

**UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO**

**DIPARTIMENTO DI INFORMATICA**

**CORSO DI LAUREA MAGISTRALE**

**IN**

**DATA SCIENCE**

**Caso di studio di**

**Gestione di dati strutturati e non strutturati**

**ANALISI DELL’INQUINAMENTO IN PUGLIA**

**Docenti: Studente:**

**Mario Alessandro Bochicchio Marco Casamassima 730346**

**Corrado Loglisci**

**Anno Accademico 2022-2023**

Sommario

[Obiettivi 2](#_Toc134891004)

[Fonti dati e diagramma architettura 2](#_Toc134891005)

[ETL 2](#_Toc134891006)

[Modellazione 2](#_Toc134891007)

[Diagramma ER 2](#_Toc134891008)

[Diagramma Relazionale 2](#_Toc134891009)

[Stima della dimensione database 3](#_Toc134891010)

[Queries 3](#_Toc134891011)

[Presentazione dei risultati 3](#_Toc134891012)

[Matrice delle responsabilità 3](#_Toc134891013)

[Bibliografia-Sitografia 3](#_Toc134891014)

# Obiettivi

L'inquinamento atmosferico, causato dalle attività umane, rappresenta un fattore di rischio ampiamente riconosciuto per la salute umana e per gli ecosistemi. Negli Stati occidentali, la lotta contro l'inquinamento atmosferico ha una storia di oltre sessant'anni di studi e ricerche volti a comprendere i meccanismi degli effetti dannosi degli inquinanti, nonché a individuare strategie e tecniche di mitigazione. Le reti di monitoraggio costituiscono il principale strumento per la valutazione della qualità dell'aria, intesa come il complesso di attività mirate a verificare il rispetto dei valori limite e l'ottenimento degli obiettivi stabiliti al fine di prevenire, eliminare o ridurre gli effetti negativi dell'inquinamento atmosferico sulla salute umana e sull'ecosistema in un determinato territorio statale [1].

Secondo uno studio dell’Organizzazione Mondiale della Sanità (OMS) ogni anno muoiono circa 7 milioni di persone a causa dell’inquinamento ambientale [2] e circa 9 persone su 10 al mondo respirano aria con elevati livelli di inquinanti [3] che causano malattie respiratorie e polmonari, leucemia, problemi cardiovascolari, malattie cardiache, ictus, cancro ai polmoni, ecc.

Un indicatore che può essere utilizzato per valutare la qualità dell’aria è la quantità del particolato atmosferico, ossia una miscela complessa di particelle solide e liquide di sostanze organiche ed inorganiche sospese in aria come, ad esempio, alcuni metalli tossici (piombo, cadmio e nichel) che hanno un’elevata tossicità [4]. In generale quanto più piccola è la dimensione delle particelle tanto maggiore è la loro capacità di penetrare nei polmoni e dunque di produrre effetti dannosi sulla salute umana. Il particolato è suddiviso in base al diametro aerodinamico:

* Particelle PM10: con diametro aerodinamico inferiore a 10 µm, penetrano nel tratto superiore delle vie aeree (cavità nasali, faringe e laringe);
* Particelle PM2.5: con diametro aerodinamico inferiore a 2.5 µm, possono giungere fino alle parti inferiori dell’apparato respiratorio.

Le sorgenti di queste particelle possono essere di tipo naturale (erosione del suolo, vulcani, incendi boschivi, dispersione di pollini, etc.) o antropogenico (industrie, riscaldamento, traffico veicolare e processi di combustione in generale) [5].

Secondo l’OMS, per il particolato non è possibile definire un valore limite al di sotto del quale non si verificano nella popolazione effetti sulla salute: per questo motivo le concentrazioni di PM10 e PM2.5 nell'aria dovrebbero essere mantenute a livelli più bassi possibile [6]. Tuttavia, le linee guida dell'OMS sulla qualità dell'aria, aggiornate al 2019, stabiliscono delle soglie di concentrazioni di particolato atmosferico consigliate da raggiungere. In particolare:

* Per il PM2.5: le concentrazioni medie annuali non devono superare i 10 µg/m3, mentre l’esposizione media giornaliera non deve superare i 25 µg/m3 per più di 3 - 4 giorni all'anno.
* Per il PM10: le concentrazioni medie annuali non devono superare i 20 µg/m3, mentre l’esposizione media giornaliera non deve superare i 50 µg/m3 per più di 3 - 4 giorni all'anno.

Come precedentemente menzionato, il traffico veicolare rappresenta una delle fonti di particolato PM2.5 e PM10, il quale assume un carattere sempre più urgente e grave, soprattutto nelle grandi città dove l'uso predominante dell'automobile privata influisce sulla mobilità delle persone in modo eccessivo. Pertanto, risulta essenziale individuare soluzioni atte a ridurre il numero di veicoli circolanti nelle aree urbane, promuovendo parallelamente modalità di trasporto più sostenibili, al fine di mitigare le emissioni di gas serra e di sostanze dannose per l'ambiente. Un'opzione di particolare interesse potrebbe essere rappresentata dalla condivisione di veicoli (automobili elettriche, bici o monopattini), il quale si configura come un complemento al trasporto, contribuendo così alla diminuzione del numero di veicoli privati in circolazione e dei chilometri percorsi. Uno studio condotto dall'Università di Palermo, condotto da Marco Migliore et al. [7], sull'impatto ambientale del car sharing a Palermo, mette in luce come questa forma di condivisione possa costituire una soluzione per ridurre l'inquinamento ambientale nelle aree urbane.

Lo scopo del presente elaborato consiste nel creare un database contenente dati raccolti dalle stazioni di monitoraggio riguardanti le concentrazioni di particolato PM2.5 e PM10 nel territorio della regione Puglia, nonché il numero di decessi potenzialmente attribuibili all'inquinamento ambientale e un’analisi mirata a identificare le città urbane maggiormente colpite da tale problematica prendendo in considerazione lo studio effettuato da Alessia Longo e Eleonora Miccoli [8] dal titolo *Studio e analisi della correlazione tra inquinamento ambientale e mortalità nella popolazione italiana*.

I principali stakeholder individuati per questo progetto risultano essere gli analisti delle imprese di condivisione di veicoli, i quali potranno avvalersi di tale database per determinare le aree pugliesi in cui operare.

# Fonti dati e diagramma architettura

Il database utilizzato in questo studio è stato creato a partire da dataset provenienti da diverse fonti di dati, tra cui:

* il numero di morti nelle diverse province pugliesi, estratto dal sito dell'Istituto Nazionale di Statistica (ISTAT);
* la popolazione delle varie città pugliesi, estratta dal sito ISTAT;
* i valori di PM10 e PM2.5 registrati dalle diverse stazioni di rilevamento dell'inquinamento sul territorio pugliese, provenienti dal sito dell’Agenzia Regionale per la Prevenzione e la Protezione dell’Ambiente (ARPA).

