CITS5504 Data Warehousing

Project 1 - Data Warehousing Design, Association Rule Mining

Zhenlong Ma [23811627], Xiheng Li [22853272]

Contents

1 Introduction and Motivation

Road safety is a major concern in many countries around the world. According to the European Commission's 2023 report, Australia has a road fatality rate of 4.8 deaths per 100,000 people—more than double that of Iceland, which has the world's safest roads at 2.1 deaths per 100,000 people [1]. In July 2024, Australia had recorded 761 deaths, with a 12-month total of 1327 deaths - representing an increase 10% from the previous year [2].

These alarming statistics emphasize the urgent need for data-driven strategies to understand and mitigate fatal road accidents in Australia. Governments, transport authorities, and policy makers need accurate and interpretable information to support decisions that can improve road safety and reduce fatalities.

This project aims to contribute to that goal by developing a data warehouse system based on real-world crash and fatality data sets from the Australian Road Deaths Database (ARDD). Through this system, we support the following.

- **Historical data integration** via a well-designed data warehouse schema.
- **Data analytics and visualization** using multidimensional queries and dashboards.
- **Association rule mining** to uncover hidden patterns and correlations related to fatal crashes.
- **Policy recommendations** based on data insights and results of rule mining.

Integration of data warehousing and data mining techniques empowers stakeholders with actionable insights to support more informed and effective road safety interventions.

1.1 Overview of Datasets

This project uses a range of real-world datasets related to traffic fatalities in Australia. The core datasets are as follows:

- **Fatal Crashes — December 2024**: Provides detailed records of crash events including location, vehicle involvement, time, and environmental factors.
- **Fatalities — December 2024**: Contains individual-level fatality data such as age, gender, road user type, and involvement in the crash.

To enrich the analysis, we also incorporate the following demographic and geographic datasets:

- **Dwelling Count Data (2021)**: Extracted from the Australian Bureau of Statistics (ABS) TableBuilder, this dataset provides the number of dwellings by Local Government Area (LGA).
- **Population Estimates (2001–2023)**: Published by ABS, this data set includes population data by LGA, significant urban area (SUA) and remoteness area, which helps contextualize mortality rates.

All datasets are cleaned and integrated into a unified data warehouse schema to support multi-dimensional analysis and data mining.

2 Business Queries, Dimension Hierarchies, and StarNet Footprints

To support road safety policy and insight-driven decision-making, we developed a data warehouse with a Galaxy Schema structure, using two central fact tables and eight shared dimension tables. In this section, we present the business queries our data warehouse can support, along with the associated dimensions and measures. This is followed by the analysis of concept hierarchies and the rationale behind the schema design.

2.1 Business Questions

- **Query 1:** What is the number of fatalities for different combinations of time of day and road type?
- **Query 2:** Which age groups and road user roles (e.g., pedestrian, driver) are most associated with fatalities?
- **Query 3:** Which months have the highest number of fatal crashes, and is there a seasonal pattern? Would analyzing time of day enhance this insight?
- **Query 4:** During holidays such as Christmas and Easter, which road types and time periods see more fatal crashes?
- **Query 5:** Are there significant differences in road safety levels across Australian states?
- **Query 6:** How have fatalities from different types of crashes changed over time (by year) across different speed categories?

2.2 Business Query Mapping

Table 1: Business Queries, Keywords, Dimensions and Measures

Queries	**Keywords**	**Potential Dimensions**	**Measurements**
Query 1	Time of day, Road type	Time, Road	Number of Fatalities
Query 2	Age group, Road user	Person	Number of Fatalities
Query 3	Month, Time of day	Time, Date	Number of Fatalities
Query 4	Holiday, Time of day, Road type	Holiday, Time, Road	Number of Fatalities
Query 5	State, Fatality Rate	Location	Death Rate per 100k
Query 6	Speed category, Crash type (Single, Multiple), year	Speed, Crash Type, date	Number of Fatalities

2.3 Dimension Analysis and Concept Hierarchies

There are 8 dimension tables. Each dimension table was carefully designed to support OLAP operations such as roll-up, drill-down, slicing, and dicing. Below we present the concept hierarchy and role of each dimension in the analytical process.

