University of Pisa
Laboratory of Data Science
A.Y 2024-2025

LDS Project

Submitted to:
Prof. Anna Monreale

Submitted by:
Alessandro Carella
Sara Hoxha
Rafael Ignacio Urbina Hincapie

December 22, 2024

Contents

1 Data Understanding & Cleaning

This section describes the steps we've taken to understand the Chicago traffic crash data, which was organized into three files `People.csv` , `Vehicles.csv`, and `Crashes.csv`. The data cleaning was performed over these files, and the results were saved into three respective files with the suffix '_Preprocessed'.

1.1 People Data

The *People* table provides detailed information about individuals involved in traffic crashes in Chicago. It includes information such as person's address of residence, their age, their actions during the crash, etc. To better understand the data, we reviewed the data types and unique values for each column which gave us an idea of what the dataset included. We explored potential duplicates but found none. However, we noticed that certain columns, such as `AGE`, were incorrectly represented as floating-point numbers instead of integers. Anther peculiarity was found in the `SEX` column, where a lot of rows had the value "X". Additionally, some columns included specific values to represent "Unknown," but also included missing values. We identified 14 columns with missing values. We utilized visualizations, including bar plots and histograms, which revealed some interesting trends, particularly regarding the `AGE` column. We discovered that a large number of entries, close to 20k, had an age recorded as 0 or had ages that seemed unusually young for drivers.

Based on the findings from data exploration on the `People` table, we decided to implement the following changes to clean the data:

1. **Use the `CITY` column to determine the `STATE` column when the latter is empty.** Initially, we considered using GeoCode API to fill the missing values, yet due to potential API usage limits and time constraints, we opted to use an external CSV file containing a comprehensive list of U.S. cities and their corresponding states. As such, we use `CITY` to infer the `STATE` by referencing the aforementioned external dataset.

2. **Map NaN `'SEX'` values → `"U"` (Unknown).** The `SEX` column sometimes contains `'U'` or missing values, which indicate unknown gender. We decided to map NaN values to `"U"` to maintain consistency and signify an unknown gender. As for the uncommon value "X", we decided to leave as-is, as it may refer to a person who doesn't wish to specify their sex intentionally.

3. **For columns that have already an "Unknown" value value, fill missing observations with this value.** The following mapping was used: For `CITY`, `DRIVER_ACTION`, `DRIVER_VISION`, `PHYSICAL_CONDITION`, and `EJECTION` → `UNKNOWN`. For `AIRBAG_DEPLOYED` → `DEPLOYMENT UNKNOWN` and for `SAFETY_EQUIPMENT` → `USAGE UNKNOWN`.

4. **Convert `VEHICLE_ID` & `AGE` to an integer.** As they were found to be stored as floats, we will convert both to integers to standardize the data and avoid any future processing issues.

5. **Add new value `'N/A'` for missing values in `DRIVER_VISION` & `DRIVER_ACTION` for passengers.** For rows that meet this condition, we will fill these missing values with `'N/A'` which stands for `Non Applicable`. This ensures that we don't have missing data in driver-related columns for passengers.

6. **Split `CRASH_DATE` into `DAY`, `MONTH`, `YEAR`, and `TIME` columns.** The `CRASH_DATE` column contains both the date and time information and to make the data more granular and easier to analyze, we will split this column into the four separate columns.

7. **Use NaN value for `AGE` when `AGE` is less than 10 and `PERSON_TYPE` is `DRIVER`.** This

is based on the assumption that anyone under the age of 10 cannot possibly be the driver, and these entries likely represent errors or anomalies.

8. **Set `CITY` to `Unknown` when `CITY` is numeric, has a length less than 2, or starts with `UNK`.**We will set the `CITY` value to `Unknown` in these cases as they are likely to represent errors, misspellings or placeholders.

9. **Set `STATE` as Unknown when `CITY` is UNKNOWN or `STATE` == XX**. This ensures that the `STATE` column remains consistent in representing missing or unknown data.

