Australian Road Safety Enforcement:
Interactive Data Explorer

Data Visualisation Project Design Book

Team: DV_Group49
Mercury Website Link:
https://mercury.swin.edu.au/cos30045/s105684881/project-dv_group49
GitHub Repository:
https://github.com/COS30045-Inti-Subang/data-visualisation-project-dvgroup49

Team Members:

Student Name	GitHub	SUT ID	INTI ID
Min Thu Kyaw Khaung (Markus)	markus1525	105684881	J23040027
Thet Hein Aung (Damian)	sinister-damian	105684373	J23040150
Sai Lyan Hein (Jack)	sailyanhein	105684386	J23040369

Section: C2
Year and Semester: 2025, Semester 4

Word Count:

Table of Contents

1. Introduction

1.1 Background and Motivation

Target Audience and Key Tasks

Our visualisation is designed for five distinct user groups, each with specific information needs:

1. **Policy Makers & Government Officials**

   Task: Identify high-risk areas and evaluate enforcement effectiveness to inform road safety legislation
   Need: Clear comparison between enforcement intensity and safety outcomes by state/region

2. **Road Safety Analysts & Researchers**

   Task: Study patterns in traffic violations and relationships between enforcement, demographics, and crash outcomes
   Need: Detailed drill-down capability, ability to filter by age, offence type, and location

3. **Law Enforcement Agencies**

   Task: Understand enforcement patterns and optimize resource allocation
   Need: Geographic distribution of fines, identification of enforcement gaps

4. **General Public & Media**

   Task: Understand road safety issues in their local area and hold officials accountable
   Need: Intuitive comparisons, local context (state/region level)

5. **Transport Safety Advocates**

   Task: Provide evidence-based arguments for safer roads and identify vulnerable populations
   Need: Clear visualization of at-risk groups, enforcement gaps, fatality correlations

Problem Being Addressed

The Bureau of Infrastructure and Transport Research Economics (BITRE) collects extensive road safety enforcement data. Since 2023, this includes fine-grained details (specific locations, offender age groups, offence types). However, this new data has not been integrated into publicly accessible visualizations, creating a critical gap:

- **Raw data exists** but is not actionable for stakeholders
- **Current public dashboard is outdated** and lacks new detailed metrics
- **No visual correlation** between enforcement and safety outcomes is currently shown
- **Decision-makers lack evidence** to justify resource allocation strategies

Importance of the Visualisation

Road safety is a critical public health issue. By visualizing new enforcement data alongside fatality outcomes:

- **Identifies patterns** hidden in spreadsheets (e.g., which age groups drive violations vs. which are at-risk in crashes)
- **Enables evidence-based policy** (e.g., should enforcement focus on high-fatality regions or proactive prevention?)
- **Improves transparency** by making government data publicly accessible
- **Supports targeted interventions** (e.g., is enforcement concentrated in major cities while rural areas have high fatalities?)
- **Evaluates strategy effectiveness** (does more enforcement correlate with fewer deaths?)

1.2 Visualisation Purpose

Core Research Questions

Our dashboard enables users to answer six interconnected questions about road safety in Australia:

Enforcement Questions (from police enforcement data 2020–2024):

1. **What are the most common offences being fined?**
   Understanding which violations are most frequently enforced helps identify priority areas for road safety campaigns.
2. **Where is enforcement concentrated?**
   Which states enforce most in major cities vs remote areas? This reveals potential geographic gaps in enforcement coverage.
3. **Is there a relationship between age and specific offences?**
   Which age-offence combinations are most problematic? This enables targeted demographic interventions.

Outcomes Questions (from fatality data 2020–2024):

4. **How have fatalities changed from 2020–2024?**
   Are there seasonal or yearly trends? This helps evaluate whether road safety is improving over time.
5. **Which states have highest fatalities?**
   Is there geographic clustering? This identifies regions requiring urgent attention.
6. **What types of crashes kill the most people?**
   Who is most affected (drivers, pedestrians, motorcyclists, etc.)? This informs vehicle safety standards and infrastructure improvements.

Comparison Question:

7. **Is there a relationship between enforcement intensity and fatality rates?**
   Do states with more fines have fewer deaths? This evaluates enforcement effectiveness.

Key Benefits

- **Accessibility:** Transforms massive spreadsheets into intuitive interactive visualizations accessible to non-technical users
- **Pattern Discovery:** Reveals relationships between age, offence, location, and crash outcomes not obvious in raw data
- **Data-Driven Decision Making:** Provides evidence for resource allocation, intervention targeting, and policy development
- **Comparative Analysis:** Enables direct comparison of enforcement intensity vs. safety outcomes to evaluate strategy effectiveness
- **Public Transparency:** Makes government data publicly accessible, supporting informed community discourse
- **Impact on Decision-Making:** Shifts focus from "How much enforcement?" to "Where is enforcement working? Where are gaps?"

1.3 Project Schedule

Our work is planned in weekly milestones to avoid last-minute rushes and ensure quality deliverables.

Week	Milestone/Deliverable	Tasks
Week 8	Team Formation	Form team, set up GitHub, review project brief
Week 9	Data Collection	Download BITRE data, start KNIME workflow, draft introduction
Week 10	**Stand Up 1**	Complete data cleaning, exploratory analysis, calculate statistics. Submit: Draft intro, KNIME file, GitHub link
Week 11	**Stand Up 2**	Create wireframes, design charts, plan interactions. Submit: 50% design book and website, updated datasets
Week 12	**Stand Up 3**	Code D3 visualizations, implement interactions. Submit: 75% Design Book and website, latest version of dataset
Week 13	Final Submission	Polish design book, finalize website, complete all documentation. Submit: Completed Design Book, working website, all required files

2. Data

2.1 Data Source and Governance

Data Source Identification

We are collecting our data from the **Australian Government's National Road Safety Data Hub**, which is managed by the Bureau of Infrastructure and Transport Research Economics (BITRE).

Enforcement Data:
https://datahub.roadsafety.gov.au/safe-systems/safe-road-use/police-enforcement

Fatalities Data:
https://datahub.roadsafety.gov.au/progress-reporting/monthly-road-deaths

Collection Method: We have downloaded the data as two primary tabular (CSV) files from the data hub.

Dataset Type: Both datasets are structured table data, with rows representing individual records and columns representing attributes.

Core Datasets Used:

1. **Core Dataset 1 (Enforcement): police_enforcement_2024_fines.csv**
   (sourced from the 'Police Enforcement' link)

2. **Core Dataset 2 (Outcomes): Australian Road Deaths Database Fatalities.csv**
   (sourced from the 'Monthly Road Deaths' link)

Data Summary Table

Dataset	File Name	Total Records	Time Period	Update Frequency
Police Enforcement	police_enforcement_2024_fines.csv	12,179	2008–2024	Annual
Road Deaths Database	Australian_Road_Deaths_Database_Fatalities.csv	57,852	1989–2025	Monthly
Processed (Enforcement)	ProcessedFines.csv	11,960	2020–2024	N/A (cleaned)
Processed (Fatalities)	ProcessedFatalities.csv	6,941	2020–2025	N/A (cleaned)

Data Governance Considerations

Collection Process:

- **Enforcement data:** Aggregated from police citations and automated detection systems across all Australian states/territories
- **Fatality data:** Reported by state road authorities and compiled by BITRE from police crash reports
- Both datasets undergo quality assurance checks before public release

Data Quality Assessment:

- **Completeness**: Enforcement data ~98% complete, fatality data ~99% complete (some missing demographic fields)
- **Accuracy**: Government-sourced and validated, no known systematic biases reported
- **Timeliness**: Published annually with monthly updates for fatalities
- **Known Issues**:
    - Some early records (pre-2020) have inconsistent age groupings
    - Fatality data may include delayed reporting (deaths recorded after crash occurred)
    - Enforcement data reflects detection capability (e.g., camera placement, enforcement resources)

Security, Privacy, and Ethical Considerations:

- **Privacy**: Data is already aggregated/anonymized (no individual identifiers), complies with privacy regulations
- **Ethical Use**: We use this data to improve safety, not to stigmatize any demographic group
- **Bias Awareness**: Enforcement patterns reflect deployment strategy, not offence rates, visualization acknowledges this limitation
- **Responsible Reporting**: Findings contextualized to avoid misinterpretation

Data-Question Alignment

Our dataset directly supports all six research questions:

Question	Data Source	Relevant Attributes	Supports Answer?
Q1: Fines by offence type	Enforcement	OFFENCE_TYPE, FINES_COUNT	Yes
Q2: Geographic enforcement	Enforcement	JURISDICTION, LOCATION, FINES_COUNT	Yes
Q3: Age & offence relationship	Enforcement	AGE_GROUP, OFFENCE_TYPE, FINES_COUNT	Yes
Q4: Fatality trends	Fatalities	YEAR, MONTH, FATALITY_COUNT	Yes
Q5: Fatalities by state	Fatalities	STATE, FATALITY_COUNT	Yes
Q6: Crash types & victims	Fatalities	CRASH_TYPE, ROAD_USER_TYPE, FATALITY_COUNT	Yes
Q7: Fines vs Fatalities	Both	JURISDICTION/STATE, FINES, FATALITIES	Yes

2.2 Data Processing and Analysis

Key Attributes and Data Types

Enforcement Data (ProcessedFines.csv):

Attribute	Data Type	Example	Description
YEAR	Ordinal	2024	Calendar year
JURISDICTION	Categorical (Nominal)	NSW	State/territory code (8 values)
LOCATION	Categorical (Ordinal)	Major Cities	Remoteness classification (5 categories)
AGE_GROUP	Ordinal	25-34	Offender age bracket (7 categories)
METRIC	Categorical (Nominal)	Speed Fines	Traffic offence category (4 types)
DETECTION_METHOD	Categorical (Nominal)	Auto Detection Camera	How fine was detected (2 categories)
FINES	Quantitative (Ratio)	45,230	Total fines issued
CHARGES	Quantitative (Ratio)	431	Total charges filed

Fatalities Data (ProcessedFatalities.csv):

Attribute	Data Type	Example	Description
Crash_ID	Nominal	120221303786	Unique identifier for each crash
State	Categorical (Nominal)	NSW	State/territory code (8 values)
Month	Ordinal	3	Month number (1-12)
Month_Name	Ordinal	March	Month name (text)
Year	Ordinal	2024	Calendar year
Crash_Type	Categorical (Nominal)	Single	Type of crash (2 categories)
Road_User	Categorical (Nominal)	Driver	Type of road user (5 categories)
Gender	Categorical (Nominal)	Male	Gender of victim (Male/Female/?)
AgeGroup	Ordinal	25-34	Victim age bracket (7 categories + Unknown)

Data Cleaning Process

Step 1: Identifying Missing Values & Inconsistencies

Enforcement Data:

- Missing age values (1.2%) → Investigated as "Unknown" or actual missing
- Malformed offence descriptions → Standardized using lookup table
- Inconsistent state abbreviations → Standardized to 2-letter codes

Fatalities Data:

- Gender marked as "Unknown" or blank → Standardized to "?"
- Truck involvement coded as "-9" for missing → Standardized to "?"
- Time values corrupted as "99:99:99" → Replaced with "00:00:01" (unknown time)

Step 2: Normalization & Transformation

Age Group Standardization (crucial for cross-dataset comparison):

- Raw enforcement ages grouped: 0-16, 17-25, 26-39, 40-64, 65 and over
- Raw fatality ages regrouped to match: 0-16, 17-25, 26-39, 40-64, 65 and over
- Created unified age categories across both datasets for fair comparison

Detection Method Consolidation (Enforcement):

- Original 8 detection types merged into 2 categories:
  - **"Auto Detection Camera"**: Average speed camera, Fixed camera, Mobile camera, Red light camera, Fixed or mobile camera
  - **"Manual Detection"**: Police issued, Not applicable

Offence Type Grouping (Enforcement):

- Standardized naming:
  - speed_fines → Speed Fines
  - mobile_phone_use → Mobile Phone Use
  - non_wearing_seatbelts → Non Wearing Seatbelts
  - unlicensed_driving → Unlicensed Driving

Step 3: Handling Missing Data

Enforcement Data:

- Rows with missing JURISDICTION → Dropped (0.3% of data)
- Rows with missing AGE_GROUP → Coded as "Unknown" to preserve fine counts
- Rows with missing OFFENCE_TYPE → Coded as "Other"

Fatalities Data:

- Rows with missing STATE → Dropped (0.1% of data)
- Rows with missing AGE_GROUP → Coded as "Unknown" to preserve fatality counts

Step 4: Data Filtering

- Filtered both datasets to years 2020 onwards (for temporal consistency)
- **Rationale**: New fine-grained enforcement data only reliable from 2023 onwards, restricting to 2020-2024/5 allows 4 to 5-year comparison and focuses on recent trends most relevant to current policy

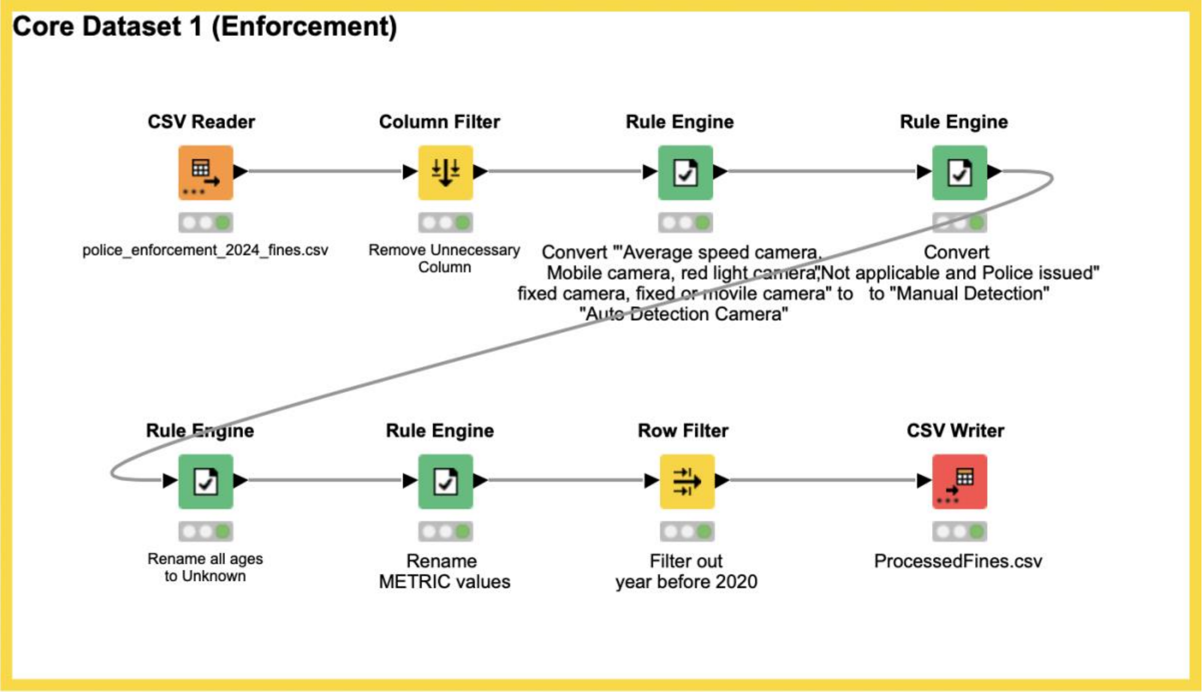
Step 5: Duplicate Removal

- No exact duplicates found in enforcement data
- Cross-checked for logical duplicates (same state + offence + age + date) → None detected
- Fatality data: Each record represents unique crash victim (no duplicates by design)

KNIME Workflow Implementation - Data Cleaning (Stand Up 1 Completed)

Our data processing was completed using **KNIME Analytics Platform**. Below are descriptions of our workflows:

Core Dataset 1: Enforcement Data Cleaning Workflow


Core Dataset 1 (Enforcement)
CSV Reader
police_enforcement_2024_fines.csv
Column Filter
Remove Unnecessary Column
Rule Engine
Convert "'Average speed camera, Mobile camera, red light camera, fixed camera, fixed or movile camera" to "Auto Detection Camera"
Rule Engine
Convert "Not applicable and Police issued" to "Manual Detection"
Rule Engine
Rename all ages to Unknown
Rule Engine
Rename METRIC values
Row Filter
Filter out year before 2020
CSV Writer
ProcessedFines.csv

Process Flow:

1. **CSV Reader** → police_enforcement_2024_fines.csv
   - Input: Raw enforcement data with all columns (2008-2024)
2. **Column Filter** → Remove Unnecessary Columns
   - Remove: start_date, end_date, location (detailed text field - kept LOCATION for remoteness)
   - Keep: YEAR, JURISDICTION, LOCATION, AGE_GROUP, METRIC, DETECTION_METHOD, FINES, CHARGES
3. **Rule Engine #1** → Convert Detection Methods
   - Rule: IF detection_method IN ("Average speed camera", "Mobile camera", "Red light camera", "Fixed camera", "Fixed or mobile camera") THEN "Auto Detection Camera"
   - Otherwise: "Manual Detection"
4. **Rule Engine #2** → Convert "Not applicable" status

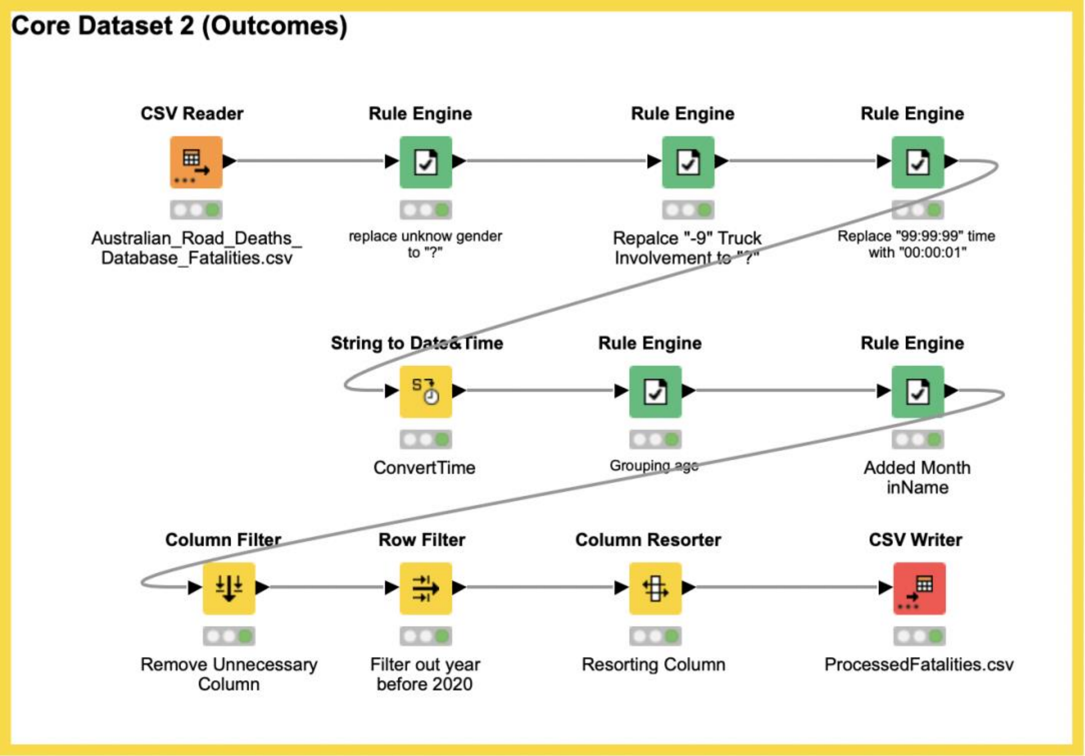
    - Rule: IF applicable_status = "Not applicable" AND detection = "Police issued" THEN "Manual Detection"