1. `dim_location` This dimension provides geographic context to crashes. It supports two structured hierarchies—`state` → `sa4_name` and `state` → `lga_name` —as well as one flat hierarchy based on `remoteness_area`, which categorizes regions into major cities, inner regional, outer regional, remote, and very remote [3].

Here, `state` refers to the Australian state or territory, `sa4_name` corresponds to the Statistical Area Level 4 defined by the ABS for broad regional analysis [4], and `lga_name` refers to the Local Government Area for fine-grained local insights [5]. Although `remoteness_area` is not part of a drill-down hierarchy, it serves as an independent classification dimension useful for cross-sectional comparison between urban and rural zones.

In addition to hierarchical attributes, this dimension includes several numeric attributes: `dwelling_records`, `population_2023_lga`, and `population_2023_remoteness`. These quantitative fields provide population and housing context to support demographic normalization and rate calculations [6; 7].

From the perspective of the StarNet schema, this dimension gives rise to three analytical paths: `All` → `state` → `sa4_name`, `All` → `state` → `lga_name`, and `All` → `remoteness_area`, each supporting different levels of geographic granularity.

2. `dim_crash_type` This dimension represents a flat classification of crash structures, such as "Single Vehicle" or "Multiple Vehicle." Unlike hierarchical dimensions, `crash_type` contains mutually exclusive categories with no inherent drill-down levels. It supports analytical queries that aim to identify which crash types are associated with higher fatality counts and which may warrant targeted policy interventions.

In the StarNet model, this dimension follows a simple path: `All` → `crash_type`, enabling straightforward comparisons across different crash categories without requiring further hierarchical expansion.

3. `dim_date` This dimension supports temporal analysis of crash events. It contains two concept hierarchies:

- `year` → `quarter` → `month`, which enables exploration of long-term and seasonal patterns;
- `day_type` → `day_of_week_name`, which distinguishes between weekdays and weekends for analyzing weekly trends.

These temporal hierarchies allow users to analyze crash patterns across different time resolutions, from yearly trends to day-of-week behavior.

4. `dim_holiday` This dimension captures binary indicators of whether a crash occurred during the `christmas_period` or the `easter_period`. As holiday seasons typically feature increased travel volumes, this dimension is important for queries assessing elevated risks during public holidays and informing targeted policy interventions.

Although the two Boolean fields themselves (`christmas_period`, `easter_period`) do not form a hierarchical structure, the surrogate key `holiday_id` effectively represents distinct holiday states. For example, `holiday_id` = 1 corresponds to the Christmas period (Christmas = `true`, Easter = `false`), `holiday_id` = 2 denotes a non-holiday period (both = `false`), and `holiday_id` = 3 marks the Easter period (Easter = `true`, Christmas = `false`).

In the StarNet model, `holiday_id` serves as a flat dimension with the hierarchy path: `All` → `holiday_id`, enabling comparisons across different holiday conditions without nested levels.

`5. dim_person` This dimension describes individual-level attributes, including raw `age`, derived `age_group` (e.g., 0–17, 18–25, etc.), `gender`, and `road_user` (e.g., pedestrian, passenger, driver). It enables analysis of demographic risk groups, helping to identify vulnerable populations in road safety.

A concept hierarchy exists between `age` and `age_group`, where continuous age values are grouped into predefined intervals for aggregate-level analysis. In addition to the age-based hierarchy, `gender` and `road_user` each form independent flat dimensions. These attributes are mutually exclusive and do not possess internal drill-down levels, but they play critical roles in classification and filtering during analytical queries.

In the StarNet schema, this dimension contributes to three separate dimensional paths:

- `age` → `age_group`→ `All`
- `gender` → `All`
- `road_user` → `All`

These parallel dimensional lines allow analysts to explore fatality patterns by age segments, gender categories, and user roles independently or in combination with other dimensions.

`6. dim_time` This time-of-day dimension classifies crashes into broader periods such as “Morning,” “Evening,” or “Night.” It supports the detection of peak-risk periods within a day and is particularly useful when cross-analyzed with road type or user role. As it is a flat dimension without internal drill-down levels, it forms a simple hierarchy in the StarNet schema from `time_of_day` → `All`, enabling aggregation of crash records either by specific time periods or at the daily total level.

