

# Simulazione d'esame 05/06/2019

Si consideri il data-set "denver\_crime" che riporta una serie di oltre 300k eventi criminosi tra il 2014 ed il 2017 nella città di Denver (Colorado, USA). Il data-set contiene un'unica tabella 'events' (vedi la figura nella pagina successiva), ed è tratto dell'originale db costantemente aggiornato e disponibile al sito web <https://www.denvergov.org/opendata/dataset/city-and-county-of-denver-crime>.

Si intende costruire un'applicazione JavaFX che permetta di interrogare tale base dati e ricavare informazioni utili alla distribuzione delle forze di polizia. L'applicazione dovrà svolgere le seguenti funzioni:

## PUNTO 1

- Si permetta all'utente di selezionare dal menu a tendina un anno tra quelli presenti nel database. Si usi come riferimento la colonna 'reported\_date'.
- Alla pressione del bottone 'rete cittadina' si costruisca un grafo pesato non orientato che connette 7 vertici, ciascuno corrispondente ad uno dei distretti di Denver (campo 'district\_id'). Per ogni distretto si considerino tutti gli eventi dell'anno selezionato, e, usando le coordinate 'geo\_lon' e 'geo\_lat' si determini il **centro geografico dei crimini di ogni distretto**. Ogni coppia di stretti è collegata da un arco il cui peso sia pari alla distanza tra i centri dei due distretti connessi, espressa in km. Per calcolare tale distanza si faccia uso della libreria *simplelatlng*<sup>1</sup>, già inclusa nel progetto.
- Si visualizzi, per ciascuno dei 7 distretti, l'elenco dei distretti adiacenti, ordinata in base alla distanza crescente.

=> media coordinate

## PUNTO 2

- Si permetta all'utente di selezionare una specifica data (giorno e mese) dell'anno già considerato ed inserire un numero N (compreso tra 1 e 10) di agenti disponibili. Vogliamo valutare il numero N di agenti utili a gestire gli eventi criminosi di Denver tramite una simulazione dei loro spostamenti durante la giornata selezionata.
- Si ipotizzi che la centrale di polizia di Denver si trovi nel distretto a minore criminalità dell'anno in corso e che tutti gli agenti siano originariamente in centrale al momento del primo evento del giorno. Al verificarsi di un evento (ordinati per 'reported\_date'), la centrale sceglie l'agente da inviare sul posto, selezionando l'agente libero più vicino. Ciascun agente libero può muoversi ad una velocità di 60 km/h, tenendo conto della distanza tra i distretti calcolata nel punto 1. Il tempo necessario a gestire un evento (tempo di permanenza nel distretto con l'evento) è, **con uguale probabilità**, di 1 o 2 ore per eventi di tipo 'offense\_category\_id=all\_other\_crimes', mentre il tempo necessario a gestire qualsiasi altro tipo di 'offense\_category\_id' è di 2 ore. Una volta che l'evento è stato gestito l'agente rimane sul posto e risulta disponibile ad occuparsi di un nuovo evento. Un evento si considera "mal gestito" se l'agente arriva nel distretto dell'evento oltre 15 minuti dal suo orario in 'reported\_date'.

<sup>1</sup> <http://javadoc.com/com.javadocmd/simplelatlng/1.3.0/com/javadocmd/simplelatlng/package-summary.html>

- c. Il simulatore dovrà produrre il numero degli eventi “mal gestiti” in funzione della data specificata e del numero di agenti N.

Nella realizzazione del codice, si lavori a partire dalle classi (Bean e DAO, FXML) e dal database contenuti nel progetto di base. È ovviamente permesso aggiungere o modificare classi e metodi.

Tutti i possibili errori di immissione, validazione dati, accesso al database, ed algoritmici devono essere gestiti, non sono ammesse eccezioni generate dal programma.

events	
🔑	incident_id: BIGINT
🔑	offense_code: INTEGER
🔑	offense_code_extension: INTEGER
🔑	offense_type_id: VARCHAR(30)
🔑	offense_category_id: VARCHAR(28)
🔑	reported_date: DATETIME
🔑	incident_address: VARCHAR(97)
🔑	geo_lon: DOUBLE
🔑	geo_lat: DOUBLE
🔑	district_id: INTEGER
🔑	precinct_id: INTEGER
🔑	neighborhood_id: VARCHAR(26)
🔑	is_crime: INTEGER
🔑	is_traffic: INTEGER