Immagine che contiene diagramma, testo, linea, schermata

Descrizione generata automaticamente

# ETL

In questo paragrafo si descrive la pipeline ETL (estrazione, trasformazione e caricamento) effettuata[[1]](#footnote-1) per il popolamento del database:

## Estrazione

Come menzionato nel paragrafo precedente, i dati utilizzati sono stati estratti dai siti Arpa Puglia e Istat. In particolare:

* I dati relativi al numero di morti nelle diverse province pugliesi sono stati estratti dal seguente link <http://dati.istat.it/Index.aspx?DataSetCode=DCIS_CMORTE1_RES> in formato csv. Le query utilizzate per l’estrazione sono:
  + Territorio: sono state selezionate le caselle relative alle province pugliesi;
  + Tipo dato: è stata selezionata la casella “morti”;
  + Sesso: sono state selezionate le caselle “maschi” e “femmine”;
  + Causa Iniziale di morte – European Short List[[2]](#footnote-2): sono state selezionate le caselle “tumori”, “malattie endocrine, nutrizionali e metaboliche”, “malattie del sistema nervoso e organi di senso”, “malattie del sistema circolatorio”, “malattie del sistema respiratorio” e “malattie dell’apparato digerente”;
  + Seleziona periodo: sono stati selezionati gli anni 2019 e 2020;
  + Per le dimensioni Pivot:
    - Filtra dimensioni: è stato inserito “Tipo dato”;
    - Dimensione verticale: è stato inserito “Territorio”;
    - Dimensione orizzontale: sono stati inseriti “Causa iniziale di morte - European Short List”, “Seleziona periodo” e “Sesso”.

La query è stata salvata nel seguente link [http://dati.istat.it//Index.aspx?QueryId=60835](http://dati.istat.it/Index.aspx?QueryId=60835). Nella Figura 1 è riportato il risultato della query.

Immagine che contiene testo, schermata, numero, software

Descrizione generata automaticamente

Figura Risultato della query sul dataset delle morti

* I dati relativi alla popolazione risedente nei diversi comuni pugliesi sono stati estratti dal seguente link [http://dati.istat.it/Index.aspx?DataSetCode=DCIS\_POPRES1#](http://dati.istat.it/Index.aspx?DataSetCode=DCIS_POPRES1) in formato csv. Le query utilizzate per l’estrazione sono:
  + Territorio: sono state selezionate le caselle relative ai comuni pugliesi;
  + Tipo di indicatore demografico: è stata selezionata la casella “popolazione inizio periodo”;
  + Sesso: sono state selezionate le caselle “maschi” e “femmine”;
  + Seleziona periodo: sono stati selezionati gli anni 2019 e 2020;
  + Per le dimensioni Pivot:
    - Filtra dimensioni: è stato inserito “Tipo di indicatore demografico”;
    - Dimensione verticale: sono stati inseriti “Territorio” e;
    - Dimensione orizzontale: sono stati inseriti “Seleziona periodo” e “Sesso”.

La query è stata salvata nel seguente link <http://dati.istat.it//Index.aspx?QueryId=60836>. Nella Figura 2 è riportato il risultato della query.

* I dati relativi ai confini geografici dei comuni e delle province sono stati estratti dal seguente link <https://www.istat.it/it/archivio/222527>. Sono stati considerati i dati relativi al 2019.

Immagine che contiene testo, schermata, numero, software

Descrizione generata automaticamente

Figura Risultato della query sul dataset della popolazione

* I dati relativi alle registrazioni effettuate dalle stazioni di rilevazione di PM2.5 e PM10 in Puglia sono stati estratti dal seguente link <http://old.arpa.puglia.it/web/guest/meta-aria> in formato csv. Dopo aver cliccato su “Dati per centralina”, è stata utilizzata la seguente query per l’estrazione:
  + Seleziona la provincia: è stato selezionato “Tutto”;
  + Seleziona l’inquinante: è stato selezionato “Tutto”;
  + Report Format: è stato selezionato “csv”;
  + Seleziona la data o un intervallo di date: sono stati selezionati i giorni dal 01-01-2019 al 31-12-2020.

## Trasformazione

La fase di trasformazione è stata eseguita con i seguenti file[[3]](#footnote-3):

* Comuni.ipynb per i comuni;
* Decessi.ipynb per i decessi;
* Indicatori.ipynb per gli indicatori;
* Popolazioni.ipynb per le popolazioni;
* Province.ipynb per le province;
* Stazioni, Rilevazioni, Rileva.ipynb per le stazioni, rilevazioni e rileva.

Di seguito sono riportate le descrizioni dei file.

### Comuni

Importazione delle librerie pandas e geopandas per la gestione dei dataframe

import pandas as pd  
import geopandas as gpd

Configurazione dei path dei dataset regioni e comuni

path\_regioni = f'../dataset/dati\_spaziali/regioni/regioni.shp'  
path\_comuni = f'../dataset/dati\_spaziali/comuni/comuni.shp'

Lettura dei dataset

comuni = gpd.read\_file(path\_comuni)  
regioni = gpd.read\_file(path\_regioni)

Estrazione dei comuni pugliesi utilizzando il codice della regione Puglia estratto dal dataset delle regioni

codice\_puglia = regioni[regioni['DEN\_REG'] == 'Puglia']['COD\_REG'].values[0]  
comuni\_puglia = comuni[comuni['COD\_REG'] == codice\_puglia]  
comuni\_puglia.reset\_index(inplace=True, drop=True)

Ridenominazione delle colonne del dataframe relativo ai comuni pugliesi

comuni\_puglia = comuni\_puglia[['COD\_PROV', 'PRO\_COM', 'COMUNE', 'geometry']]  
comuni\_puglia.rename(columns={'COMUNE': 'nome',  
 'COD\_PROV': 'id\_prov',  
 'PRO\_COM': 'id\_comune'},   
 inplace=True)

Esportazione del dataset trasformato

comuni\_puglia.to\_file('../dataset/dati\_spaziali/comuni/comuni\_finale.shp', index=None)

### Decessi

Importazione delle librerie pandas e geopandas per la gestione dei dataframe

import pandas as pd  
import geopandas as gpd

Configurazione dei path dei dataset decessi e province

path\_decessi = '../dataset/decessi/decessi\_malattie\_2019\_2020.csv'  
path\_province = f'../dataset/dati\_spaziali/province/province.shp'

Lettura dei dataset

province = gpd.read\_file(path\_province)  
decessi = pd.read\_csv(path\_decessi)

Si considerano solo le colonne ‘Territorio’, ‘Sesso’, ‘Causa iniziale di morte – European Short List’, ‘TIME’ e ‘Value’ del dataset decessi

columns = [‘Territorio’, ‘Sesso’, ‘Causa iniziale di morte – European Short List’, ‘TIME’, ‘Value’]  
decessi = decessi[columns]

Creazione del dataframe finale

decessi\_finale = pd.DataFrame(columns = ['nome\_malattia', 'numero\_decessi\_maschi', 'numero\_decessi\_femmine', 'anno', 'id\_provincia'])

Il dataset iniziale contiene gli attributi Sesso e Value. Di seguito si fondono queste due colonne in numero\_decessi\_maschi e numero\_decessi\_femmine.