`7. dim_vehicle` This dimension describes the involvement of high-risk vehicles such as buses, heavy rigid trucks, and articulated trucks. Each vehicle type is represented by a boolean flag indicating whether it was involved in a given crash. Since these boolean fields (`bus_involvement`, `heavy_rigid_truck_involvement`, `articulated_truck_involvement`) do not form a drill-down hierarchy, they cannot directly be modeled as hierarchical levels. Instead, the surrogate key `vehicle_id` is used as a flat dimension that uniquely represents each possible combination of these involvement flags. In the StarNet schema, this results in a simple hierarchy: `vehicle_id` → `All`, enabling aggregated analysis by vehicle involvement patterns across the dataset.

`8. dim_road` This dimension captures two core attributes: `road_type` (e.g., highway, urban, rural) and the numeric `speed_limit`, which is further categorized into `speed_category` (Very Low, Low, Medium, High, Very High). A concept hierarchy exists between `speed_limit` and `speed_category`, where continuous speed values are grouped into broader descriptive zones to facilitate comparison and policy-oriented analysis.

The classification is based on commonly enforced speed zones across Australian states [8]. For instance, school zones and pedestrian-heavy areas often fall below 30 km/h (Very Low), residential areas are generally capped at 50 km/h (Low), urban arterial roads range between 60–70 km/h (Medium), and rural roads or state highways typically range from 80–100 km/h (High). Freeways and remote roads may reach up to 110–130 km/h, thus defined as Very High.

As such, the `dim_road` table contributes two analytical dimensions in the StarNet schema. The first is a concept hierarchy: `speed_limit` → `speed_category`→ `All`; the second is a flat classification through road_type→ All. These enable multidimensional analysis of crash risk patterns by infrastructure type and traffic speed environment.

Fact Tables: fact_fatal_crash and fact_person_fatality The data warehouse adopts a Galaxy Schema with two core fact tables at different levels of granularity.

9. fact_fatal_crash stores one record per crash event, capturing aggregated attributes such as number_fatalities and crash_count. It links to several dimension tables including dim_date, dim_holiday, dim_location, dim_road, dim_vehicle, and dim_crash_type.

10. fact_person_fatality, in contrast, provides individual-level detail with one record per fatality. It enables detailed demographic and contextual analysis by linking to additional dimensions such as dim_person and dim_time, in addition to those shared with fact_fatal_crash.

This dual-grain design provides analytical flexibility: high-level metrics and trends can be drawn from fact_fatal_crash, while deeper insights into victim characteristics and contributing factors can be derived from fact_person_fatality.

2.4 StarNet diagrams and the query footprints

Based on the previously defined concept hierarchies, the StarNet diagram constructed for this project consists of 14 distinct analytical lines, each representing a dimension path rooted in the fact tables and supporting aggregation or slicing operations for business analysis.

These 14 paths, organized from the most granular level up to All, are:

- **From dim_date:**
  - month → quarter → year → All
  - day_type → day_of_week_name→ All
- **From dim_location:**
  - lga_name → state → All
  - sa4_name → state → All
  - remoteness_area → All
- **From dim_road:**
  - speed_limit → speed_category → All
  - road_type → All
- **From dim_person:**
  - age → age_group → All
  - gender → All
  - road_user → All
- **From dim_time:**
  - time_of_day → All
- **From dim_crash_type:**

  - crash_type → All

- **From** dim_holiday:
  - holiday_id → All
- **From** dim_vehicle:
  - vehicle_id → All

Each analytical line reflects either a structured hierarchy or a flat dimension used in the queries throughout this project. By organizing dimensions from the lowest level of detail up to the most general, the StarNet diagram facilitates efficient OLAP-style exploration and clearly illustrates the dimensional footprint of the business questions addressed.