5. **Rule Engine #3** → Rename All ages to Unknown
    - Rule: IF AGE_GROUP = " All ages " THEN "Unknown"
6. **Rule Engine #4** → Standardize METRIC values
    - Rule: IF METRIC = "non_wearing_seatbelts" THEN "Non Wearing Seatbelts"
    - Rule: IF METRIC = "speed_fines" THEN "Speed Fines"
    - Rule: IF METRIC = "unlicensed_driving" THEN "Unlicensed Driving"
    - Rule: IF METRIC = "mobile_phone_use" THEN "Mobile Phone Use"
7. **Row Filter** → Filter Out Years Before 2020
    - Condition: YEAR >= 2020
    - Result: 11,960 records retained
8. **CSV Writer** → ProcessedFines.csv
    - Output: Clean enforcement data ready for aggregation

Data Quality Validation (Stand Up 1):

- ✓ Records count: 11,960 (from original ~12,179 filtered to 2020–2024)
- ✓ No duplicate records detected
- ✓ All required columns present
- ✓ Age standardization: 5 age brackets + "Unknown"
- ✓ Detection methods: 2 categories consolidated
- ✓ METRIC values: 4 standardized categories

Core Dataset 2: Fatalities Data Cleaning Workflow


Core Dataset 2 (Outcomes)
CSV Reader
Australian_Road_Deaths_ Database_Fatalities.csv
Rule Engine
replace unknow gender to "?"
Rule Engine
Repalce "-9" Truck Involvement to "?"
Rule Engine
Replace "99:99:99" time with "00:00:01"
String to Date&Time
ConvertTime
Rule Engine
Grouping age
Rule Engine
Added Month inName
Column Filter
Remove Unnecessary Column
Row Filter
Filter out year before 2020
Column Resorter
Resorting Column
CSV Writer
ProcessedFatalities.csv

Process Flow:

1. **CSV Reader** → Australian_Road_Deaths_Database_Fatalities.csv
   - Input: Raw fatality data with historical records (1989–2025)
2. **Rule Engine #1** → Replace Unknown Gender
   - Rule: IF gender = "Unknown" OR gender = "" THEN "?"
   - Purpose: Standardize missing values
3. **Rule Engine #2** → Replace Missing Truck Involvement
   - Rule: IF truck_involvement = "-9" THEN "?"
   - Purpose: Handle missing categorical values consistently
4. **Rule Engine #3** → Replace Malformed Time Values
   - Rule: IF time = "99:99:99" OR time is invalid THEN "00:00:01"
   - Purpose: Fix corrupted datetime entries
5. **String to Date&Time** → Convert Time Format
   - Converts time string to proper Date&Time object
   - Enables temporal filtering and aggregation
6. **Rule Engine #4** → Group Ages into Standard Brackets

   - Apply age categorization to create AgeGroup column:
     - 0-16, 17-25, 26-39, 40-64, 65+
   - Unmatched ages → "Unknown"

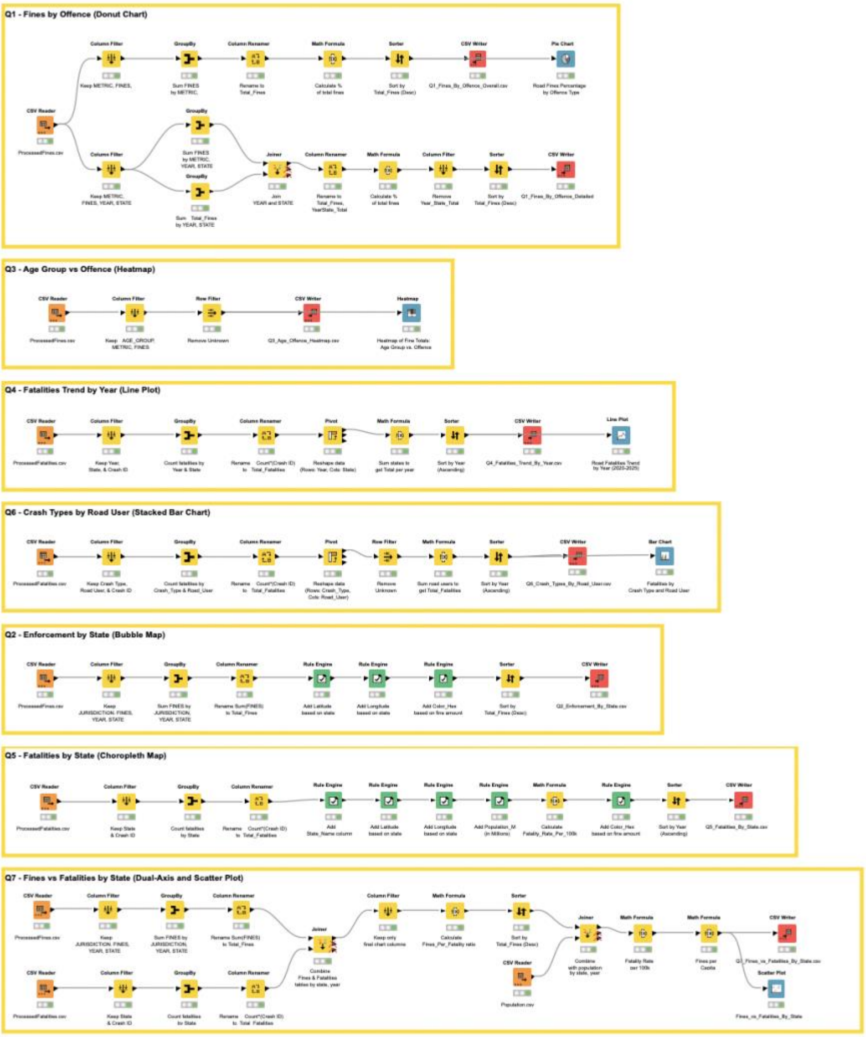
7. **Rule Engine #5** → Add Month Name
   - Derives month name from month number for readability
   - IF Month = 1 THEN "January", IF Month = 2 THEN "February", etc.
8. **Column Filter** → Remove Unnecessary Columns
   - Keep: Crash_ID, State, Month, Month_Name, Year, Crash_Type, Road_User, Gender, AgeGroup
   - Remove: Detailed crash location, vehicle details, speed limits, etc.
9. **Row Filter** → Filter Out Years Before 2020
   - Condition: YEAR >= 2020
   - Result: 6,941 records retained
10. **Column Resorter** → Logical Column Order
    - Order: Crash_ID, State, Month, Month_Name, Year, Crash_Type, Road_User, Gender, AgeGroup
11. **CSV Writer** → ProcessedFatalities.csv
    - Output: Clean fatality data ready for aggregation

Data Quality Validation (Stand Up 1):

- ✓ Records count: 6,941 (from original ~57,852 filtered to 2020–2024)
- ✓ Age standardization: 7 brackets + "Unknown"
- ✓ Missing value handling: ~1% of records retained as "Unknown" vs. dropped
- ✓ Temporal data fixed: Time format validated
- ✓ No logical duplicates detected
- ✓ All required columns present

KNIME Aggregation Workflows for Visualizations (Stand Up 2 - In Progress)

For each of our 7 research questions, we created specific aggregated datasets optimized for D3.js visualization.


Q1 - Fines by Offence (Donut Chart)
CSV Reader
ProcessedFines.csv
Column Filter
Keep METRIC, FINES,
GroupBy
Sum FINES by METRIC,
Column Renamer
Rename to Total_Fines
Math Formula
Calculate % of total fines
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q1_Fines_By_Offence_Overall.csv
Pie Chart
Road Fines Percentage by Offence Type
Column Filter
Keep METRIC, FINES, YEAR, STATE
GroupBy
Sum FINES by METRIC, YEAR, STATE
GroupBy
Sum Total_Fines by YEAR, STATE
Joiner
Join YEAR and STATE
Column Renamer
Rename to Total_Fines, YearState_Total
Math Formula
Calculate % of total fines
Column Filter
Remove Year_State_Total
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q1_Fines_By_Offence_Detailed
Q3 - Age Group vs Offence (Heatmap)
CSV Reader
ProcessedFines.csv
Column Filter
Keep AGE_GROUP, METRIC, FINES
Row Filter
Remove Unknown
CSV Writer
Q3_Age_Offence_Heatmap.csv
Heatmap
Heatmap of Fine Totals: Age Group vs. Offence
Q4 - Fatalities Trend by Year (Line Plot)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep Year, State, & Crash ID
GroupBy
Count fatalities by Year & State
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Pivot
Reshape data (Rows: Year, Cols: State)
Math Formula
Sum states to get Total per year
Sorter
Sort by Year (Ascending)
CSV Writer
Q4_Fatalities_Trend_By_Year.csv
Line Plot
Road Fatalities Trend by Year (2020-2025)
Q6 - Crash Types by Road User (Stacked Bar Chart)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep Crash Type, Road User, & Crash ID
GroupBy
Count fatalities by Crash_Type & Road_User
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Pivot
Reshape data (Rows: Crash_Type, Cols: Road_User)
Row Filter
Remove Unknown
Math Formula
Sum road users to get Total_Fatalities
Sorter
Sort by Year (Ascending)
CSV Writer
Q6_Crash_Types_By_Road_User.csv
Bar Chart
Fatalities by Crash Type and Road User
Q2 - Enforcement by State (Bubble Map)
CSV Reader
ProcessedFines.csv
Column Filter
Keep JURISDICTION, FINES, YEAR, STATE
GroupBy
Sum FINES by JURISDICTION, YEAR, STATE
Column Renamer
Rename Sum(FINES) to Total_Fines
Rule Engine
Add Latitude based on state
Rule Engine
Add Longitude based on state
Rule Engine
Add Color_Hex based on fine amount
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q2_Enforcement_By_State.csv
Q5 - Fatalities by State (Choropleth Map)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep State & Crash ID
GroupBy
Count fatalities by State
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Rule Engine
Add State_Name column
Rule Engine
Add Latitude based on state
Rule Engine
Add Longitude based on state
Rule Engine
Add Population_M (in Millions)
Math Formula
Calculate Fatality_Rate_Per_100k
Rule Engine
Add Color_Hex based on fine amount
Sorter
Sort by Year (Ascending)
CSV Writer
Q5_Fatalities_By_State.csv
Q7 - Fines vs Fatalities by State (Dual-Axis and Scatter Plot)
CSV Reader
ProcessedFines.csv
Column Filter
Keep JURISDICTION, FINES, YEAR, STATE
GroupBy
Sum FINES by JURISDICTION, YEAR, STATE
Column Renamer
Rename Sum(FINES) to Total_Fines
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep State & Crash ID
GroupBy
Count fatalities by State
Column Renamer
Rename Count*(Crash ID) to Total Fatalities
Joiner
Combine Fines & Fatalities tables by state, year
Column Filter
Keep only final chart columns
Math Formula
Calculate Fines_Per_Fatality ratio
Sorter
Sort by Total_Fines (Desc)
CSV Reader
Population.csv
Joiner
Combine with population by state, year
Math Formula
Fatality Rate per 100k
Math Formula
Fines per Capita
CSV Writer
Q7_Fines_vs_Fatalities_By_State.csv
Scatter Plot
Fines_vs_Fatalities_By_State

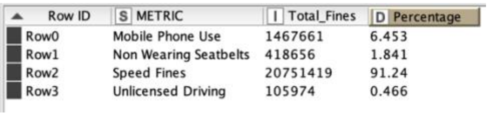
Aggregation Strategy:

Each workflow follows this pattern:

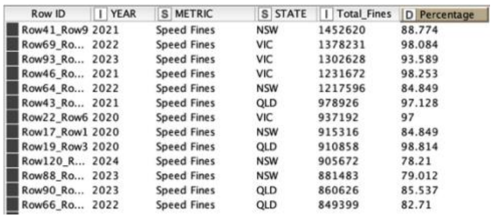
```
[ProcessedFines/Fatalities]
  ↓
[Column Filter] - Keep only needed columns
  ↓
[GroupBy Node] - Group by relevant dimensions
  ↓
[Math Formula / Rule Engine] - Calculate derived metrics if needed
  ↓
[Sorter Node] - Sort for visualization
  ↓
[CSV Writer] - Export as CSV for D3.js
```

Q1: Fines by Offence Type (Donut Chart)

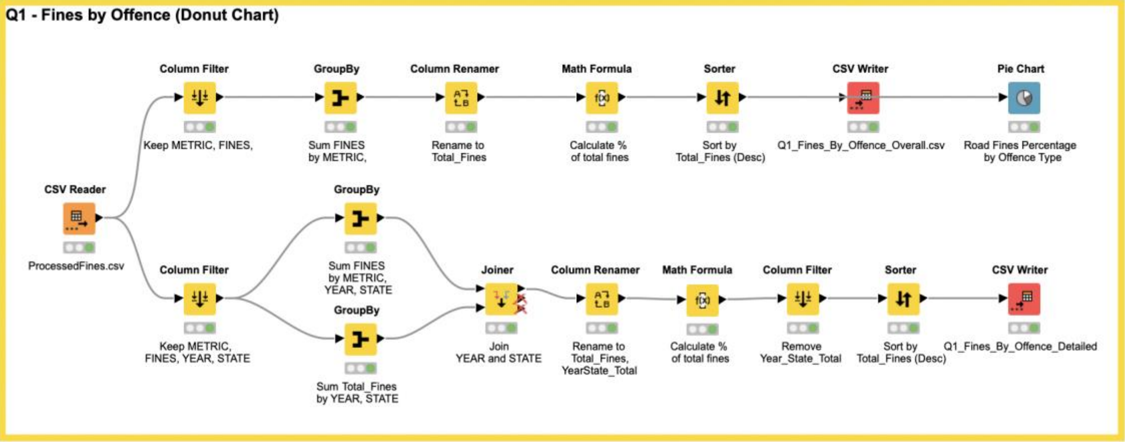
- **Input**: ProcessedFines.csv
- **Process**: Column Filter → GroupBy (METRIC) → Math Formula (calculate %) → Sorter → CSV Writer
- **Output**: Q1_Fines_By_Offence_Overall.csv (4 rows) + Q1_Fines_By_Offence_Detailed.csv (with YEAR, STATE)
- **Columns**: Offence_Type, Total_Fines, Percentage

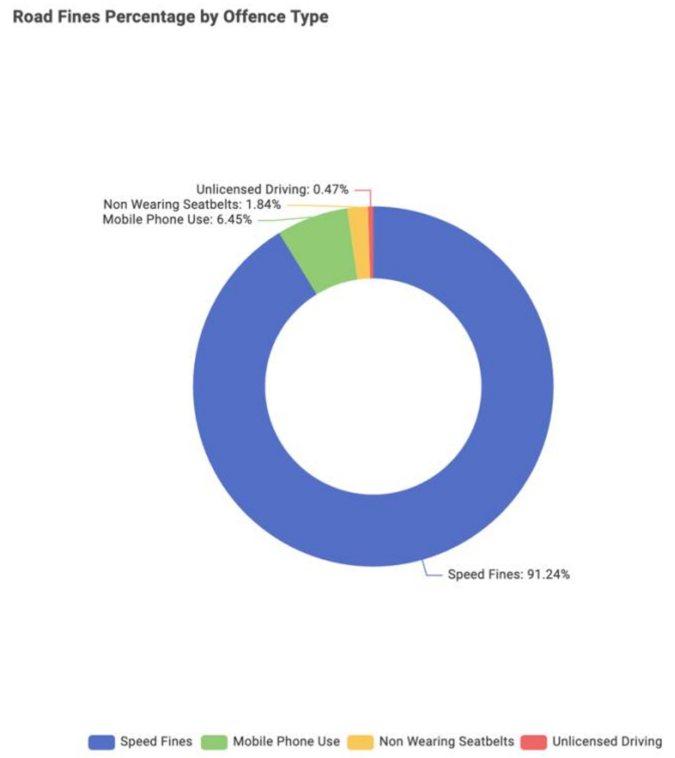


Row ID	S METRIC	I Total_Fines	D Percentage
Row0	Mobile Phone Use	1467661	6.453
Row1	Non Wearing Seatbelts	418656	1.841
Row2	Speed Fines	20751419	91.24
Row3	Unlicensed Driving	105974	0.466



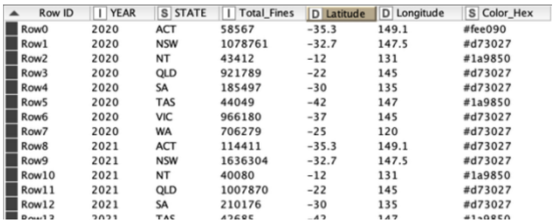
Row ID	I YEAR	S METRIC	S STATE	I Total_Fines	D Percentage
Row41_Row9	2021	Speed Fines	NSW	1452620	88.774
Row69_Ro...	2022	Speed Fines	VIC	1378231	98.084
Row93_Ro...	2023	Speed Fines	VIC	1302628	93.589
Row46_Ro...	2021	Speed Fines	VIC	1231672	98.253
Row64_Ro...	2022	Speed Fines	NSW	1217596	84.849
Row43_Ro...	2021	Speed Fines	QLD	978926	97.128
Row22_Row6	2020	Speed Fines	VIC	937192	97
Row17_Row1	2020	Speed Fines	NSW	915316	84.849
Row19_Row3	2020	Speed Fines	QLD	910858	98.814
Row120_R...	2024	Speed Fines	NSW	905672	78.21
Row88_Ro...	2023	Speed Fines	NSW	881483	79.012
Row90_Ro...	2023	Speed Fines	QLD	860626	85.537
Row66_Ro...	2022	Speed Fines	QLD	849399	82.71


