UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO

DIPARTIMENTO DI INFORMATICA

CORSO DI LAUREA IN DATA SCIENCE A.A. 2023/2024

RELAZIONE DI PROGETTO

CASO DI STUDIO

CORSO DI GESTIONE DI DATI STRUTTURATI E NON STRUTTURATI

DOCENTI

Prof. Mario Alessandro Bochicchio

Prof. Corrado Loglisci

STUDENTI

Detomaso Giacomo

(e-mail: g.detomaso7@studenti.uniba.it)

Detomaso Gabriele

(e-mail: g.detomaso6@studenti.uniba.it)

Sommario

Gestione dati strutturati 3
Obiettivi 3
Fonti dati individuate e diagramma architettura 3
ETL 5
Fonte dati D01 5
Fonte dati D02 5
Fonte dati D03 6
Fonte dati D04 6
Fonte dati D05 6
Fonte dati D06 6
Fonte dati D07 7
Fonte dati D08 7
Fonte dati D09 7
Fonte dati D10 7
Diagramma ER e modello relazionale, stima sulle dimensioni del DB 8
Queries di analisi dei dati 9
Premessa: query di supporto 9
Q1 9
Q2 10
Q3 10
Q4 10
Q5 10
Q6 10
Q Final 11
Presentazione dei risultati 11
Presentazione Q1 11
Presentazione Q2 12
Presentazione Q3 12
Presentazione Q4 13
Presentazione Q5 13
Presentazione Q6 13
Presentazione Q Final 14
Commento finale 14
Matrice delle responsabilità 15
Sitografia 15

Gestione dati strutturati

Obiettivi

Un imprenditore che opera nel campo dei **B&B**, gestendone uno, con ottimi risultati a Peschici, necessita di **espandere** la propria attività in una grande città estera in modo da avviare l'apertura di una **catena di B&B**. In particolare, è rimasto molto colpito dalle opportunità che la città di **New York** *sembra* offrire e ha messo in evidenza due questioni essenziali:

1. Il luogo più vantaggioso dove aprire un nuovo B&B;
2. Le caratteristiche consigliate (idealmente), intese come servizi e costi, da assegnare al suddetto B&B.

In definitiva nell'ambito di questa analisi di dati strutturati si vuole:

- Analizzare le realtà già esistenti (competitors) sul territorio al fine di poterne determinare i fattori di successo attraverso:
  - Numero di recensioni;
  - Compliance con nuove regolamentazioni;
  - Servizi offerti;
  - Prezzi generali.
- Analizzare i Borough di NYC, ordinandoli in base a diversi fattori critici per l'attività quali:
  - Safety rate (tasso di criminalità);
  - Competitors (numerosità dei bnb nello specifico Borough);
  - Presenza di POI (numerosità dei Punti Di Interesse).
- Analizzare l'andamento, **attuale**, del mercato immobiliare individuando:
  - Aree di NYC con un prezzo degli appartamenti inferiore o uguale alla media, avendo cura di riportare i m^2 medi delle case vendute;

Fonti dati individuate e diagramma architettura

Le fonti dati scelte per il progetto sono elencate nelle seguenti tabelle, dove è presente un link per il download e un codice identificativo.

Nome file	Link	Codice
listings.csv	http://data.insideairbnb.com/united-states/ny/new-york-city/2023-11-01/visualisations/listings.csv	D01
stops.csv	https://www.kaggle.com/datasets/monsieurwagner/nyctransit?select=stops.csv	D02
[district_name].csv	https://www.nyc.gov/site/finance/taxes/property-rolling-sales-data.page	D03
NYPD_Arrest_2023.csv	https://www.kaggle.com/datasets/justinpakzad/nypd-arrests-2023-dataset	D04
nyc_borough_boundaries.zip	https://data.cityofnewyork.us/City-Government/2020-Neighborhood-Tabulation-Areas-NTAs-Tabular/9nt8-h7nd	D05
nyc_roads.zip	https://data.cityofnewyork.us/City-Government/NYC-Street-Centerline-CSCL-/exjm-f27b	D06
nyc_parks.zip	https://data.cityofnewyork.us/Recreation/Parks-Properties/enfh-gkve	D07
nyc_bus_stops_shelters.zip	https://data.cityofnewyork.us/Transportation/Bus-Stop-Shelters/qafz-7myz	D08
nyc_points_of_Interest.zip	https://data.cityofnewyork.us/City-Government/Points-Of-Interest/rxuy-2muj	D09

nyc_borough.zip	https://data.cityofnewyork.us/City-Government/Borough-Boundaries/tqmj-j8zm	D10

Tabella 1

Le fonti dati, elencate precedentemente sono descritte sinteticamente nella seguente tabella.

Codice	Descrizione	Formato dati	Numerosità
D01	Dati sulle prenotazioni di AirBnB a NYC, aggiornate al 2023.	Numerico, stringa, spaziale	10^4
D02	Dati sulle ubicazioni di fermate della metro di New York City	Numerico, stringa, spaziale	10^3
D03	Dati sulle case vendute nell'area di NYC nel corso del 2023, correlate con le tasse relative in base al distretto.	Numerico, stringa, spaziale, date	10^4
D04	Dati relativi agli arresti e relativi crimini avvenuti in NYC nel 2023 e relativa ubicazione spaziale	Numerico, stringa, spaziale, date	10^6
D05	Dati relativi ai perimetri dei 5 distretti di NYC	Numerico, stringa, spaziale	5
D06	Aree di tabulazione dei quartieri (NTA) del 2020 per i 5 distretti di NYC.	Numerico, stringa, spaziale	10^2
D07	Dati relativi alla rete stradale di NYC.	Numerico, stringa, spaziale	10^6
D08	Dati relativi ai perimetri dei parchi dei 5 distretti di NYC	Numerico, stringa, spaziale	10^3
D09	Dati relativi ai punti di fermata dei BUS pubblici di NYC	Numerico, stringa, spaziale	10^3
D10	Dati relativi ai punti di interesse per la città dei NYC nei 5 distretti	Numerico, stringa, spaziale	10^5

Tabella 2

Di seguito è riportato lo schema dell'architettura:

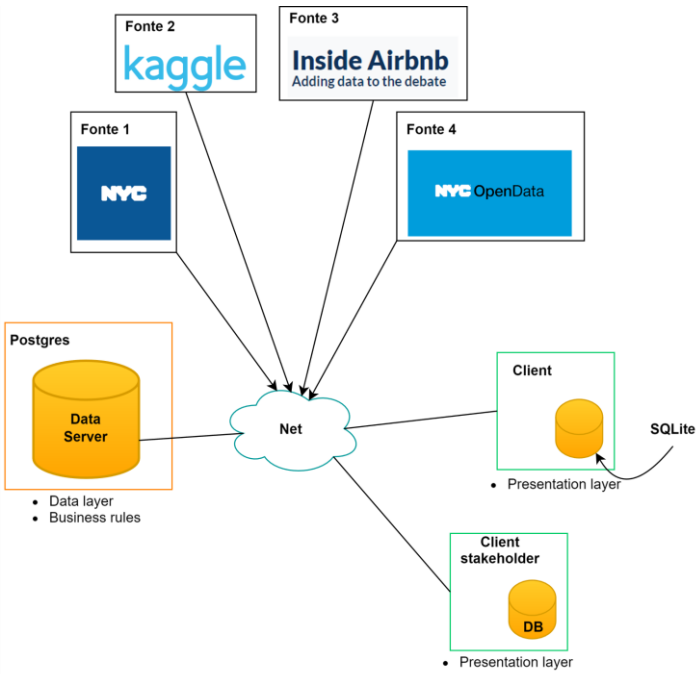


Figura 1

ETL

Nota: Le tabelle contenenti dati provenienti da file CSV sono state create nel file DDL/ddl_csv.sql.

Nota: Le tabelle contenenti dati provenienti da shapefile sono state create nel file DDL/ddl_shapefile.sql.

Fonte dati D01

Nota: la tabella *listings* è una tabella di appoggio per caricare i dati da CSV su database.

Operazione	Descrizione	File di progetto
O01	Estrapolazione di attributi significativi dalla colonna bnb_names usando espressioni regolari. (39k righe elaborate)	*ETL/listings_etl.ipynb*
O02	Inserimento dei dati nella tabella **rental_units** a partire dalla tabella *listings.* (39k righe inserite)	*DML/csv/dml_listings.sql*
O03	Conversione dell'attributo *availability_rate_365* in valore percentuale (39k righe elaborate) e assegnazione del valor medio ove la percentuale è uguale a 0. (13.5k righe elaborate)	*DML/csv/dml_listings.sql*
O04	Inserimento dei dati nella tabella **room_configuration** raggruppando la tabella *listings* per *room_type*, *n_beds*, *n_baths*, *is_bath_shared* (40 righe inserite).	*DML/csv/dml_listings.sql*
O05	Inserimento dei dati nella tabella **rental_fares** raggruppando la tabella *listings* per price e minimum_nights (3.5k righe inserite).	*DML/csv/dml_listings.sql*
O06	Inserimento dei dati nella tabella **rental_resumes** mappando, a partire dalla tabella *listings* le room_configurations e rental_fares con i rispettivi id (39k righe inserite).	*DML/csv/dml_listings.sql*
O07	1. Separazione ed estrazione degli attributi *room_type ed n_rooms* (39k righe elaborate). 2. Assegnazione del valore 1 alle camere di tipo 'Studio' (12 righe elaborate).	*DML/csv/dml_listings.sql*
O08	Inserimento dei dati nella tabella **hosts** raggruppando i vari host per *id* e *nome dalla tabella listings*. (29k righe inserite).	*DML/csv/dml_listings.sql*

Tabella 3

Fonte dati D02

Nota: la tabella *subway_stops_temp* è una tabella di appoggio per caricare i dati da CSV su database.