for p in set(decessi['Territorio']):  
 decessi\_territorio = decessi[decessi['Territorio'] == p]  
 for c in set(decessi\_territorio['Causa iniziale di morte - European Short List']):  
 decessi\_territorio\_malattia = decessi\_territorio[decessi\_territorio['Causa iniziale di morte - European Short List'] == c]  
 anni = set(decessi['TIME'])  
 for a in anni:  
 decessi\_territorio\_malattia\_maschi = decessi\_territorio\_malattia[decessi\_territorio\_malattia['Sesso'] == 'maschi']  
 decessi\_territorio\_malattia\_maschi\_anno = decessi\_territorio\_malattia\_maschi[decessi\_territorio\_malattia\_maschi['TIME'] == a]['Value'].values[0]  
   
 decessi\_territorio\_malattia\_femmine = decessi\_territorio\_malattia[decessi\_territorio\_malattia['Sesso'] == 'femmine']  
 decessi\_territorio\_malattia\_femmine\_anno = decessi\_territorio\_malattia\_femmine[decessi\_territorio\_malattia\_femmine['TIME'] == a]['Value'].values[0]  
 id\_provincia = province[province['DEN\_UTS'] == p]['COD\_PROV'].values[0]  
   
 row = [c, decessi\_territorio\_malattia\_maschi\_anno, decessi\_territorio\_malattia\_femmine\_anno, a, id\_provincia]  
 decessi\_finale.loc[len(decessi\_finale)] = row

Esportazione del dataset trasformato

decessi\_finale.to\_csv('..\dataset\decessi\decessi\_finale.csv', index=False)

### Indicatori

Importazione delle librerie pandas per la gestione dei dataframe

import pandas as pd

Creazione del dataframe finale contenente i dati

indicatori = {  
 'sigla': ['PM2.5', 'PM10'],  
 'nome': ['Particolato PM2.5', 'Particolato PM10'],  
 'soglia\_media\_giornaliera': [10, 20],  
 'giorni\_superamento': [4, 4],  
 'soglia\_media\_annuale': [25, 50]  
}  
indicatori\_df = pd.DataFrame(indicatori)

Esportazione del dataset

indicatori\_df.to\_csv('..\dataset\indicatori\indicatori\_finale.csv', index=False)

### Popolazioni

Importazione delle librerie pandas per la gestione dei dataframe

import pandas as pd

Configurazione dei path dei dataset decessi e province

path\_popolazioni = f'../dataset/popolazioni/popolazioni.csv'

Lettura del dataset

popolazioni = pd.read\_csv(path\_popolazioni)

Si considerano solo le colonne ‘ITTER107’, ‘Sesso’, ‘TIME’, ‘Value’ del dataset popolazioni

popolazioni = popolazioni[[‘ITTER107’, ‘Sesso’, ‘TIME’, ‘Value’]]

Creazione del dataframe con colonne 'id\_comune', 'anno', 'numero\_maschi', 'numero\_femmine'

popolazioni\_finale = pd.DataFrame(columns = ['id\_comune', 'anno', 'numero\_maschi', 'numero\_femmine'])

Il dataset iniziale contiene gli attributi Sesso e Value. Di seguito si fondono queste due colonne in numero\_maschi e numero\_femmine

for p in set(decessi['Territorio']):  
 decessi\_territorio = decessi[decessi['Territorio'] == p]  
 for c in set(decessi\_territorio['Causa iniziale di morte - European Short List']):  
 decessi\_territorio\_malattia = decessi\_territorio[decessi\_territorio['Causa iniziale di morte - European Short List'] == c]  
 anni = set(decessi['TIME'])  
 for a in anni:  
 decessi\_territorio\_malattia\_maschi = decessi\_territorio\_malattia[decessi\_territorio\_malattia['Sesso'] == 'maschi']  
 decessi\_territorio\_malattia\_maschi\_anno = decessi\_territorio\_malattia\_maschi[decessi\_territorio\_malattia\_maschi['TIME'] == a]['Value'].values[0]  
   
 decessi\_territorio\_malattia\_femmine = decessi\_territorio\_malattia[decessi\_territorio\_malattia['Sesso'] == 'femmine']  
 decessi\_territorio\_malattia\_femmine\_anno = decessi\_territorio\_malattia\_femmine[decessi\_territorio\_malattia\_femmine['TIME'] == a]['Value'].values[0]  
 id\_provincia = province[province['DEN\_UTS'] == p]['COD\_PROV'].values[0]  
   
 row = [c, decessi\_territorio\_malattia\_maschi\_anno, decessi\_territorio\_malattia\_femmine\_anno, a, id\_provincia]  
 decessi\_finale.loc[len(decessi\_finale)] = row

Esportazione del dataset trasformato

decessi\_finale.to\_csv('..\dataset\decessi\decessi\_finale.csv', index=False)

### Province

Importazione delle librerie pandas e geopandas per la gestione dei dataframe

import pandas as pd  
import geopandas as gpd

Configurazione dei path dei dataset regioni, province e comuni

path\_path\_regioni = f'../dataset/dati\_spaziali/regioni/regioni.shp'  
path\_province = f'../dataset/dati\_spaziali/province/province.shp'

Lettura dei dataset

regioni = gpd.read\_file(path\_regioni)  
province = gpd.read\_file(path\_province)

Estrazione delle province pugliesi

codice\_puglia = regioni[regioni['DEN\_REG'] == 'Puglia']['COD\_REG'].values[0]  
province\_puglia = province[province['COD\_REG'] == codice\_puglia]  
province\_puglia.reset\_index(inplace=True, drop=True)

Rinominazione delle colonne del dataframe

province\_puglia = province\_puglia[['DEN\_UTS', 'COD\_PROV', 'geometry']]  
province\_puglia.rename(columns={'DEN\_UTS': 'nome',   
 'COD\_PROV': 'id\_prov'},   
 inplace=True)

Esportazione del dataset trasformato

province\_puglia.to\_file('../dataset/dati\_spaziali/province/province\_finale.shp', index=None)

### Stazioni, rilevazioni, rileva

Importazione delle librerie pandas e geopandas per la gestione dei dataframe

import pandas as pd  
import geopandas as gpd

Configurazione dei path dei dataset comuni e quelli relativi alle rilevazioni di particolato PM2.5 e PM10

path\_rilevazioni\_pm2\_5 = f'../dataset/rilevazioni/rilevazioni\_PM2\_5.csv'  
path\_rilevazioni\_pm10 = f'../dataset/rilevazioni/rilevazioni\_PM10.csv'  
path\_comuni = f'../dataset/dati\_spaziali/comuni/comuni.shp'

Creazione dei dataframe finali per stazioni, rileva e rilevazioni

stazioni = pd.DataFrame(columns= ['nome', 'lat', 'long', 'id\_comune'])  
rileva = pd.DataFrame(columns= ['nome\_stazione', 'sigla\_indicatore'])  
rilevazioni = pd.DataFrame(columns= ['data', 'valore', 'sigla\_indicatore', 'nome\_stazione'])

Lettura del dataset comuni per assegnare le stazioni ai relativi comuni

comuni = gpd.read\_file(path\_comuni)

