'reason_lookup' is comprised of two hard coded variables, 'reason_id' and 'reason'. This table allows you to join to the eviction_reason table if you want your query to include the actual reason for eviction rather than just the 'reason_id'.

The 'eviction_notice' table is the main table, comprised of spatial temporal data and unique identifiers. Notably, several columns were not transfered to this table from the original table because they were derivative of other necessary columns (i.e. neighborhood, supervisor_district).

The 'eviction_reason' table is created last because it's primary key relies on variables from the previously mentioned tables. This table's primary key is a combination of eviction_id and reason_id. It is populated by 19 individual statements that insert the eviction_id where the reason columns from the original dataset = 'true', and hardcoded with the corresponding reason_id from the reason_lookup table.

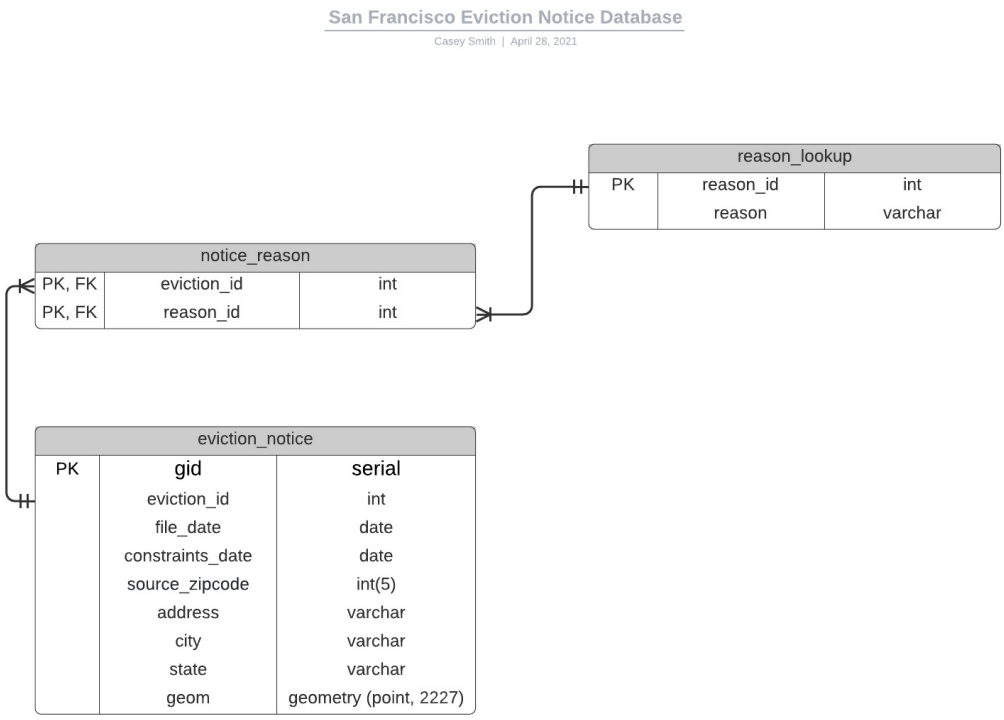

Figure 1: IMAGE

Analysis

1. Between 2014-2018, what was the average number of evictions per tract in tracts with a significant Hispanic population, compared to the average of tracts overall?

First let's take a look at all San Francisco tracts:

```
select avg(evictions) as  avg_evictions_per_tract
 from (
        select d.geoid as tract, count(n.geom) as evictions, d.geom
        from acs_demographic d
        left join eviction_notice n
        on st_contains(st_transform(d.geom, 4326),         st_transform(n.geom, 4326))
            and extract(year from n.file_date) > 2013
            and extract(year from n.file_date) < 2019
        group by d.geoid, d.geom)
    as counts;
```

Now let's confine our search to tracts with a Hispanic population of greater than 30 percent:

```
select avg(evictions) as  avg_evictions_per_tract
    from (
           select d.geoid as tract, count(n.geom) as evictions, pct_hispanic_e ,d.geom
           from acs_demographic d
           left join eviction_notice n
           on st_contains(st_transform(d.geom, 2227), st_transform(n.geom, 2227))
               where d.pct_hispanic_e > 30
               and extract(year from n.file_date) > 2013
               and extract(year from n.file_date) < 2019
           group by d.geoid, d.geom, pct_hispanic_e)
       as counts;
```

Results: Between the years 2014 and 2018, the average number of evictions per tract is 47.9. When we confine the search to include only tracts with a Hispanic population of greater than 30%, the average rises to 66.1.