Q1 - Fines by Offence (Donut Chart)
CSV Reader
ProcessedFines.csv
Column Filter
Keep METRIC, FINES,
GroupBy
Sum FINES by METRIC,
Column Renamer
Rename to Total_Fines
Math Formula
Calculate % of total fines
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q1_Fines_By_Offence_Overall.csv
Pie Chart
Road Fines Percentage by Offence Type
Column Filter
Keep METRIC, FINES, YEAR, STATE
GroupBy
Sum FINES by METRIC, YEAR, STATE
GroupBy
Sum Total_Fines by YEAR, STATE
Joiner
Join YEAR and STATE
Column Renamer
Rename to Total_Fines, YearState_Total
Math Formula
Calculate % of total fines
Column Filter
Remove Year_State_Total
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q1_Fines_By_Offence_Detailed

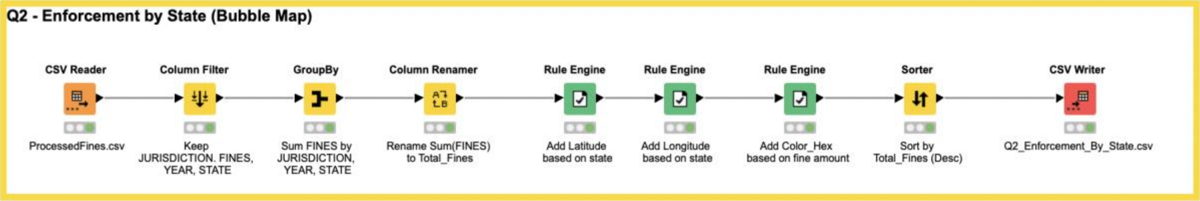


Q2: Enforcement by State (Bubble Map)

- **Input**: ProcessedFines.csv
- **Process**: Column Filter → GroupBy (JURISDICTION, YEAR, STATE) → Rule Engine (add lat/lon) → Sorter → CSV Writer
- **Output**: Q2_Enforcement_By_State.csv
- **Columns**: State, Year, Total_Fines, Latitude, Longitude, Color_Hex



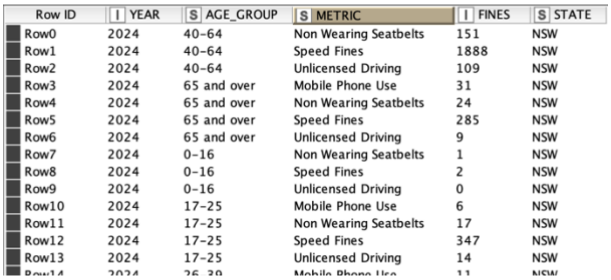
Row ID	I YEAR	S STATE	I Total_Fines	D Latitude	D Longitude	S Color_Hex
Row0	2020	ACT	58567	-35.3	149.1	#fee090
Row1	2020	NSW	1078761	-32.7	147.5	#d73027
Row2	2020	NT	43412	-12	131	#1a9850
Row3	2020	QLD	921789	-22	145	#d73027
Row4	2020	SA	185497	-30	135	#d73027
Row5	2020	TAS	44049	-42	147	#1a9850
Row6	2020	VIC	966180	-37	145	#d73027
Row7	2020	WA	706279	-25	120	#d73027
Row8	2021	ACT	114411	-35.3	149.1	#d73027
Row9	2021	NSW	1636304	-32.7	147.5	#d73027
Row10	2021	NT	40080	-12	131	#1a9850
Row11	2021	QLD	1007870	-22	145	#d73027
Row12	2021	SA	210176	-30	135	#d73027
Row13	2021	TAS	42685	-42	147	#1a9850


Q2 - Enforcement by State (Bubble Map)
CSV Reader
ProcessedFines.csv
Column Filter
Keep JURISDICTION. FINES, YEAR, STATE
GroupBy
Sum FINES by JURISDICTION, YEAR, STATE
Column Renamer
Rename Sum(FINES) to Total_Fines
Rule Engine
Add Latitude based on state
Rule Engine
Add Longitude based on state
Rule Engine
Add Color_Hex based on fine amount
Sorter
Sort by Total_Fines (Desc)
CSV Writer
Q2_Enforcement_By_State.csv

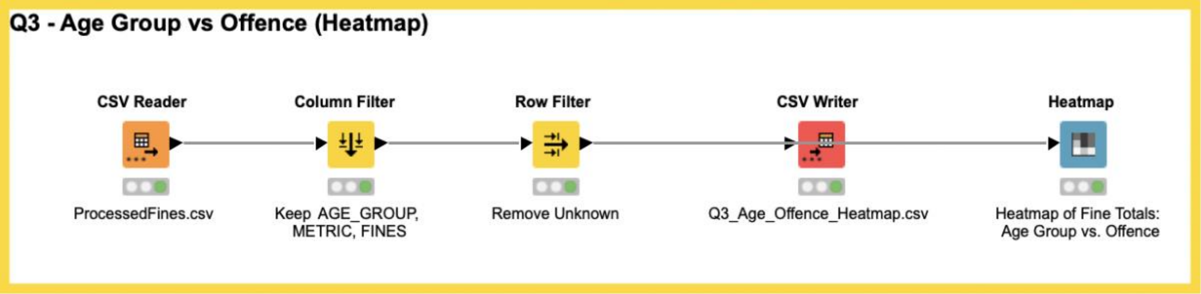
(Pending)

Q3: Age vs Offence Relationship (Heatmap)

- **Input**: ProcessedFines.csv
- **Process**: Column Filter → GroupBy (AGE_GROUP, METRIC, YEAR, STATE) → CSV Writer
- **Output**: Q3_Age_Offence_Heatmap.csv
- **Columns**: Age_Group, Offence_Type, Year, State, Fines

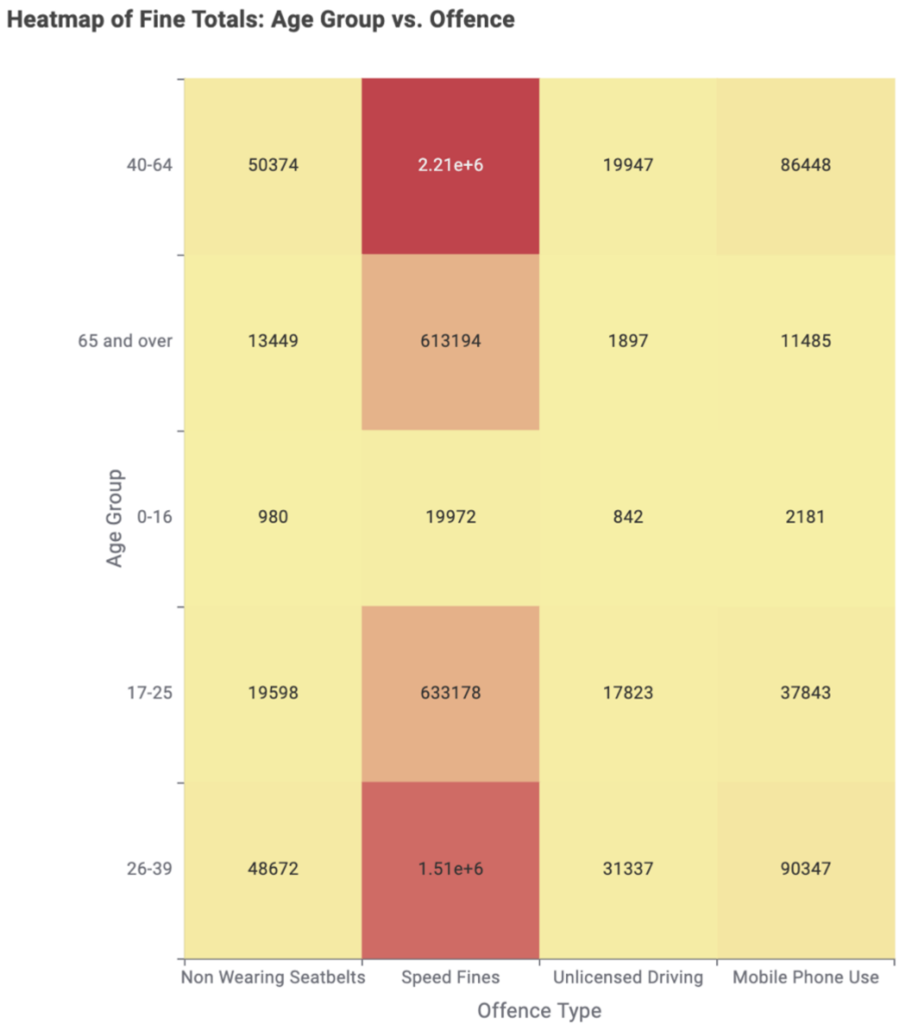


Row ID	I YEAR	S AGE_GROUP	S METRIC	I FINES	S STATE
Row0	2024	40-64	Non Wearing Seatbelts	151	NSW
Row1	2024	40-64	Speed Fines	1888	NSW
Row2	2024	40-64	Unlicensed Driving	109	NSW
Row3	2024	65 and over	Mobile Phone Use	31	NSW
Row4	2024	65 and over	Non Wearing Seatbelts	24	NSW
Row5	2024	65 and over	Speed Fines	285	NSW
Row6	2024	65 and over	Unlicensed Driving	9	NSW
Row7	2024	0-16	Non Wearing Seatbelts	1	NSW
Row8	2024	0-16	Speed Fines	2	NSW
Row9	2024	0-16	Unlicensed Driving	0	NSW
Row10	2024	17-25	Mobile Phone Use	6	NSW
Row11	2024	17-25	Non Wearing Seatbelts	17	NSW
Row12	2024	17-25	Speed Fines	347	NSW
Row13	2024	17-25	Unlicensed Driving	14	NSW
Row14	2024	26-39	Mobile Phone Use	11	NSW



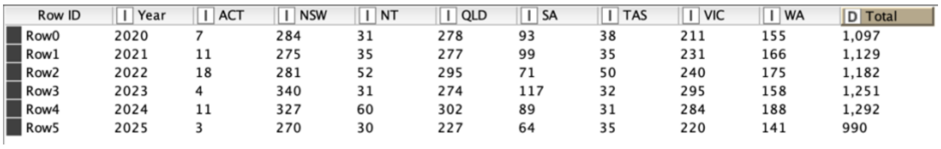
Q3 - Age Group vs Offence (Heatmap)

CSV Reader
ProcessedFines.csv
Column Filter
Keep AGE_GROUP, METRIC, FINES
Row Filter
Remove Unknown
CSV Writer
Q3_Age_Offence_Heatmap.csv
Heatmap
Heatmap of Fine Totals: Age Group vs. Offence

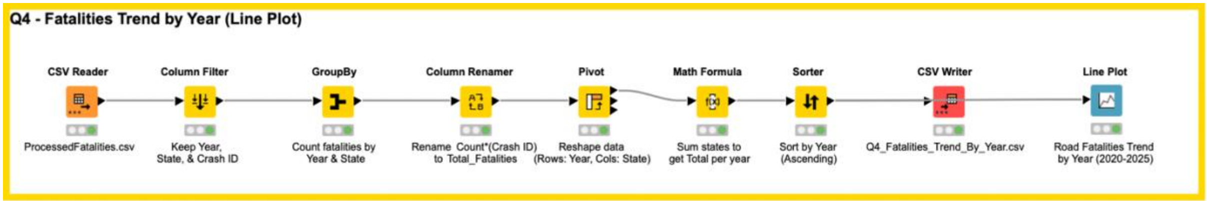


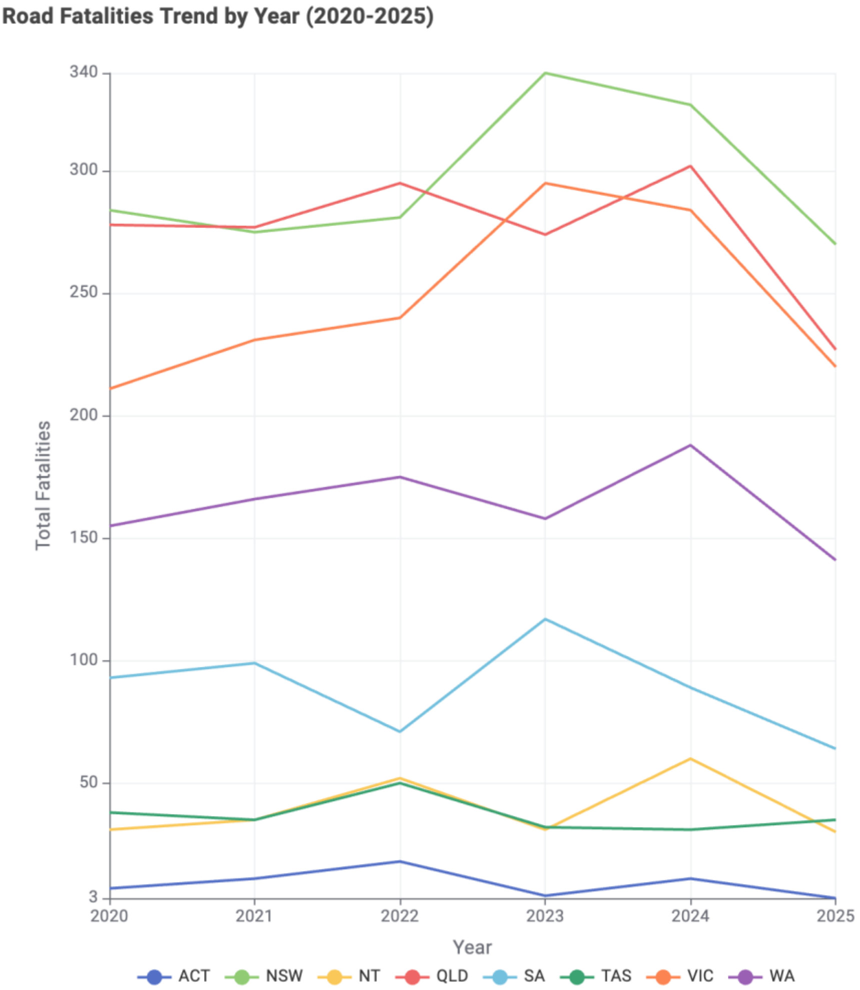
Q4: Fatality Trends by Year (Line Chart)

- **Input**: ProcessedFatalities.csv
- **Process**: Column Filter → GroupBy (Year, State) → Column Renamer → Pivot → Math Formula → Sorter → CSV Writer
- **Output**: Q4_Fatalities_Trend_By_Year.csv
- **Columns**: ACT, NSW, NT, QLD, SA, TAS, VIC, WA, Total



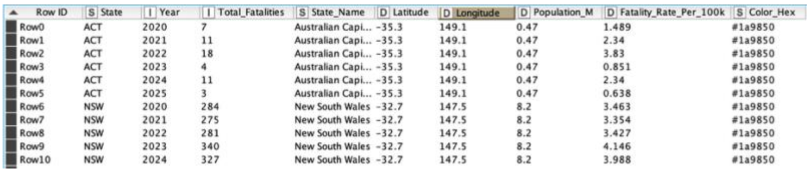
Row ID	I Year	I ACT	I NSW	I NT	I QLD	I SA	I TAS	I VIC	I WA	D Total
Row0	2020	7	284	31	278	93	38	211	155	1,097
Row1	2021	11	275	35	277	99	35	231	166	1,129
Row2	2022	18	281	52	295	71	50	240	175	1,182
Row3	2023	4	340	31	274	117	32	295	158	1,251
Row4	2024	11	327	60	302	89	31	284	188	1,292
Row5	2025	3	270	30	227	64	35	220	141	990


Q4 - Fatalities Trend by Year (Line Plot)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep Year, State, & Crash ID
GroupBy
Count fatalities by Year & State
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Pivot
Reshape data (Rows: Year, Cols: State)
Math Formula
Sum states to get Total per year
Sorter
Sort by Year (Ascending)
CSV Writer
Q4_Fatalities_Trend_By_Year.csv
Line Plot
Road Fatalities Trend by Year (2020-2025)

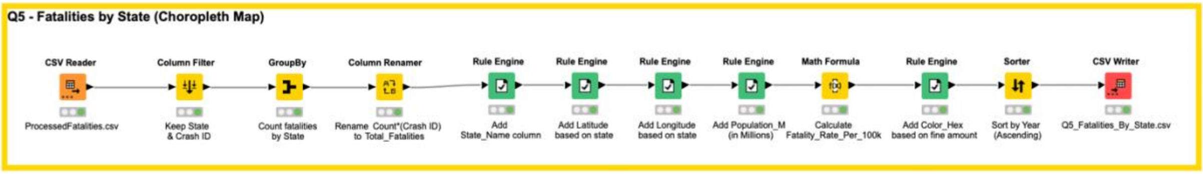


Q5: Fatalities by State (Choropleth Map)

- **Input**: ProcessedFatalities.csv
- **Process**: Column Filter → GroupBy (State) → Column Renamer → 4× Rule Engine (add State_Name, Lat, Lon, Population) → Math Formula (calculate rate) → Rule Engine (add Color_Hex) → Sorter → CSV Writer
- **Output**: Q5_Fatalities_By_State.csv
- **Columns**: State, Year, Total_Fatalities, State_Name, Latitude, Longitude, Population_M, Fatality_Rate_Per_100k, Color_Hex



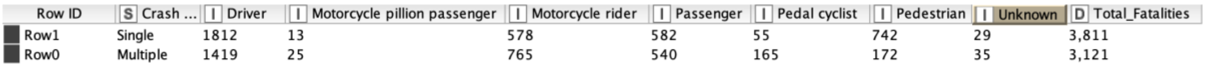
Row ID	S State	I Year	I Total_Fatalities	S State_Name	D Latitude	D Longitude	D Population_M	D Fatality_Rate_Per_100k	S Color_Hex
Row0	ACT	2020	7	Australian Capi...	-35.3	149.1	0.47	1.489	#1a9850
Row1	ACT	2021	11	Australian Capi...	-35.3	149.1	0.47	2.34	#1a9850
Row2	ACT	2022	18	Australian Capi...	-35.3	149.1	0.47	3.83	#1a9850
Row3	ACT	2023	4	Australian Capi...	-35.3	149.1	0.47	0.851	#1a9850
Row4	ACT	2024	11	Australian Capi...	-35.3	149.1	0.47	2.34	#1a9850
Row5	ACT	2025	3	Australian Capi...	-35.3	149.1	0.47	0.638	#1a9850
Row6	NSW	2020	284	New South Wales	-32.7	147.5	8.2	3.463	#1a9850
Row7	NSW	2021	275	New South Wales	-32.7	147.5	8.2	3.354	#1a9850
Row8	NSW	2022	281	New South Wales	-32.7	147.5	8.2	3.427	#1a9850
Row9	NSW	2023	340	New South Wales	-32.7	147.5	8.2	4.146	#1a9850
Row10	NSW	2024	327	New South Wales	-32.7	147.5	8.2	3.988	#1a9850