1.2 Vehicles Data

The vehicles table provides comprehensive information about the vehicles involved in crashes. Each record in this table represents a distinct unit involved in a crash, with multiple units potentially being involved in the same incident. This granular structure allows for detailed analysis of multi-vehicle accidents. The table contains the following interesting attributes, such as `UNIT_TYPE`: Categorizes the operational status of the unit at the time of the crash, including classifications such as 'DRIVER', 'PARKED', or 'NON-MOTOR VEHICLE', `LIC_PLATE_STATE`: A two-letter state code indicating where the vehicle is registered, helping track interstate involvement in crashes, `VEHICLE_DEFECT`: Indicates any mechanical or structural defects that may have contributed to the crash, etc... Prior to loading the data into our data warehouse, several standardization and cleaning operations were performed on the Vehicles dataset. The primary focus was on handling missing values (NaN) and standardizing inconsistent entries. The following modifications were implemented:

1. **Missing Values Standardization:** Multiple attributes containing NaN values were standardized to 'UNKNOWN' for consistency and better querying capabilities. This included: UNIT_TYPE, VEHICLE_DEFECT, VEHICLE_TYPE, VEHICLE_USE, VEHICLE_DIRECTION, MANEUVER, and FIRST_CONTACT_POINT.

2. **License Plate State:** Both NaN values and 'XX' entries in LIC_PLATE_STATE were consolidated to 'UNKNOWN' to maintain consistency in representing unidentified states.

3. **Vehicle Make and Model:** Duplicate entries caused by typographical errors were identified and consolidated. The MODEL field contained both 'UNKNOWN' and 'UKNOW' which were standardized to 'UNKNOWN'.

4. **Vehicle Year Cleaning:** Years recorded beyond 2024 were marked as 'UNKNOWN' as they represent impossible future dates. A special case was identified where '999' appeared in the data, potentially representing a typo for '1999'.

1.3 Crashes Data

The *Crashes* dataset provides information about traffic incidents in Chicago, including spatial, temporal, and severity-related aspects of crashes. We cleaned the data by performing the following: **Handling Incorrect Values** Two entries in the `RD_NO` column had lower-case letters, differently from the usual format. These anomalies (`hz273623` and `hz125235`) were corrected to align with the dataset's standard.

Imputation of Missing Values Several columns contained missing data that required careful consideration. For the `REPORT_TYPE` column, 4,996 out of 257,925 entries lacked values. Since reliable inference methods were unavailable and the page on the Chicago police department website indicated missing values as `AMENDED`, these entries were left as missing. The `STREET_DIRECTION` column had two missing values, and attempts to impute them using the `TRAVEL_DIRECTION` column from the Vehicles dataset were unsuccessful, so these values were

retained. Initially there were many missing values for the columns LATITUDE, LONGITUDE, POINT and BEAT_OF_OCCURRENCE. Using the fields STREET_NAME and STREET_NO and the API available at this link we were able to obtain most of the missing values for LATITUDE, LONGITUDE and, obviously POINT. The STREET_NAME column had one missing value. This record also lacked LATITUDE and LONGITUDE, making it impossible to identify the corresponding street name. Using the found values for the 3 columns previously listed we were able to fill almost all missing values in the column BEAT_OF_OCCURRENCE by finding perfect match of the column POINT in other records. Just one record misses the field BEAT_OF_OCCURRENCE and is in an unique point. For the MOST_SEVERE_INJURY column, 7 missing values were identified. No discernible correlation with other injury-related columns or predictors was found, so these entries were left blank.

Derived Features and Enhancements A new column, DELTA_TIME_CRASH_DATE_POLICE_REPORT_DATE, was created to quantify the time difference between crash occurrence (CRASH_DATE) and police notification (DATE_POLICE_NOTIFIED). This feature was expressed in the format "days minutes seconds." Numeric columns such as NUM_UNITS, INJURIES_TOTAL, and various injury subcategories were converted to integer type for consistency.