Query Description: Both of the above queries follow the same format. The subquery counts the number of evictions in each tract using the function ST_CONTAINS. In this case, ST_CONTAINS uses the tract geometry to identify which point geometries (eviction notices) fall within the tract's geometry. Then COUNT(n.geom) finds the total number of rows (eviction notices) for each tract. In order to find the average of the count column (aliased as 'evictions'), this must be written as a subquery because aggregate functions cannot be nested within each other i.e. AVG(COUNT(n.geom)).

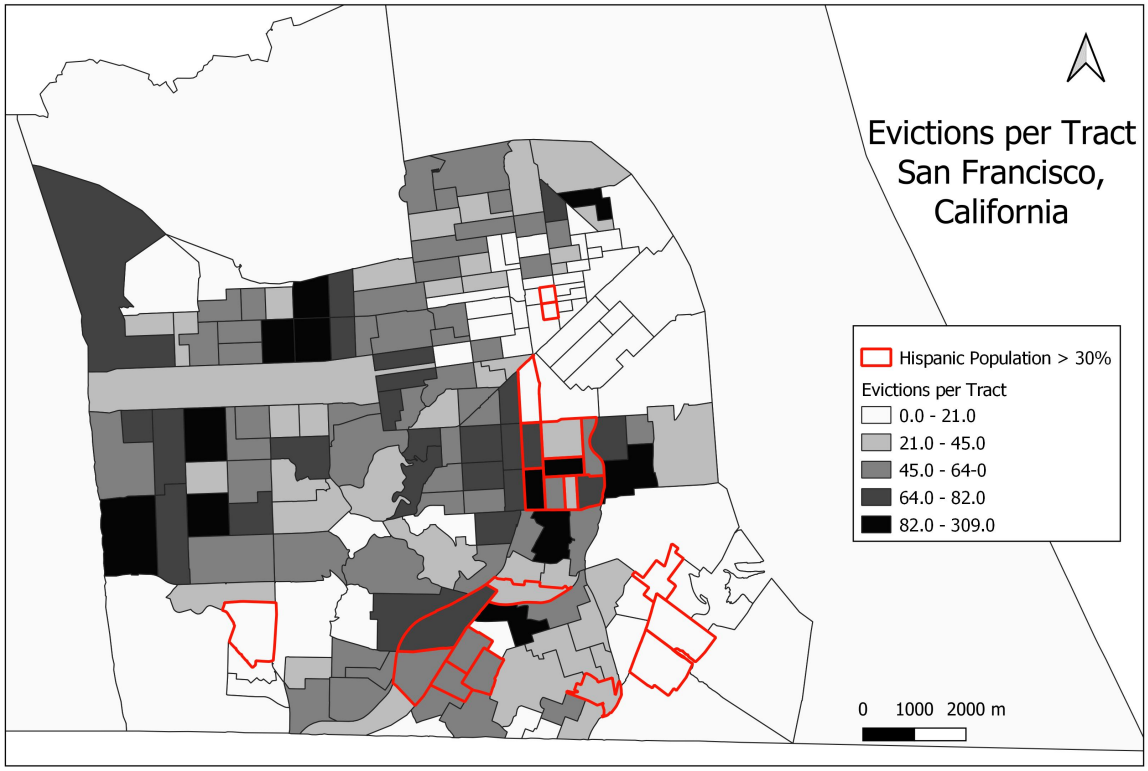

Figure 2: IMAGE

2. Between 2014-2018, what was the average number of evictions with the reason 'non_payment' or 'late_payments' per tract? What about tracts with a Median Income below $25,000?