Operazione	Descrizione	File di progetto
O01	Eliminazione dell'indice dal file csv (1.5k righe eliminate).	*ETL/subway_stops_etl.ipynb*
O02	Inserimento dei dati nella tabelle **subway_stops**, (1.5k righe inserite)	*DML/csv/dml_subway_stops.sql*
O03	Eliminazione delle tuple che fanno riferimento a fermate Nord e (1k righe eliminate).	*DML/csv/dml_subway_stops.sql*

Tabella 2

Fonte dati D03

Nota: la tabella *house_sales_temp* è una tabella di appoggio per caricare i dati da CSV su database.

Operazione	Descrizione	File di progetto
O01	Standardizzazione degli indirizzi delle case (73.2k righe elaborate).	*ETL/rolling_sales.ipynb*
O02	Geocoding degli indirizzi (73.2k righe elaborate).	*ETL/rolling_sales.ipynb*
O03	Inserimento dei dati nella tabella **house** a partire dalla tabella *house_sales_temp*, escludendo le tuple dove il valore relativo ai m^2 è mancante (39.3k righe inserite).	DML/csv/dml_house_sales.sql
O04	Standardizzazione a zero (passaggio di proprietà) per i prezzi delle case con valore <= 10 (17.3k righe elaborate).	DML/csv/dml_house_sales.sql

Tabella 3

Fonte dati D04

Nota: la tabella *nypd_arrests* è una tabella di appoggio per caricare i dati da CSV su database.

Operazione	Descrizione	File di progetto
O01	Formattazione della data con un tipo compatibile con SQL (170k righe elaborate).	*ETL/ NYPD_Arrest_2023_etl.ipynb*
O02	Inserimento dei dati nella tabella **crimes** a partire dalla tabella *nypd_arrests.* (64 righe inserite)	*DML\csv \dml_nypd_arrests.sql*
O03	Inserimento dei dati nella tabella **arrests** a partire tabella *nypd_arrests*, mappando la descrizione (univoca) dell'arresto con l'id del tipo del crimine (valorizzando quindi la chiave esterna). (170k righe inserite)	*DML\csv \dml_nypd_arrests.sql*

Tabella 4

Fonte dati D05

Nota: *boroughs_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dati nella tabella **boroughs**, a partire dalla tabella *boroughs_temp* (5 righe inserite).	*DML\shapefiles\dml_borough.sql*
O02	Mapping del numero del borough con il suo codice letterale (5 righe elaborate).	*DML\shapefiles\dml_borough.sql*

Tabella 5

Fonte dati D06

Nota: *neighborhoods_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dei dati nella tabelle **neighborhoods** a partire dalla tabella *neighborhoods_temp*. (262 righe inserite).	*DML\shapefiles\dml_neighborhood.sql*
O02	Mapping del nome del borough con il codice letterale (262 righe elaborate).	*DML\shapefiles\dml_neighborhood.sql*

Tabella 6

Fonte dati D07

Nota: *roads_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dei dati nella tabella **roads** a partire dalla tabella *roads_temp*. (121.5k righe inserite).	*DML/shapefiles/dml_roads.sql*
O02	Mapping del numero del borough con il suo codice letterale (121.5k righe elaborate).	*DML/shapefiles/dml_roads.sql*
O03	Mapping dello status number delle strade con la descrizione letterale (121.5k righe elaborate).	*DML/shapefiles/dml_roads.sql*

Tabella 7

Fonte dati D08

Nota: *parks_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dei dati nella tabella **parks**, a partire dalla tabella *parks_temp* (2k righe inserite)	*DML/shapefiles/dml_parks.sql*
O02	Eliminazione delle tuple con codici che non rappresentano parchi (200 righe eliminate).	*DML/shapefiles/dml_parks.sql*

Tabella 8

Fonte dati D09

Nota: *bus_stops_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dei dai nella tabella **bus_stops**, a partire dalla tabella *bus_stops_temp*. (3.3k righe inserite).	*DML\shapefiles\dml_bus_stops.sql*

Tabella 9

Fonte dati D10

Nota: *poi_temp* è una tabella di appoggio per caricare i dati da shapefile su database.

Operazione	Descrizione	File di progetto
O01	Inserimento dei dai nella tabella **poi_types**, selezionando in maniera distinta tutti i tipi di facilities (13 righe inserite).	*DML\shapefiles\dml_POI.sql*
O02	Inserimento dati nella tabella **poi**, a partire dalla tabella *poi_temp* (16k righe inserite).	*DML\shapefiles\dml_POI.sql*
O03	Eliminazione di particolari tipi di poi dalla tabella poi. (4k righe eliminate)	*DML\shapefiles\dml_POI.sql*

Tabella 10

Di seguito sono riportate alcune note:

- Per le tabelle ottenute dagli shapefiles sono state riportate, in questa sezione, **solo e soltanto** le operazioni “strutturate”, non sono state incluse operazioni di DML effettuate usando la geometria della tabella;

- Per le tabelle ottenute da CSV non è riportata, in questa sezione, la conversione di latitudine e longitudine in punti geometrici ("coordinates"), in quanto sfrutta concetti relativi alla parte "non strutturata";
- Le constraints sono state definite, al termine del processo di DML nel file DDL/ddl_constraints.sql.

Diagramma ER e modello relazionale, stima sulle dimensioni del DB

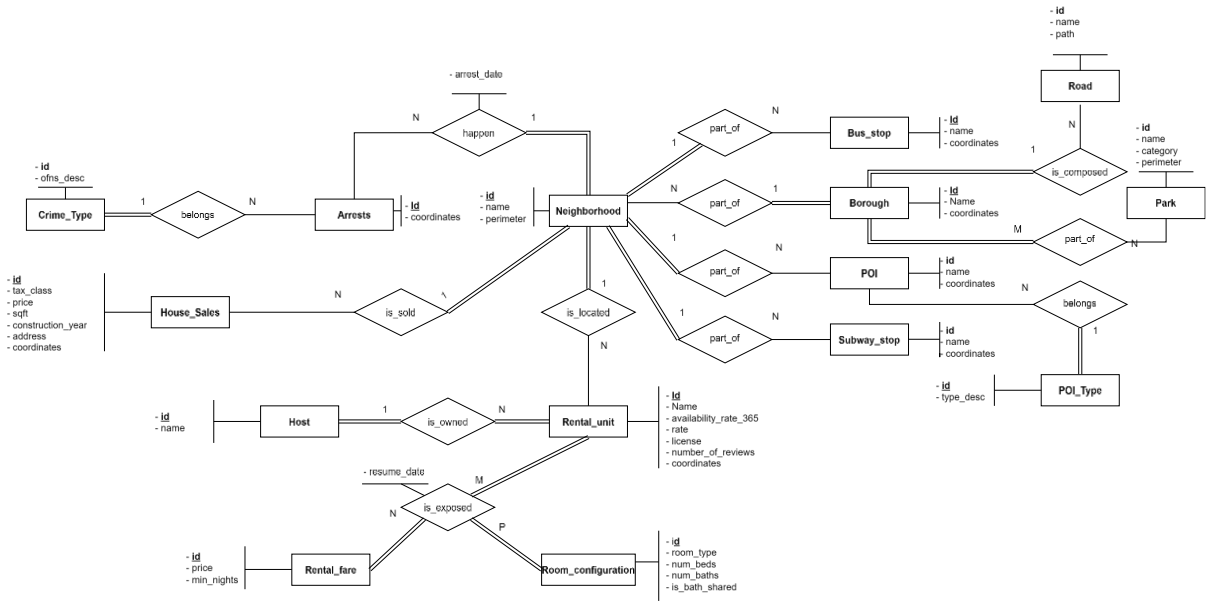


Figura 2

Nota: l'entità House_Sales conterrà le case di NYC vendute nell'anno 2023, non sono inerenti quindi alle strutture BnB dell'entità Rental_unit.

Di seguito è presente il **modello relazionale**. In grassetto sono indicati le **chiavi primarie**, in corsivo con sottolineatura tratteggiata le *chiavi esterne*, inoltre, il nome della chiave esterna in corsivo è lo stesso della tabella a cui il vincolo fa riferimento (ma al singolare e non al plurale).

- **Rental_fares**: (**id**, price, min_nights)*;*
- **Room_configurations**: (**id**, room_type, n_rooms, num_beds, num_baths, is_bath_shared)*;*
- **Rental_units**: (**Id**, name, availability_rate_365, rate, rewiew_number, coordinates, *neighborhood*, *host*)*;*
- **Rental_resumes**: (***rental_fare***, ***rental_unit***, ***room_configuration***, resume_date);
- **Hosts**: (**id**, min_nights);
- **Neighborhoods**: (**id**, name, perimeter, *borough*);
- **House_sales**: (**id**, tax_class, price, sqft, construction_year, address, coordinates, *neighborhood*)*;*
- **Crimes**: (**id**, ofns_desc);
- **Arrests**: (**id**, coordinates, *crime*, *neighborhood*);
- **Bus_stops**: (**id**, name, coordinates, *neighborhood*);
- **Boroughs**: (**id**, name, perimeter, *positioning*)*;*
- **Positionings**: (***borough***, ***park***);
- **Parks**: (**id**, name, perimeter)*;*
- **Roads**: (**id**, name, path, *borough*);

- **Pois**: (**id**, name, coordinates, *neighborhood, poi_type*);
- **Poi_types**: (**id**, *type_desc*);
- **Subway_stop**: (**id**, name, coordinates, *neighborhood*).