Lettura ai dataset relativi alle rilevazioni di particolato PM2.5 e PM10. Vengono scartate alcune righe perché è presente una descrizione dei dataset

rilevazioni\_pm2\_5 = pd.read\_csv(path\_rilevazioni\_pm2\_5,   
 sep=';',   
 skiprows=[0, 1, 2, 3 , 4, 5, 6, 8],  
 skipfooter=1,  
 engine='python',   
 index\_col=False)  
rilevazioni\_pm10 = pd.read\_csv(path\_rilevazioni\_pm10,  
 sep=';',   
 skiprows=[0, 1, 2, 3, 4, 5, 7],  
 skipfooter=1,  
 engine='python',   
 index\_col=False)

Si considerano solo le colonne 'NomeCentralina', 'DataRilevazione ', 'Comune', 'Sigla ', 'Valore' del dataset rilevazioni\_pm2\_5 e rilevazioni\_pm10 e ridenominazione delle colonne

columns = ['Nome', 'Data', 'Comune', 'Sigla', 'Valore']  
rilevazioni\_pm2\_5 = rilevazioni\_pm2\_5[['NomeCentralina ', 'DataRilevazione ', 'Comune ', 'Sigla ', 'Valore ']]  
rilevazioni\_pm2\_5.columns = columns  
  
rilevazioni\_pm10 = rilevazioni\_pm10[['NomeCentralina ', 'DataRilevazione ', 'Comune ', 'Sigla ', 'Valore ']]  
rilevazioni\_pm10.columns = columns

Nei file csv delle rilevazioni, al cambio della centralina viene ripetuto l'header, l'inquinante e la provincia. Vengono quindi rimosse queste righe

rilevazioni\_pm2\_5 = rilevazioni\_pm2\_5[rilevazioni\_pm2\_5["Nome"].str.contains('Provincia:|NomeCentralina|Inquinante:') == False]  
rilevazioni\_pm2\_5.reset\_index(inplace=True, drop=True)  
rilevazioni\_pm10 = rilevazioni\_pm10[rilevazioni\_pm10["Nome"].str.contains('Provincia:|NomeCentralina|Inquinante:') == False]  
rilevazioni\_pm10.reset\_index(inplace=True, drop=True)  
  
rilevazioni\_pm2\_5["Nome"] = rilevazioni\_pm2\_5["Nome"].apply(lambda x: x.split("-")[1].strip().replace(" ", " "))  
rilevazioni\_pm10["Nome"] = rilevazioni\_pm10["Nome"].apply(lambda x: x.split("-")[1].strip().replace(" ", " "))

Estrazione della latitudine e longitudine delle stazioni da OpenSteetMap. Se non sono disponibili le coordinate, vengono considerate quelle dei comuni

import requests  
import json

def get\_lat\_long(stazione, comune):  
 s = stazione.replace(' ', '+')  
 c = comune.replace(' ', '+')  
 request = requests.get(f'https://nominatim.openstreetmap.org/search?country=Italia&city={comune}&street={stazione}&format=jsonv2&limit=1')  
 json\_object = json.loads(request.content)  
 if len(json\_object) == 0:  
 request = requests.get(f'https://nominatim.openstreetmap.org/search?country=Italia&city={comune}&format=jsonv2&limit=1')  
 json\_object = json.loads(request.content)  
 return json\_object[0]['lat'], json\_object[0]['lon']

Preparazione dei dataset finali di stazioni e rileva

stazioni\_pm2\_5\_comuni\_nomi = set(zip(rilevazioni\_pm2\_5['Nome'].values, rilevazioni\_pm2\_5['Comune'].values))  
for s,c in stazioni\_pm2\_5\_comuni\_nomi:  
 comune = c.strip()  
 if len(comuni[comuni['COMUNE'] == comune]['PRO\_COM'].values) != 0:  
 if stazioni[stazioni['nome'] == s].shape[0] == 0:  
 id\_comune = comuni[comuni['COMUNE'] == comune]['PRO\_COM'].values[0]  
 lat, long = get\_lat\_long(s, comune)  
 row = [s, lat, long, id\_comune]  
 stazioni.loc[len(stazioni.index)] = row  
  
 row = [s, 'PM2.5']  
 rileva.loc[len(rileva.index)] = row

stazioni\_pm10\_comuni\_nomi = set(zip(rilevazioni\_pm10['Nome'].values, rilevazioni\_pm10['Comune'].values))  
for s,c in stazioni\_pm10\_comuni\_nomi:  
 comune = c.strip()  
 if len(comuni[comuni['COMUNE'] == comune]['PRO\_COM'].values) != 0:  
 if stazioni[stazioni['nome'] == s].shape[0] == 0:  
 id\_comune = comuni[comuni['COMUNE'] == comune]['PRO\_COM'].values[0]  
 lat, long = get\_lat\_long(s, comune)  
 row = [s, lat, long, id\_comune]  
 stazioni.loc[len(stazioni.index)] = row  
  
 row = [s, 'PM10']  
 rileva.loc[len(rileva.index)] = row

Preparazione del dataset finale di rilevazioni

for i in range(rilevazioni\_pm2\_5.shape[0]):  
 data = rilevazioni\_pm2\_5.loc[i, 'Data'][:10]  
 valore = rilevazioni\_pm2\_5.loc[i, 'Valore']  
  
 sigla\_indicatore = rilevazioni\_pm2\_5.loc[i, 'Sigla']  
 nome\_stazione = rilevazioni\_pm2\_5.loc[i, 'Nome']  
 row = [data, valore, sigla\_indicatore, nome\_stazione]  
 rilevazioni.loc[len(rilevazioni.index)] = row

for i in range(rilevazioni\_pm10.shape[0]):  
 data = rilevazioni\_pm10.loc[i, 'Data'][:10]  
 valore = rilevazioni\_pm10.loc[i, 'Valore']  
  
 sigla\_indicatore = rilevazioni\_pm10.loc[i, 'Sigla']  
 nome\_stazione = rilevazioni\_pm10.loc[i, 'Nome']  
 row = [data, valore, sigla\_indicatore, nome\_stazione]  
 rilevazioni.loc[len(rilevazioni.index)] = row

Esportazione dei dataset trasformati

stazioni.to\_csv('../dataset/stazioni/stazioni\_finale.csv', index=False)  
rilevazioni.to\_csv('../dataset/rilevazioni/rilevazioni\_finale.csv', index=False)  
rileva.to\_csv('../dataset/rileva/rileva\_finale.csv', index=False)

## Caricamento

Dopo aver creato il database InquinamentoAriaPuglia, è stata aggiunta l’estensione postgis e sono stati importati i file shapefile in PgAdmin4 utilizzando il software PostGIS 2.0 Shapefile and DBF Loader Exporter.

Immagine che contiene testo, schermata, schermo, software

Descrizione generata automaticamente

Di seguito è riportata la fase di Data Definition Language.

L’importazione degli shapefile relativi alle province e ai comuni crea nuove tabelle nel database. Vengono quindi ridenominate le tabelle create con l’importazione, modificate le chiavi primarie e create le chiavi esterne alle altre tabelle. Inoltre, per permettere la visualizzazione delle stazioni, delle province e delle regioni all’interno di una mappa, è stato cambiato il sistema di riferimento dell’attributo spaziale delle province e regioni da 32632 a 4326.