First, let's look at the overall average:

```
select avg(evictions) as  avg_evictions_per_tract
 from (
        select d.geoid as tract, count(n.geom) as evictions, d.geom
        from acs_demographic d
        left join eviction_notice n
        on st_contains(st_transform(d.geom, 2227), st_transform(n.geom, 2227))
        join eviction_reason r
        on n.eviction_id = r.eviction_id
        where reason_id < 3
            and extract(year from n.file_date) > 2013
            and extract(year from n.file_date) < 2019
        group by d.geoid, d.geom
        )
    as counts;
```

Now let's take a look at the tracts with a median income of less than $25,000:

```
    select avg(evictions) as  avg_evictions_per_tract
    from (
            select d.geoid as tract, count(n.geom) as evictions ,d.geom
            from acs_demographic d
            left join eviction_notice n
            on st_contains(st_transform(d.geom, 2227), st_transform(n.geom, 2227))
            join eviction_reason r
            on n.eviction_id = r.eviction_id
            where reason_id < 3
                and d.median_income_e < 25000
                and extract(year from n.file_date) > 2013
                and extract(year from n.file_date) < 2019
            group by d.geoid, d.geom
            )
        as counts;
```

Results: Between the years 2014-2018, the average number of eviction notices for nonpayment or late payment of rent across all San Francisco tracts was 5.39. When the query is changed to include only tracts with a median income of less than $25,000, that number nearly doubles to 10.33. Now certainly, income and ability to pay rent are correlated variables, so this is not surprising. In our next set of questions we will look at whether this phenomenon applies to all reasons for eviction.

Query Description: The queries here resemble those in Question #1, and for that reason I will not reiterate the functions of ST_CONTAINS and the subquery. Instead I will expand on the additional join that takes place, with the eviction_reason table. As you will notice there are now three tables involved in this query instead of two. The third table is the eviction_reason table, which encodes each eviction notice with a reason for eviction. non_payment and late_payments are encoded as reason_id 1 and 2, respectively, which is why the WHERE statement utilizes the language 'reason_id < 3'. I will also take this opportunity to discuss the function ST_TRANSFORM, which I left out of the query description in Question #1. ST_TRANSFORM returns a new geometry with its coordinates transformed to a different spatial reference system, in this case SRID 2227, which is California State Plane - Zone 3. It is a cheap function, and is extremely useful when one is trying to avoid taking up too much memory.

3. Now that we have looked at the correlation between non_payment/late_payment evictions and low-income tracts, let's look at the average number of all evictions in tracts with a median income less than $25,000. What about tracts with a median income greater than $120,000?

First let's take a look at low-income tracts:

```
select avg(evictions) as  avg_evictions_per_tract
 from (
        select d.geoid as tract, count(n.geom) as evictions, d.geom
        from acs_demographic d
        left join eviction_notice n
        on st_contains(st_transform(d.geom, 4326), st_transform(n.geom, 4326))
        join eviction_reason r
        on n.eviction_id = r.eviction_id
        where d.median_income_e < 25000
            and extract(year from n.file_date) > 2013
            and extract(year from n.file_date) < 2019
        group by d.geoid, d.geom, d.median_income_e
        order by d.median_income_e
        )
    as counts;
```

Now, let's take a look at high-income tracts:

```
select avg(evictions) as  avg_evictions_per_tract
 from (
        select d.geoid as tract, count(n.geom) as evictions, d.geom
        from acs_demographic d
        left join eviction_notice n
        on st_contains(st_transform(d.geom, 4326), st_transform(n.geom, 4326))
        join eviction_reason r
        on n.eviction_id = r.eviction_id
        where d.median_income_e > 120000
            and extract(year from n.file_date) > 2013
            and extract(year from n.file_date) < 2019
        group by d.geoid, d.geom, d.median_income_e
        order by d.median_income_e
        )
    as counts;
```

Results: The results from the from the first query in Question #1 will help us here, if we look back, we'll see that the average number of evictions per tract across all SF tracts was 47.9. From the first query above we see that there is a slight increase, to 52.14, in the average number of evictions in low-income tracts. While our second query tells us that high-income tracts have a significantly lower average number of evictions, at 38.9.

Query Description: Since we already calculated the average number of evictions per tract across all SF tracts in Question #1, we only need to query tracts with the low-income and high-income constraints the respective WHERE clauses.