Nota: per tutelare la conformità con lo schema concettuale, anche nello schema logico sono stati riportati attributi non relazionali. Questi ovviamente non saranno manipolati in nessuna delle parti riguardanti la parte relazionale del caso di studio.

ENTITÀ	STIMA DIMENSIONE (byte)	ENTITÀ	STIMA DIMENSIONE (byte)
Rental_fares	49 MB	**Bus_stops**	1 GB
Room_configurations	1,4 MB	**Boroughs**	50 B
Rental_units	14,7 GB	**Positionings**	21,5 MB
Rental_resumes	2,3 GB	**Parks**	1 GB
Hosts	1,5 GB	**Roads**	39,8 GB
Neighborhoods	15,9 MB	**Pois**	10 GB
House_sales	28,3 GB	**Poi_types**	0,6 MB
Crimes	17 MB	**Subway_stops**	159 MB
Arrests	7,8 GB		

Tabella 11

Queries di analisi dei dati

Di seguito è riportata una descrizione e il relativo codice delle query di analisi dati effettuate.

Premessa: query di supporto

Lo stakeholder è interessato nell'analizzare le unità di affitto con valutazione che spaziano dal "medio" (3.5) al "eccellente" (5.0) al fine di comprendere a quale fascia di valutazione ambire (rate). Infatti, come ulteriore richiesta ci ha indicato di associare al rate una fascia. Le indicazioni per l'attribuzione di quest'ultima sono le seguenti:

Denominazione fascia	Rate di partenza	Rate di arrivo
AVERAGE	3.5	4.09
GOOD	4.1	4.49
VERY GOOD	4.5	4.79
EXECELLENT	4.8	5.00

Tabella 12

Nota: per la "la legge dei grandi numeri" [1], al fine di individuare le unità di affitto rilevanti per la nostra analisi, abbiamo convenuto nel considerare solo quelle con un *numero di recensioni pari almeno a 50*.

Q1

Individuare l'importanza del rispetto delle nuove norme dello stato di New York, in NYC, in materia di affitti a breve termine (*almeno 30 giorni se non si possiede una licenza*) [2]. Lo stakeholder vuole verificare:

- Il numero di unità di affitto che **non rispettano** le norme di durata minima senza licenza.
- Il numero di unità di affitto che **rispettano** le norme di durata minima senza licenza.
- Il numero di unità di affitto che posseggono una licenza (e di conseguenza rispettano le norme di durata minima).

Tale analisi va fatta, sia per **TUTTE** le unità di affitto, al fine di individuare il trend globale, che per quelle individuate nella premessa.

Q2

Lo stakeholder vuole valutare le caratteristiche ideali di un'unità di affitto, *per ognuna delle fasce di rate* indicate nella premessa. Le caratteristiche che si vogliono valutare sono:

- Prezzo medio e disponibilità media della struttura su 365 giorni;
- Numero medio di letti e bagni;
- *Tendenza* a: disporre di una licenza per affitti a breve termine, disporre o meno di bagni condivisi, disporre di camere di tipo 'bedroom' o 'studio'.

Q3

Lo stakeholder vuole valutare quale dei borough è il più indicato per aprire una nuova unità di affitto tenendo conto del *tasso di arresti* che li caratterizza, in particolare è richiesto un ordinamento dal meno al più pericoloso.

Q4

Lo stakeholder ha richiesto di consultare la densità dei competitors per ogni borough. Nello specifico vorrebbe valutare quante sono le unità di affitto già presenti sul territorio che rispettino determinati parametri come:

- La fascia di rate;
- Le configurazioni delle stanze e le tariffe più gettonate associate ai bnb ideali individuati in Q3.

Q5

Al fine di individuare con maggiore precisione quale borough si presta meglio all'apertura di nuovi bnb, è necessario individuare il numero di punti di interesse che potrebbero rappresentare un'attrattiva per la clientela. In particolare, lo stakeholder vuole valutare, per ogni borough, la numerosità di:

- Servizi di: sanità, commerciali e di ricreazione;
- Fermate di trasporto pubblico, in particolare metro e bus.

Q6

Il cliente ha richiesto un'analisi approfondita sulle vendite degli appartamenti per ogni borough di NYC, al fine di confrontare:

- La probabilità di acquistare una struttura a prezzo minore della media di acquisto del Borough;
- I m^2 delle medi delle suddette case vendute per lo specifico borough;
- Il prezzo *consigliato* di acquisto, calcolato come media dei prezzi di tutte le case il cui prezzo è inferiore alla media dello specifico borough;
- Il prezzo medio delle case per ogni borough.

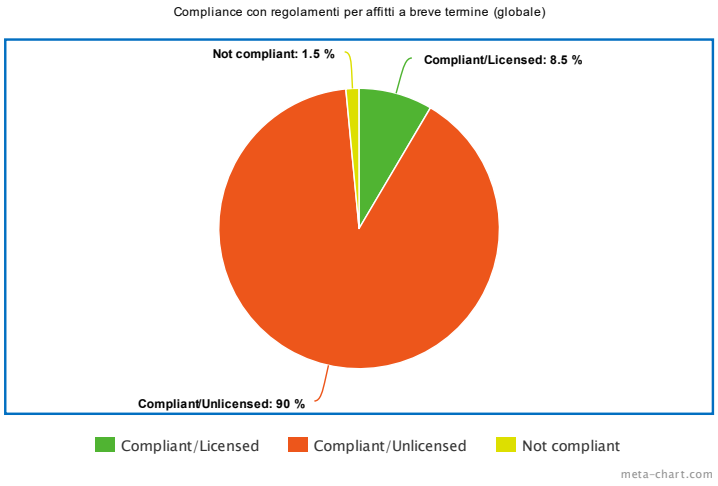
Nota: Gli appartamenti presi in considerazione sono quelli che hanno una class tax uguale ad 1 o 2 [3] in quanto 3 e 4 indicano strutture commerciali o fabbriche (non destinate all'uso ricercato dallo stakeholder). Inoltre, non sono state considerate case con prezzo uguale a zero, in quanto indicanti un semplice passaggio di proprietà senza esborso di denaro.

Q Final

Lo stakeholder è interessato a valutare, senza considerare spese successive, quanti mesi sarebbero necessari, lavorando il 70% dell'anno, per coprire l'esborso iniziale per l'acquisto di una casa da utilizzare come unità di affitto. Si vuole analizzare questa situazione analizzando i prezzi medi, sia in un area di un chilometro dal centro della zona prescelta (Q7 non strutturata), sia considerando i prezzi medi globali, per la fascia excellent (Q2 strutturata) e le relative disponibilità medie annue.

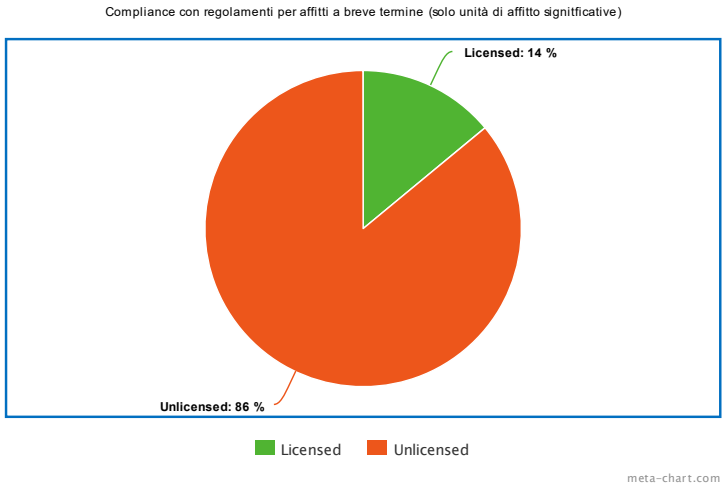
Presentazione dei risultati

Presentazione Q1



Risultati Q1 - 1

Nel primo grafico sono riportate le percentuali di unità d'affitto che risultano o meno **compliance** con il nuovo regolamento su affitti a breve termine, calcolati su **tutte** quelle presenti a NYC. Si può affermare con certezza, quindi, che il mercato si è adattato ai nuovi regolamenti, proponendo affitti di durata minima di 30gg (il 90% delle unità propone questa soluzione).



Risultati Q1 - 2

Il secondo grafico, riportato di seguito, definisce la compliance ai regolamenti di NYC delle unità d'affitto nel range di valutazione (rate) definito con lo stakeholder (3.5 – 5 con almeno 50 recensioni). La tendenza notata dal primo grafico è confermata anche per questo range ridotto. In aggiunta si può notare che tutte le unità di affitto presentano, rispettano i regolamenti, a

dimostrazione (ovvia) che in una grande città come NYC per puntare ad avere “successo” è necessario rispettare le normative vigenti.

