Build the Datawarehouse

[Laboratory for Data Science Project Pt1]

Authors: *Davide Ricci, Alessandro Mastrorilli*

Professors: *Roberto Pellungrini, Anna Monreale, Cristiano Landi*
Master's degree: Data Science & Business Informatics

Introduction

Lo scopo di questa prima parte progettuale è quello di creare e popolare un database partendo da diversi file(in formato csv,json,xml) e conseguendo opportune operazioni e trasformazioni seguendo la metodologia ETL utilizzando Python. In particolare nei vari assignment incrementali si richiede di riprodurre lo schema di dati seguente:

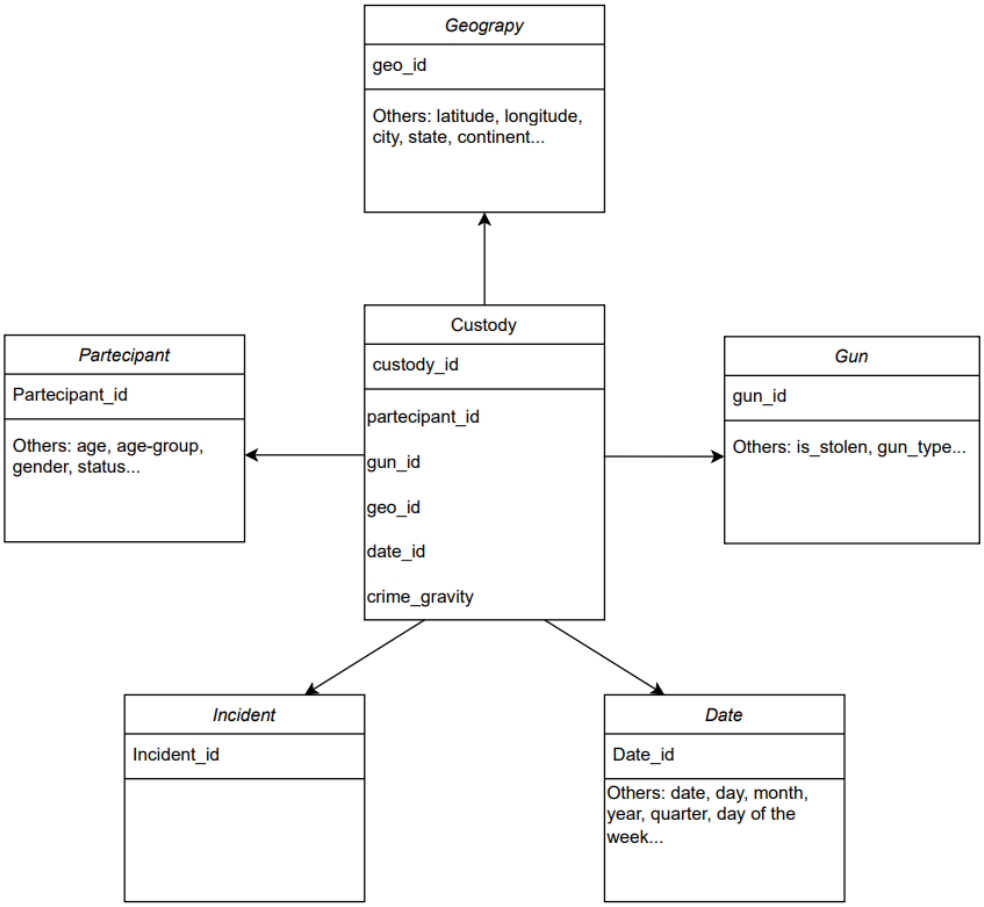

Figura 1: Datawarehouse schema

Assignment 0

Il primo task di BI richiede di progettare e implementare lo schema di dati proposto utilizando SQL Server Management Studio. Come si può vedere dalla figura seguente è stata definita la fact table ”Custodies” contenente, oltre alla chiave primaria e all'attributo crime_gravity, i riferimenti di chiave esterna verso le altre tabelle dimensionali. In particolare per la dimensione ”Incident”, essendo priva di informazioni aggiuntive oltre all'identificativo, si è deciso di trattarla come dimensione degenere e di inserire il relativo attributo all'interno della tabella del fatto senza creare la rispettiva tabella dimensionale. Inoltre per la definizione dei tipi di dati sono state effettuate le seguenti scelte principali:

- per gli identificativi di chiave primaria ed esterna è stato utilizzato il tipo int. In particolare per le PK si è deciso di non settare l'autoincremento e di gestire gli ID lato client, invece per le FK la possibilità di assumere valori nulli.
- per gli attributi di tipo stringa è stato scelto il tipo nvarchar con una lunghezza massima di 15 o 50 caratteri a seconda della caratteristica.

- l'attributo relativo alla data è stato definito come date in quanto nel nostro caso di studio la caratteristica relativa al tempo è costante(00:00:00) per tutte le istanze.