Limitations and Unresolved Missing Data The REPORT_TYPE column retained 4,996 missing values, while STREET_DIRECTION and STREET_NAME had two and one missing entries, respectively. The BEAT_OF_OCCURRENCE column had one unresolved value, and the MOST_SEVERE_INJURY column retained seven missing entries due to a lack of predictive relationships with other variables. Additionally, one record missing LATITUDE, LONGITUDE, LOCATION, and STREET_NAME could not be resolved because no viable recovery method was identified. While these missing cases represent a very small proportion of the dataset, they were retained without further imputation to preserve the dataset's integrity.

2 Data Warehouse Schema

The data warehouse schema resembles a snowflake design, consisting of a central fact table and multiple dimension tables, which are further normalized. This structure was chosen to optimize query performance and reduce data redundancy, especially when dealing with changing or repeated attribute values across different records which we found to be the case. Below is a breakdown of the schema and the rationale behind the design choices.

2.0.1 Fact Table

The central fact table is DamageReimbursement. It contains the transactional data related to reimbursements for damages caused by crashes, as such the most useful information for the insurance agency.It has a composite primary key, comprised of three foreign keys Crash_ID, Person_ID, and Vehicle_ID linking the fact tables to the respective dimensions. The attribute Cost_Category classifies the reimbursement based on its value, and the measure Cost quantifies the total reimbursement. This structure was chosen because it enables efficient tracking of reimbursements related to specific crashes, vehicles, and people. The foreign keys ensure a comprehensive relationship with other tables and ability to delve into more details if needed. The Cost measure allows for aggregation and analysis of total reimbursement amounts, and the Cost_Category attribute allows segmentation by value categories.

2.0.2 Dimensions

As part of our design, several dimensions were structured to provide a clear and efficient representation of the data. In particular, the `Crash` dimension is normalized into three sub-dimensions: `CrashLocation`, `CrashCondition`, and `Injury`. This normalization approach was primarily chosen to avoid redundancy, such as the repeating location data for multiple crashes. Moreover, it improves data integrity, and enables more granular analysis of specific aspects of a crash if needed. The `DateTime` dimension is shared between two dimensions: `Crash` and `Vehicle`, which allows both to reference temporal attributes consistently, and thus ensuring consistency.

CrashLocation stores information about the geographic and physical location of each crash. Primary key is `Crash_Location_ID`. Key attributes include: `Street_No`, `Latitude`, `Longitude` etc...

CrashCondition captures information about the conditions surrounding each crash, such as weather, road conditions, and traffic control. Primary key is `Crash_Condition_ID`. This dimension allows us to analyze crashes based on various environmental and situational factors, which can provide insights into crash risk and/or eligibility for reimbursement.

Injury contains numerical data on the types and severity of injuries sustained during the crash. Primary key is `Injury_ID`. It is composed of measures, where the main ones include: `Injuries_Total`, `Injuries_Fatal`, `Injuries_No_Indication`, etc..

DateTime stores temporal information about crashes. Primary key is `DateTime_ID`. Attributes are`Day`, `Month`, `Year`, `Time`. It allows for time-based analysis, helping us understand patterns in crash frequency.

Person contains detailed information about the individuals involved in a crash. Primary key is `Person_ID`. Attributes include demographic information such as `Sex`, `Age`, whether safety measures were used `Safety_Equipment`, etc...

Vehicle stores details about the vehicles involved in the crash. Primary key is `Vehicle_ID`. It has attributes like `Make`, `Model`, `Defect`, etc.. The dimension helps us identify patterns related to specific vehicle types, defects and their potential influence to crash severity.