Presentazione Q2

rental_unit_band	avg_price	avg_availability_	mode_license	avg_number_of_	avg_number_of_baths	mode_bath_shared	mode_room_type
average	202.29$	0.77	True	1	1	False	Bedroom
execellent	199.14$	0.60	False	1	1	False	Bedroom
good	196.17$	0.66	False	1	1	False	Bedroom
very good	165.91$	0.64	False	1	1	False	Bedroom

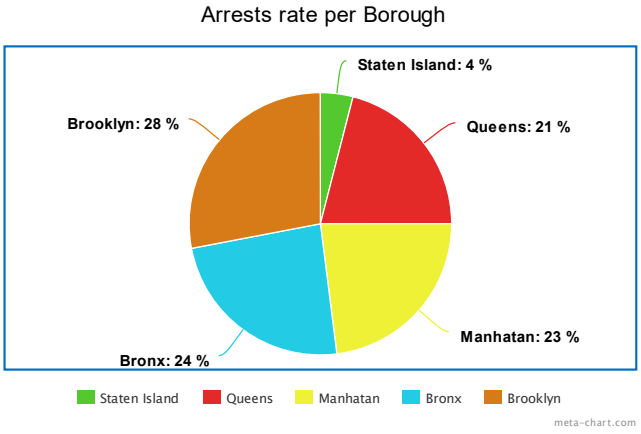
Risultati Q2 - 1

Nella tabella sono state riportate, per ogni fascia di rate (definita durante l’analisi iniziale), le caratteristiche che dovrebbe avere un BnB ideale per condurre una fiorente attività. È importante sottolineare che grazie a tale informazione lo stakeholder ha deciso di che la fascia più appropriata per i suoi interessi è la ‘**Execellent’**. È arrivato a tale conclusione inquanto, come si può facilmente dedurre dai risultati della query, mettendo a confronto le diverse fasce si ha che tutte le caratteristiche ideali sono uguali escluse avg_price avg_availability e mode_license.

Sarebbe pertanto preferibile puntare ad un rate più alto possibile visto che:

- il prezzo medio per notte che si aggira intorno ai 200 euro (simile ad average o good);
- la disponibilità annua è la più bassa tra le varie fasce, il che significa che l’unità di affitto risulta **disponibile** per la prenotazione il 60% dell’anno.

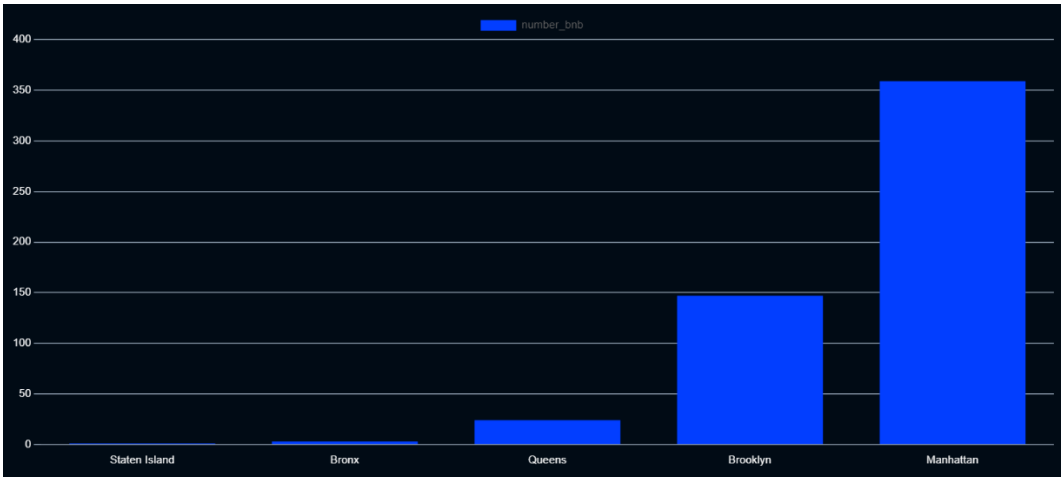
Presentazione Q3



Risultati Q3 - 1

Il grafico mostra che Staten Island è il borough con un tasso di arresti molto inferiore rispetto agli altri 4. Per questi ultimi il punteggio calcolato si attesta tra il 21% e il 28%.

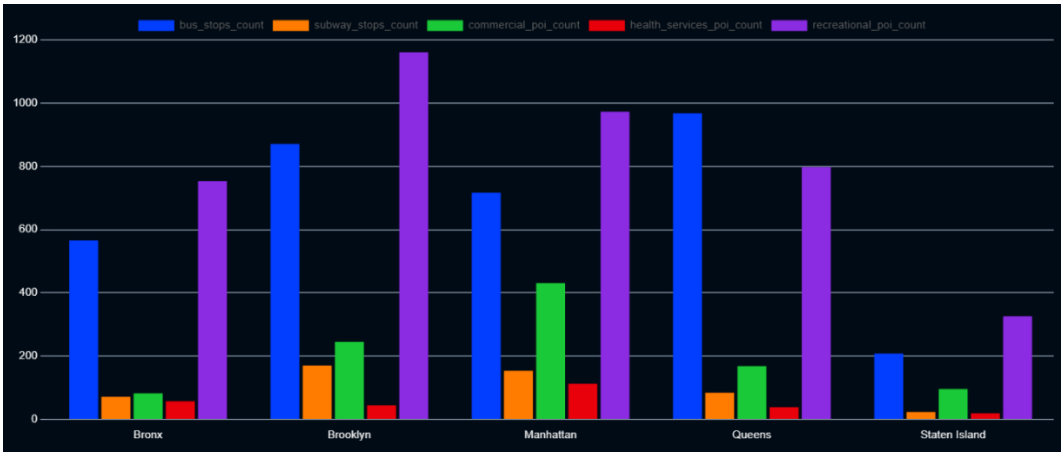
Presentazione Q4



Risultati Q4 - 1

Il grafico mostra, per ogni Borough, la numerosità dei BnB che rispettano tutte le caratteristiche del BnB ideale della fascia **Execellent** individuate nella Q2. Questo permette di classificare i Borough secondo il livello di concorrenza.

Presentazione Q5



Risultati Q5 - 1

Il grafico mostra, per ogni Borough, la numerosità delle fermate di autobus e metropolitane e POI inerenti al commercio, alla sanità e indirizzate ad attività ricreative. Questa presentazione permette di analizzare i Borough individuando quelli con le caratteristiche più utili all'apertura di una nuova attività di successo.

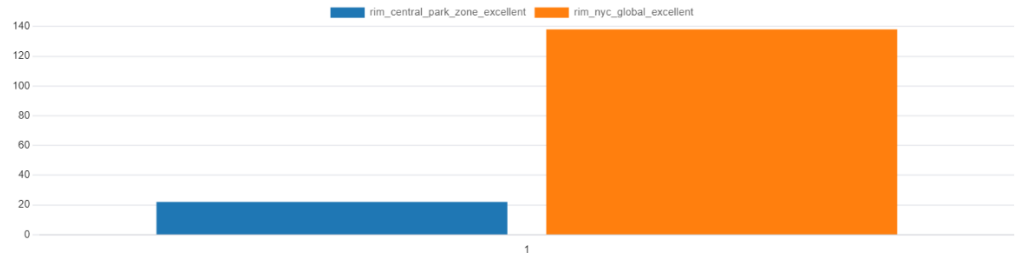
Presentazione Q6

boroughA	house_sold_low_avg	avg_sqft	recommended price ($)	avg_price_borough ($)
Bronx	90%	263 m^2	67195996	120869360
Manhattan	76%	649 m^2	246786605	710965708
Brooklyn	71%	228 m^2	88325540	157865744
Queens	61%	298 m^2	64092230	93784743
Staten Island	59%	294 m^2	51945680	70604660

Presentazione Q6 - 1

La tabella mostra, per ogni Borough, la percentuale di case vendute con un prezzo minore del prezzo di vendita medio del Borough di riferimento. Come informazione aggiuntiva è riportata anche una media dei m^2 considerando le sole case con prezzo di vendita minore.

Presentazione Q Final



Presentazione Q-Final 1

Dopo aver individuato l'area di NYC più indicata per aprire un BnB, lo stakeholder ha espresso un'ultima richiesta, ovvero analizzare qual è il tempo stimato per recuperare l'investimento iniziale (l'acquisto dell'appartamento).

Lo stakeholder a sua volta (non avendo un criterio chiaro per la scelta di tale prezzo) ha richiesto di prevedere il tempo di recupero dell'investimento valutando:

- Il prezzo medio dei bnb a partire dal centro della sotto area '103 Upper East Side-Lenox Hill-Roosevel Island (MN)', con un raggio di un chilometro (Q8 non strutturata);
- Il prezzo medio dei bnb di fascia 'Excellent' dislocati in tutta NYC (Q2 strutturata).

Fatta tale premessa è ora possibile dedurre dal grafico che il prezzo nella sotto area, essendo consistentemente maggiore, comporterà un tempo (espresso nel grafico in mesi) di recupero dell'investimento inziale sensibilmente più rapido (circa 2 anni con 1251.00 $ a notte) rispetto a quello ottenuto utilizzando il prezzo medio della fascia 'Excellent' (circa 12 anni con 199.14 $ a notte).