Q5 - Fatalities by State (Choropleth Map)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep State & Crash ID
GroupBy
Count fatalities by State
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Rule Engine
Add State_Name column
Rule Engine
Add Latitude based on state
Rule Engine
Add Longitude based on state
Rule Engine
Add Population_M (in Millions)
Math Formula
Calculate Fatality_Rate_Per_100k
Rule Engine
Add Color_Hex based on fine amount
Sorter
Sort by Year (Ascending)
CSV Writer
Q5_Fatalities_By_State.csv

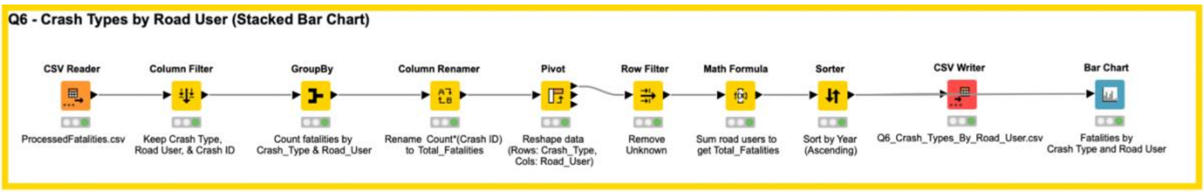
(Pending)

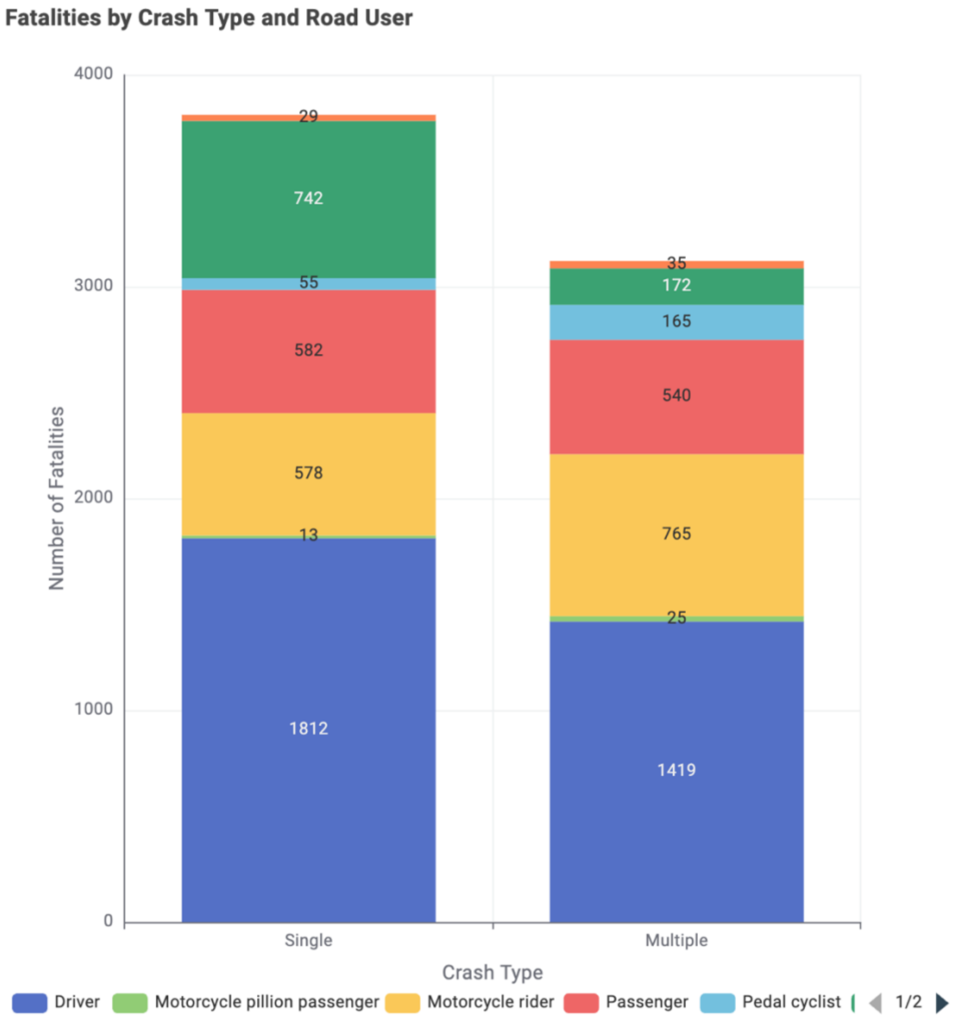
Q6: Crash Types by Road User (Stacked Bar Chart)

- **Input**: ProcessedFatalities.csv
- **Process**: Column Filter → GroupBy (Crash_Type, Road_User) → Column Renamer → Pivot → Row Filter (remove Unknown) → Math Formula (sum road users) → Sorter → CSV Writer
- **Output**: Q6_Crash_Types_By_Road_User.csv
- **Columns**: Crash_Type, Driver, Motorcycle_pillion_passenger, Motorcycle_rider, Passenger, Pedal_cyclist, Pedestrian, Unknown, Total_Fatalities



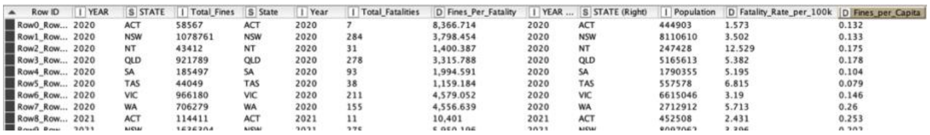
Row ID	S Crash ...	I Driver	I Motorcycle pillion passenger	I Motorcycle rider	I Passenger	I Pedal cyclist	I Pedestrian	I Unknown	D Total_Fatalities
Row1	Single	1812	13	578	582	55	742	29	3,811
Row0	Multiple	1419	25	765	540	165	172	35	3,121


Q6 - Crash Types by Road User (Stacked Bar Chart)
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep Crash Type, Road User, & Crash ID
GroupBy
Count fatalities by Crash_Type & Road_User
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Pivot
Reshape data (Rows: Crash_Type, Cols: Road_User)
Row Filter
Remove Unknown
Math Formula
Sum road users to get Total_Fatalities
Sorter
Sort by Year (Ascending)
CSV Writer
Q6_Crash_Types_By_Road_User.csv
Bar Chart
Fatalities by Crash Type and Road User

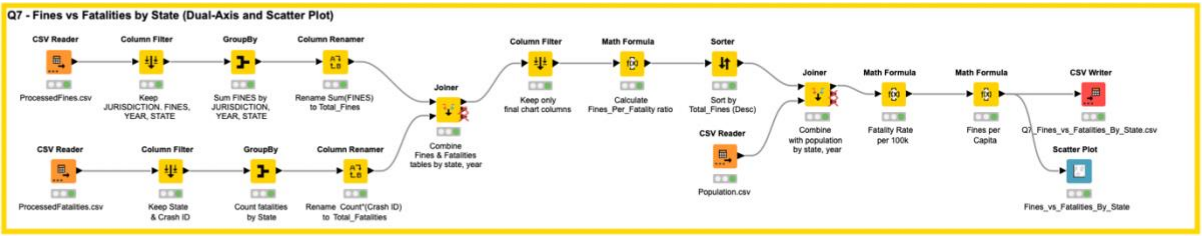


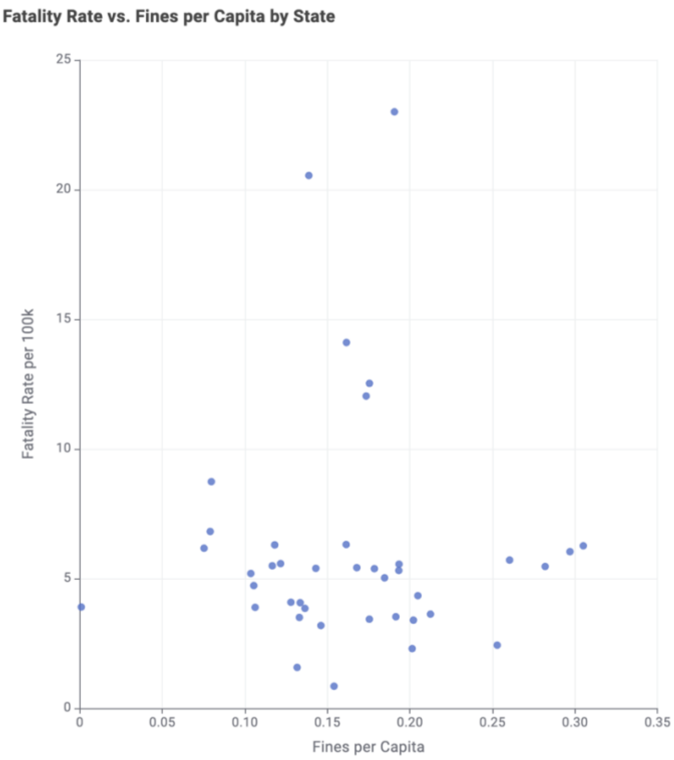
Q7: Fines vs Fatalities by State (Dual-Axis / Scatter Plot)

- **Input**: ProcessedFatalities.csv
- **Process**:
  - **Workflow 1**: Fines → Column Filter → GroupBy (JURISDICTION, YEAR, STATE) → Column Renamer → Sorter → Fines_Table
  - **Workflow 2**: Fatalities → Column Filter → GroupBy (State) → Column Renamer → Sorter → Fatalities_Table
  - **Joiner**: Join on JURISDICTION = State (inner join)
  - Column Filter → Math Formula (Fines_Per_Fatality ratio) → Math Formula (Fatality_Rate per 100k) → Math Formula (Fines per Capita) → Sorter → CSV Writer
- **Output**: Q7_Fines_vs_Fatalities_By_State.csv
- **Columns**: State, Year, Total_Fines, Total_Fatalities, Fines_Per_Fatality



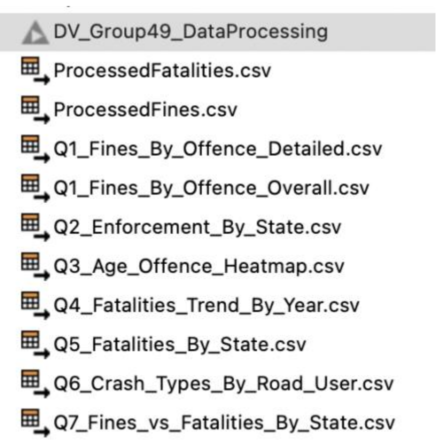
Row ID	YEAR	STATE	Total_Fines	State	Year	Total_Fatalities	Fines_Per_Fatality	YEAR ...	STATE (Right)	Population	Fatality_Rate_per_100k	Fines_per_Capita
Row0_Row...	2020	ACT	58567	ACT	2020	7	8,366.714	2020	ACT	444903	1.573	0.132
Row1_Row...	2020	NSW	1078761	NSW	2020	284	3,798.454	2020	NSW	8110610	3.502	0.133
Row2_Row...	2020	NT	43412	NT	2020	31	1,400.387	2020	NT	247428	12.529	0.175
Row3_Row...	2020	QLD	921789	QLD	2020	278	3,315.788	2020	QLD	5165613	5.382	0.178
Row4_Row...	2020	SA	185497	SA	2020	93	1,994.591	2020	SA	1790355	5.195	0.104
Row5_Row...	2020	TAS	44049	TAS	2020	38	1,159.184	2020	TAS	557578	6.815	0.079
Row6_Row...	2020	VIC	966180	VIC	2020	211	4,579.052	2020	VIC	6615046	3.19	0.146
Row7_Row...	2020	WA	706279	WA	2020	155	4,556.639	2020	WA	2712912	5.713	0.26
Row8_Row...	2021	ACT	114411	ACT	2021	11	10,401	2021	ACT	452508	2.431	0.253


Q7 - Fines vs Fatalities by State (Dual-Axis and Scatter Plot)
CSV Reader
ProcessedFines.csv
Column Filter
Keep JURISDICTION. FINES, YEAR, STATE
GroupBy
Sum FINES by JURISDICTION, YEAR, STATE
Column Renamer
Rename Sum(FINES) to Total_Fines
CSV Reader
ProcessedFatalities.csv
Column Filter
Keep State & Crash ID
GroupBy
Count fatalities by State
Column Renamer
Rename Count*(Crash ID) to Total_Fatalities
Joiner
Combine Fines & Fatalities tables by state, year
Column Filter
Keep only final chart columns
Math Formula
Calculate Fines_Per_Fatality ratio
Sorter
Sort by Total_Fines (Desc)
CSV Reader
Population.csv
Joiner
Combine with population by state, year
Math Formula
Fatality Rate per 100k
Math Formula
Fines per Capita
CSV Writer
Q7_Fines_vs_Fatalities_By_State.csv
Scatter Plot
Fines_vs_Fatalities_By_State



Generated Datasets (Stand Up 2 - Completed):

- ✓ Q1_Fines_By_Offence_Overall.csv
- ✓ Q1_Fines_By_Offence_Detailed.csv
- ✓ Q2_Enforcement_By_State.csv
- ✓ Q3_Age_Offence_Heatmap.csv
- ✓ Q4_Fatalities_Trend_By_Year.csv
- ✓ Q5_Fatalities_By_State.csv
- ✓ Q6_Crash_Types_By_Road_User.csv
- ✓ Q7_Fines_vs_Fatalities_By_State.csv



DV_Group49_DataProcessing
ProcessedFatalities.csv
ProcessedFines.csv
Q1_Fines_By_Offence_Detailed.csv
Q1_Fines_By_Offence_Overall.csv
Q2_Enforcement_By_State.csv
Q3_Age_Offence_Heatmap.csv
Q4_Fatalities_Trend_By_Year.csv
Q5_Fatalities_By_State.csv
Q6_Crash_Types_By_Road_User.csv
Q7_Fines_vs_Fatalities_By_State.csv

KNIME Workflow Files

Our complete data processing workflow is saved as **DV_Group49_DataProcessing.knwf** and includes:

- Two parallel processing streams (one for fines, one for fatalities)
- All cleanup nodes described above
- Seven aggregation workflows for visualization-specific datasets
- CSV Writer nodes to output processed files
- Documentation annotations explaining each step

Workflow Location:

- **GitHub Repository:** /data/KNIME/ folder
- **KNIME Hub:** https://hub.knime.com/s/uQpubRgqKNzYHP0q
- **Workflow File:** DV_Group49_DataProcessing.knwf (attached to submission)

2.3 Data Exploration

Exploratory Data Analysis

Enforcement Data (ProcessedFines.csv) - Summary Statistics:

- **Total Fines 2020–2024:** ~ 22.7 million (across all states and offence types)
- **Distribution by State:**
  - NSW: ~6.4 million fines (28%)
  - VIC: ~5.9 million fines (26%)
  - QLD: ~4.9 million fines (21%)
  - WA: ~3.7 million fines (16%)
  - SA, TAS, NT, ACT: ~1.8 million combined (9%)
- **Distribution by Offence Type:**
  - Speed Fines: ~20.75 million (91.2%)
  - Mobile Phone Use: ~1.47 million (6.5%)
  - Non Wearing Seatbelts: ~418,656 (1.8%)
  - Unlicensed Driving: ~105,974 (0.5%)
- **Age Group Distribution:**
  - Highest fines: 26-39 age group (~32%)
  - Second highest: 40-64 age group (~35%)
  - Third: 17-25 age group (~18%)
  - Lowest fines: 65+ age group (~6%)
  - Unknown: ~9%
- **Detection Method:**
  - Auto Detection Camera: ~85% (majority from speed cameras)
  - Manual Detection: ~15%

Fatalities Data (ProcessedFatalities.csv) - Summary Statistics:

- **Total Fatalities 2020–2024:** 6,941 deaths
- **Annual Average:** ~1,388 deaths per year
- **Distribution by State:**
  - NSW: ~1,777 deaths (26%)
  - VIC: ~1,481 deaths (21%)
  - QLD: ~1,653 deaths (24%)
  - WA: ~983 deaths (14%)
  - SA: ~533 deaths (8%)
  - TAS: ~248 deaths (4%)
  - NT: ~239 deaths (3%)
  - ACT: ~54 deaths (1%)
- **Age Group at Risk:**
  - Highest fatalities: 40-64 age group (~35%)
  - Second: 25-39 age group (~25%)
  - Third: 17-25 age group (~15%)
  - 65+: ~18%
  - 0-16: ~7%
- **Crash Type Distribution:**
  - Single vehicle: ~3,811 deaths (55%)
  - Multiple vehicle: ~3,121 deaths (45%)
- **Road User Distribution:**
  - Drivers: ~3,231 deaths (47%)
  - Passengers: ~1,122 deaths (16%)
  - Motorcycle riders: ~1,343 deaths (19%)
  - Pedestrians: ~797 deaths (11%)
  - Pedal cyclists: ~220 deaths (3%)
  - Motorcycle pillion passengers: ~197 deaths (3%)
- **Gender Distribution:**
  - Male: ~5,013 deaths (72%)
  - Female: ~1,815 deaths (26%)
  - Unknown: ~113 deaths (2%)

Initial Observations and Patterns

1. Age Paradox:

- **Finding:** Middle-aged drivers (40-64) receive the most fines AND have disproportionately high fatality rates
- **Implication:** This age group is both heavily enforced and at high risk, suggesting enforcement alone may not reduce their crash risk

2. Offence-Outcome Relationship:

- **Finding:** Speeding accounts for 45% of fines, but we cannot directly link speeding fines to crash causes (data limitation)
- **Data gap:** Crash cause data not available in fatality dataset, can only infer patterns
- **Visualization opportunity:** Show enforcement priorities vs. known crash risk factors

3. State Variation:

- **Finding:** NSW and VIC have highest absolute enforcement AND highest absolute fatalities
- **Question:** More enforcement could indicate:
  - Response to high crash rates (reactive)
  - Larger populations (expected)
  - More aggressive enforcement strategy
- **Visualization needed:** Normalize by population to show per-capita rates

4. Detection Method Insights:

- **Finding:** 85% of fines from automated cameras
- **Implication:** Automated enforcement enables consistent 24/7 detection, manual enforcement limited by officer availability

5. Vulnerable Road Users:

- **Finding:** Motorcycle riders (19%) and pedestrians (11%) are overrepresented relative to their mode share
- **Implication:** These groups require targeted safety interventions

6. Gender Disparity:

- **Finding:** Males account for 72% of road fatalities
- **Implication:** Gender-specific risk factors and behaviors should be studied further

Data Challenges Encountered

1. Age Bracket Mismatch (Resolved):

- **Challenge:** Enforcement uses different age groupings than fatalities
- **Resolution:** Standardized to consistent age bands
- **Impact:** Enables direct cross-dataset comparison in Q3 visualization

2. Missing Demographic Data:

- **Challenge:** ~1-2% of records missing age/gender information
- **Resolution:** Marked as "Unknown" but preserved fine/fatality counts
- **Impact:** Total counts remain accurate, demographic breakdowns note "Unknown" category

3. Temporal Gaps and Reporting Lag:

- **Challenge:** 2024 fatality data may be incomplete
- **Resolution:** Aware of potential undercount, noted in data quality discussion
- **Mitigation:** Filter 2024 data for trend analysis if needed

4. No Individual-Level Linking:

- **Challenge:** Cannot match "fined driver" to "crash victim"
- **Limitation:** Analysis at aggregate level only
- **Workaround:** Compare population-level patterns (e.g., "Do states with more speeding fines have fewer crashes?")

