UNIVERSITÀ DI PISA

Department of Computer Science

Master programme in Data Science and Business Informatics

Decision Support Systems
Module II: Laboratory of Data Science

Project report

Miccoli Martin
Montinaro Aldo
Poiani Marco

A.A. 2024/2025

Contents

Introduction

Part 1

The goal of the assignments of Part 1 is to design and populate a database using data from given CSV files and perform a series of tasks to manipulate and analyze this data. We are required to use python scripts and additionally implement solutions using SQL Server Integration Services (SSIS), with a focus on minimizing the use of SQL commands within the nodes, relying primarily on native SSIS nodes for computations.
This project revolves around traffic incidents in Chicago, using a simplified dataset derived from Kaggle. The goal is to simulate a Decision Support System tailored for an insurance company. The dataset consists of three main files:

1. **Crashes.csv**: contains detailed records of traffic incidents occurring between January 2014 and January 2019 in Chicago, with additional information about the incidents' causes, road properties, and injuries reported.

2. **People.csv**: provides details about individuals involved in the traffic incidents, including their sex, age, and city of residence.

3. **Vehicles.csv**: contains information about the vehicles involved in the incidents, including data collected by the police post-incident.

Part 2

Project Assignment - Part 1

2.1 Data Understanding and Cleaning

Assignment 1-2

The **Crashes** dataset contains 257925 records with 36 attributes with information about road incidents in Chicago between January 18, 2014 and January 12, 2019. The data types found are int, float and object (mainly strings but also datetimes formatted as strings). There are no duplicate rows, but there are missing values that have been handled using different approaches:

- REPORT_TYPE: binary column which contains 1.94% of nulls; cannot be retrieved and we decided to keep null values in order to not assign incorrect labels.
- STREET_DIRECTION: contains cardinal directions, only 2 missing values that cannot be retrieved.
- BEAT_OF_OCCURRENCE: 4 nulls that can be retrieved by spatial join with official police beat partition from Chicago Data Portal [1].
- MOST_SEVERE_INJURY: 7 nulls, not available online.
- LATITUDE, LONGITUDE, LOCATION: first two contains 0.4% of nulls, while LOCATION contains 1022 nulls. All of them can be retrieved using reverse geocoding.

A candidate key for the records is the column **RD_NO** which contains 257925 unique values (no missing) with equal frequency 1 and constant length 8. We can get some initial insights from Crashes: by looking at distributions of days and hours of accidents we see that the most of them occurred on a Friday, during the month of Ocrober and around 3/4 PM; in 79.64% of cases weather is CLEAR and most of the accidents happened on DRY roadway surface; the LOCATION with the highest number of records corresponds to a place near Chicago-O'Hare International Airport.

The Crashes data cleaning was handled by `main.py` script that orchestrates the cleaning workflow calling objects (and corresponding functions) from `data_cleaning.py` script. The libraries used in this step are: `csv`, `time`, `geopy`, `shapely`, `h3`. The classes containing cleaning functions are:

- `DataProcessor`: contains loading and saving functions, dates processing (we separated CRASH_DATE into day, month, quarter, year, hour and minute) and the correction of NUM_UNITS values (based on number of UNIT_NO in the corresponding data in Vehicles.csv;
- `DataGeocoder`: by geocoding and reverse geocoding retrieves geospatial information.
- `SpatialOperator`: performs spatial joins to retrieve police beats and gets H3 encoding to enlarge spatial information.
- `OutCorrection`: indentifies recorded location that falls outside the geographical boundaries of the city of Chicago and correct them using reverse geocoding.

Main script takes Crashes, Vehicles, PoliceBeats and generates a clean and updated version of Crashes.

- For the People Data Cleaning in `People_Data_Cleaning.py`, demographic corrections and consistency checks were carried out. Missing values in key columns like PERSON_ID and VEHICLE_ID were either filled or raised as errors. The CRASH_DATE column was parsed into multiple components, such as YEAR, MONTH, and HOUR, to facilitate temporal analysis. City names were standardized, with ZIP codes used to infer missing information, and invalid entries were removed. The DAMAGE values were standardized, with missing entries defaulted to 500.00, ensuring uniformity across records.
- The Vehicles cleaning was managed using `Vehicles_data_cleaning.py`: a custom DataFrame class was developed to validate and clean the vehicle data. This included checking the consistency of each row's column count and cleaning extraneous details from the MODEL column to make vehicle names consistent. Missing VEHICLE_ID values were filled with a default of -1 to ensure consistency.

2.2 Data Warehouse Schema

Assignment 3

We identify as a fact table a table called **DamageToUser** that will contain, in addition to the foreign keys for the dimensions, the measures DAMAGE (containing the estimated cost of the damage) and NUM_UNITS (containing the number of units involved in the incident associated with the user under consideration); each fact (identified by DTUID) will represent a new damage associated to a user involved in an accident. We came to this choice mainly because we noticed, in the People dataset, the presence of totally identical rows (thus same person with the same characteristics) in which only the value of the associated DAMAGE changed.

2.3 Data Preparation and Uploading

Assignment 4

Assignment 5

Assignment 6

2.4 SSIS Business Questions

Project Assignment - Part 2

Bibliography

[1] *Boundaries - Police Beats (current) — City of Chicago — Data Portal — data.cityofchicago.org*. `https://data.cityofchicago.org/d/aerh-rz74`. [Accessed 01-12-2024].