3 Data Uploading

In regards to assignment 6, given the constraints imposed by the relationships between tables in our data warehouse schema, it was not feasible to obtain an exact 10% sample of the data without compromising referential integrity. To address this, we employed a top-down sampling approach. Parent tables (without foreign keys) were sampled using SSIS's percentage sampling operation to achieve a 10% subset. Child tables (with foreign key dependencies) were filtered to include only the records associated with the sampled parent data. While this approach resulted in less than 10% of the data in the child tables, it ensured the database's referential integrity. Uniform sampling across all tables was not viable, as it would have disregarded hierarchical dependencies, leading to inconsistencies such as orphaned records or invalid references. This method, while imperfect in its yield, offered several advantages as referential integrity was preserved and the sampled dataset remained coherent and suitable for query testing.

4 Queries

4.1 Assignment 6a

In this query, we aim to analyze the frequency of crash involvement for all participants across different years to identify individuals with higher crash frequencies on a yearly basis. To achieve this, we begin with the Person table, extracting key demographic details such as ID, Type, Sex, and Age. We then link this data to the DamageReimbursement table to retrieve crash IDs, followed by the Crash table to obtain crash dates, and finally, the DateTime table to extract the year of the crash. The data is aggregated by counting the number of crashes for each participant per year, generating a "Total Crashes" metric. The results are sorted by year and total crashes to provide a clear chronological overview of crash frequency patterns. Finally, the results are stored in the "Query_6_result" table.

4.2 Assignment 7a

This query calculates a day-night crash index per police beat by comparing vehicle counts in nighttime (9 PM–8 AM) vs. daytime (8 AM–9 PM) incidents. To implement this, we extract relevant columns from the Crash(Number_of_Units), CrashLocation(Beat_of_occurrence), and DateTime tables, joining them sequentially on their IDs. A derived column transformation creates a Time_Category field ("Day" or "Night") based on timestamps. Data is split by Time_Category, aggregated by Beat_of_occurrence, and summed for vehicle counts (Number_of_Units_Day and Night). The final ratio calculation is performed using a derived column transformation that handles potential division by zero cases, setting the index to 0 when no daytime crashes exist. The results are sorted by the calculated index and stored in the "Query_7_result" table.

4.3 Assignment 8a

This query calculates the ratio of people under 21 to those over 21 for each quarter, weather condition, and beat of occurrence, analyzing how seasonality, weather, and location in Chicago affect crash proportions by age. This helps insurance companies understand young drivers' behavior and improve decision-making. To implement this, we join the DamageReimbursement, Crash, CrashLocation, CrashCondition, DateTime, and Person tables using SSIS lookup operations to extract relevant data. After counting individuals over and under 21, we aggregate by the required columns, thus creating a new column containing the ratio. The results are sorted and stored in the "Query_8_result" table.

4.4 Assignment 9a

Building on the previous analysis of day-night and age-based ratios, we propose a query to examine the ratio of older individuals (over 60) involved in interstate movements in Chicago. This includes analyzing whether they reside in Chicago and whether their vehicles are registered in other states. Additionally, we aim to identify significant quarterly variations, incorporating seasonality into the analysis. To implement this, we use lookup transformations to retrieve only the necessary attributes from dimensional tables, avoiding full joins. After classifying individuals as over or under 60, we aggregate the data by quarter and license plate registration. The ratio is calculated, sorted using a sorting operator, and saved in the "Query_9_result" table.

5 OLAP Cube

The data cube is designed to store and analyze traffic crash data in Chicago, using as a source the data warehouse that stores this data. The primary objective is to support an insurance company in examining costs, identifying high-risk groups, and understanding geographic and demographic trends to refine their policies. The data cube is structured to offer detailed insights into traffic incidents, including demographic data of individuals involved, vehicle details, crash specifics, and time-related attributes.

Dimensions The data cube is organized into dimensions and measure groups. The dimensions include `Person, Crash, Vehicle, and DateTime`. In Table 1 you can see the dimensions and their attributes.