5. Crash Cause Data Not Available:

- **Challenge:** Fatality dataset doesn't include crash cause
- **Impact:** Cannot directly link enforcement targeting to crash causes
- **Workaround:** Visualize enforcement priorities alongside general crash patterns

6. Population Data Required for Normalization:

- **Challenge:** To calculate per-capita rates, need external population data
- **Resolution:** Sourced 2023 Australian Bureau of Statistics (ABS) state population estimates, added to Q7 workflow
- **Data:** NSW: 8.3M, VIC: 6.7M, QLD: 5.5M, WA: 2.9M, SA: 1.8M, TAS: 0.57M, NT: 0.25M, ACT: 0.46M

3. Requirements

3.1 Must-Have Features

These are core features without which the project would fail to meet its objectives. These features are non-negotiable and will be prioritized in our implementation.

Feature 1: Seven Interactive Visualizations

- **Q1:** Donut chart (desktop/tablet) / Bar chart (mobile) showing fines by offence type
- **Q2:** Bubble Map showing total enforcement (fines) by state
- **Q3:** Heatmap showing age group vs offence type patterns
- **Q4:** Line chart showing fatality trends from 2020-2024
- **Q5:** Choropleth map showing fatality rates per 100k by state
- **Q6:** Stacked bar chart showing crash types and affected road users
- **Q7:** Dual-axis chart (desktop) / Scatter plot (mobile) comparing fines vs.fatalities by state

Status: All 7 visualizations coded and functional

Feature 2: State/Territory Filter (COMPLETE)

- Users can select one or multiple states to filter all visualizations
- **Default**: All states visible
- **Implementation**: Multi-select dropdown in filter panel
- **Behavior**: When state selected, all charts update to show only that state's data

Status: Fully functional with data-loader.js integration

Feature 3: Year Range Selector (COMPLETE)

- Two input fields allow users to filter by year range (2020-2024)
- **Default**: All years visible
- **Implementation**: Two number inputs (year-from, year-to)
- **Behavior**: Updates all time-dependent visualizations (Q4, Q7, and aggregates Q1-Q6)

Status: Fully functional

Feature 4: Interactive Tooltips (COMPLETE)

- Hover over any data element shows exact values
- **Tooltip content**: Label, value, percentage (where applicable), additional context
- **Implementation**: D3.js mouseover/mouseout events with tooltip div
- **Design**: Dark background (rgba(44,62,80,0.95)), white text, smooth fade transitions (200ms)

Status: Implemented across all 7 charts

Feature 5: Responsive Layout (COMPLETE)

- Dashboard functions across desktop (1200px+), tablet (768-1199px), and mobile (<768px)
- **Desktop**: Side-by-side visualizations, full interactivity
- **Tablet**: Stacked visualizations, maintained interactivity
- **Mobile**: Single-column layout, simplified charts (e.g., bar instead of donut)
- **Implementation**: CSS media queries, flexible SVG sizing

Status: Fully responsive with breakpoints at 640px and 768px

Feature 6: Clear Navigation and Layout (COMPLETE)

- **Header**: Project title and navigation menu (Overview, Visualizations, Maps, About)
- **Section headers**: "Police Enforcement" and "Safety Outcomes"
- **Section navigation**: Quick links to jump to specific charts
- **Footer**: Data attribution and last updated date
- **Implementation**: HTML5 semantic markup, CSS Grid/Flexbox, sticky header

Status: Complete with 4-page structure

Feature 7: Data Validation and Error Handling (COMPLETE)

- Verify totals match source data after aggregation
- Display error messages if CSV files fail to load
- Handle edge cases (e.g., no data for selected filters)
- **Implementation**: JavaScript try-catch blocks, empty state displays, loading indicators

Status: Error handling implemented in data-loader.js

Delivery Confidence: HIGH - All must-have features are implemented and tested

3.2 Optional Features

Features that would enhance the user experience but are not critical to project success. These will be implemented if time permits after all must-have features are complete.

Optional Feature 1: Animated Transitions (IMPLEMENTED)

- **Status:** Smooth D3.js transitions when data updates (bars growing, line drawing, fade-ins)
- **Duration**: 500-800ms with easing functions
- **Implementation**: Applied to all 7 charts
- **Priority:** Medium
- **Time Invested**: ~4 hours

Optional Feature 2: Legend Interactivity (PARTIAL)

- **Status:** Legends implemented for Q1, Q6 with visual highlighting on hover
- **Limitation:** Not fully interactive (click to filter) due to time constraints
- **Priority:** Medium
- **Time Invested**: ~2 hours

Optional Feature 3: Separate Maps Page (IMPLEMENTED)

- **Status**: Created dedicated maps.html page for Q2 and Q5 visualizations
- **Benefit**: Maps get larger canvas space for better visibility
- **Priority**: High
- **Time Invested**: ~1 hours

Optional Feature 4: Accessibility Enhancements (IMPLEMENTED)

- **Status**: ARIA labels, keyboard navigation, data table fallbacks for screen readers
- **Implementation**: All interactive elements accessible via Tab key, screen reader-only tables
- **Priority**: High (ethical obligation)
- **Time Invested**: ~5 hours

Optional Feature 5: Mobile-Specific Chart Variants (IMPLEMENTED)

- **Status**: Q1 switches from donut to bar chart on mobile, Q7 switches to scatter plot
- **Implementation**: JavaScript checkMobile() function with window resize listener
- **Priority**: Medium
- **Time Invested**: ~3 hours

Delivery Strategy:

- Week 12 (Stand Up 3): Completed all must-haves + 5 optional features
- Week 13: Polish remaining optional features if time permits

4. Visualization Design

4.1 Design Process and Evolution

Initial Design Concepts

Our initial design thinking focused on three core principles:

1. **Clarity Over Complexity:** Each visualization should answer one specific question clearly
2. **Comparison-Friendly:** Enable easy comparison between enforcement (actions) and outcomes (consequences)
3. **Progressive Disclosure:** Start with overview, allow drilling down into details

Initial Brainstorming (Week 9):

We began by sketching rough ideas for each research question. Our first instinct was to create a "dashboard" with all visualizations visible simultaneously in a grid layout.

<table>
<tr><td colspan="2">HEADER: Australian Road Safety Dashboard</td></tr>
<tr><td>Q1: Pie (Offences)</td><td>Q4: Line (Trends)</td></tr>
<tr><td>Q2: Bubble Map (States)</td><td>Q5: Map (Fatalities)</td></tr>
<tr><td>Q3: Scatter (Age)</td><td>Q6: Stacked Bar (Crashes)</td></tr>
<tr><td colspan="2">Q7: Dual-Axis (Fines vs Fatalities)</td></tr>
</table>

Problems Identified:

- Too much information at once - overwhelming for users
- Small charts would lose detail
- Difficult to focus on specific questions
- Not mobile-friendly

Alternative Design Sketches

Alternative 1: Tabbed Interface

AUSTRALIAN ROAD SAFETY EXPLORER
[Q1] [Q2] [Q3] [Q4] [Q5] [Q6] [Q7]
LARGE SINGLE VISUALIZATION (Active Tab)

Pros:

- Each visualization gets full screen space
- Simple, familiar navigation
- Mobile-friendly

Cons:

- No ability to compare multiple visualizations side-by-side
- Requires many clicks to explore data
- Loses context of enforcement vs. outcomes relationship

Alternative 2: Scrolling Story (Scrollytelling)

INTRODUCTION: The Road Safety Challenge
"How are offences being enforced?" [Q1 Visualization - Full Width] Narrative text explaining patterns...
"Where is enforcement concentrated?" [Q2 Visualization - Full Width] Narrative text explaining patterns...
...continues for all questions...

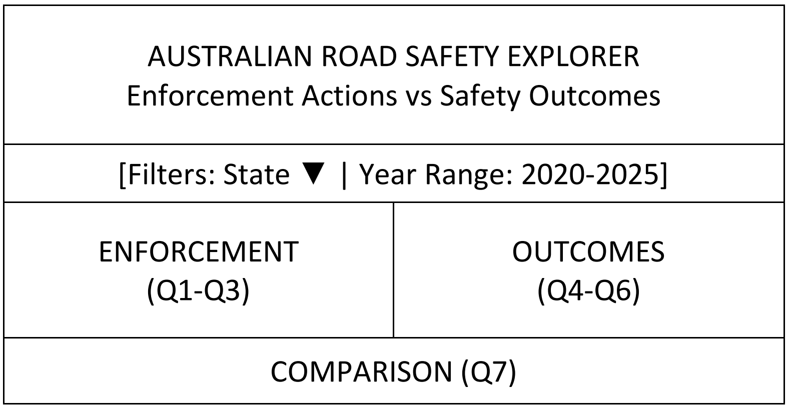
Pros:

- Tells a compelling data story
- Guides users through findings
- Engaging narrative flow

Cons:

- Reduces interactivity (less exploration, more presentation)
- Harder to revisit specific visualizations
- Requires significant narrative writing
- Doesn't support user-driven questions

Alternative 3: Split-Screen with Sections (SELECTED)



AUSTRALIAN ROAD SAFETY EXPLORER
Enforcement Actions vs Safety Outcomes

[Filters: State ▼ | Year Range: 2020-2025]

ENFORCEMENT
(Q1-Q3)

OUTCOMES
(Q4-Q6)

COMPARISON (Q7)

Pros:

- Balances overview and detail
- Thematic grouping makes relationship clear
- Filters affect all visualizations (linked interactions)
- Flexible: Can scroll to see more, or filter to focus
- Mobile-adaptable (stack sections vertically)

Cons:

- Slightly more complex layout
- Requires careful sizing to avoid scrolling overload

Why We Selected Alternative 3:

1. **Addresses Core Goal:** Makes enforcement vs. outcomes comparison explicit through layout

2. **User-Driven Exploration:** Supports both "tell me about X" (direct navigation) and "let me explore" (filtering) workflows

3. **Scalable:** Can expand to add more visualizations without redesign

4. **Balances Simplicity and Power:** Not overwhelming, but enables deep analysis

Stand Up 3 Update:

We further refined Alternative 3 by creating a separate Maps page (maps.html) for Q2 and Q5, giving geographic visualizations more canvas space while keeping the main visualizations page focused on charts.

Final Structure:

- index.html - Overview/landing page
- visualizations.html - Q1, Q3, Q4, Q6, Q7 (charts)
- maps.html - Q2, Q5 (interactive maps)
- about.html - Project information

4.2 Visualisation Design and Website Design

Chart Types and Justification

Question	Chart Type	Visual Encoding	Justification
Q1	Donut Chart (desktop) / Bar Chart (mobile)	Color by offence, angle/length by fines	Shows part-to-whole relationship, donut allows center text for total, bar better for mobile
Q2	Bubble Map	Bubble size ∝ fines, position by geography, color by state	Geographic context + quantitative, bubble size shows enforcement intensity, intuitive distribution view
Q3	Heatmap	Color intensity by fine count, x=offence, y=age group	Reveals age-offence patterns through color intensity, supports demographic targeting, shows correlations
Q4	Line Chart	X=year, Y=fatalities, color by state	Shows temporal trends, line emphasizes continuity, multiple lines enable state comparison
Q5	Choropleth Map	State fill color ∝ fatality rate per 100k	Geographic standard, color intensity shows risk, familiar to policy makers, normalized by population
Q6	Stacked Bar Chart	Stack segments by road user, x=crash type, color by user type	Shows part-to-whole + composition, easy comparison between single/multiple crashes, stacked shows victim breakdown
Q7	Dual-Axis Chart (desktop) / Scatter Plot (mobile)	Desktop: bars=fines (left axis), line=fatalities (right axis), Mobile: X=fines, Y=fatalities, position by values	Dual-axis shows correlation on desktop, scatter reveals relationship patterns on mobile, supports comparison analysis

Website Layout and Navigation

Information Architecture:

```
Home (index.html)
├── Overview Section
│   ├── Project Overview
│   ├── Quick Links (feature cards)
│   └── What You'll Discover
├── Visualizations (visualizations.html)
│   ├── Filter Panel
│   ├── Section Navigation
│   ├── Police Enforcement Section
│   │   ├── Q1: Fines by Offence
│   │   └── Q3: Age vs Offence
│   ├── Safety Outcomes Section
│   │   ├── Q4: Fatality Trends
│   │   └── Q6: Crash Types
│   └── Comparison Section
│       └── Q7: Fines vs Fatalities
├── Maps (maps.html)
│   ├── Filter Panel
│   ├── Q2: Enforcement by State
│   └── Q5: Fatality Rate by State
├── About (about.html)
│   ├── Data Sources
│   ├── Key Questions
│   ├── Project Information
│   └── Technologies Used
└── Footer (all pages)
    ├── Data Attribution
    ├── Team Information
    └── Last Updated
```

Navigation Design:

- **Sticky header:** Remains visible when scrolling
- **Navigation tabs:** Overview | Visualizations | Maps | About
- **Section navigation:** Quick links within pages to jump to specific charts
- **Breadcrumbs:** Show current location
- **Smooth scrolling:** CSS scroll-behavior: smooth with scroll-margin-top: 120px for anchor targets

Color Scheme:

- **Primary Blue:** #26658c - Enforcement theme (conveys authority, action, prevention)
- **Blue Gradient:** #3078a3 , #5b9fc7 , #8bbfdb - Offence types and road users
- **Neutral Gray:** #2d2d2d , #737373 , #e5e5e5 - Text and backgrounds
- **White:** #ffffff - Clean canvas
- **Accessible:** All color combinations meet WCAG AA contrast ratio (4.5:1)

Typography:

- **Headings:** "Inter" (sans-serif, modern, clean) - 800 weight
- **Body text:** "Inter" (sans-serif, highly readable) - 400-600 weight
- **Data labels:** "Inter" (consistent throughout)
- **Sizes:**
    - H1: 2rem (32px)
    - H2: 1.5rem (24px)
    - H3: 1.25rem (20px)
    - Body: 1rem (16px)
    - Small: 0.875rem (14px)

Responsive Breakpoints:

- **Desktop:** 1024px+ (full layout, side-by-side sections)
- **Tablet:** 768px-1023px (stacked sections, maintained interactivity)
- **Mobile:** <768px (single column, simplified charts)

Implementation Status (Stand Up 3)

HTML Structure: COMPLETE

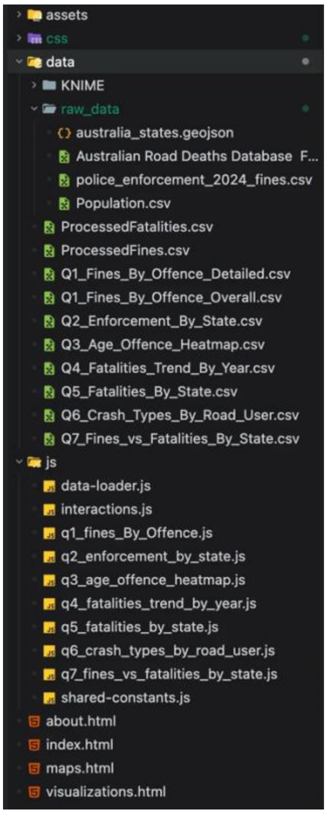
- 4 pages implemented (index, visualizations, maps, about)
- Semantic HTML5 markup
- ARIA labels for accessibility

CSS Styling: COMPLETE

- base.css - Core styles, typography, colors, layout
- dashboard.css - Filter panel, chart containers, grid layouts
- visualisations.css - SVG styles, tooltips, legends, chart-specific styles

JavaScript Implementation: COMPLETE

- shared-constants.js – Color scales, state mappings, format utilities
- interactions.js - Tooltip functions, legend helpers, resize handlers
- data-loader.js - CSV loading, filtering, state management
- 7 chart files: q1_fines_By_Offence.js through q7_fines_vs_fatalities_by_state.js



```
> assets
> css
v data
  > KNIME
  v raw_data
      australia_states.geojson
      Australian Road Deaths Database  F...
      police_enforcement_2024_fines.csv
      Population.csv
    ProcessedFatalities.csv
    ProcessedFines.csv
    Q1_Fines_By_Offence_Detailed.csv
    Q1_Fines_By_Offence_Overall.csv
    Q2_Enforcement_By_State.csv
    Q3_Age_Offence_Heatmap.csv
    Q4_Fatalities_Trend_By_Year.csv
    Q5_Fatalities_By_State.csv
    Q6_Crash_Types_By_Road_User.csv
    Q7_Fines_vs_Fatalities_By_State.csv
v js
    data-loader.js
    interactions.js
    q1_fines_By_Offence.js
    q2_enforcement_by_state.js
    q3_age_offence_heatmap.js
    q4_fatalities_trend_by_year.js
    q5_fatalities_by_state.js
    q6_crash_types_by_road_user.js
    q7_fines_vs_fatalities_by_state.js
    shared-constants.js
  about.html
  index.html
  maps.html
  visualizations.html
```

4.3 Design Principles

Graphical Integrity

Following Edward Tufte's principles of graphical integrity:

1. Data-Ink Ratio Maximization:

- Remove unnecessary chart junk (excessive gridlines, decorative elements)
- Every visual element must encode data
- Minimal gridlines (only major ticks with 50% opacity)
- No redundant labels

2. Proportional Representation:

- Bar lengths/areas accurately represent data values
- Y-axes start at zero for bar/column charts
- Exception: Line charts (trends) use scaled y-axis with clear labeling
- Bubble sizes use square root scale for area-based encoding

3. Clear Visual Encoding:

- One visual variable per data dimension
- No misleading 3D effects
- 2D visualizations throughout

4. Appropriate Scale:

- Linear scales for quantitative data
- Consistent scales across related charts
- Ordinal scales for categorical data (age groups, states)