**ALTER TABLE comuni\_finale RENAME TO comuni;**

**ALTER TABLE province\_finale RENAME TO province;**

**ALTER TABLE comuni DROP COLUMN gid;**

**ALTER TABLE comuni ADD CONSTRAINT comuni\_pkey primary key (id\_comune);**

**ALTER TABLE province DROP COLUMN gid;**

**ALTER TABLE province ADD CONSTRAINT province\_pkey primary key (id\_prov);**

**ALTER TABLE comuni ALTER COLUMN id\_comune type integer;**

**ALTER TABLE comuni ALTER COLUMN id\_prov type integer;**

**ALTER TABLE comuni ADD FOREIGN KEY (id\_prov) REFERENCES province(id\_prov) ON UPDATE CASCADE ON DELETE CASCADE;**

**ALTER TABLE comuni**

**ALTER COLUMN geom TYPE geometry(MULTIPOLYGON, 4326) USING ST\_Transform(ST\_SetSRID(geom,32632),4326);**

**ALTER TABLE province**

**ALTER COLUMN geom TYPE geometry(MULTIPOLYGON, 4326) USING ST\_Transform(ST\_SetSRID(geom,32632),4326);**

Di seguito sono riportate le creazioni delle restanti tabelle

**CREATE TABLE decessi(**

**id\_decessi SERIAL PRIMARY KEY,**

**nome\_malattia VARCHAR(100) NOT NULL,**

**numero\_decessi\_maschi INT NOT NULL,**

**numero\_decessi\_femmine INT NOT NULL,**

**anno INT NOT NULL,**

**id\_prov INT NOT NULL,**

**FOREIGN KEY (id\_prov) REFERENCES province(id\_prov)**

**ON UPDATE CASCADE**

**ON DELETE CASCADE**

**);**

**CREATE TABLE popolazioni(**

**id\_popolazioni SERIAL PRIMARY KEY,**

**numero\_maschi INT NOT NULL,**

**numero\_femmine INT NOT NULL,**

**anno INT NOT NULL,**

**id\_comune INT NOT NULL,**

**FOREIGN KEY (id\_comune) REFERENCES comuni(id\_comune)**

**ON UPDATE CASCADE**

**ON DELETE CASCADE**

**);**

**CREATE TABLE stazioni(**

**nome VARCHAR(100) PRIMARY KEY,**

**geom GEOMETRY(Point, 4326),**

**id\_comune INT NOT NULL,**

**lat numeric NOT NULL,**

**long numeric NOT NULL,**

**FOREIGN KEY (id\_comune) REFERENCES comuni(id\_comune)**

**ON UPDATE CASCADE**

**ON DELETE CASCADE**

**);**

**CREATE TABLE indicatori(**

**sigla VARCHAR(10) PRIMARY KEY,**

**nome VARCHAR(100) NOT NULL,**

**soglia\_media\_giornaliera FLOAT NOT NULL,**

**giorni\_superamento INT NOT NULL,**

**soglia\_media\_annuale FLOAT NOT NULL**

**);**

**CREATE TABLE rileva(**

**sigla\_indicatore VARCHAR(10),**

**nome\_stazione VARCHAR(100),**

**PRIMARY KEY(sigla\_indicatore, nome\_stazione),**

**FOREIGN KEY (sigla\_indicatore) REFERENCES indicatori(sigla),**

**FOREIGN KEY (nome\_stazione) REFERENCES stazioni(nome)**

**ON UPDATE CASCADE**

**ON DELETE CASCADE**

**);**

**CREATE TABLE rilevazioni(**

**id\_rilevazioni SERIAL PRIMARY KEY,**

**data\_rilevazione DATE NOT NULL,**

**valore FLOAT,**

**sigla\_indicatore VARCHAR(10),**

**nome\_stazione VARCHAR(100),**

**FOREIGN KEY (sigla\_indicatore) REFERENCES indicatori(sigla),**

**FOREIGN KEY (nome\_stazione) REFERENCES stazioni(nome)**

**ON UPDATE CASCADE**

**ON DELETE CASCADE**

**);**

Al termine della fase di DDL, è stata effettuata la fase di Data Manipulation Language (DML) in cui sono stati importati i dati preparati nella fase di trasformazione

COPY decessi(nome\_malattia,numero\_decessi\_maschi,numero\_decessi\_femmine,anno,id\_prov)  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\decessi\decessi\_finale.csv'  
DELIMITER ','  
CSV Header;  
  
  
COPY popolazioni(id\_comune, anno,numero\_maschi,numero\_femmine)  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\popolazioni\popolazioni\_finale.csv'  
DELIMITER ','  
CSV HEADER;  
  
COPY stazioni(nome,lat,long,id\_comune)  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\stazioni\stazioni\_finale.csv'  
DELIMITER ','  
CSV Header;  
**UPDATE** stazioni **SET** geom = ST\_SETSRID(ST\_MakePoint(**long**, lat),4326);  
  
  
COPY indicatori  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\indicatori\indicatori\_finale.csv'  
DELIMITER ','  
CSV HEADER;  
  
  
COPY rileva(nome\_stazione, sigla\_indicatore)  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\rileva\rileva\_finale.csv'  
DELIMITER ','  
CSV HEADER;  
   
  
COPY rilevazioni(data\_rilevazione,valore,sigla\_indicatore,nome\_stazione)  
FROM 'C:\Users\Marco\progetto\_dati\_strutturati\dataset\rilevazioni\rilevazioni\_finale.csv'  
DELIMITER ','  
CSV HEADER;

# Modellazione

In questo capitolo vengono riportati il modello concettuale, con un diagramma E-R, e il modello logico, con un diagramma relazionale.

## Diagramma ER

Il modello concettuale di un database rappresenta in modo astratto la realtà di interesse, indipendentemente dal tipo di sistema di gestione del database (DBMS) utilizzato. Questo livello si avvicina di più alla visione umana e si distacca dai dettagli tecnici relativi alla sua implementazione. La realtà viene modellata mediante l'uso di tipi di entità e tipi di relazione che le collegano.