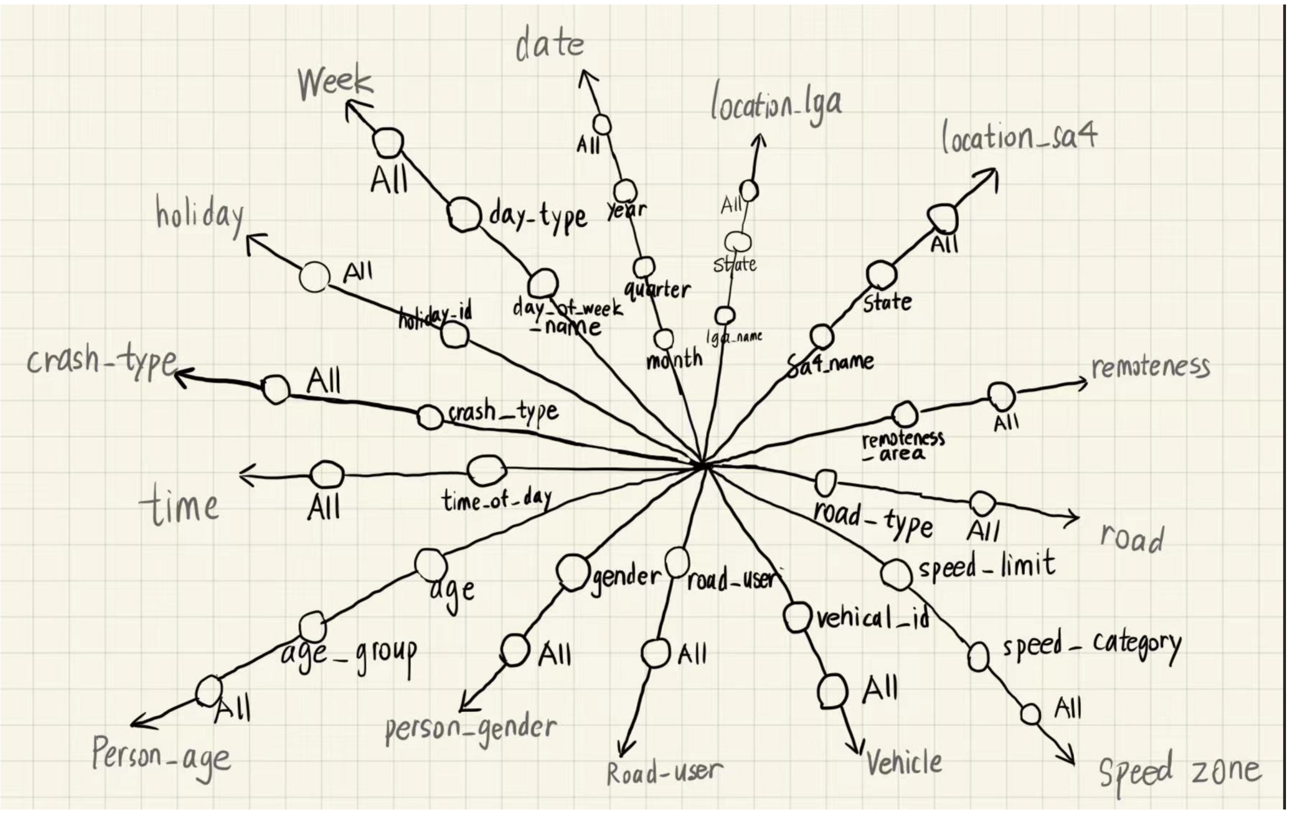


Figure 1: Starnet diagram

Query 1: What is the number of fatalities for different combinations of time of day and road type?

To address this query, we utilized the `road_type` field from the road dimension and the `time_of_day` field from the time dimension. For better data readability and focus, only records from the year 2024 were selected, which required filtering by the `year` attribute from the date dimension. All other dimensions were set to "All".