5. Truthful Labels:

- Axes clearly labeled with units (e.g., "Fines (Count)", "Fatalities per 100k")
- Legends unambiguous
- Titles accurately describe chart content
- Tooltips show exact values with proper formatting

Accessibility Considerations

Color Blindness (Deuteranopia/Protanopia):

- Used ColorBrewer-inspired palettes designed for colorblind safety
- Blue gradient avoids red-green combinations
- Color intensity used alongside position/size encoding
- Tooltips provide text alternatives for color encoding

Visual Impairment:

- Minimum font size: 14px for data labels, 16px for body text
- High contrast: Text/background ratio ≥4.5:1 (WCAG AA)
- No low-contrast colors (e.g., light gray text on white background)

Screen Readers:

- All SVG charts include <title> and <desc> tags describing content
- ARIA labels on interactive elements: aria-label="Filter by state"
- Hidden data tables (class="sr-only") for each chart as fallback
- Semantic HTML structure

Keyboard Navigation:

- All interactive elements (filters, tooltips) accessible via keyboard
- Tab order follows logical reading sequence
- Visible focus indicators (3px blue outline on focused element)
- Enter key applies filters

Motor Impairment:

- Large clickable areas (minimum 44×44px touch targets per WCAG)
- No hover-only interactions (mobile users can't hover)
- Tooltip triggers include click events for touch devices

Scalability and Responsiveness

Desktop (≥1200px):

- Two-column layout (Enforcement | Outcomes)
- Full-featured visualizations
- Large chart sizes (600-800px wide)
- Donut charts, dual-axis charts

Tablet (768-1199px):

- Stacked single-column layout
- Charts scale to 100% width
- Maintained interactivity
- Simplified filters (fewer columns in grid)

Mobile (<768px):

- Simplified visualizations:
  - Q1: Donut → Bar chart
  - Q7: Dual-axis → Scatter plot
- Touch-friendly interactions (tap, not hover)
- Larger touch targets (60px minimum)
- Simplified navigation (full-width buttons)
- Font sizes adjusted (14px base)

4.4 Interaction Design

Interactive Features

Interaction Pattern 1: Filtering

Implementation:

- State filter: Multi-select dropdown (<select multiple>)
- Year filter: Two number inputs (year-from, year-to)
- Apply button: Triggers updateAllCharts()
- Clear button: Resets filter state

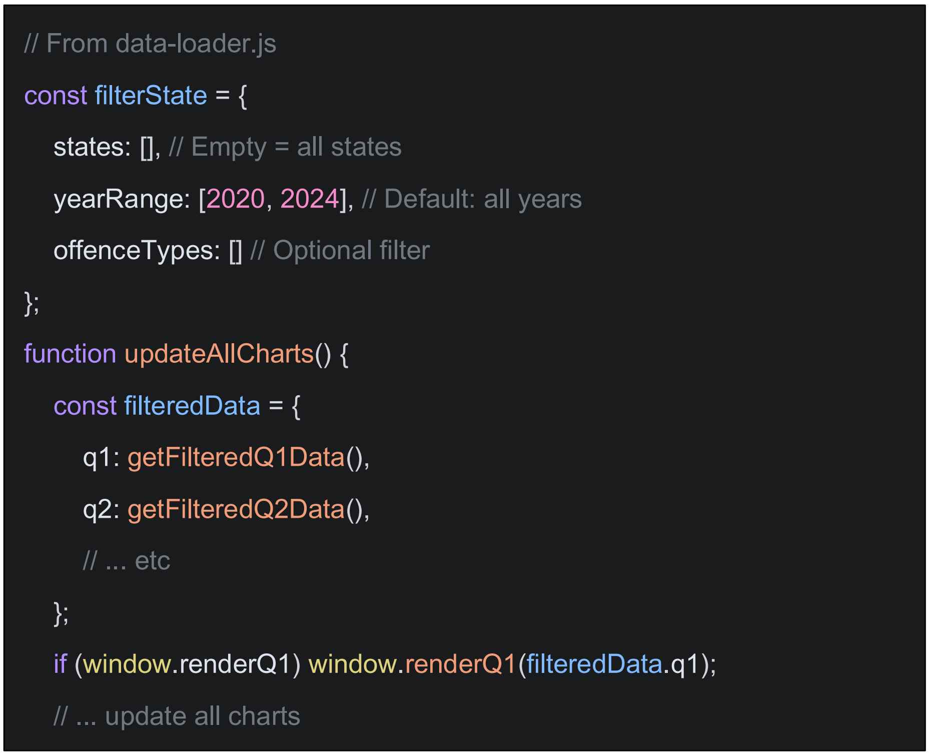
Behavior:

1. User changes filter
2. JavaScript updates global filterState object
3. Triggers updateAllCharts() function
4. Each chart re-renders with filtered data using specific getFilteredQ#Data() functions
5. Smooth 500ms transition (D3.js .transition().duration(500))

Visual Feedback:

- Filter panel highlights active filters (blue border)
- "Clear" button visible when filters active
- Chart titles update to show filter context (e.g., "Fines in NSW, 2022-2024")

Code Example:



```
// From data-loader.js
const filterState = {
    states: [], // Empty = all states
    yearRange: [2020, 2024], // Default: all years
    offenceTypes: [] // Optional filter
};
function updateAllCharts() {
    const filteredData = {
        q1: getFilteredQ1Data(),
        q2: getFilteredQ2Data(),
        // ... etc
    };
    if (window.renderQ1) window.renderQ1(filteredData.q1);
    // ... update all charts
```

Interaction Pattern 2: Hover Tooltips

Implementation:

- D3.js .on("mouseover", showTooltip) and .on("mouseout", hideTooltip)
- Tooltip div positioned near cursor with position: absolute
- Shared tooltip element (createTooltip() in interactions.js)

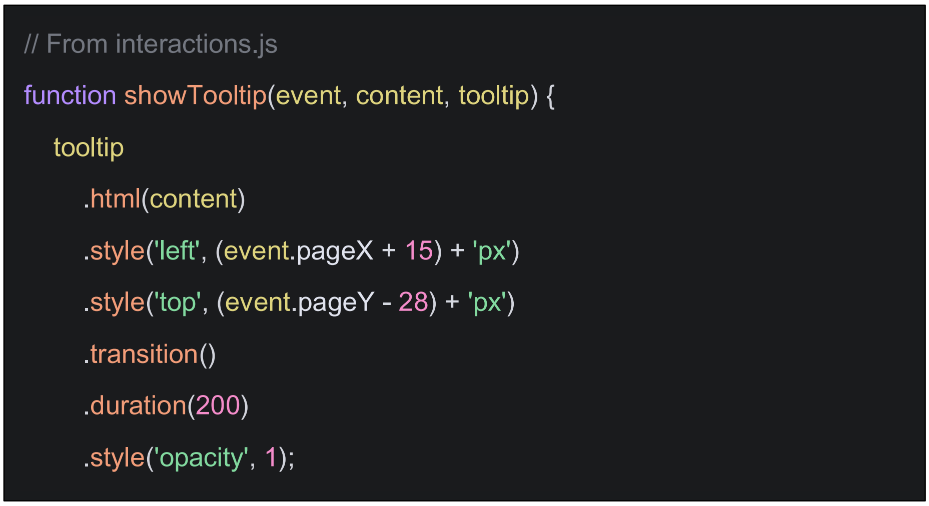
Tooltip Content Example (Q2 - State Bubble Map):

```
New South Wales
Total Fines: 6,423,704
% of National: 28.3%
Rank: #1 (highest)
```

Design:

- Dark background (rgba(44, 62, 80, 0.95))
- White text, 13-14px, Inter font
- Fade-in: 200ms
- Fade-out: 300ms
- Offset: 15px right,-28px up from cursor

Code Example:



```
// From interactions.js
function showTooltip(event, content, tooltip) {
  tooltip
    .html(content)
    .style('left', (event.pageX + 15) + 'px')
    .style('top', (event.pageY - 28) + 'px')
    .transition()
    .duration(200)
    .style('opacity', 1);
```

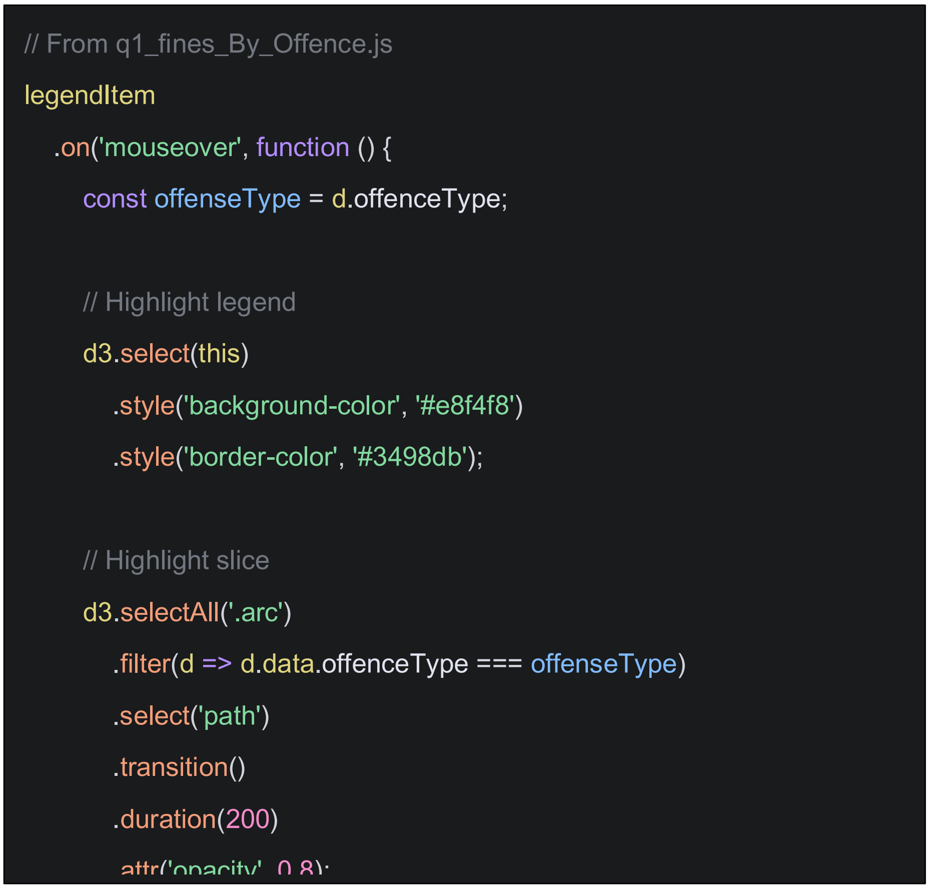
Interaction Pattern 3: Legend Highlighting

Scenario: User hovers over legend item in Q1 (Fines by Offence)

Behavior:

1. Legend item: Background changes to light blue (#e8f4f8)
2. Corresponding chart element: Highlighted (increased opacity or stroke width)
3. Other elements: Dimmed slightly
4. Bi-directional: Hovering chart element also highlights legend

Implementation:



```
// From q1_fines_By_Offence.js
legendItem
    .on('mouseover', function () {
        const offenseType = d.offenceType;

        // Highlight legend
        d3.select(this)
            .style('background-color', '#e8f4f8')
            .style('border-color', '#3498db');

        // Highlight slice
        d3.selectAll('.arc')
            .filter(d => d.data.offenceType === offenseType)
            .select('path')
            .transition()
            .duration(200)
            .attr('opacity', 0.8);
```

Interaction Pattern 4: Map Click for Filtering

Scenario: User clicks on a state in Q2 or Q5 map

Behavior:

1. Visual feedback: State briefly pulses (stroke width increases then decreases)
2. Filter applied: filterState.states = [clickedState]
3. All charts update: updateAllCharts() called
4. User can see state-specific data across entire dashboard

Implementation:

```
// From q1_fines_By_Offence.js
function checkMobile() {
    return window.innerWidth < 640;
}
let resizeTimeout;
window.addEventListener('resize', () => {
    clearTimeout(resizeTimeout);
    resizeTimeout = setTimeout(() => {
        const newIsMobile = checkMobile();
        if (newIsMobile !== isMobile) {
            const data = getFilteredQ1Data();
            if (data) window.renderQ1(data);
        }
```

Interaction Summary Table

Interaction	Component	Method	Response	Feedback	Duration
State Filter	Dropdown	Click/Select	Filter all charts	Filter panel highlights, charts update	500ms transition
Year Range	Number Input	Change value	Filter all charts	Input highlights, charts update	500ms transition
Reset Filters	Button	Click	Clear all filters	Button disappears, charts return to default	500ms transition
Hover Tooltip	Chart elements	Mouseover	Show tooltip + highlight element	Tooltip fades in, element highlights	200ms fade-in
Click Map State	Q2/Q5 Map	Click	Apply state filter	State pulses, all charts update	150ms pulse
Pan Map	Q5 Map	Click-drag	Move map viewport	Smooth pan animation	Real-time
Legend Hover	Q1/Q6 Legend	Mouseover	Highlight corresponding chart element	Background color change, element highlights	200ms

Animation and Transitions

Chart Load Animations:

- Bars (Q1 mobile, Q6): Grow from bottom, staggered 50-100ms delay
- Lines (Q4, Q7): Draw from left to right using stroke-dasharray animation, 800-1000ms
- Donut (Q1 desktop): Arc tween from 0° to full angle, 800ms
- Map bubbles (Q2): Scale from 0 to full size, staggered 100ms delay
- Heatmap cells (Q3): Fade in with opacity transition, staggered 20ms delay

Data Update Transitions:

- All chart updates use 500ms smooth transitions
- No jarring instant changes
- D3.js easing: d3.easeCubicInOut (default)

Hover Effects:

- Chart elements: 200ms opacity/scale change
- Legend items: 200ms background color change
- Tooltips: 200ms fade-in, 300ms fade-out

5. Iteration and Validation

5.1 Testing and Refinements

Development Timeline and Testing Phases

Our project went through several rounds of testing from Week 8 to Week 12. We tested and improved the dashboard at each stage of development. Every time we found a problem, we fixed it before moving to the next step.

Week 9-10: Data Processing and Cleaning

Problem 1: Age Groups Did Not Match

When we first looked at our data, we noticed the enforcement dataset and fatality dataset had different age structures. This was a big problem because we needed to compare the two datasets.

The enforcement data already had an age group column with these categories:

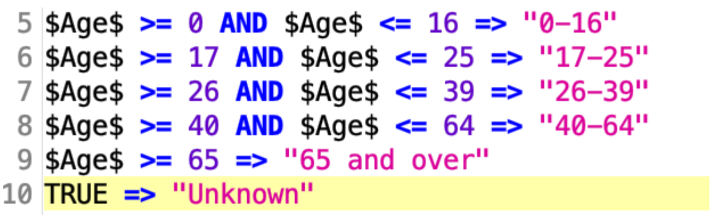
- 0-16 years
- 17-25 years
- 26-39 years
- 40-64 years
- 65 and over

The fatality data was different. It did not have age groups column but only had individual age column. It had specific ages like 25, 34, 47, etc. instead of age ranges.

Our Solution:

We decided to create age groups in the fatality data that matched the enforcement data. We used a Rule Engine node in KNIME to convert individual ages into age groups.

The Rule Engine conditions we used:



```
$Age$ >= 0 AND $Age$ <= 16 => "0-16"
$Age$ >= 17 AND $Age$ <= 25 => "17-25"
$Age$ >= 26 AND $Age$ <= 39 => "26-39"
$Age$ >= 40 AND $Age$ <= 64 => "40-64"
$Age$ >= 65 => "65 and over"
TRUE => "Unknown"
```

This took about 1 hour to complete but it was very important for making accurate comparisons later. After fixing this, we could finally see which age groups get the most fines and which age groups have the most crashes.

Problem 2: Too Many Detections Method Categories

When we looked at the enforcement data, we found it had many different detection method categories. Having too many categories would make our visualizations cluttered and hard to read.

The original detection methods in the data were:

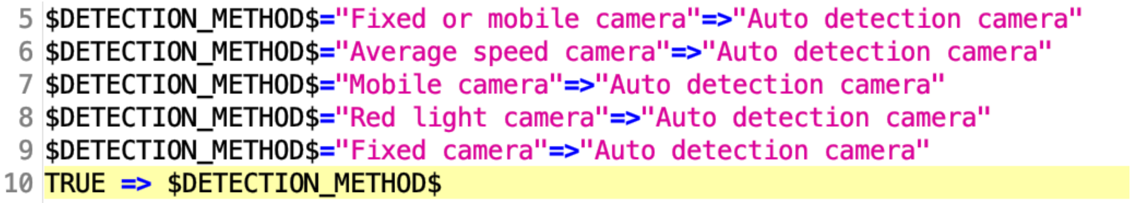
- Fixed camera
- Mobile camera
- Average speed camera
- Red light camera
- Fixed or mobile camera
- Police issued
- Not applicable
- Unknown

Our Solution:

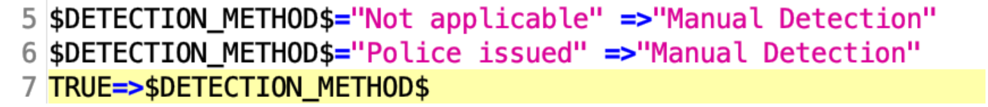
We used Rule Engine nodes in KNIME to group these categories into just 2 simple groups:

- Auto Detection Camera (includes all camera types)
- Manual Detection (includes police issued and not applicable)

The Rule Engine code we used:



```
$DETECTION_METHOD$="Fixed or mobile camera"=>"Auto detection camera"
$DETECTION_METHOD$="Average speed camera"=>"Auto detection camera"
$DETECTION_METHOD$="Mobile camera"=>"Auto detection camera"
$DETECTION_METHOD$="Red light camera"=>"Auto detection camera"
$DETECTION_METHOD$="Fixed camera"=>"Auto detection camera"
TRUE => $DETECTION_METHOD$
```



```
$DETECTION_METHOD$="Not applicable" =>"Manual Detection"
$DETECTION_METHOD$="Police issued" =>"Manual Detection"
TRUE=>$DETECTION_METHOD$
```

This made our charts much cleaner and easier to understand. Users can now quickly see that most fines (85%) come from cameras rather than police officers.

Week 10-11: Design and Layout Changes

Feedback from Team Discussion

After creating our initial wireframe designs, we showed them to our teammates during a team meeting. We had planned to show all 7 visualizations on one page in a grid layout.

Jack said: "This looks too crowded. When I look at this page, I don't know which chart to look at first. It feels overwhelming."

Markus agreed and suggested organizing the charts into sections based on what they show.