Immagine che contiene diagramma, testo, linea, Piano

Descrizione generata automaticamente

### Dizionario dei dati

Di seguito viene descritto il diagramma E-R

* Entità **Stazione**: stazione di rilevamento degli inquinanti. Ha come attributi:
  + Nome: primary key. Nome della stazione di rilevamento;
  + Posizione: posizione geografica della stazione.
* Entità **Comune**: comune della Puglia. Ha come attributi:
  + ID\_Comune: primary key. Codice identificativo del comune;
  + Nome: nome del comune;
  + Posizione: posizione geografica del comune.
* Relazione **Appartiene\_A**: relazione che lega le entità Stazione e Comune con una relazione N:1 con partecipazione totale di Stazione. Indica l’appartenenza della stazione al comune.
* Entità **Popolazione**: popolazione registrata dal comune. Ha come attributi:
  + ID\_Popolazione: primary key. Identificativo della popolazione;
  + Numero\_maschi: numero di maschi nella popolazione;
  + Numero\_femmine: numero di femmine nella popolazione;
  + Anno: anno a cui si riferisce la popolazione.
* Relazione **Ha**: relazione che lega le entità Comune e Popolazione con una relazione 1:N con partecipazione totale di entrambe le entità.
* Entità **Provincia**: provincia della puglia. Ha come attributi:
  + ID\_Provincia: primary key. Codice identificativo della provincia;
  + Nome: nome della provincia;
  + Posizione: posizione geografica della provincia.
* Relazione **Situato\_in**: relazione che lega le entità Comune e Provincia con una relazione N:1 con partecipazione totale di entrambe le entità.
* Entità **Decessi**: decessi registrati. Ha come attributi:
  + ID\_Decessi: primary key. Codice identificativo dei decessi;
  + Nome\_malattia: nome della malattia a cui si riferiscono i decessi;
  + Numero\_decessi\_maschi: numero di maschi morti;
  + Numero\_decessi\_femmine: numero di femmine morte;
  + Anno: anno a cui si riferiscono i decessi registrati.
* Relazione **Registra**: relazione che lega le entità Provincia e Decessi con una relazione 1:N con partecipazione totale di entrambe le entità.
* Entità **Rilevazione**: rilevazione dell’indicatore di inquinamento effettata dalla stazione. Ha come attributi:
  + ID\_Rilevazione: primary key. Codice identificativo della rilevazione;
  + Data: data in cui è stata effettuata la rilevazione;
  + Valore: valore di inquinante rilevato.
* Relazione **Effettua**: relazione che lega le entità Stazione e Rilevazione con una relazione 1:N con partecipazione totale di Rilevazione.
* Entità **Indicatore**: indicatore d’inquinamento. Ha come attributi:
  + Sigla: primary key. Sigla dell’indicatore d’inquinamento;
  + Nome: Nome dell’indicatore d’inquinamento;
  + Soglia\_media\_giornaliera: soglia giornaliera dell’inquinante che l’OMS consiglia di non superare;
  + Soglia\_media\_annuale: soglia della media di valori annuali dell’inquinante che l’OMS consiglia di non superare;
  + Giorni\_superamento: giorni che l’OMS considera per la soglia media giornaliera.
* Relazione **Relativo\_a**: relazione che lega le entità Rilevazione e Indicatore con una relazione N:1 con partecipazione totale di Rilevazione.

Relazione **Rileva**: relazione che lega le entità Stazione e Indicatore con una relazione N:N.

## Diagramma Relazionale

Si procede ora dalla fase concettuale a quella logica, che si allontana dalla prospettiva dell'utente e si avvicina maggiormente alla macchina. Il modello logico converte le entità del modello concettuale in un insieme di tabelle, ciascuna delle quali contiene una o più chiavi primarie e vincoli di integrità referenziale con altre tabelle.

Immagine che contiene testo, schermata, Carattere, diagramma

Descrizione generata automaticamente

## Stima della dimensione database

Di seguito si riportano le stime sulla crescita delle tabelle del database dall’anno 2019 all’anno 2028. Per comuni, province e indicatori si ipotizza che le righe non cambino nel tempo.

1. **Decessi**: il dataset risultante dalla fase di trasformazione dell'ETL è composto da 72 righe e contiene dati relativi a due anni. Si stima quindi che ogni anno vengano aggiunte 36 righe al dataset.

Immagine che contiene testo, schermata, diagramma, Diagramma

Descrizione generata automaticamente

1. **Popolazioni**: il dataset risultante dalla fase di trasformazione dell'ETL è composto da 514 righe e contiene dati relativi a due anni. Ipotizzando che non varia il numero di province nel tempo, si stima che ogni anno vengano aggiunte 257 righe al dataset.

Immagine che contiene testo, schermata, diagramma, Diagramma

Descrizione generata automaticamente

1. **Stazioni**: il dataset risultante dalla fase di trasformazione dell'ETL è composto da 79 righe. Si ipotizza che ogni 3 anni si aggiunga 1 stazione nella regione Puglia.

Immagine che contiene testo, schermata, Rettangolo, linea

Descrizione generata automaticamente

1. **Rileva**: il dataset risultante dalla fase di trasformazione dell'ETL è composto da 111 righe. Mantenendo costante il numero di indicatori a 2, il numero di righe di questa tabella può variare da #stazioni\*1 a #stazioni\*2. Nel seguente grafico viene mostrato il numero di righe che variano se si aggiungono solo stazioni che rilevano due indicatori.

Immagine che contiene testo, schermata, Diagramma, linea

Descrizione generata automaticamente

1. **Rilevazioni**: Il dataset risultante dalla fase di trasformazione dell'ETL è composto da un totale di 79594 righe e contiene dati relativi a un periodo di due anni. Si ipotizza quindi, che nel corso del 2019 sono state aggiunte 39797 righe al dataset. È importante notare che ogni anno, ogni stazione effettua in media circa 365 rilevazioni per ogni indicatore che monitora. Ipotizzando che si aggiunga una stazione che rileva il massimo numero di indicatori, cioè due, ogni tre anni, ogni anno si aggiungeranno 39797 righe, e ogni tre anni si aggiungeranno anche le righe corrispondenti alle nuove stazioni.

Immagine che contiene testo, schermata, diagramma, Diagramma

Descrizione generata automaticamente

# Queries

Di seguito sono riportati esempi di query che gli stakeholder potrebbero eseguire sul database:

1. Calcolare il rapporto tra la popolazione e il numero di decessi per ogni provincia in cui il numero di giorni in cui il livello giornaliero degli indicatori ha superato la soglia raccomandata dall'OMS è superiore al limite consentito.

**CREATE** **OR** **REPLACE** **FUNCTION** public.rapporto\_popolazione\_morti\_soglia\_giornaliera(anno integer, indicatore varchar)  
 **RETURNS** **TABLE**(provincia varchar, rapporto\_popolazione\_morti float)   
 **LANGUAGE** 'plpgsql'  
  