Dimension	Attributes	Notes
Person	'Person ID', 'Injury Classification', 'Physical Condition', 'Age', 'Type', 'Sex', 'Driver Action', 'Driver Vision', 'Age Group', 'City', 'State'	The Person ID is formatted as 'PersonID-Type-Sex-Age' for clarity and readability. Age Group contains groups 'Under 18', 'Over 65', '18-35', '36-50', '51-65'.
Crash	'Crash ID', 'Crash Date ID', 'Police Notified Date ID', 'Crash Location ID', 'Crash Condition ID', 'Injury ID', 'Primary Contributory Cause', 'Secondary Contributory Cause', 'Difference Between Crash Date & Police Notified' 'Beat of Occurrence', 'Street', 'Street No', 'Longitude', 'Latitude'	For clarity and readability, Crash ID is formatted as 'CrashID-Primary Contributory Cause', Crash Location ID as 'CrashLocationId-Street-StreetNo', Crash Condition ID as 'CrashConditionID-Weather Condition-LightningCondition', Injury ID as Injuries Total.
Vehicle	'Vehicle ID', 'Vehicle Type'	The Vehicle ID is formatted as 'VehicleID-Make-Model' for clarity and readability.
DateTime	'Date Time ID', 'Day', 'Year', 'Month Name', 'Time'	The Date Time ID is formatted as 'Year-Month-Day' for clarity and readability. Month Name represents textual names of months corresponding to their numeric values (1–12). Time is also displayed in a 24H format. It's a role-playing dimension and represents: Crash Date and Police Notified Date.

Table 1: Dimensions and their Attributes

Hierarchies The `DateTime` dimension includes a hierarchy on the date which is specified as Year → Month → Day → Time. As per OLAP recommendations, the access to these hierarchical attributes (Year, Month, Day, Time) is not visible to end-users.

Measures The measure groups include `Damage Reimbursement` and `Person`. Damage Reimbursement captures financial data, including Cost, Damage Reimbursement Count, and Average Cost, which are essential for assessing the financial impact of crashes. The Person measure group provides insights into the number and type of individuals involved in crashes, including Person Count, Count of Person Type, and Fatal Crashes.

Key Relationships The cube establishes key relationships between different dimensions. Damage Reimbursement links to Person, Crash, and Vehicle, providing a holistic view of the incident and its impact. It also connects to DateTime, through Crash, thus allowing for a temporal analysis of incidents.

After building and defining the structure, the data was processed and the cube was deployed on the OLAP server, allowing us to run MDX queries and perform data analysis directly within the cube environment.

6 MDX Queries

6.1 Query 2

For assignment 2, we had to show the total damage costs for each location and each month, as well as the total. In the query, we use `WITH MEMBER` to define custom grand totals for both months and locations, aggregating totals across all entries. We used these members since it allowed us to give a clear name to the column, instead of using the default "All", and specify their placement in the result set (in the end instead of the top), rendering it more readable and clear. The `Hierarchize` function is used to correctly order the months, listing the individual months first and followed by the total for the entire year. For locations, we use the Filter function to exclude the All member, ensuring only individual locations are shown, and then we add the grand total for locations which replaces the default All member.

6.2 Query 4

For assignment 4, the objective was to show, for every location, damage costs increase or decrease, percentage-wise, relative to the previous year. In our query we used two `WITH MEMBER`, the first one is to name the damage cost of the previous year, the second one is to create the change in terms of the percentage of damage. To finish we are showing the cost and the damage percentage change on the columns and the crash location in the rows. for the specific year selected we are showing the year 2017.

6.3 Query 6

For assignment 6, in this query the goal was to show for each vehicle type and each year, show the information and the (total) damage costs of the person with the highest reported damage. In this one we selected on the columns only the Total cost which is made of the sum of the **currentmember** of pearson for their cost, then for the rows we have a nested function where we are taking only the **non empty** values generating the **combination** between year and vehicle type to then extract the **top count** of this in terms of each **currentmember** of year and of vehicle type, this **top count** is regarding each person **top 1** and measuring the cost. the result of this nested function will be what we have in our rows.