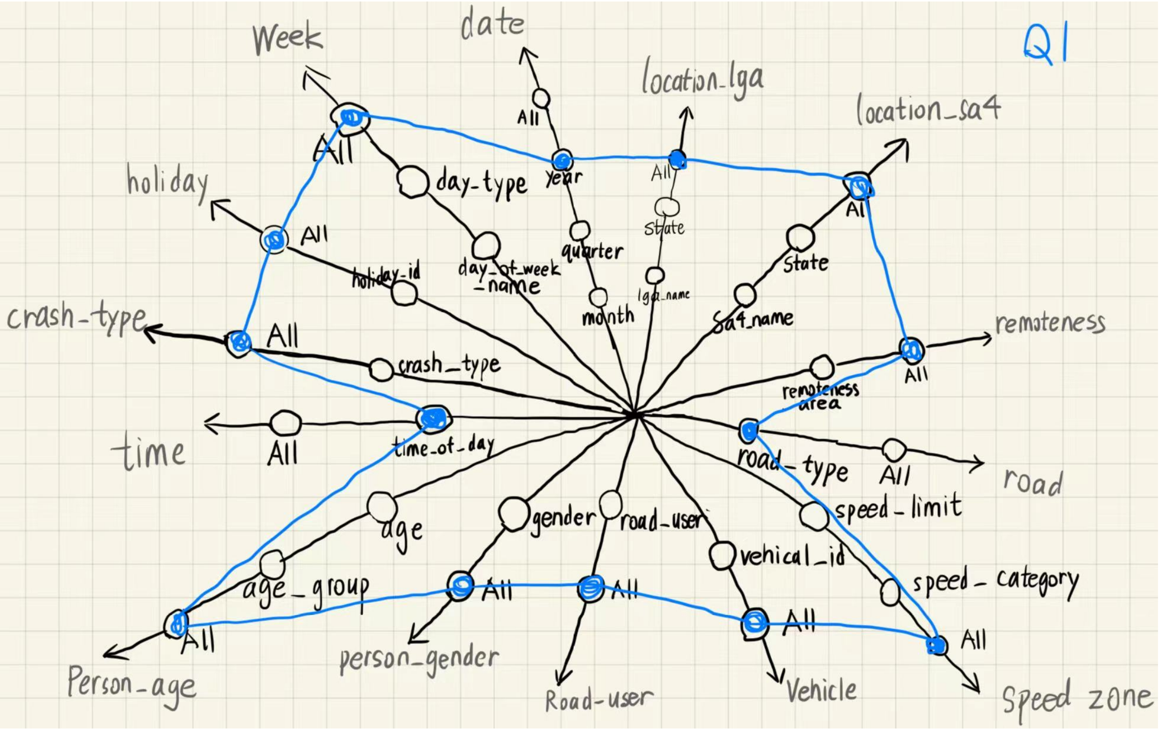


Figure 2: StarNet footprint for Query 1

Query 2: Which age groups and road user roles (e.g., pedestrian, driver) are most associated with fatalities?

This query utilizes the `road_user` and `age_group` fields from the person dimension to identify high-risk demographic and role-based patterns. To maintain temporal consistency and data clarity, the analysis is restricted to the year 2024 using the `year` attribute from the date dimension. All other dimensions were set to "All".