Commento finale

Tramite un'analisi di mercato per permettere allo stakeholder di approfondire la sua conoscenza sul nuovo territorio è apparso chiaro che la licenza non è necessaria per aprire un nuovo BnB purchè si adatti la propria attività ad affitti di minimo 30 giorni. Successivamente, analizzando le fasce di riferimento, lo stakeholder ha meglio compreso quale tipo di modello di business seguire.

Il passo successivo ha riguardato un'approfondita analisi di ogni Borough tramite la quale lo stakeholder ha constatato che nonostante Staten Island sia un luogo con un bassissimo livello di arresti e una scarsa competizione presenta allo stesso tempo il minor numero di elementi di attrattiva (Es. POI) essenziali per l'apertura di un attività di successo. Bronx invece presenta un trasso di arresti troppo alto a fronte di una scarsità di elementi di attrattiva.

Per i motivi indicati precedentemente lo stakeholder ha ritenuto di prendere in considerazione Manhattan, Brooklyn e Queens:

- Tutti e tre i borough presentano un tasso di arresto simile;
- A livello di presenza di elementi di interesse nel borough, vista anche la sua natura turistica, Manhattan risulta la scelta più indicata, seguita immediatamente di Brooklyn, nonostante sia quello **con più competizione**;

- Il prezzo delle case, oltre il 76% delle case a Manhattan è venduto ad un prezzo inferiore alla media, con un numero di metri quadri medio maggiore rispetto ai borough concorrenti;
- Il **prezzo di acquisto consigliato per le case di Manhattan** è, ovviamente, il maggiore tra tutti i borough. Nonostante ciò, essendo questo dato una media delle case che presentano prezzo inferiore alla media generale del borough, non è escluso trovare case a prezzo decisamente inferiore rispetto a quello consigliato.
- Basandoci sul prezzo delle case e vista la similarità per quanto concerne aspetti di presenza di elementi di interesse, i borough consigliati per l'apertura dell'attività dello stakeholder sono Brooklyn e Manhattan, con il secondo favorito sul primo vista la sua imponente attrattiva turistica.

Nota: i criteri per una scelta definita sul borough nel quale aprire la propria attività necessità di operazioni spaziali che permettano, in conclusione di individuare la zona geografica migliore. Tale analisi è riportata nella parte non strutturata di questo caso di studio.

Matrice delle responsabilità

OPERAZIONI	GABRIELE DETOMASO	GIACOMO DETOMASO
Definizione degli obiettivi dell'analisi	X	X
Definizione fonti dati	X	X
Descrizione esaustiva delle fonti dati	X	
Schema architettura	X	
Schema E/R		X
Modello logico	X	
ETL fonti dati: D01, D02, D09, D10	X	
ETL fonti dati: da D03 a D08		X
Definizione di vincoli di integrità referenziale		X
Query di premessa	X	X
Query di analisi: Q1 a Q3		X
Query di analisi: Q4 a Q6	X	
Query Finale	X	X
Presentazione dei risultati	X	X

Tabella 13

Sitografia

[1] probabilitycourse, «Law of Large Numbers,» 2024. [Online]. Available: https://www.probabilitycourse.com/chapter7/7_1_1_law_of_large_numbers.php.

[2] Mayor's Office of Special Enforcement, «Registration and Requirements for Short-Term Rentals,» 5 3 2023. [Online]. Available: https://rules.cityofnewyork.us/wp-content/uploads/2022/12/FINAL-RULES-GOVERNING-REGISTRATION-AND-REQUIREMENTS-FOR-SHORT-TERM-RENTALS-1.pdf. [Consultato il giorno 2 2024].

[3] Department of Finance - NYC gov, «Definitions of Property Assessment Terms,» 2024. [Online]. Available: https://www.nyc.gov/site/finance/property/definitions-of-property-assessment-terms.page.

Appendici

In questa appendice sono riportati **solo** pezzi di codice **significativo**.

Codici per ETL

DDL per file CSV

```sql
-- This table will load CSV data about bnb listings
CREATE TABLE listings (
    id BIGINT,
    name VARCHAR(255),
    host_id INT,
    host_name VARCHAR(255),
    neighbourhood_group VARCHAR(255),
    neighbourhood VARCHAR(255),
    latitude FLOAT,
    longitude FLOAT,
    bnb_type VARCHAR(255),
    price INT,
    minimum_nights INT,
    number_of_reviews INT,
    last_review DATE,
    reviews_per_month FLOAT,
    calculated_host_listings_count INT,
    availability_365 DECIMAL,
    number_of_reviews_ltm INT,
    license VARCHAR(255),
    rate DECIMAL,
    room_type VARCHAR(255),
    n_beds INTEGER,
    n_baths INTEGER,
    is_bath_shared BOOLEAN
);
```

Blocco di codice 1

Il codice riportato qui sopra indica la creazione della tabella listings, la quale è stata importata da un file .CSV.

Per tutti i file .CSV è stato utilizzato lo stesso modus operandi inquanto è stata creata una tabella con lo stesso nome del file da cui sono state successivamente estratte le colonne utili riportate all'interno dello schema ER.

DDL file csv

Questi blocchi di codice mostrano il DDL per le tabelle create dalle fonti dati CSV.

```sql
-- This table will contain every Airbnb rental unit in NYC
CREATE TABLE IF NOT EXISTS rental_units (
    id BIGINT,
    name VARCHAR(50) NOT NULL,
    availability_rate_365 DECIMAL,
    rate DECIMAL,
    number_of_reviews INTEGER,
    latitude DECIMAL,
    longitude DECIMAL,
    host INTEGER NOT NULL,
    license BOOLEAN DEFAULT FALSE,
    neighborhood INTEGER,
    coordinates geometry(Point)
);

-- This table will contain every host for the renal units
CREATE TABLE IF NOT EXISTS hosts (
    id INTEGER,
    name VARCHAR(50)
);
```

Blocco di codice 2

```sql
-- This table will contain every possible room configuration for the rental units
CREATE TABLE IF NOT EXISTS room_configurations (
    id SERIAL,
    room_type VARCHAR(20),
    n_beds INTEGER,
    n_baths INTEGER,
    n_rooms INTEGER,
    is_bath_shared BOOLEAN
);

-- This table will contain every possible rental fare for the rental units
CREATE TABLE IF NOT EXISTS rental_fares (
    id SERIAL,
    minimum_nights INTEGER NOT NULL,
    price DECIMAL NOT NULL
);

-- This table resume the listing of a rental unit
CREATE TABLE IF NOT EXISTS rental_resumes (
    id SERIAL,
    rental_unit BIGINT NOT NULL,
    room_configuration INTEGER NOT NULL,
    rental_fare INTEGER NOT NULL,
    resume_date DATE
);
```

Blocco di codice 3

```sql
-- This table will contain the arrests made in NYC
CREATE TABLE IF NOT EXISTS arrests (
    id INTEGER,
    arrest_date DATE NOT NULL,
    latitude DECIMAL,
    longitude DECIMAL,
    neighborhood INTEGER,
    coordinates geometry(Point),
    crime INTEGER NOT NULL
);

-- This table will contain NYC subway stops
CREATE TABLE IF NOT EXISTS subway_stops (
    id VARCHAR(4),
    name VARCHAR(50) NOT NULL,
    latitude DECIMAL,
    longitude DECIMAL,
    neighborhood INTEGER,
    coordinates geometry(Point)
 );
```

Blocco di codice 4

```sql
-- This table will contain the house sales in NYC during 2023
CREATE TABLE IF NOT EXISTS house_sales (
    id SERIAL,
    tax_class CHAR(2),
    sqft DECIMAL NOT NULL,
    price DECIMAL,
    construction_year INTEGER,
    address VARCHAR(100),
    latitude DECIMAL,
    longitude DECIMAL,
    neighborhood INTEGER,
    coordinates geometry(Point)
);
```

Blocco di codice 5

DDL Shapefiles

Questi blocchi di codice mostrano il DDL per le tabelle create dalle fonti dati CSV.

```sql
-- This table will contain every borough in NYC
CREATE TABLE IF NOT EXISTS boroughs (
    id CHAR(2),
    name VARCHAR(50),
    perimeter geometry(MultiPolygon, 4326)
);

-- This table will contain every neighborhoods
CREATE TABLE IF NOT EXISTS neighborhoods (
    id SERIAL,
    name VARCHAR(100),
    borough CHAR(2),
    perimeter geometry(MultiPolygon, 4326)
);

-- This tables will contain every POIs' types
CREATE TABLE IF NOT EXISTS poi_types (
    id INTEGER,
    type_desc VARCHAR(50)
);

-- This tables will contain every POI
CREATE TABLE IF NOT EXISTS poi (
    id SERIAL,
    name VARCHAR(100),
    domain VARCHAR(50),
    poi_type INTEGER,
    neighborhood INTEGER,
    coordinates geometry(Point, 4326)
);
```

Blocco di codice 6

```sql
-- This table will contain every bus stop
CREATE TABLE IF NOT EXISTS bus_stops (
    id SERIAL,
    name VARCHAR(50),
    corner VARCHAR(100),
    neighborhood INTEGER,
    coordinates geometry(Point, 4326)
);

-- This table will contain information about NYC roads
CREATE TABLE IF NOT EXISTS roads (
    id SERIAL,
    traffic_direction CHAR(2),
    name VARCHAR(50),
    status VARCHAR(20),
    borough CHAR(2),
    path geometry(LineString, 4326)
);

CREATE INDEX roads_idx
    ON roads
    USING GIST(path);
```