6.4 Query 7

For assignment 7, the objective was to calculate and display the median and maximum time delays (delta time) between when crashes occurred and were reported for each beat of occurrence. The delta time was not present in the original dataset and was calculated earlier in the

project. This analysis provides insights into typical and extreme reporting delays across beats. We defined a measure, **TotalDelaySeconds** to compute the total delay in seconds by extracting and converting the components (days, hours, minutes, and seconds) from the Difference Between Crash Date And Police Notified field. The median and maximum delays were then calculated using the MEDIAN and MAX functions, respectively. These values were formatted into human-readable strings (e.g., "X days, Y hours, Z minutes, W seconds") for clarity, with null handling to exclude invalid entries. The query displayed beats in descending order of the median delay to prioritize significant results. This query highlights reporting efficiency at a granular level, showing typical delays through the median and identifying extreme outliers via the maximum delay. The results allow for targeted analysis of beats with the longest delays, we belive this might be an useful insight for the insurance company because we detected a big difference between different beats in the reporting behaviour.

6.5 Query 8a

For assignment 8, the goal was to identify the most frequent cause of crashes for each year, calculate the associated total damage costs, and determine the overall most frequent crash cause across all years. To prioritize causes, the analysis weighted the primary crash contributing factor twice as much as the secondary factor. This weighting was computed by calculating the measure **WeightedCauseCount**. The most frequent cause per year was determined using the 'TOPCOUNT' function, which ranked crash causes by their weighted frequency within each year. The total damage costs for the top cause per year were aggregated in the measure **TotalCostPerYearTopCause**, filtering for entries matching the most frequent cause. Similarly, the overall most frequent cause across all years was identified by ranking crash causes using 'TOPCOUNT' without restricting the time context, stored in the measure **MostFrequentCauseOverall**. The query results displayed, for each year, the most frequent crash cause (**MostFrequent CausePerYear**), its total damage costs (**TotalCostPerYearTopCause**), and the overall most frequent crash cause (**MostFrequentCauseOverall**). This provided a concise yet detailed view of crash patterns and their financial impact, both annually and across all years.

7 Dashboards

7.1 Geographical Dashboard

For Assignment 9, the dashboard, illustrated in Figure 1, was designed to showcase the geographical distribution of total damage costs by vehicle category, offering interactive insights into spatial and categorical trends. This dashboard is equipped with two primary filters on the left-hand side, allowing users to refine their analysis by selecting specific beats of occurrence and vehicle types to focus on. These filters enable stakeholders to dynamically adjust the displayed data to suit their investigative focus.

The dashboard layout is divided into two primary columns of visualizations. The middle column begins with a treemap at the top, highlighting the total damage costs for the selected vehicle categories. This visual provides a quick, hierarchical overview of cost distributions, enabling users to grasp relative impacts at a glance. Beneath the treemap, a complementary bar plot displays the same data, withh a beat of occurrence comparison context, in a more detailed, comparative format, helping to underline the specific vehicle categories with higher damage costs for each beat of occurrence.

On the right-hand side, the map visualization provides spatial insights, focusing on the Chicago area by default when all beats of occurrence are selected. This map dynamically updates to reflect the chosen beats, allowing users to explore specific regions of interest and identify areas with higher concentration of damage costs.

The dashboard emphasizes interactivity and user-driven exploration. Users can manipulate filters to adapt the displayed data, enabling nuanced analyses of relationships between geographical locations and vehicle-related damage costs. This adaptability ensures the dashboard is not only informative but also highly customizable to meet diverse analytical needs.