Appendix 1 - Obtaining and Loading Data

SF Open Data - Eviction Notices

The SF Eviction Notice dataset can be obtained from the above link and exported as a CSV. The dataset can be imported into your database using ogr2ogr or DB Manager in QGIS. In the dataset's original form, the locations are provided as latitude and longitude in a single column, separated by commas. In order to transform the lat/lon into point geometry, they will need to be separated into two individual columns. The

dataset is also available as a shapefile, which would allow you to skip this step. In the case that the shapefile format is used instead of the CSV file, the ETL script in Appendix 2 will need to be altered accordingly, as it was written for the CSV format.

For analysis purposes I have provided a sample of Median Income and Race/Ethnicity demographics in a file with San Francisco tract geometry. This data was selected from the 2018 5-year ACS. The table can be uploaded to your database via ogr2ogr or DB Manager in QGIS, just as the original SF Eviction Notice dataset was. Upon being uploaded, the data types of the demographic columns will need to be changed from text type to int and float type. This will **not** be included in the scripts.

Appendix 2 - Data Normalization Script

```
    set search_path = eviction, public;

--PART 1: Clean the original table 'original_notice'
--1. Empty Values --> NULL
--1a. 'location' values empty --> NULL
update original_notice
set location = null
where location not ilike '%,%';

--1b. 'constraint_date' values empty --> NULL
update original_notice
set constraints_date = null
where constraints_date not ilike '%/%';

--1c. 'file_date' values empty --> NULL
update original_notice
set file_date = null
where file_date not ilike '%/%';

--1d. 'source_zipcode' values empty --> NULL
update original_notice
set source_zipcode = null
where source_zipcode ilike '';

--2. Split 'location' column into two 'point_x', 'point_y' columns
--2a. Create 'clean_original_notice' table (only difference is 'location' column split)
drop table if exists clean_original_notice;
create table clean_original_notice (
gid serial,
eviction_id text,
address text,
city text,
state text,
source_zipcode text,
file_date text,
non_payment text,
breach text,
nuisance text,
illegal_use text,
failure_to_sign_renewal text,
access_denial text,
unapproved_subtenant text,
owner_move_in text,
```

```
demolition text,
capital_improvement text,
substantial_rehab text,
ellis_act_withdrawal text,
condo_conversion text,
roommate_same_unit text,
other_cause text,
late_payments text,
lead_remediation text,
development text,
good_samaritan_ends text,
constraints_date text,
supervisor_district text,
neighborhood text,
point_x double precision,
point_y double precision
);

--2b. Populate 'clean_original_notice' table
insert into clean_original_notice (gid, eviction_id, address, city, state, source_zipcode, file_date, n
select gid::serial,
eviction_id,
address,
city,
state,
source_zipcode,
file_date,
non_payment,
breach,
nuisance,
illegal_use,
failure_to_sign_renewal,
access_denial,
unapproved_subtenant,
owner_move_in,
demolition,
capital_improvement,
substantial_rehab,
ellis_act_withdrawal,
condo_conversion,
roommate_same_unit,
other_cause,
late_payments,
lead_remediation,
development,
good_samaritan_ends,
constraints_date,
supervisor_district,
neighborhood,
trim(split_part(location, ',',1), '('::double precision
    as point_x,
trim(split_part(location, ',', 2), ')'::double precision
    as point_y
from original_notice
```

```
;

--PART 2: ETL

--3a. Create 'reason_lookup' table
drop table if exists reason_lookup cascade;
CREATE TABLE "reason_lookup" (
  "reason_id" int ,
  "reason" varchar,
  PRIMARY KEY ("reason_id")
);

--3b. Populate 'reason_lookup' table
insert into reason_lookup (reason_id, reason)
values (1, 'non_payment'),
(2, 'late_payments'),
(3, 'nuisance'),
(4, 'illegal_use'),
(5, 'failure_to_sign_renewal'),