Blocco di codice 7

```sql
-- This table will contain every NYC park
CREATE TABLE IF NOT EXISTS parks (
    id SERIAL,
    name VARCHAR(100),
    category VARCHAR(50),
    boroughs CHAR(2)[],
    perimeter geometry(MultiPolygon, 4326)
);

CREATE INDEX parks_idx
    ON parks
    USING GIST(perimeter);

-- This table will contain the boroughs where each park belongs
CREATE TABLE IF NOT EXISTS positionings (
    borough CHAR(2),
    park INTEGER
);
```

Blocco di codice 8

Fonte dati D01

```sql
-- 1)
INSERT INTO rental_units (id, name, availability_rate_365, rate,
                          number_of_reviews, latitude, longitude,
                          host, license)
    SELECT id, name, availability_365, rate,
           number_of_reviews, latitude, longitude,
           host_id, set_licence(license)
    FROM listings;

-- 2)
INSERT INTO room_configurations(room_type, n_beds, n_baths, is_bath_shared)
    SELECT room_type, n_beds, n_baths, is_bath_shared
    FROM listings
    GROUP BY room_type, n_beds, n_baths, is_bath_shared;

-- 3)
INSERT INTO hosts (id, name)
    SELECT host_id AS id, host_name AS name
    FROM listings
    WHERE name IS NOT NULL
    GROUP BY host_id, host_name;

-- 4)
INSERT INTO rental_fares(price, minimum_nights)
    SELECT price, minimum_nights
    FROM listings
    GROUP BY price, minimum_nights;
```

Blocco di codice 9

Popolamento delle tabelle rental_units, room_configurations, rental_fares e house_sales(O02, O04, O05 e O08).

```sql
-- 5)
INSERT INTO rental_resumes(rental_unit, room_configuration, rental_fare, resume_date)
    SELECT l.id AS rental_unit, rg.id AS room_configuration,
           rf.id AS rental_fare, CURRENT_DATE AS resume_date
    FROM listings l, room_configurations rg, rental_fares rf
    WHERE l.room_type = rg.room_type AND l.n_beds = rg.n_beds AND
          l.n_baths = rg.n_baths AND l.is_bath_shared = rg.is_bath_shared AND
          l.price = rf.price AND l.minimum_nights = rf.minimum_nights;
```

Blocco di codice 10

Popolamento della della tabella rental_resumes (O06).

```sql
/*
    This function is used to dind the number of rooms
    for each room configuration
*/
CREATE OR REPLACE FUNCTION find_n_room()
RETURNS VOID AS
$$ BEGIN
    FOR i IN 1..26 LOOP
        UPDATE room_configurations
        SET n_rooms = I
        WHERE room_type LIKE Concat(CAST(i AS VARCHAR),' %');
   END LOOP;

END $$
LANGUAGE plpgsql;
```

Blocco di codice 11

```sql
-- Find the number of rooms for each room_configuration
SELECT find_n_room();

-- Set the room type to bedroom where the number of rooms is known
-- since, by default, only Studio room have a null n_rooms column
UPDATE room_configurations
SET room_type = 'Bedroom'
WHERE n_rooms IS NOT NULL;

-- Set n_rooms to 1 to Studio rooms
UPDATE room_configurations
SET n_rooms = 1
WHERE room_type LIKE 'Studio';
```

Blocco di codice 12

Definizione della funzione find_n_room() la quale estrae dalla colonna room_type il numero di stanze della configurazione utilizzata dall'unità di affitto. Assegnazione del valore standard 1 a tutte le camere di tipo 'Studio' (O07).

```sql
/*
    This function is used to set a license boolean value
    inside the rental units.
*/
CREATE OR REPLACE FUNCTION set_licence(VARCHAR(255))
RETURNS BOOLEAN AS
$$
    DECLARE is_licensed BOOLEAN;

    BEGIN
        IF $1 IS NOT NULL THEN
            SELECT TRUE INTO is_licensed;
        ELSE
            SELECT FALSE INTO is_licensed;
        END IF;

        RETURN is_licensed;
    END
$$
LANGUAGE plpgsql;
```

Blocco di codice 13

Funzione usata in fase di popolamento per settare il possesso o meno di una licenza.

```sql
-- Cast the column to decimal with 2 decimal digits
UPDATE rental_units
SET availability_rate_365 = ROUND(
    CAST(availability_rate_365 AS DECIMAL)/365, 2
);

-- Set availability_rate_365 as the avg of this column when
-- the value is zero
UPDATE rental_units
SET availability_rate_365 = (SELECT ROUND(AVG(availability_rate_365), 2)
                             FROM rental_units
                             WHERE availability_rate_365 <> 0)
WHERE availability_rate_365 = 0; -- unknown
```

Blocco di codice 14

Conversione in valore percentuale dell'attributo avalability_rate_365 e assegnazione del valor medio ove la percentuale è uguale a 0 (O02).

```regex
n_beds = df['name'].str.extract(r'([\d\.]+) bed[s]?\s*')
```

Blocco di codice 15

In questo blocco è riportata l'espressione regolare utilizzata per estrarre il numero di letti appartenenti a ciascun bnb. Tutte le altre operazioni di estrazione (numero bagni, bagni condivisi, nome bnb) sono simili, differiscono solo per l'espressione regolare utilizzata (O01).

Fonte dati D02

```
df.to_csv('../out/subway_stops.csv', index=False)
```

Blocco di codice 16

Eliminazione dell'indice dal file CSV (operazione eseguita per ogni file CSV, ma essendo ripetitiva è stata riportata solo in questo caso) (O01).

```
-- 0) Insert data into table
INSERT INTO subway_stops (id, name, latitude, longitude)
    SELECT  stop_id, stop_name, stop_lat, stop_lon
    FROM subway_stops_temp;
```

Blocco di codice 17

Popolamento della tabella subway_stops (O02).

```
/*
   Every subway stop in the imported dataset is differentiated in terms of its id
   in nord and sud, but the relative coordinates are always equals.
   These duplicates are eliminated
*/
DELETE
FROM subway_stops
WHERE id LIKE '%N' OR id LIKE '%S';
```

Blocco di codice 18

Eliminazione delle tuple che riportano la specializzazione della fermata della metropolitana (nord/sud), lasciando nel database solo l afermata con descrizione generale (O03).

Fonte dati D03

```python
def adjust_street_format(x: str):
    fap = re.split(pattern=r'\s(STREET|AVENUE)', string=x)[0] # first part of the address

    suffix_dict = {'1': 'ST', '2': 'ND', '3': 'RD'}

    street_number = re.split(pattern='\s', string=fap)[-1]

    # Obtains the correct suffix to concatenate
    th_condition = (
        # Conditions on last number of the street
        int(street_number[-1]) >= 4 or
        int(street_number[-1]) == 0 or
        # Conditions if the number end with a number between 11 and 19
        (len(street_number) >= 2 and street_number[-2] == '1')
    )

    # Selects the suffix to apply
    suffix = ('TH' if th_condition else suffix_dict[street_number[-1]])

    fap += suffix

    return fap + (' STREET' if 'STREET' in x else ' AVENUE')
```

Blocco di codice 19

Definizione della funzione adjust_street_format() per la standardizzazione degli indirizzi (O01).

```python
def geocode_address(address):
    global i

    location = geocoder.geocode(address)

    if location is not None:
        response = f'{location.latitude},{location.longitude}' # type: ignore
    else:
        response = None

    return response
```

Blocco di codice 20

Definizione della funzione di geocoding geocode_address() per individuare le coordinate geografiche dell'indirizzo dato in input (O02).

```sql
-- 0) Insert data into table
INSERT INTO house_sales(tax_class, sqft, price, construction_year, address, latitude, longitude)
    SELECT TAX_CLASS_AT_PRESENT, LAND_SQUARE_FEET, SALE_PRICE,
           YEAR_BUILT, address, latitude, longitude
    FROM house_sales_temp
    WHERE LAND_SQUARE_FEET IS NOT NULL;
```

Blocco di codice 21

Creazione della tabella house_sales popolandola solo con le tuple aventi anche i m^2 (O03).

```sql
-- Where the price is so low, it is considered as a
-- property swap (which is indicated by 0 for price)
UPDATE house_sales
SET price = 0
WHERE price = 10
```

Blocco di codice 22

Assegnazione di un valore standard per le case con prezzo di vendita <= 10 (O04).

Fonte dati D04

```python
df['ARREST_DATE'] = pd.to_datetime(df['ARREST_DATE'])
df['ARREST_DATE'] = df['ARREST_DATE'].dt.strftime('%Y/%m/%d')
```

Blocco di codice 23

Conversione dell'attributo contenente la data di arresto in modo da renderla compatibile con i tipi standard di SQL (O01).

```sql
-- Insert data into table
INSERT INTO crimes(description)
    SELECT DISTINCT OFNS_DESC
    FROM nypd_arrests;

INSERT INTO arrests (id, arrest_date, crime, latitude, longitude)
    SELECT ARREST_KEY, arrest_date, ct.id, latitude, longitude
    FROM nypd_arrests na, crimes ct
    WHERE ct.description = na.OFNS_DESC -- map the description with its id
```

Blocco di codice 24

Creazione della tabella crimes (O02) e arrests (O03)

Fonti dati: DDL shapefiles da D05 a D10

Per queste fonti dati le modalità di popolamento sono le medesime. Pertanto, è riportato lo screen dell'istruzione INSERT INTO solo per i boroughs, in quanto il modus operandi è il medesimo per le altre tabelle.

```sql
-- 0) Insert data into table
INSERT INTO boroughs (id, name, perimeter)
    SELECT boro_code, boro_name, geom
    FROM boroughs_temp;
```

Blocco di codice 25

DML shapefile: esempio operazioni di mapping

```sql
-- 4) Map borough numeric code to its literal
WITH borough_mapping AS (
    SELECT digit, literal
    FROM (VALUES ('1', 'MN'), ('2', 'BX'), ('3', 'BK'), ('4', 'QN'), ('5', 'SI'))
    AS boro_codes_literal (digit, literal)
) UPDATE boroughs n
    SET id = (
        SELECT literal
        FROM borough_mapping bcm
        WHERE n.id = bcm.digit
    );
```

Blocco di codice 26

I mapping effettuati e descritti per le varie fonti dati, hanno **tutti** il formato riportato nel blocco di codice inserito precedentemento.