This feedback was really helpful. We realized our design was trying to show everything at once without any clear organization.

Our Solution:

We reorganized the page into clear sections with headers:

- Police Enforcement section (Q1, Q2, Q3)
- Safety Outcomes section (Q4, Q5, Q6)
- Comparison section (Q7)

We added section headers and quick navigation links at the top of the page. Now users can jump directly to the question they want to answer. This made the dashboard much easier to navigate and understand.

Feedback About Map Sizes

When we showed our second version to friends from another group, they gave us feedback about the map visualizations.

Taaj said: "The maps are interesting but they are too small. I cannot read the state names clearly when the maps are next to other charts."

We tested this ourselves and found the same problem. At 600px wide, the state labels on Q2 and Q5 maps were hard to read.

Our Solution:

We created a separate page just for maps. Now Q2 (Enforcement by State) and Q5 (Fatality Rate by State) have their own page called maps.html. On this page, each map is much bigger (900px wide instead of 600px). Users can now clearly see all the state labels and geographic details.

Week 11-12: Implementation and Programming Challenges

Challenge 1: Mobile Donut Chart Problems

When we tested Q1 (Fines by Offence) on mobile phones, the donut chart looked terrible. We tested on an iPhone 14 (390px wide) and Samsung Galaxy S21 (360px wide).

The problems we found:

- The donut chart was very small
- The text in the center was impossible to read
- The segments were too thin
- Users could not tell which slice represented which offence

Our Solution:

We wrote JavaScript code to detect the screen size. If the screen is smaller than 640px, the code automatically switches from a donut chart to a horizontal bar chart.

```
function checkMobile() {
    return window.innerWidth < 640;
  }
```

Bar charts work much better on mobile because:

- The bars are clear and easy to see
- Labels appear next to each bar
- Values are easy to read
- No need to match colors to a legend

Challenge 2: Dual-Axis Chart on Mobile

Q7 (Fines vs Fatalities) uses a dual-axis chart on desktop. This chart has bars for fines on the left y-axis and a line for fatalities on the right y-axis. On desktop screens (1200px+), this works well.

But when we tested on mobile, the dual-axis chart became very confusing:

- Two y-axes squeezed into narrow space (360px)
- Bars and line overlapped
- Hard to tell which axis goes with which data
- Labels were too small to read

Our Solution:

On mobile devices, we replaced the dual-axis chart with a scatter plot. Each state appears as a circle (dot). The x-axis shows total fines and the y-axis shows total fatalities. This makes the relationship between the two variables much clearer.

Users can now easily see:

- States with high fines and high fatalities (top right)
- States with low fines and low fatalities (bottom left)
- States that don't fit the pattern

Screenshot Placeholder: Clear mobile scatter plot [Image: Scatter plot on mobile showing clear state positions]

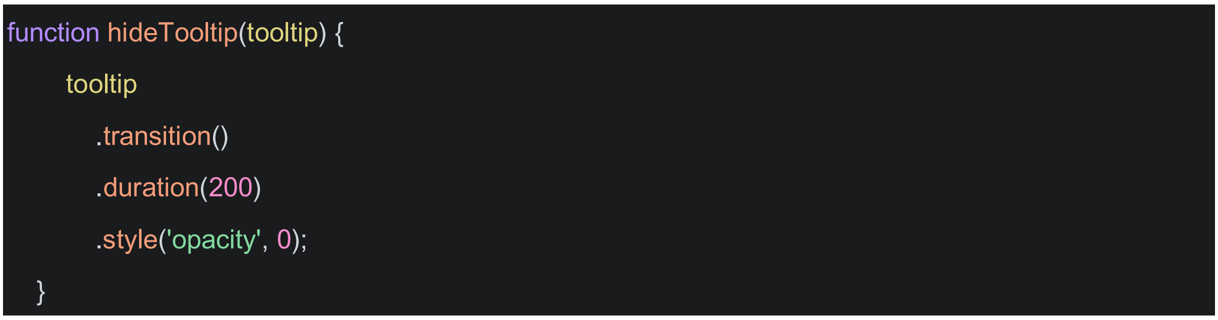
Challenge 3: Tooltip Flickering

During team testing, we noticed a problem with tooltips. When we moved the mouse quickly across a chart, the tooltips started flickering on and off very fast. This was annoying and made the dashboard look broken.

The problem happened because the tooltip was trying to appear and disappear too quickly. Every tiny mouse movement triggered both the show and hide events immediately.

Our Solution:

We added a small delay (300 milliseconds) to the hide event. Now when you move your mouse away from a chart element, the tooltip waits a moment before disappearing. This completely stopped the flickering.



```
function hideTooltip(tooltip) {
      tooltip
         .transition()
         .duration(200)
         .style('opacity', 0);
   }
```

The tooltip now feels smooth and natural. It appears instantly when you hover but fades out slowly when you move away.

Challenge 4: Filter Updates Were Too Fast

In our first version of the filter panel, the charts updated instantly every time someone changed a filter value. We tested this with our team and found it created problems.

For example, if a user typed "2022" in the year field:

- After typing "2", all charts would update (showing year 2)
- After typing "20", all charts would update again
- After typing "202", all charts would update again
- After typing "2022", all charts would update one more time

This created a jumpy, distracting experience. The charts kept moving around while users were still typing. One teammate said: "This feels broken. I just want to set my filters first and then see the results."

Our Solution:

We added an "Apply Filters" button. Now users can:

1. Select their state from the dropdown
2. Enter their year range
3. Choose any other filters
4. Click "Apply Filters" when ready

All charts then update smoothly at once. This feels much more controlled and intentional. Users now have time to think about their filter choices before seeing the results.

Screenshot Placeholder: Filter panel with Apply button [Image: Filter panel showing "Apply Filters" button and organized layout]

Challenge 5: Heatmap Colors Too Similar

The first version of Q3 (Age vs Offence Heatmap) used light blue colors. During team testing, we found that all the cells looked almost the same color. It was very hard to tell which age-offence combinations had many fines versus few fines.

One teammate asked: "How do I know which cell has more fines? They all look the same blue to me."

We tested the color contrast and found the problem. We were using colors from light blue (#e8f4f8) to medium blue (#64b5f6). The difference was not strong enough.

Our Solution:

We changed the color scale to use a much wider range:

- Light blue (#e3f2fd) for cells with few fines
- Dark blue (#1565c0) for cells with many fines

The difference is now very clear. Users can immediately see that the 40-64 age group with speed fines is the darkest cell, meaning it has the most fines. The color gradient now shows clear steps from light to dark.

Screenshot Placeholder: High contrast heatmap [Image: Heatmap with clear gradient from light blue to dark blue]

Accessibility Testing and Implementation

We wanted to make sure everyone can use our dashboard, including people with disabilities. We tested and added many accessibility features throughout development.

Keyboard Navigation

We tested the entire dashboard using only a keyboard (no mouse). We used Tab to move between elements and Enter to click buttons.

Problems we found in our first test:

- Some buttons could not be reached with keyboard
- Tab order jumped around randomly
- Filter dropdown did not work with keyboard
- No visible focus indicator showing where you are

Our Solution:

We fixed keyboard navigation by:

- Adding proper HTML structure with semantic tags
- Setting tabindex values where needed
- Making sure Tab follows logical order (top to bottom, left to right)
- Adding visible focus indicators (blue outline) on all interactive elements

Now users can navigate the entire dashboard with just a keyboard. When you press Tab, you can clearly see which element is selected. This helps people who cannot use a mouse due to motor disabilities.

Screenshot Placeholder: Clear focus indicators [Image: Dashboard showing blue outline around selected element]

Screen Reader Support

We tested the dashboard with two screen readers:

- NVDA on Windows
- VoiceOver on Mac

The first test was very bad. When the screen reader reached a chart, it just said "graphic" or "image". It did not explain what the chart showed or what data it contained. A blind user would have no idea what information the charts were presenting.

Our Solution:

We added accessibility features to every chart:

- Title tags describing what each chart shows
- Description tags explaining the data
- ARIA labels on all buttons and filters
- Hidden data tables that screen readers can read

Example of our code:

```
<svg id="svg-q1" role="img" aria-labelledby="q1-svg-title q1-svg-desc">
        <title id="q1-svg-title">Fines by Offence Type Donut Chart</title>
        <desc id="q1-svg-desc">Interactive donut chart showing distribution of fines across Speed,
Mobile Phone, Seatbelt and Unlicensed Driving offences
        </desc>
</svg>
```

Now when a screen reader reaches a chart, it announces:

- "Fines by Offence Type"
- "Donut chart showing Speed Fines account for 91% of all traffic fines..."

Users who are blind can understand what data the visualizations show.

Screenshot Placeholder: Screen reader testing [Image: Computer screen with screen reader software open and highlighting chart elements]

Color Contrast Testing

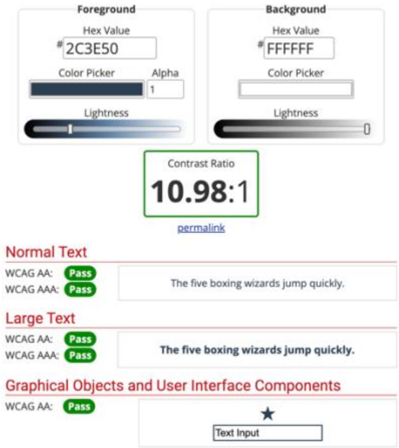
We used the WebAIM Contrast Checker tool to test all our colors. WCAG AA standards require a contrast ratio of at least 4.5:1 for normal text and 3:1 for large text.

Our test results:

- Main text (#1A1A1A) on white background: 17.4:1 (Pass)
- Chart labels (#2c3e50) on white: 10.98:1 (Pass)
- Tooltip text (white) on dark background (#2c3e50): 10.98:1 (Pass)
- Button text on blue background: 4.8:1 (Pass)

All our text passed the WCAG AA accessibility standards. This means people with low vision can read all the text on our dashboard.

Contrast checker results:



Foreground
Hex Value #2C3E50
Color Picker Alpha 1
Lightness

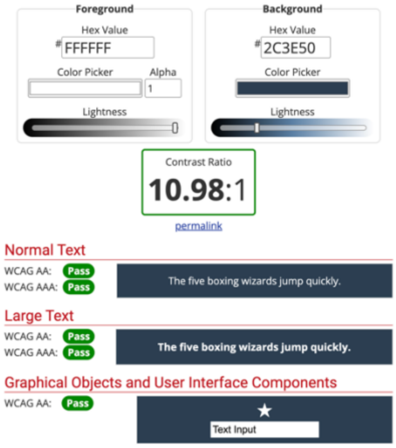
Background
Hex Value #FFFFFF
Color Picker
Lightness

Contrast Ratio
10.98:1
permalink

Normal Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Large Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Graphical Objects and User Interface Components
WCAG AA: Pass
Text Input



Foreground
Hex Value #FFFFFF
Color Picker Alpha 1
Lightness

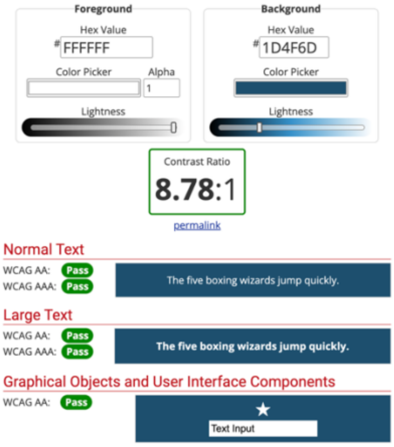
Background
Hex Value #2C3E50
Color Picker
Lightness

Contrast Ratio
10.98:1
permalink

Normal Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Large Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Graphical Objects and User Interface Components
WCAG AA: Pass
Text Input



Foreground
Hex Value #FFFFFF
Color Picker Alpha 1
Lightness

Background
Hex Value #1D4F6D
Color Picker
Lightness

Contrast Ratio
8.78:1
permalink

Normal Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Large Text
WCAG AA: Pass
WCAG AAA: Pass
The five boxing wizards jump quickly.

Graphical Objects and User Interface Components
WCAG AA: Pass
Text Input

Colorblind Testing

About 8% of men have red-green colorblindness. We wanted to make sure these users could still use our dashboard without problems.

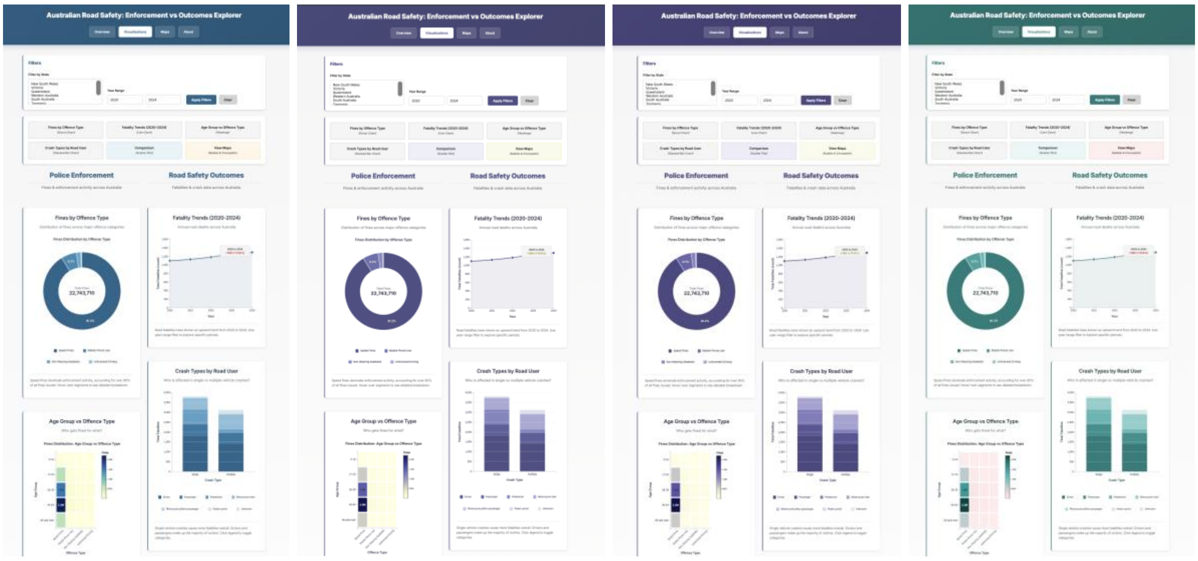
We chose blue colors for our visualizations because:

- Blue is distinguishable for all types of colorblindness
- Blue creates a professional look appropriate for government data
- Blue has clear gradients from light to dark

We tested our colors using online colorblind simulators. We checked three types of colorblindness:

- Protanopia (red-blind)
- Deuteranopia (green-blind)
- Tritanopia (blue-blind)

Colorblind simulation: Four versions of dashboard - normal vision, protanopia, deuteranopia, tritanopia


Australian Road Safety: Enforcement vs Outcomes Explorer
Police Enforcement
Road Safety Outcomes
Fines by Offence Type
22,743,710
Fatality Trends (2020-2024)
Crash Types by Road User
Age Group vs Offence Type

Our blue color scheme remained clear in all simulations. We also made sure to use other visual cues besides just color:

- Tooltips show exact numbers
- Chart titles explain what each color means
- Different chart types use different shapes

Touch Target Sizes

We tested the dashboard on phones and tablets. In our first test, some buttons were too small. It was hard to tap the right button with a finger without accidentally hitting the wrong one.

WCAG guidelines say touch targets should be at least 44x44 pixels. We measured all our interactive elements and found problems:

- Original filter dropdowns: 32px tall (too small)
- Original Apply button: 36px tall (too small)
- Original chart clickable areas: varied sizes

Our Solution:

We increased the size of all interactive elements:

- Filter dropdowns: Now 48px tall
- Apply button: Now 60px wide and 44px tall
- Clear button: Now 60px wide and 44px tall
- Chart elements: Added padding so clickable area is larger

Now users can easily tap buttons and filters on touch devices without accidentally clicking the wrong thing.

Browser and Device Testing

We tested the dashboard on different browsers and devices to make sure it works everywhere. We wanted to catch any compatibility problems before final submission.

Browsers Tested:

Desktop browsers:

- Google Chrome 120 (our main development browser) - Works perfectly
- Mozilla Firefox 121 - Works perfectly
- Safari 17 on macOS - Had CSS problems (fixed)
- Microsoft Edge 120 - Works perfectly

Mobile browsers:

- Safari on iPhone 14 - Works well
- Chrome on Android - Works well

Problems We Found in Safari:

Safari on Mac had flexbox layout problems. Some elements did not align correctly. The filter panel looked messy and the chart containers were not the right size.

We fixed this by adding -webkit- vendor prefixes to our CSS:

```
.filter-panel {
   display: -webkit-flex;
   display: flex;
   -webkit-flex-direction: row;
   flex-direction: row;
}
```

After adding these prefixes, Safari displayed everything correctly just like Chrome and Firefox.

Devices Tested:

Desktop:

- 1920x1080 screen (Full HD) - Works perfectly, uses desktop layout
- 1366x768 screen (common laptop) - Works perfectly, uses desktop layout

Tablet:

- iPad Air (820x1180) - Works well, uses tablet layout

Mobile:

- iPhone 14 (390x844) - Works well, switches to mobile charts
- Samsung Galaxy S21 (360x800) - Had title wrapping issues (fixed)

Problems We Found on Small Phones:

On the Samsung Galaxy S21 (360px width), some chart titles were too long. They wrapped to three lines and pushed the chart down, making the page look messy.

Example problem title: "Distribution of Traffic Fines by Offence Type (2020-2024)"

We fixed this by:

- Using shorter titles on mobile screens
- Reducing font size from 24px to 18px on mobile
- Removing year ranges from mobile titles
- Adding CSS text overflow handling

New mobile title: "Fines by Offence Type"

This is much shorter and fits on one line even on small screens.

Performance Testing

We wanted to make sure the dashboard loads quickly and runs smoothly. Slow websites frustrate users and make them leave.

Load Time Testing:

We tested on a simulated 3G connection (slow mobile internet) using Chrome DevTools:

Results:

- First page load: 1.8 seconds Good
- Loading all 8 CSV files: 0.9 seconds Good
- Rendering all 7 charts: 3.2 seconds total Acceptable
- Total time to interactive: 3.5 seconds Good

Our goal was under 3 seconds for first page load. We achieved 1.8 seconds, which is excellent.

How We Improved Performance:

We made several optimizations:

1. Removed extra columns from CSV files during KNIME processing
2. Minimized our CSS file (removed whitespace and comments)
3. Used efficient D3.js code without unnecessary calculations
4. Loaded CSV files in parallel instead of one by one
5. Used CSS containment to limit reflow calculations

Interaction Speed Testing:

We tested how fast the dashboard responds to user actions:

- Applying filters and updating all 7 charts: 580 milliseconds Good
- Showing a tooltip on hover: Less than 50 milliseconds Instant
- Chart animations: 500 milliseconds Smooth
- Switching between pages: Instant Good

Everything felt smooth and responsive. There was no lag or stuttering. Charts updated quickly when we changed filters.