(6, 'access_denial'),
(7, 'unapproved_subtenant'),
(8, 'owner_move_in'),
(9, 'demolition'),
(10, 'capital_improvement'),
(11, 'substantial_rehab'),
(12, 'ellis_act_withdrawl'),
(13, 'condo_conversion'),
(14, 'roommate_same_unit'),
(15, 'other_cause'),
(16, 'breach'),
(17, 'lead_remediation'),
(18, 'development'),
(19, 'good_samaritan_ends')
;

--4a. Create main table 'eviction_notice'
drop table if exists eviction_notice cascade;
CREATE TABLE "eviction_notice"(
  "gid" serial,
  "eviction_id" varchar,
  "file_date" date,
  "constraints_date" date,
  "source_zipcode" int,
  "address" varchar,
  "city" varchar,
  "state" varchar,
  "geom" geometry(point, 2227),
  PRIMARY KEY ("gid"),
  UNIQUE ("eviction_id")
);

--4b. Populate 'eviction_notice' table
insert into eviction_notice (eviction_id, file_date, constraints_date, source_zipcode, address, city, s
select
```

```
eviction_id::varchar,  'int'
file_date::date,
constraints_date::date,
source_zipcode::int,
address::varchar,
city::varchar,
state::varchar,
st_transform(st_setsrid(st_point(point_y, point_x), 4326), 2227) as geom --California State Plane, Zone
from clean_original_notice;

--5a. Create eviction_reason table
drop table if exists eviction_reason cascade;
create table "eviction_reason" (
"eviction_id" varchar references eviction_notice(eviction_id),
"reason_id" int references reason_lookup(reason_id),
 primary key (eviction_id, reason_id)
);

--5b. Populate 'eviction_reason' table
--Non-payment, 1
INSERT INTO eviction_reason
SELECT eviction_id, 1
FROM clean_original_notice
WHERE non_payment = 'true';


--late_payment, 2
INSERT INTO eviction_reason
SELECT eviction_id, 2
FROM clean_original_notice
WHERE late_payments = 'true';

--nuisance, 3
INSERT INTO eviction_reason
SELECT eviction_id, 3
FROM clean_original_notice
WHERE nuisance = 'true';

--illegal_use, 4
INSERT INTO eviction_reason
SELECT eviction_id, 4
FROM clean_original_notice
WHERE illegal_use = 'true';

--failure_to_sign_renewal, 5
INSERT INTO eviction_reason
SELECT eviction_id, 5
FROM clean_original_notice
WHERE failure_to_sign_renewal = 'true';

--access_denial, 6
INSERT INTO eviction_reason
SELECT eviction_id, 6
FROM clean_original_notice
```

```
WHERE access_denial = 'true';

--unapproved_subtenant, 7
INSERT INTO eviction_reason
SELECT eviction_id, 7
FROM clean_original_notice
WHERE unapproved_subtenant = 'true';

--owner_move_in, 8
INSERT INTO eviction_reason
SELECT eviction_id, 8
FROM clean_original_notice
WHERE owner_move_in = 'true';

--demolition, 9
INSERT INTO eviction_reason
SELECT eviction_id, 9
FROM clean_original_notice
WHERE demolition = 'true';

--capital_improvement, 10
INSERT INTO eviction_reason
SELECT eviction_id, 10
FROM clean_original_notice
WHERE capital_improvement = 'true';

--substantial_rehab, 11
INSERT INTO eviction_reason
SELECT eviction_id, 11
FROM clean_original_notice
WHERE substantial_rehab = 'true';

--ellis_act_withdrawl, 12
INSERT INTO eviction_reason
SELECT eviction_id, 12
FROM clean_original_notice
WHERE ellis_act_withdrawal = 'true';

--condo_conversion, 13
INSERT INTO eviction_reason
SELECT eviction_id, 13
FROM clean_original_notice
WHERE condo_conversion = 'true';

--roommate_same_unit, 14
INSERT INTO eviction_reason
SELECT eviction_id, 14
FROM clean_original_notice
WHERE roommate_same_unit = 'true';

--other_cause, 15
INSERT INTO eviction_reason
SELECT eviction_id, 15
FROM clean_original_notice
```

```
WHERE other_cause = 'true';

--breach, 16
INSERT INTO eviction_reason
SELECT eviction_id, 16
FROM clean_original_notice
WHERE breach = 'true';

--lead_remediation, 17
INSERT INTO eviction_reason
SELECT eviction_id, 17
FROM clean_original_notice
WHERE lead_remediation = 'true';

--development, 18
INSERT INTO eviction_reason
SELECT eviction_id, 18
FROM clean_original_notice
WHERE development = 'true';

--good_samaritan_ends, 19
INSERT INTO eviction_reason
SELECT eviction_id, 19
FROM clean_original_notice
WHERE good_samaritan_ends = 'true';
```