Codici per queries di analisi dati

Query di premessa

```sql
-- This function is used to add a band name to a rate interval
CREATE OR REPLACE FUNCTION add_band_to_rental_unit_rate(rate DECIMAL)
RETURNS VARCHAR(50) AS
$$
    DECLARE band VARCHAR(50);
    BEGIN
        IF rate BETWEEN 3.50 AND 4.09 THEN
            SELECT 'average' INTO band;
        ELSEIF rate BETWEEN 4.10 AND 4.49 THEN
            SELECT 'good' INTO band;
        ELSEIF rate BETWEEN 4.50 AND 4.79 THEN
            SELECT 'very good' INTO band;
        ELSEIF rate BETWEEN 4.80 AND 5.00 THEN
            SELECT 'execellent' INTO band;
        END IF;

        RETURN band;
    END
$$
LANGUAGE plpgsql;
```

Blocco di codice 27

La funzione restituisce una fascia rate, in base al valore numerico di rate fornito. I band rate sono definiti con le stesse indicazioni fornite in Q2.

```sql
CREATE OR REPLACE VIEW significant_rental_units AS (
    SELECT  id, name, rate, number_of_reviews, availability_rate_365, license, neighborhood,
            add_band_to_rental_unit_rate(rate) AS rental_unit_band
    FROM    rental_units ru
    WHERE   ru.number_of_reviews > 50 AND ru.rate BETWEEN 3.5 AND 5
    ORDER BY ru.rate DESC, ru.number_of_reviews DESC
);

SELECT * FROM significant_rental_units;
```

Blocco di codice 28

La prima query seleziona tutte le unità di affitto significative, in base ai criteri forniti dallo stakeholder. Essendo questo risultato usato in numerose altre query, è stata definita una **vista**.

Q1

```sql
-- This function count the number of compliant bnb according to the input parameter
CREATE OR REPLACE FUNCTION count_compliances(is_compliant BOOLEAN, is_global_ru BOOLEAN)
RETURNS INTEGER AS
$$
    DECLARE comp_count INTEGER;
    DECLARE COMPLIANT_LIMIT INTEGER := 30;

    BEGIN
        IF is_global_ru = TRUE THEN
            CREATE TEMPORARY TABLE no_licenses_units AS (
                SELECT ru.id, rf.minimum_nights
                FROM rental_units ru, actual_resumes ar, rental_fares rf
                WHERE ru.id = ar.rental_unit AND rf.id = ar.rental_fare AND ru.license = false
            );
        ELSE
            CREATE TEMPORARY TABLE no_licenses_units AS (
                SELECT ru.id, rf.minimum_nights
                FROM significant_rental_units ru, actual_resumes ar, rental_fares rf
                WHERE ru.id = ar.rental_unit AND rf.id = ar.rental_fare AND ru.license = false
            );
        END IF;
```

Blocco di codice 29

```sql
        IF is_compliant THEN
            SELECT count(*) INTO comp_count
            FROM no_licenses_units nlu
            WHERE nlu.minimum_nights >= COMPLIANT_LIMIT;
        ELSE
            SELECT count(*) INTO comp_count
            FROM no_licenses_units nlu
            WHERE nlu.minimum_nights < COMPLIANT_LIMIT;
        END IF;

        DROP TABLE no_licenses_units;

        RETURN comp_count;
    END
$$
LANGUAGE plpgsql;
```

Blocco di codice 30

Questa funzione (inserita in due screen) è usata per contare il:

- Numero rental units che rispettano nuovi regolamenti senza non aventi licenza

- Numero rental units che non rispettano i nuovi regolamenti
- Numero rental units che rispettano i nuovi regolamenti aventi licenza

Il parametro **is_compliant** indica se il conteggio va effettuato su bnb che rispettano il regolamento o meno, mentre il parametro **is_global_ru** indica se effettuare il conteggio su TUTTE le rental units oppure su quelle significative presenti nella vista logica **significant_rental_units**.

```
-- Globally
SELECT count_compliances(TRUE, TRUE)
            AS compliant_bnb_without_licence,
       count_compliances(FALSE, TRUE)
            AS not_compliant_bnb_without_licence,
       (SELECT COUNT(*) FROM rental_units) - (count_compliances(TRUE, TRUE) + count_compliances(FALSE, FALSE))
            AS compliant_with_license;

-- Only SRU
SELECT count_compliances(TRUE, FALSE)
            AS compliant_bnb_without_licence,
       count_compliances(FALSE, FALSE)
            AS not_compliant_bnb_without_licence,
       (SELECT COUNT(*) FROM significant_rental_units) - (count_compliances(TRUE, FALSE) + count_compliances(FALSE, FALSE))
            AS compliant_with_license;
```

Query 1

Le query proposte mostrano sia i risultati su tutte le rental units che su quelle significative.

Q2

```
CREATE OR REPLACE FUNCTION count_by_rate_bands(bnb_band VARCHAR(50))
RETURNS INTEGER AS
$$
    DECLARE n INTEGER;

    BEGIN
        IF bnb_band LIKE 'average' THEN
            SELECT COUNT(*) INTO n FROM temp_mode tm WHERE tm.rate BETWEEN 3.50 AND 4.09;
        ELSEIF bnb_band LIKE 'good' THEN
            SELECT COUNT(*) INTO n FROM temp_mode tm WHERE tm.rate BETWEEN 4.10 AND 4.49;
        ELSEIF bnb_band LIKE 'very good' THEN
            SELECT COUNT(*) INTO n FROM temp_mode tm WHERE tm.rate BETWEEN 4.50 AND 4.79;
        ELSEIF bnb_band LIKE 'execellent' THEN
            SELECT COUNT(*) INTO n FROM temp_mode tm WHERE tm.rate BETWEEN 4.80 AND 5.00;
        END IF;

        DROP TABLE temp_mode;

        RETURN n;
    END
$$
LANGUAGE plpgsql;
```

Blocco di codice 31

La funzione lavora su una bnb band (fascia di rate) come input, e su una tabella temporanea **speciale** definita nelle funzioni mostrate successivamente. Restituisce un conteggio delle righe di temp_mode.

```
CREATE OR REPLACE FUNCTION count_bath_shared(bath_shared BOOLEAN, bnb_band VARCHAR(50))
RETURNS INTEGER AS
$$
    DECLARE n INTEGER;

    BEGIN
        CREATE TEMPORARY TABLE IF NOT EXISTS temp_mode AS (
            SELECT rr.id, sru.rate
            FROM room_configurations rc, rental_resumes rr, significant_rental_units sru
            WHERE rc.id = rr.room_configuration AND sru.id = rr.rental_unit AND
                  rc.is_bath_shared = bath_shared
        );

        RETURN count_by_rate_bands(bnb_band);
    END
$$
LANGUAGE plpgsql;
```

Blocco di codice 32

Le funzioni che iniziano per **count** creano questa tabella speciale in base ai parametri di input e restituiscono il conteggio adoperando la funzione definita nel blocco di codice 22. Questo conteggio è usato per definire la moda di queste colonne rappresentanti dati categorici.

Tali funzioni sono state definite per i bagni, tipi di camere e licenze e hanno la medesima struttura.

```
-- This result will be reused in other queries
CREATE OR REPLACE VIEW ideal_rental_unit_per_band_rate AS (
    SELECT  rental_unit_band,
            ROUND(AVG(rf.price), 2) AS avg_price,
            ROUND(AVG(sru.availability_rate_365), 2) AS avg_availability_rate_365,
            (count_licensed_ru(TRUE, rental_unit_band) >=
             count_licensed_ru(FALSE, rental_unit_band)) AS mode_license,
            CAST(AVG(n_beds) AS INTEGER) AS avg_number_of_beds,
            CAST(AVG(n_baths) AS INTEGER) AS avg_number_of_baths,
            (count_bath_shared(TRUE, rental_unit_band) >=
             count_bath_shared(FALSE, rental_unit_band)) AS mode_bath_shared,
            CASE
            -- First case of mode room type
            WHEN (count_room_type('Bedroom', rental_unit_band) >=
                  count_room_type('Studio', rental_unit_band)) IS TRUE THEN 'Bedroom'
            -- Second case of mode room type
            WHEN (count_room_type('Bedroom', rental_unit_band) >=
                  count_room_type('Studio', rental_unit_band)) IS FALSE THEN 'Studio'
            END AS mode_room_type
    FROM    significant_rental_units sru, rental_fares rf, room_configurations rc, rental_resumes rr
    WHERE   sru.id = rr.rental_unit AND rf.id = rr.rental_fare AND rc.id = rr.room_configuration
    GROUP BY rental_unit_band
);

SELECT *
FROM ideal_rental_unit_per_band_rate
ORDER BY avg_price DESC;
```

Query 2

La grandezza di questa query è dovuta alla presenza di numerosi attributi nelle select. Nella SELECT sono state effettuate numerose aggregazioni sulle informazioni richieste dallo stakeholder, raggruppando per fascia di rate (**rental_unit_band**). L'uso delle funzioni, definite in precedenza, è usato per valutare la moda delle features categoriche.