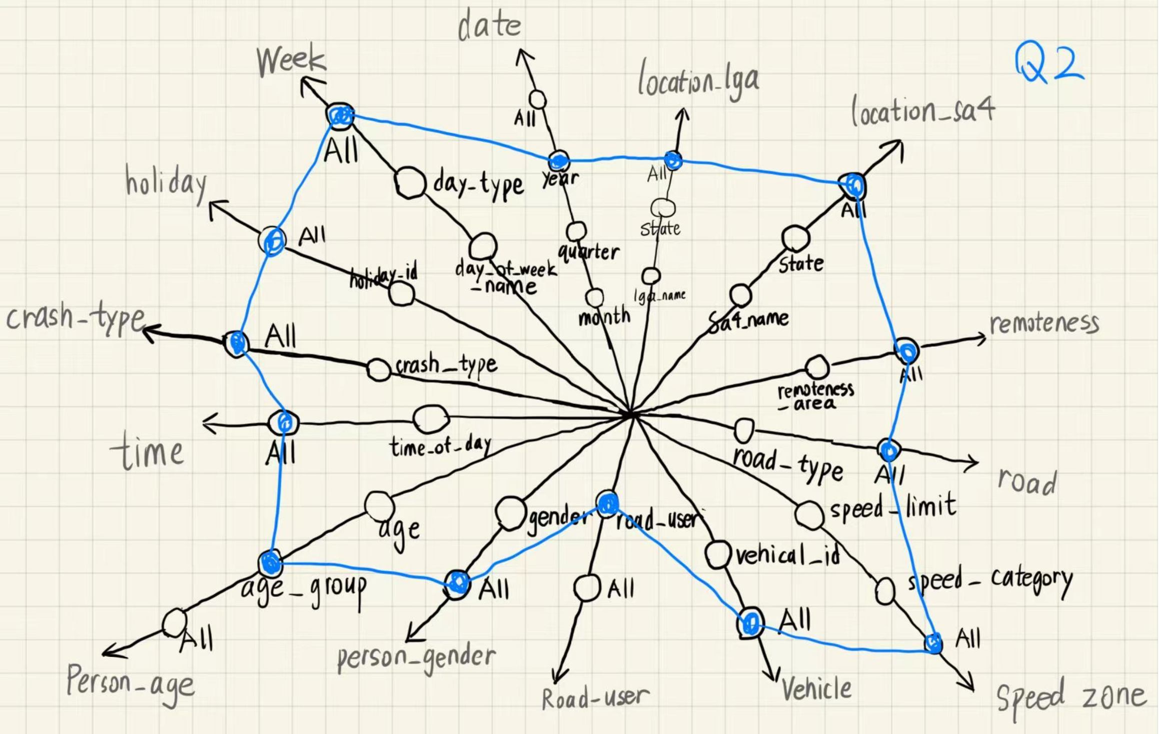


Figure 3: StarNet footprint for Query 2

Query 3: Which months have the highest number of fatal crashes, and is there a seasonal pattern? Would analyzing time of day enhance this insight?

This query uses the `time_of_day` field from the time dimension and the `month` field from the date dimension. Although the analysis is filtered to the year 2024, `month` is prioritized over `year` since it is a lower-level attribute. All other dimensions are set to “All”.