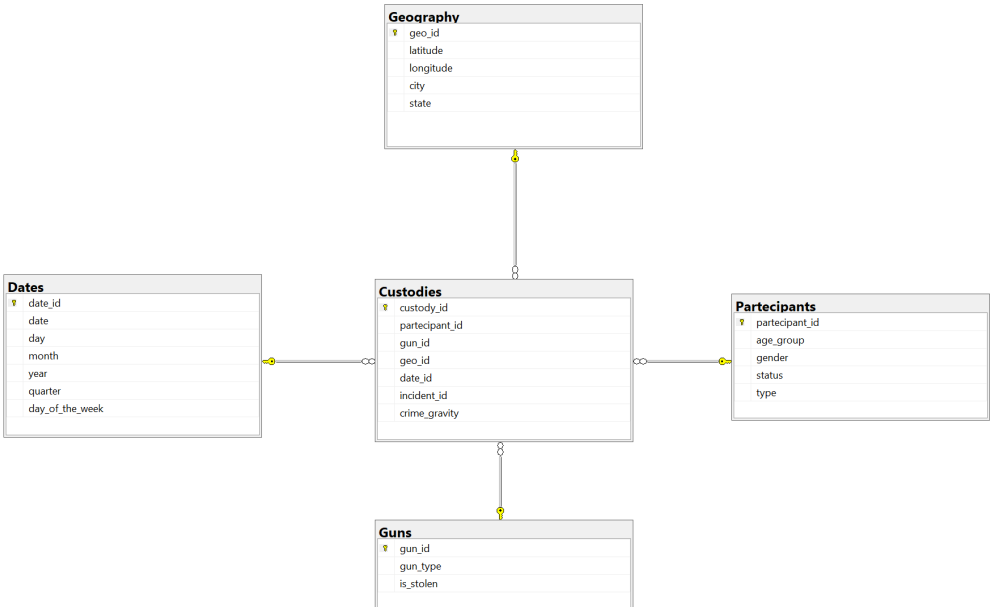

Figura 2: Database schema

Assignment 1

Il secondo task richiede di dividere il contenuto dei file Police.csv e dates.xml in sei tabelle separate e di calcolare il crime_gravity, misura principale del Datawarehouse, sfruttando le informazioni contenute nei file addizionali json. Come anticipato nella sezione precedente si è deciso di non implementare la tabella "Incidents" ma di trattarla come dimensione degenere e di inserirla quindi come caratteristica aggiuntiva nella tabella "Custodies". Dopo una prima fase di data understanding in cui si è riscontrata la presenza di valori "Unknown" nelle colonne gun_type e gun_stolen, si è deciso di sostituire tali valori con "NULL" e di considerare invece ammissibili i valori "Irrelevant". Tale modifica è stata implementata nello script crime_gravity.py che, come suggerisce il nome, calcola anche la relativa caratteristica per ogni istanza del file Police generando infine un nuovo file Police_crime.csv contenente una colonna aggiuntiva che rappresenta la gravità del crimine associata ad ogni fatto. Tale file sarà la sorgente di tutti i successivi script python per la generazione delle tabelle richieste. Si precisa che la prima colonna di ogni tabella contiene l'identificativo che, se non presente nel file Police.csv, viene generato incrementalmente a partire da 0. Di seguito viene mostrata la metodologia conseguita per la generazione di ciascuna tabella :

- Dates: per ogni istanza, le informazioni relative alla data sono state estratte dal file dates.xml utilizzando la relativa chiave presente nel file principale. Sono state poi definite apposite funzioni per estrarre gli attributi year,quarter,month,day, day of the week.
- Guns: la tabella è caratterizzata dagli attributi gun_type e gun_stolen estratti direttamente dal file principale
- Partecipants: le informazioni relative a gender,status,type, age_group dell'entità "partecipant" sono state estratte seguendo la logica della precedente tabella.
- Geography: per generare la tabella contenente le caratteristiche relative alla città e stato in cui si è verificato il fatto si è deciso di utilizzare un dataset esterno "uscities.csv" disponibile al seguente indirizzo us-cities. In particolare , sono stati conseguiti i seguenti passi :

  - Estrazione coppie univoche (latitudine,longitudine) dal file principale per ogni fatto.
  - Estrazione informazioni su latitudine,longitudine,città e stato dal file uscities.csv
  - Dal momento che il match tra le due coppie di (lat,long) estratte può non essere immediato , si è deciso di calcolare la loro distanza di Manhattan, considerando una threshold iniziale pari a 0.0001. Il mapping della città e dello stato avviene solamente se i risultati restituiti dopo i calcoli , sono inferiori o uguali al valore della soglia, altrimenti il threshold viene incrementato di 0.0001 finchè tutti i fatti non sono stati processati. Per cercare di ottenere un risultato più preciso è stato definito anche un secondo script per la generazione della tabella richiesta che sfrutta la funzione "great_circle", presente nella libreria geopy, per calcolare la distanza minima sulla superficie terrestre tra due punti date le loro coordinate geografiche e restituendo le informazioni city e state relative alla distanza trovata. Purtroppo questa seconda soluzione è risultata computazionalmente inefficiente e la tabella Geography è stata definita eseguendo lo script contenente la prima soluzione.

- Custodies: La tabella relativa al fatto in analisi è stata definita iterando il file principale Police_crime.csv e andando a reperire, per ogni istanza, i corrispondenti valori di chiave esterna all'interno delle tabelle precedentemente definite. Per realizzare questa logica sono stati utilizzati opportuni dizionari contenenti come key una tupla di valori che identifica univocamente ogni record delle tabelle dimensionali e come value l'informazione di referenza esterna ricercata.

Alla fine di ogni script viene generata la conseguente tabella in formato csv che verrà poi utilizzata, nel prossimo task, per popolare il database progettato su SQL Server.

Assignment 2

L'obiettivo di questo ultimo assignment è quello di popolare il database con le tabelle create nelle sezioni precedenti. Per conseguire ciò è stato definito lo script load_data.py in cui, dopo aver instaurato la connessione al server e database, sono stati creati opportuni dizionari contenenti le informazioni delle 5 tabelle che verranno proccessati e caricati sul database. In particolare la logica di inserimento è stata definita all'interno di un blocco "try" per catturare qualsiasi potenziale eccezione scaturita a tempo di esecuzione; i relativi errori verranno poi registrati in un opportuno file errors.txt. Inoltre, per controllare il flusso di esecuzione dello script e il tempo di caricamento di ogni tabella, sono state utilizzate le librerie time e tqdm.