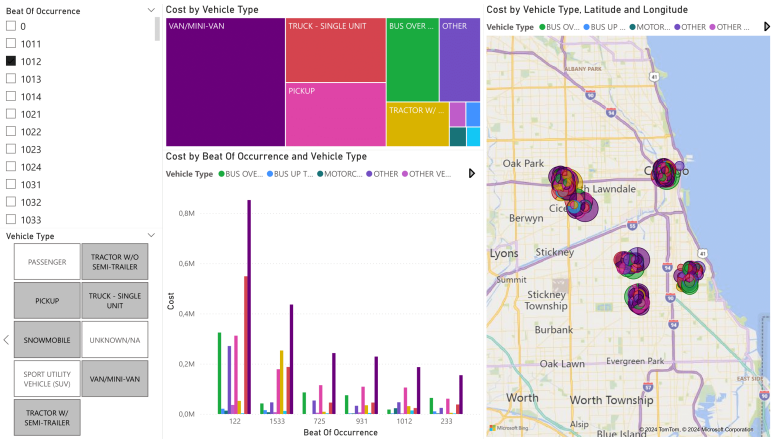

Figure 1: Geographical distribution of damage costs for vehicle categories Dashboard

7.2 Streets Dashboard

For assignment 10, the dashboard in figure 2 is built to show insights of the cost and damage reimbursement subject to the street; this can help the insurance company locate and analyze the mobility dynamics inside of the city when it is going to grant an insurance agreement or when it's going to pay for damages. Regarding the visualizations on the dashboard, we can see a conglomerate of relations between the cost and the streets where the company can gain insights by looking at total cost metrics, the average cost increase by years, the count of the damage reimboursements depending on the street, and among other, a street map showing the top 10 streets with the highest cost in chicago, these kind of visualizations put together can help determining new policies for the insurance company.

Figure 2: Streets Dashboard

7.3 People Dashboard

For assignment 11, the dashboard in Figure 3 was designed to provide actionable information about crash participants. At the top left, the pie chart titled `Proportion of Crash Participants by Physical Condition` identifies the impact of factors such as impairment due to drugs, alcohol, or medical conditions, helping business analysts understand the role of human factors in accidents and tailor risk models. The map visualizes the geographical concentrations of crashes, allowing insurers to pinpoint high-risk regions for premium adjustments. On the bottom left, the bar chart provides insight into crash costs by age group and vision impairments, revealing that younger drivers (under 18) and those with specific vision conditions incur higher costs and, as such, help identify high-risk demographics for targeted pricing. The treemap `Driver Action by Injury Breakdown` uncovers how specific driver behaviors (e.g., failure to act, distracted driving) correlate with injuries, thus showing a potential need to adjust premiums for policyholders exhibiting these patterns.

The central KPIs, participants in `Participants in Crashes, Total Damage Cost, Fatal Crashes, and Average Damage Cost` provide an immediate overview of the volume of claims and financial exposure, which is crucial for forecasting payments and ensuring sufficient reserves.

The dashboard offers three types of slicers— `Age, Sex, and Type`—that dynamically update all the plots, allowing the end user to perform detailed analysis effortlessly. The use of a cohesive blue theme was chosen to provide an aesthetically pleasing and coherent look, enhancing the overall user experience and ensuring clarity in data visualization.

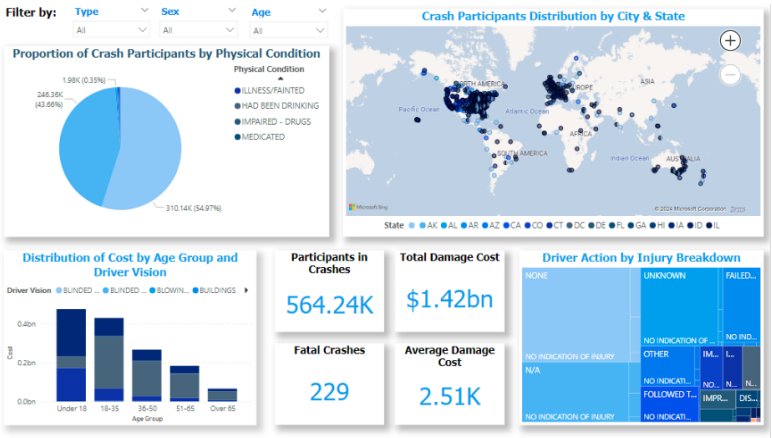

Figure 3: People Dashboard