Q3

```
/*
    Q4
    List all the borough based on the crime rate from the lower to the higher.
*/
WITH total_arrests AS (
    SELECT COUNT(*) AS tot
    FROM arrests
)   SELECT b.name AS borough,
           ROUND(CAST(COUNT(*) AS DECIMAL) / CAST((SELECT * FROM total_arrests) AS DECIMAL), 2) AS arrests_rate
    FROM neighborhoods n, arrests a, boroughs b
    WHERE n.id = a.neighborhood AND b.id = n.borough
    GROUP BY b.name
    ORDER BY arrests_rate ASC;
```

Query 3

Questa query permette di individuare il tasso di criminalità per ogni borough di NYC ordinando il tutto in base alla percentuale, dalla più bassa alla più alta.

Q4

```
/*
    Q5
    Count the bnb number in a specific borough with ideal charatteristics.
*/

SELECT  b.name AS borough, COUNT(*) AS number_bnb
FROM    ideal_rental_unit_per_band_rate iru, rental_resumes rs, rental_fares rf, rental_units ru,
        room_configurations rc, neighborhoods n, boroughs b
WHERE   rf.id = rs.rental_fare AND rc.id = rs.room_configuration AND ru.id = rs.rental_unit AND
        rf.price BETWEEN iru.avg_price-10 AND iru.avg_price+10 AND
        rc.n_beds = iru.avg_number_of_beds AND rc.n_baths = iru.avg_number_of_baths AND rc.is_bath_shared = iru.mode_bath_shared AND
        ru.availability_rate_365 BETWEEN iru.avg_availability_rate_365-0.5 AND iru.avg_availability_rate_365+0.5 AND
        ru.neighborhood = n.id AND n.borough = b.id AND iru.rental_unit_band = 'execellent'
GROUP BY b.name;
```

Query 4

La Q4 permette di contare per ogni borough di NYC, il numero di unità di affitto che rispettano tutte le caratteristiche del bnb ideale della fascia EXECELLENT individuate nella Q3.

Q5

```
CREATE OR REPLACE FUNCTION count_poi_types(poi_type INTEGER, borough_name VARCHAR(255))
RETURNS INTEGER AS
$$
    SELECT COUNT(*) AS tot
    FROM poi, neighborhoods n, boroughs b
    WHERE n.id = poi.neighborhood AND b.id = n.borough AND
          b.name = $2 AND poi.poi_type = $1;
$$
LANGUAGE SQL;

CREATE TEMPORARY TABLE IF NOT EXISTS count_bus_stops AS (
    SELECT b.name AS borough, COUNT(*) bus_stops_count
    FROM bus_stops bs, neighborhoods n, boroughs b
    WHERE n.id = bs.neighborhood AND b.id = n.borough
    GROUP BY b.name
);

CREATE TEMPORARY TABLE IF NOT EXISTS count_subway_stops AS (
    SELECT b.name AS borough, COUNT(*) subway_stops_count
    FROM subway_stops ss, neighborhoods n, boroughs b
    WHERE n.id = ss.neighborhood AND b.id = n.borough
    GROUP BY b.name
);

CREATE TEMPORARY TABLE IF NOT EXISTS count_poi AS (
    SELECT b.name AS borough,
           count_poi_types(7, b.name) AS commercial_poi_count,
           count_poi_types(10, b.name) AS health_services_poi_count,
           count_poi_types(4, b.name) AS recreational_poi_count
    FROM boroughs b
);
```

Blocco di codice 33

La funzione presente all'inizio del blocco di codice permette di calcolare il numero di poi, di una particolare tipologia presenti in un borough, di cui è specificato il nome.

Le tabelle temporanee servono per calcolare il numero di fermate di bus e quelle di metro, per ogni borough.

```
SELECT cp.borough, bus_stops_count, subway_stops_count,
       commercial_poi_count, health_services_poi_count, recreational_poi_count
FROM count_poi cp, count_subway_stops css, count_bus_stops cbs
WHERE cp.borough = css.borough AND cp.borough = cbs.borough;
```

Query 5

I risultati di questa funzione e delle tabelle temporanee sono "uniti" nella query finale attraverso la quale i conteggi sono visualizzati in un'unica "vista".

Q6

```
/*
    Q6
    For all borough find the houses percentage that were sold with a price lower than borough average price
*/
WITH avg_price_per_borough AS(
    SELECT b.name AS borough, ROUND(AVG(hs.price), 2) AS avg_price, COUNT(*) AS tot_house
    FROM house_sales hs, neighborhoods n, boroughs b
    WHERE hs.neighborhood = n.id AND n.borough = b.id AND
        (hs.tax_class LIKE '1%' OR hs.tax_class LIKE '2%') AND hs.price <> 0
    GROUP BY b.name
)
SELECT  b.name AS borough,
        (ROUND(CAST(COUNT(*) AS DECIMAL) /
               (SELECT tot_house FROM avg_price_per_borough WHERE borough = b.name), 2)) AS house_sold_low_avg,
        CEIL(CAST(AVG(hs.sqft) * 0.092903 AS DECIMAL)) AS avg_sqft,
        ROUND(AVG(hs.price), 2) AS avg_price,
        (SELECT avg_price FROM avg_price_per_borough WHERE borough = b.name) AS avg_price_borough
FROM    house_sales hs, neighborhoods n, boroughs b
WHERE   hs.neighborhood = n.id AND n.borough = b.id AND
        (hs.tax_class LIKE '1%' OR hs.tax_class LIKE '2%') AND hs.price <> 0 AND
        hs.price < (SELECT avg_price FROM avg_price_per_borough apb WHERE apb.borough = b.name)
GROUP BY b.name
ORDER BY house_sold_low_avg DESC;
```

Query 6

La Q7 usa una CTE per individuare il prezzo medio e il numero totale delle case per ogni borough. Nella SELECT vengono individuate le case vendute con prezzo di vendita inferiore al prezzo medio per ogni borough calcolato nella clausola WITH.

Per tale scopo sono selezionati il nome del borough, il **tasso di vendita** inferiore al prezzo medio, e il numero medio di metri quadri di queste case.

Da notare che solo le categorie di tax class che iniziano per 1 o 2 sono state selezionate, in quanto le categorie 3 e 4 non rappresentano delle abitazioni.

QFinal

```
-- RIM = Recover Invevestment Months
SELECT  CONCAT(s.avg_price_ex_band, '$') AS avg_price_central_park_zone_excellent,
        ROUND(s.avg_price_house_sales / (s.avg_price_ex_band * 365 * 0.70), 2) * 12 AS RIM_CENTRAL_PARK_ZONE_EXCELLENT,
        CONCAT(ir.avg_price, '$') AS avg_price_nyc_global_excellent,
        ROUND(s.avg_price_house_sales / (ir.avg_price * 365 * 0.70), 2) *12 AS RIM_NYC_GLOBAL_EXCELLENT
FROM    analyze_sub_neighborhood(103) s,
        (SELECT avg_price
         FROM   ideal_rental_unit_per_band_rate ir
         WHERE  rental_unit_band = 'excellent') AS ir;
```

Query 7

Al prospetto finale circa il recupero dall'investimento inziale per l'acquisto di una proprietà nella sotto zona 103 del neighborhood selezionato, sono stati aggiunti anche i prezzi medi in $ di B&B da seguire per le due diverse strategie di recupero.

Esempio di definizione dei constraints

```sql
-- Constraints for "boroughs" table
ALTER TABLE boroughs
    ADD CONSTRAINT pk_borough PRIMARY KEY(id);

-- Constraints for "neighborhoods" table
ALTER TABLE neighborhoods
    ADD CONSTRAINT pk_neighborhoods PRIMARY KEY(id),
    ADD CONSTRAINT fk_neighborhoods_borough FOREIGN KEY(borough) REFERENCES boroughs(id)
        ON DELETE CASCADE ON UPDATE CASCADE;

-- Constraints for "crimes" table
ALTER TABLE crimes
    ADD CONSTRAINT pk_crimes PRIMARY KEY(id);

-- Constraints for "arrests" table
ALTER TABLE arrests
    ADD CONSTRAINT pk_arrests PRIMARY KEY(id),
    ADD CONSTRAINT fk_arrest_crime FOREIGN KEY(crime) REFERENCES crimes(id)
        ON DELETE CASCADE ON UPDATE CASCADE,
    ADD CONSTRAINT fk_crimes_neighborhood FOREIGN KEY(neighborhood) REFERENCES neighborhoods(id)
        ON DELETE CASCADE ON UPDATE CASCADE;

-- Constraints for "hosts" table
ALTER TABLE hosts
    ADD CONSTRAINT pk_hosts PRIMARY KEY(id);
```

Blocco di codice 34