**AS** $**BODY**$  
**BEGIN**  
 **RETURN** **query**  
 **SELECT** **distinct**(p.nome), **CAST**(pop.popolazione **as** FLOAT) / **CAST**(d.numero\_morti **as** FLOAT) **as** rapporto\_popolazione\_morti  
 **FROM** province **as** p   
 **JOIN** (  
 **SELECT** d.id\_prov, **SUM**(d.numero\_decessi\_maschi)+**SUM**(d.numero\_decessi\_femmine) **as** numero\_morti  
 **FROM** decessi **as** d  
 **WHERE** d.anno = $1  
 **GROUP** **BY** id\_prov) **as** d  
 **ON** p.id\_prov = d.id\_prov  
 **JOIN** (**SELECT** prov.id\_prov, **SUM**(p.numero\_femmine)+**SUM**(p.numero\_maschi) **as** popolazione  
 **FROM** popolazioni **as** p   
 **JOIN** comuni **as** c **ON** p.id\_comune = c.id\_comune   
 **JOIN** province **as** prov **ON** prov.id\_prov = c.id\_prov  
 **WHERE** p.anno = $1  
 **GROUP** **BY** prov.id\_prov) **as** pop **ON** pop.id\_prov = p.id\_prov   
 **JOIN** comuni **as** c **ON** p.id\_prov = c.id\_prov  
 **JOIN** stazioni **as** s **ON** s.id\_comune = c.id\_comune  
 **JOIN** (**SELECT** ril.nome\_stazione  
 **FROM** (**SELECT** r.nome\_stazione, **COUNT**(\*) **as** numero\_giorni\_superamento  
 **FROM** rilevazioni **as** r **JOIN** indicatori **as** i   
 **ON** i.sigla = r.sigla\_indicatore  
 **WHERE** **EXTRACT**(**YEAR** **from** r.data\_rilevazione) = $1   
 **and** r.sigla\_indicatore = $2   
 **and** r.valore > i.soglia\_media\_giornaliera  
 **GROUP** **BY** r.nome\_stazione  
 ) **as** ril, indicatori **as** i  
 **WHERE** i.sigla = $2 **and** ril.numero\_giorni\_superamento > i.giorni\_superamento) **as** staz **ON** staz.nome\_stazione = s.nome   
 **ORDER** **BY** rapporto\_popolazione\_morti **ASC**;  
  
**END**  
$**BODY**$;

2020 PM10:

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

2020 PM2.5:

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

1. Calcolare in quali comuni il numero di giorni in cui il livello giornaliero degli indicatori ha superato il limite raccomandato dall'OMS supera la soglia consentita:

**CREATE** **OR** **REPLACE** **FUNCTION** public.comuni\_soglia\_giornaliera(anno integer, indicatore varchar)  
 **RETURNS** **TABLE**(comune varchar, provincia varchar)   
 **LANGUAGE** 'plpgsql'  
  
**AS** $**BODY**$  
**BEGIN**  
 **RETURN** **query**  
 **SELECT** **distinct**(c.nome), p.nome  
 **FROM** comuni **as** c   
 **JOIN** province **as** p **ON** p.id\_prov = c.id\_prov  
 **JOIN** stazioni **as** s **ON** s.id\_comune = c.id\_comune  
 **JOIN** (**SELECT** ril.nome\_stazione  
 **FROM** (**SELECT** r.nome\_stazione, **COUNT**(\*) **as** numero\_giorni\_superamento  
 **FROM** rilevazioni **as** r **JOIN** indicatori **as** i   
 **ON** i.sigla = r.sigla\_indicatore  
 **WHERE** **EXTRACT**(**YEAR** **from** r.data\_rilevazione) = $1   
 **and** r.sigla\_indicatore = $2   
 **and** r.valore > i.soglia\_media\_giornaliera  
 **GROUP** **BY** r.nome\_stazione  
 ) **as** ril, indicatori **as** i  
 **WHERE** i.sigla = $2 **and** ril.numero\_giorni\_superamento > i.giorni\_superamento) **as** staz **ON** staz.nome\_stazione = s.nome;  
  
**END**  
$**BODY**$;

2020 PM2.5

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente

2020 PM10

Immagine che contiene testo, schermata, numero, Carattere

Descrizione generata automaticamente

1. Dato un anno, ricavare i 25 comuni con più popolazione:

**CREATE** **OR** **REPLACE** **FUNCTION** public.popolazione\_comuni(anno integer)  
 **RETURNS** **TABLE**(comune varchar, popolazione\_comune integer, provincia varchar)   
 **LANGUAGE** 'plpgsql'  
  
**AS** $**BODY**$  
**BEGIN**  
 **RETURN** **query**  
 **SELECT** c.nome, pop.numero\_maschi+pop.numero\_femmine **as** popolazione, p.nome  
 **FROM** comuni **as** c   
 **JOIN** province **as** p **ON** p.id\_prov = c.id\_prov  
 **JOIN** popolazioni **as** pop **ON** c.id\_comune = pop.id\_comune  
 **WHERE** pop.anno = $1  
 **ORDER** **BY** popolazione **DESC** **LIMIT**(25);  
**END**  
$**BODY**$;

Anno 2020

Immagine che contiene testo, menu, numero, documento

Descrizione generata automaticamente

1. Data una provincia e un anno, visualizzare quante stazioni hanno superato il limite giornaliero annuo:

**CREATE** **OR** **REPLACE** **FUNCTION** public.num\_stazioni\_prov\_sup\_limite\_annuo(provincia varchar, indicatore varchar, anno integer)  
 **RETURNS** **TABLE**(num\_stazioni\_superamento bigint, num\_stazioni\_prov bigint)   
 **LANGUAGE** 'plpgsql'  
  
**AS** $**BODY**$  
**BEGIN**  
 **RETURN** **query**  
 **SELECT** \* **FROM**  
 (**SELECT** **COUNT**(\*) **as** stazioni\_sup\_limite\_giornaliero  
 **FROM** (**SELECT** r.nome\_stazione, **COUNT**(\*) **as** numero\_giorni\_superamento  
 **FROM** rilevazioni **as** r **JOIN** indicatori **as** i   
 **ON** i.sigla = r.sigla\_indicatore  
 **WHERE** **EXTRACT**(**YEAR** **from** r.data\_rilevazione) = $3  
 **and** r.sigla\_indicatore = $2  
 **and** r.valore > i.soglia\_media\_giornaliera  
 **GROUP** **BY** r.nome\_stazione  
 ) **as** ril  
 **JOIN** stazioni **as** s **ON** ril.nome\_stazione = s.nome  
 **JOIN** comuni **as** c **ON** s.id\_comune = c.id\_comune  
 **JOIN** province **as** p **ON** c.id\_prov = p.id\_prov, indicatori **as** i  
 **WHERE** p.nome = $1 **and** i.sigla = $2 **and** ril.numero\_giorni\_superamento > i.giorni\_superamento) **as** stazioni\_sup\_limite\_giornaliero,  
 (**SELECT** **COUNT**(\*) **as** num\_stazioni\_provincia  
 **FROM** province **as** p  
 **JOIN** comuni **as** c **ON** c.id\_prov = p.id\_prov  
 **JOIN** stazioni **as** s **ON** s.id\_comune = c.id\_comune  
 **WHERE** p.nome = $1) **as** num\_stazioni\_provincia;   
**END**  
$**BODY**$;

Bari PM10 2020

Immagine che contiene testo, schermata, Carattere, linea

Descrizione generata automaticamente

1. Ricavare le province con i valori massimi rilevati di un dato indicatore in un dato anno:

**CREATE** **OR** **REPLACE** **FUNCTION** public.rilevazione\_massima\_provincia(indicatore varchar, anno integer)  
 **RETURNS** **TABLE**(provincia varchar, valore\_massimo **double** **precision**)   
 **LANGUAGE** 'plpgsql'  
  
**AS** $**BODY**$  
**BEGIN**  
 **RETURN** **query**  
 **SELECT** p.nome, prov.massimo\_valore  
 **FROM**(  
 **SELECT** p.id\_prov, **MAX**(valore) **as** massimo\_valore   
 **FROM** province **as** p  
 **JOIN** comuni **as** c **ON** c.id\_prov = p.id\_prov  
 **JOIN** stazioni **as** s **ON** s.id\_comune = c.id\_comune  
 **JOIN** rilevazioni **as** r **ON** r.nome\_stazione = s.nome  
 **WHERE** r.sigla\_indicatore = $1 **and** **EXTRACT**(**YEAR** **from** r.data\_rilevazione) = $2  
 **GROUP** **BY** p.id\_prov) **as** prov   
 **JOIN** province **as** p **ON** p.id\_prov = prov.id\_prov  
 **ORDER** **BY** prov.massimo\_valore **DESC**;  
**END**  
$**BODY**$;

PM2.5 2020

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

PM10 2020

Immagine che contiene testo, schermata, Carattere, numero

Descrizione generata automaticamente

1. Visualizzare su una mappa le stazioni di rilevamento colorate in base al valore medio annuale rilevato in un dato anno.