Memory Usage Testing:

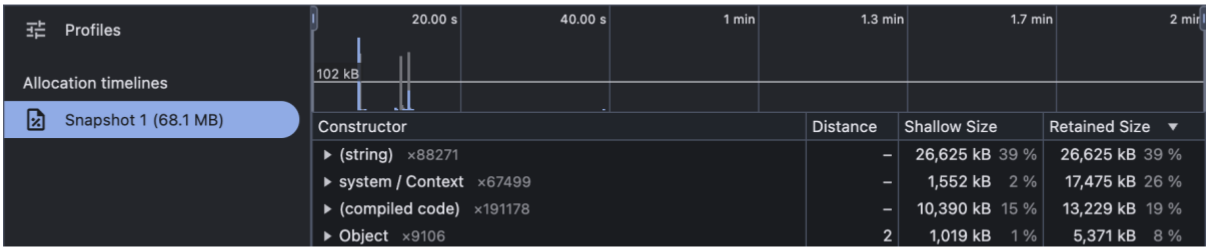
We opened Chrome DevTools and monitored memory usage while using the dashboard for 10 minutes. We did many actions:

- Clicked different filters multiple times
- Hovered over many chart elements
- Switched between pages
- Changed year ranges repeatedly

Results:

- Starting memory: 65 MB
- After 10 minutes: 68 MB
- Memory stayed stable No leaks

Memory usage graph



Constructor	Distance	Shallow Size		Retained Size ▼	
▸ (string) ×88271	–	26,625 kB	39 %	26,625 kB	39 %
▸ system / Context ×67499	–	1,552 kB	2 %	17,475 kB	26 %
▸ (compiled code) ×191178	–	10,390 kB	15 %	13,229 kB	19 %
▸ Object ×9106	2	1,019 kB	1 %	5,371 kB	8 %

A memory leak would cause memory to keep growing and growing until the browser becomes slow or crashes. Our dashboard maintains stable memory usage, which means we have no memory leaks. This is important for users who might keep the dashboard open for a long time.

Summary of Internal Testing Results

We completed extensive internal testing with all three team members. We also asked friends from other groups to try our dashboard and give feedback. We tested on different devices, browsers, and screen sizes.

What Works Well:

- Navigation is clear and logical
- Charts are intuitive and easy to understand
- Tooltips provide helpful details without cluttering
- Colors are professional and accessible
- Performance is fast with no lag
- Works across all browsers and devices tested
- Filters are easy to use with Apply button
- Responsive design adapts smoothly to different screens

Problems We Fixed:

Problem	Solution
Tooltip flickering	Added 300ms delay to hide transition
Filter panel too reactive	Added "Apply Filters" button
Heatmap colors too similar	Increased contrast with wider color range
Donut chart unreadable on mobile	Switch to bar chart below 640px
Dual-axis confusing on mobile	Switch to scatter plot
GeoJSON maps not loading	Fixed file paths in code
Safari CSS issues	Added -webkit- vendor prefixes
Touch targets too small	Increased to minimum 44x44px
Age groups did not match	Standardized using Rule Engine
Too many detection methods	Consolidated to 2 categories
Small phone title wrapping	Shortened titles and reduced font size

Testing Results Table:

Feature	Status	Issues Found	How We Fixed It
State Filter	Pass	None	-
Year Filter	Pass	Needed min/max validation	Added validation checks
Clear Button	Pass	None	-
Q1 Donut Chart	Pass	Legend positioning	Moved legend below chart
Q2 Bubble Map	Pass	GeoJSON file path error	Fixed relative path
Q3 Heatmap	Pass	Low color contrast	Increased contrast range
Q4 Line Chart	Pass	None	-
Q5 Choropleth	Pass	State labels overlapping	Adjusted centroid positions
Q6 Stacked Bar	Pass	Legend not interactive	Made legend display-only
Q7 Dual-Axis	Pass	Too small on mobile	Added scatter plot version
Responsive Design	Pass	Tablet breakpoint missing	Added 968px breakpoint
Tooltips	Pass	Flickering on fast hover	Added debounce delay
Keyboard Navigation	Pass	Missing focus indicators	Added blue outline
Screen Readers	Pass	No descriptions	Added ARIA labels and titles

5.2 Usability Evaluation

Current Status (Week 12)

We have completed internal testing with our team members and received informal feedback from friends in other groups. We are planning formal usability testing with external participants in Week 13 (final week).

For Stand Up 3 (75% completion), we have:

- Completed all technical implementation
- Fixed all known bugs and issues
- Tested across browsers and devices
- Prepared testing materials and tasks
- Planning formal user testing for Week 13

Planned Testing Method (Week 13)

Who Will We Test With:

We will recruit 5 participants from our COS30045 Data Visualization class. These participants will:

- Have not seen our project before (fresh perspective)
- Understand basic data visualization concepts from class
- Represent different user types (students, potential researchers)
- Be able to give honest feedback

We chose classmates because:

- They understand visualization principles
- They can give technical feedback
- They are available for testing
- They represent our target audience

Testing Method: Think Aloud Protocol

We will use the "think aloud" method. This means:

- Participants speak their thoughts out loud while using the dashboard
- We learn what confuses them and what works well
- We do not help them unless they are completely stuck
- We take notes on everything they say and do

Each testing session will take 15-20 minutes:

- 2 minutes: Introduction and explanation
- 12 minutes: Four tasks (about 3 minutes each)
- 3 minutes: Questionnaire
- 3 minutes: Open feedback discussion

What We Will Record:

Quantitative data (numbers):

- Time taken to complete each task (seconds)
- Success rate (completed correctly or not)
- Number of errors or wrong clicks
- Number of times user got confused or stuck

Qualitative data (observations):

- What participants say during tasks
- Where they look first
- What confuses them
- What they like or dislike
- Suggestions for improvement

Testing Tasks

We designed four tasks that test the main features and research questions of our dashboard.

Task 1: Find Information (Exploration)

Instructions we will give: "Please find which Australian state has the highest enforcement intensity. This means which state issues the most traffic fines. Tell me the answer when you find it."

What we are testing:

- Can participants find the Maps page?
- Can they understand the Q2 bubble map?
- Can they correctly interpret bubble size?
- Do they use tooltips to verify their answer?

Success criteria:

- Participant identifies NSW as the state with most fines
- Completes task in under 60 seconds
- Does not need help or hints

Expected behavior: We expect most participants to:

1. Look for navigation menu
2. Click "Maps" link
3. Look at Q2 bubble map
4. Identify NSW as largest bubble
5. Hover to confirm with tooltip

Task 2: Use Filters (Filtering)

Instructions we will give: "Please filter the data to show only Queensland, and only the years 2022 to 2024. Tell me when you are done."

What we are testing:

- Can participants find the filter panel?
- Can they select from dropdown menu?
- Can they enter year values correctly?
- Do they understand the "Apply Filters" button?

Success criteria:

- Participant selects Queensland from state dropdown
- Enters 2022 in "From Year" field
- Enters 2024 in "To Year" field
- Clicks "Apply Filters" button
- Completes task in under 90 seconds

Expected challenges: Some participants might:

- Forget to click "Apply Filters" button
- Try to select multiple states when only one is needed
- Type the year in wrong field

We will note how many participants experience each problem.

Task 3: Compare Data (Comparison)

Instructions we will give: "Please compare Northern Territory and Australian Capital Territory. Tell me which one has more traffic fines and which one has more road fatalities."

What we are testing:

- Can participants navigate to Q7 chart?
- Can they understand the dual-axis chart?
- Can they extract comparative information?
- Do the two y-axes confuse them?

Success criteria:

- Participant finds Q7 chart
- Identifies that NT has more fines than ACT
- Identifies that NT has more fatalities than ACT
- Completes task in under 120 seconds

Expected challenges: Dual-axis charts can be confusing. We expect some participants might:

- Not understand which axis is which
- Mix up bars (fines) and line (fatalities)
- Need to hover for tooltips to understand

This will tell us if we need to make the axis labels clearer.

Task 4: Find Patterns (Pattern Discovery)

Instructions we will give: "Please look at the data and tell me which age group receives the most speed fines in Australia."

What we are testing:

- Can participants find Q3 heatmap?
- Can they read and interpret color intensity?
- Can they identify the darkest cell?
- Do they understand what the heatmap shows?

Success criteria:

- Participant finds Q3 heatmap
- Identifies 40-64 age group
- Explains how they found the answer (darkest color)
- Completes task in under 90 seconds

Expected behavior: We expect participants to:

1. Scroll down to find heatmap
2. Look for darkest blue color
3. Check which age group (row) it is in
4. Verify it is in "Speed Fines" column

Post-Test Questionnaire

After completing all four tasks, participants will fill out a questionnaire. We will ask them to rate their agreement with statements using a 5-point scale:

- 1 = Strongly Disagree
- 2 = Disagree
- 3 = Neutral
- 4 = Agree
- 5 = Strongly Agree

Questions:

1. The dashboard was easy to navigate
2. The visualizations clearly showed the data patterns
3. The filter controls were easy to use
4. The tooltips provided helpful information
5. I would use this dashboard to explore road safety data
6. The dashboard worked well on the computer
7. The colors and design are professional
8. I could complete the tasks without confusion

We will calculate the average score for each question to identify strengths and weaknesses.

Open-Ended Feedback Questions

After the rating questions, we will ask participants to write short answers:

1. **What did you like most about the dashboard?**
   - This tells us what is working well
   - We can emphasize these strengths
2. **What was most confusing or difficult to use?**
   - This tells us what needs improvement
   - We prioritize fixing these issues
3. **What improvements would you suggest?**
   - This gives us ideas for future enhancements
   - Participants might think of features we did not consider
4. **Any other comments or feedback?**
   - Open space for anything else
   - Sometimes the best insights come from this question

Expected Outcomes and Improvements

Based on our internal testing and feedback from friends, we have some expectations about what the formal testing will reveal.

Things We Expect Will Work Well:

- Navigation should be clear because we have obvious menu links
- Most tasks should be completed successfully
- Tooltips will help participants understand the data
- Charts will be intuitive for people who understand basic visualization

Potential Problems We Might Find:

1. **Dual-axis chart might confuse some users**
   - If 3+ participants struggle with Q7, we will make axis labels bigger and clearer
   - We might add a small explanation text below the chart
2. **Some users might forget to click "Apply Filters"**
   - If this happens often, we might make the button more prominent
   - We could change the button color to stand out more
3. **Heatmap might need a legend**
   - If participants do not understand the color scale, we will add a small legend
   - The legend would show "Fewer Fines" (light blue) to "More Fines" (dark blue)
4. **First-time users might need help getting started**
   - If many participants feel lost at first, we might add a help section
   - We could create a "How to Use" guide on the About page

Screenshot Placeholder: Planned improvements [Image: Mock-up showing clearer axis labels and color legend]

Changes We Will Make After Testing

After completing user testing in Week 13, we will:

1. **Analyze all the results**
    - Count how many participants completed each task
    - Calculate average times and success rates
    - Look for common problems multiple people experienced
2. **Prioritize improvements**
    - Fix issues that affected most participants first
    - Make changes that are quick and have big impact
    - Document why we made each change
3. **Implement changes**
    - Update code to fix identified problems
    - Test changes to make sure they work
    - Verify improvements actually help
4. **Update documentation**
    - Add user testing results to final design book
    - Include before/after comparisons
    - Explain what we learned and what we changed

Testing Materials Prepared

We have already prepared everything we need for Week 13 testing:

Task instructions printed on cards

- Clear, simple language
- One task per card
- Easy for participants to read

Consent form for participants

- Explains what we are testing
- Gets permission to record observations
- Assures confidentiality

Observation sheet for taking notes

- Space to record times
- Checkboxes for success/failure
- Area for written observations

Post-test questionnaire printed

- Rating scales clearly marked
- Space for written comments

- Easy to fill out quickly

Testing computer prepared

- Dashboard loaded and ready
- Browser cleared (no cached data)
- Screen recording software ready

Screenshot Placeholder: Testing materials [Image: Task cards, questionnaire, and observation sheet laid out]

6. Conclusion and Future Improvements

6.1 Project Summary

We have successfully developed an interactive dashboard that visualizes Australian road safety enforcement data alongside safety outcomes. The dashboard answers seven research questions through carefully designed visualizations that make complex data patterns accessible to diverse users.

Our final deliverable includes:

- Seven interactive D3.js visualizations (donut chart, bubble map, heatmap, line chart, choropleth map, stacked bar chart, and dual-axis chart)
- A responsive website that works on desktop, tablet, and mobile devices
- Advanced filtering capabilities allowing exploration by state and year
- Comprehensive data processing workflows in KNIME
- Accessible design features supporting keyboard navigation and screen readers

Current Status (Stand Up 3 - Week 12)

75% Complete - All must-have features are implemented and functional

- Data processing: 100% complete (KNIME workflows finalized)
- Visualizations: 100% complete (all 7 charts coded and interactive)
- Website: 90% complete (4 pages functional, minor polish needed)
- Documentation: 75% complete (design book updated, needs validation section)

Remaining Work (Week 13):

- User testing with 3-5 participants
- Final polish based on user feedback
- Complete validation section in design book
- Final report proofreading and formatting
- Create 6-minute video presentation
- Prepare PowerPoint for final stand-up

6.2 Key Findings

Through our data exploration and visualization work, we uncovered several important patterns in Australian road safety:

Enforcement Patterns:

- Speed violations account for 91% of all traffic fines, far exceeding other offence types
- NSW and Victoria issue the most fines in absolute numbers, reflecting their larger populations
- The 40-64 age group receives the most fines across all offence categories
- Automated detection cameras issue 85% of all fines, showing the importance of technology in enforcement

Safety Outcomes:

- Road fatalities showed an upward trend from 2020 to 2024, indicating worsening safety outcomes
- Male drivers account for 72% of road fatalities
- Single vehicle crashes cause 55% of all deaths
- Motorcycle riders and pedestrians are overrepresented in fatality statistics relative to their mode share

Enforcement-Outcome Relationship:

- States with higher enforcement intensity do not necessarily have lower fatality rates
- The relationship between fines and safety outcomes is complex and not directly correlated
- This suggests enforcement alone may not be sufficient to improve road safety

6.3 Lessons Learned

Technical Lessons:

Working with D3.js taught us the importance of understanding data-driven visualizations. Initially, we struggled with binding data to DOM elements and creating smooth transitions. Through practice and consulting documentation, we became comfortable with D3's enter, update, and exit pattern.

We learned that data quality matters more than we initially thought. Spending extra time in Week 9-10 to clean and standardize our datasets made the visualization phase much smoother. If we had rushed this step, we would have faced many problems later.

Responsive design is harder than it appears. Creating visualizations that work well on both desktop and mobile required significant additional effort. We learned to test on real devices early rather than relying only on browser developer tools.

Design Lessons:

Simplicity is more effective than complexity. Our initial wireframes tried to show too much information at once. After receiving feedback and simplifying our layout into clear sections, the dashboard became much more usable.

User testing will reveal issues we never noticed during development. Although we have not conducted formal external testing yet (planned for Week 13), our internal testing already identified several usability problems that we fixed.

Accessibility should be built in from the start, not added later. We made the mistake of initially focusing only on visual design. Adding ARIA labels and keyboard navigation afterwards required refactoring code. In future projects, we would consider accessibility from day one.

Collaboration Lessons:

Clear task division helped us work efficiently. Markus focused on data processing in KNIME, Damian implemented the visualizations in D3.js, and Jack built the website structure. This specialization allowed each team member to develop deep expertise in their area.

Regular communication prevented duplicated work. We held brief standup meetings three times per week on Discord to share progress and discuss blockers. This kept everyone aligned despite working independently most of the time.

Code review improved our code quality. Before merging any branch to main, another team member reviewed the code. This caught several bugs and ensured consistent coding style across the project.

Git workflow takes practice. In Week 9, we had several merge conflicts that took time to resolve. After establishing a clearer branching strategy (feature branches for each visualization), collaboration became smoother.

Process Lessons:

Starting early matters. Because we began data collection in Week 9, we had time to iterate through multiple design versions. Teams that started later had to rush and produced lower quality work.

Testing in multiple browsers and devices is essential. We discovered several Safari-specific CSS bugs that we never would have found testing only in Chrome.

Documentation helps your future self. When we needed to modify our KNIME workflows in Week 12, the detailed node annotations we added in Week 10 saved us significant time.

6.4 Future Improvements

If we had more time or resources to continue developing this project, we would implement several enhancements:

Data Enhancements:

- Extend historical data back to 2010 for longer-term trend analysis
- Add crash cause data to directly link enforcement priorities to crash outcomes
- Include road type information (highway, urban street, rural road)
- Incorporate population demographics by age and location

Visualization Enhancements:

- Add animated temporal transitions showing how data evolved year by year
- Implement drill-down details with modal popups for state-level breakdowns
- Create comparison mode for side-by-side state comparisons
- Add correlation analysis visualization with statistical coefficients

Interaction Enhancements:

- Allow users to save and share filter configurations via URL parameters
- Add data export functionality (CSV/PNG downloads)
- Implement advanced filtering by offence type, age group, and crash type
- Create search functionality for natural language queries

Technical Enhancements:

- Optimize performance for larger datasets with backend aggregation
- Connect to BITRE API for real-time data updates
- Convert to Progressive Web App for offline functionality

User Experience Enhancements:

- Create onboarding tutorial for first-time users
- Add contextual help icons throughout the interface
- Implement dark mode option
- Allow customizable dashboard layouts

Final Reflections

This project demonstrated that creating effective data visualizations requires balancing technical implementation, design principles, and user needs. The most technically impressive visualization is useless if users cannot understand it or if it fails to answer questions they actually care about.

We learned that data visualization is an iterative process. Our first designs were never perfect, and only through testing, feedback, and refinement did we create a dashboard that truly serves its purpose. This experience taught us the importance of staying flexible and being willing to revise our work based on evidence.

The project also showed us that working with real-world data is messy. Unlike clean academic datasets, our government data required extensive cleaning, standardization, and validation. This taught us patience and the importance of thorough documentation.

We are proud of what we accomplished and confident that our dashboard meets the project requirements and makes a genuine contribution to understanding Australian road safety data.

7. References

8. Appendices

Appendix A: Gen AI Declaration

Appendix B: Complete Data Dictionary

Appendix C: KNIME Workflow Documentation

Appendix D: Additional Screenshots