Appendix 3 - Creating a Spatial Index

```
CREATE INDEX ON eviction_notice
USING gist (ST_Transform(geom, 2227));
```

Appendix 4 - Data Dictionary

eviction_id: (text) eviction notice unique identifier provided by San Francisco Rent Board.

file_date: (date) The date on which the eviction notice was filed with the Rent Board of Arbitration.

contraints_date: (date) In the case of certain just cause evictions like Ellis and Owner Move In, constraints are placed on the property and recorded by the City Recorder. This date represents the date through which the relevant constraints apply. You can learn more on fact sheet 4 of the Rent Board available at: http://sfrb.org/fact-sheet-4-eviction-issues

source_zipcode: (int) The zip code where the eviction notice was issued.

address: (text) The address where the eviction notice was issued. The addresses are represented at the block level.

city: (text) The city where the eviction notice was issued. In this dataset, always San Francisco.

state: (text) The state where the eviction notice was issued. In this dataset, always CA.

geom: (point, 2227) The point geometry of each eviction notice is based on geocoded latitude and longitude location of the record is at the mid block level and is represented by it's latitude and longitude. Some addresses are not well formed and do not get geocoded. These will be blank. Geocoders produce a confidence match rate. Since this field is automated, we set the match at 90% or greater. Please note, that even this rate could result in false positives however more unlikely than at lower confidence levels.

reason_id: (int) surrogate key given to each of the following reasons for eviction.

non_payment: Nonpayment of rent or frequent bounced checks.

late_payment: Habitual late payment or frequent bounced checks.

nuisance: Nuisance or substantial damage to the unit (waste), or "creating a substantial interference with the comfort, safety, or enjoyment of the landlord or other tenants in the building."

illegal_use: Illegal use of the unit. This just cause may not be used to evict a tenant from an illegal residential unit.

failure_to_sign_renewal: Termination of the rental agreement and the tenant refuses to execute a written extension for materially the same terms.

access_denial: The tenant has, after written notice to cease, refused the landlord access to the unit as required by law.

unapproved_subtenant: Unapproved subtenant (approval can be either stated or implied) is the only person still remaining in the unit (subtenant holding over).

owner_move_in: Move-in of the landlord or a close relative of the landlord (if the landlord lives in the building). The tenant has a right to relocation payments.

demolition: Demolition or removal of the unit from housing use. The tenant has a right to relocation payments.

capital_impovement: Capital improvements or rehabilitation with all the necessary permits that allows temporary removal of the unit from housing use. The tenant has the right to re-occupy the unit once the work is completed at the prior rent, adjusted by the Rent Board's allowable rent increases such as the annual rent increase. The tenant has a right to relocation payments.

substantial_rehab: "Substantial rehabilitation" of a building that is essentially uninhabitable with all the necessary permits. The tenant has a right to relocation payments.

ellis_act_withdrawl: Ellis Act evictions which require withdrawal from rental housing use all of the units in the building or a unit detached from another structure on the same lot (e.g. a cottage). Tenants evicted for this cause have a right to a relocation payment.

condo_conversion: Sale of a unit which has been converted to a condo. Seniors and permanently disabled tenants cannot be evicted for condo conversions. Tenants have a right to a 1-year lease or 120 days with relocation payments.

roommate_same_unit: if the landlord indicated if they were evicting a roommate in their unit as a grounds for eviction.

other_cause: eviction cause other than existing just causes

breach: Breach (violation) of a term of the rental agreement that has not been corrected after written notice from the landlord.

lead_remediation: Lead abatement as required by the San Francisco Health Code with temporary removal of the unit from housing use for less than 30 days. The tenant has a right to a relocation payment.

development: Demolition or to otherwise permanently remove the rental unit from housing use in accordance with the terms of a development agreement entered into by the City under Chapter 56 of the San Francisco Administrative Code.

good_samaritan_ends: Good Samaritan Occupancy Status for the tenant expires, and the landlord serves an eviction notice within 60 days after expiration of the status. (The Good Samaritan Occupancy Status is when a tenant loses their home due to a disaster and the landlord rents another temporary unit to the tenant for low rent.)