Immagine che contiene testo, schermata, linea, diagramma

Descrizione generata automaticamente

PM10 2020

Immagine che contiene mappa, testo, atlante

Descrizione generata automaticamente

# Presentazione dei risultati

L'obiettivo di questo lavoro consisteva nella produzione di un database destinato all'utilizzo da parte degli analisti delle imprese di condivisione di veicoli, al fine di determinare l'area della Puglia in cui operare, basandosi sui dati rilevati relativi ai livelli di PM2.5 e PM10 durante gli anni 2019-2020. Si è tenuto conto del fatto che risulta più agevole proporre i propri servizi nelle zone maggiormente inquinate. I dati sull'inquinamento e sulla mortalità utilizzati in questo progetto sono stati estratti da due fonti principali: l'Arpa Puglia e l'Istat. Dopo una fase di pre-elaborazione condotta in Python, i dataset sono stati importati nel database management system (DBMS) PostgresSQL mediante l'utilizzo del software PgAdmin4, sul quale sono state eseguite le query. Inoltre, è stato utilizzato il software Knime, che, collegandosi al database, ha consentito di visualizzare su una mappa le stazioni di rilevamento, evidenziate mediante colori in base ai valori medi annuali rilevati. È stata inoltre effettuata una stima della crescita del database, tenendo in considerazione i dati dal 2019 al 2028. Per quanto concerne i lavori futuri, sarà possibile aggiornare i dati relativi ad altri anni, consentendo così di mappare l'evoluzione dell'inquinamento nel corso del tempo. Inoltre, un'ulteriore possibile evoluzione potrebbe essere l’estensione di tale database a tutto il territorio italiano, europeo e internazionale, allo scopo di ottenere un quadro di analisi più completo ed esaustivo. I dataset e i file relativi alla fase di trasformazione dell’ETL, al DDL, al DML, al QL utilizzati in questo lavoro sono disponibili al seguente repository GitHub: <https://github.com/marcocasa/progetto_dati_strutturati>.

# Strumenti utilizzati:

* Python 3.9.13: per la fase di pre-processing;
* PGAdmin4, PostGIS 2.0 Shapefile and DBF Loader Exporter, PostgreSQL 9.6 e postgis 2.5.3: per le fasi di DDL, DML e QL;
* QGIS Desktop 3.28.3: per la visualizzazione grafica delle stazioni, province e comuni.

# Bibliografia

|  |  |
| --- | --- |
| [1] | S. n. p. l. p. dell'ambiente, «PARTE-INIZIALE-STATO-E-TREND.pdf,» 02 12 2020. [Online]. Available: https://www.snpambiente.it/wp-content/uploads/2020/12/PARTE-INIZIALE-STATO-E-TREND.pdf. |
| [2] | «500 actions to take, including 82 measures to reduce both outdoor and indoor air pollution that can help prevent 7 million premature deaths worldwide,» 2 09 2021. [Online]. Available: https://www.who.int/multi-media/details/500-actions-to-take-including-82-measures-to-reduce-both-outdoor-and-indoor-air-pollution-that-can-help-prevent-7-million-prematuer-deaths-worldwide. |
| [3] | «9 out of 10 people worldwide breathe polluted air, but more countries are taking action,» 2 05 2018. [Online]. Available: https://www.who.int/news/item/02-05-2018-9-out-of-10-people-worldwide-breathe-polluted-air-but-more-countries-are-taking-action#:~:text=Air%20pollution%20levels%20remain%20dangerously,outdoor)%20and%20household%20air%20pollution.. |
| [4] | A. R. P. A. V. d'Aosta, «ll particolato atmosferico - PM10 e PM2.5,» [Online]. Available: https://www.arpa.vda.it/it/aria/l-inquinamento-atmosferico/2536-il-particolato-atmosferico. |
| [5] | A. Lombardia, «PM10 E PM2,5,» [Online]. Available: https://www.arpalombardia.it/Pages/Aria/Inquinanti/PM10-PM2,5.aspx. |
| [6] | «PM10 - Particolato atmosferico o polveri sottili,» [Online]. Available: https://www.issalute.it/index.php/la-salute-dalla-a-alla-z-menu/p/pm10-particolato-atmosferico-o-polveri-sottili#effetti-sulla-salute. [Consultato il giorno 28 02 2018]. |
| [7] | M. Migliore, G. D'Orso e D. Caminiti, «The environmental benefits of carsharing: the case study of Palermo,» *Transportation Research Procedia,* vol. XXXXVIII, pp. 2127-2139, 2020. |
| [8] | A. Longo e E. Miccoli, «Studio e analisi della correlazione tra inquinamento ambientale e mortalità nella popolazione italiana». |
| [9] | R. News, «A Trento a 30 all'ora, ecco il piano del Comune per la sicurezza,» [Online]. Available: https://www.rainews.it/tgr/trento/articoli/2023/04/a-trento-a-30-allora-ecco-il-piano-del-comune-per-la-sicurezza-1a2c52be-39f6-4c89-97b7-7d4866fc3b07.html. |
| [10] | I. Macchi, «In Italia crescono le città con limite a 30 km/h,» [Online]. Available: https://www.alvolante.it/news/italia-crescono-citta-limite-30-kmh-381060. |
| [11] | N. I. o. E. H. Sciences, «Air Pollution and Your Health,» [Online]. Available: https://www.niehs.nih.gov/health/topics/agents/air-pollution/index.cfm. [Consultato il giorno 06 05 2023]. |

1. I dati relativi al numero di morti sono disponibili fino all’anno 2020 e quelli relativi alla popolazione a partire dall’anno 2019. Sono stati quindi considerati solo i dati relativi a questi due anni. [↑](#footnote-ref-1)
2. Secondo il National Institute of Environmental Health Sciences, alcune delle malattie causate dall'inquinamento ambientale includono il cancro, le malattie cardiovascolari, le malattie respiratorie, il diabete mellito, l'obesità e i disturbi riproduttivi, neurologici e del sistema immunitario [9]. [↑](#footnote-ref-2)
3. <https://github.com/marcocasa/progetto_dati_strutturati> [↑](#footnote-ref-3)