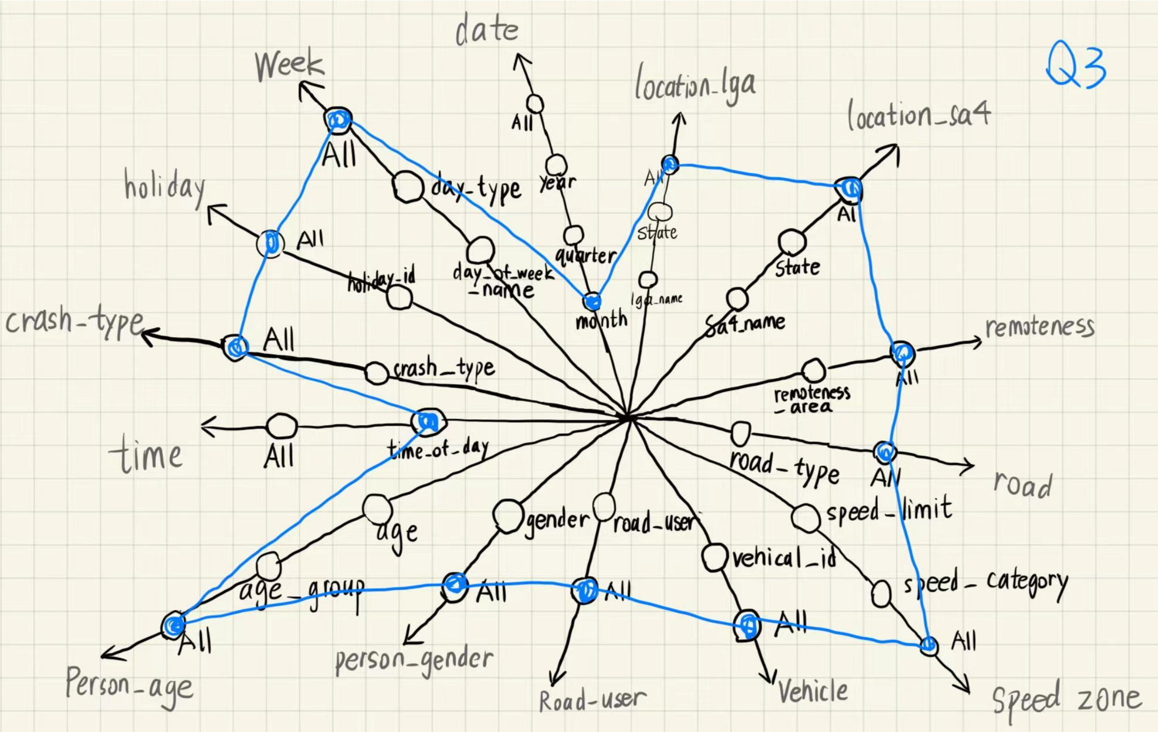


Figure 4: StarNet footprint for Query 3

Query 4: During holidays such as Christmas and Easter, which road types and time periods see more fatal crashes?

Since the query focuses on crashes during Christmas or Easter, records are filtered using `holiday_id` values 1 and 3, requiring selection of this node. Additionally, `road_type` from the road dimension and `time_of_day` from the time dimension are included. All other dimensions are set to “All”.

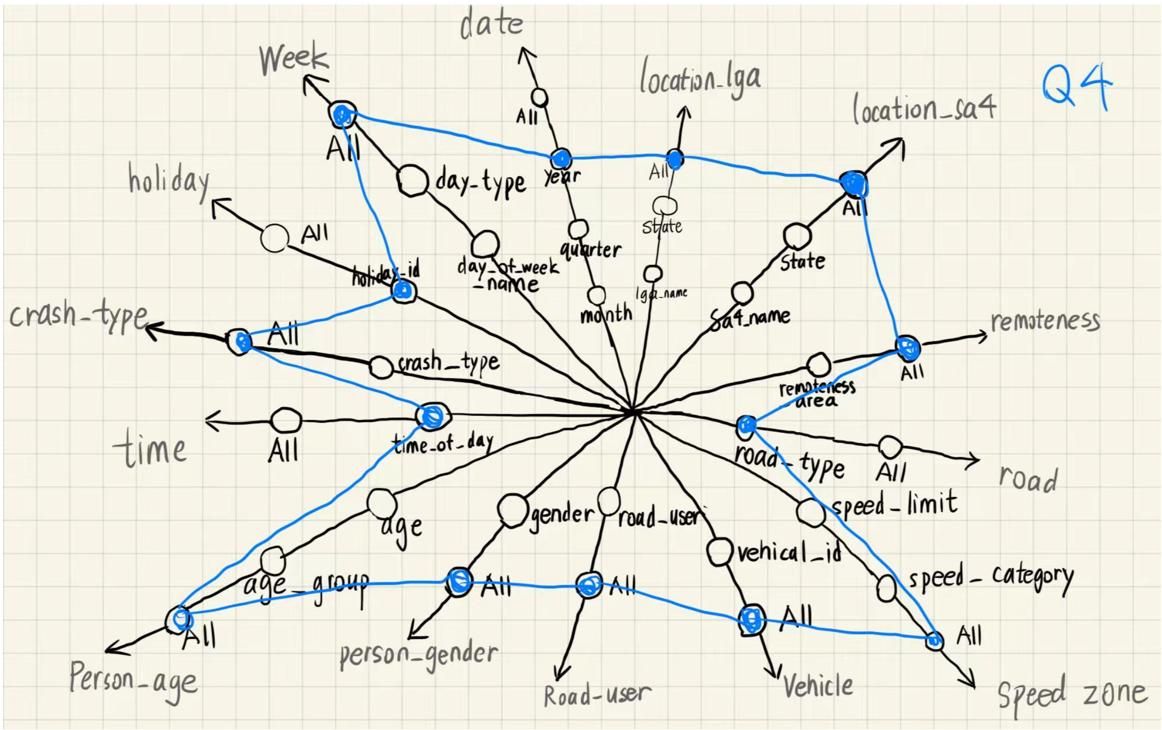


Figure 5: StarNet footprint for Query 4

Query 5: Are there significant differences in road safety levels across Australian states?

Since the latest population data in the resource table is for 2023, the analysis is limited to that year to calculate fatality rates. Therefore, the `year` node from the date dimension is selected. As the question focuses only on states, the `state` node is included from both location dimensions.

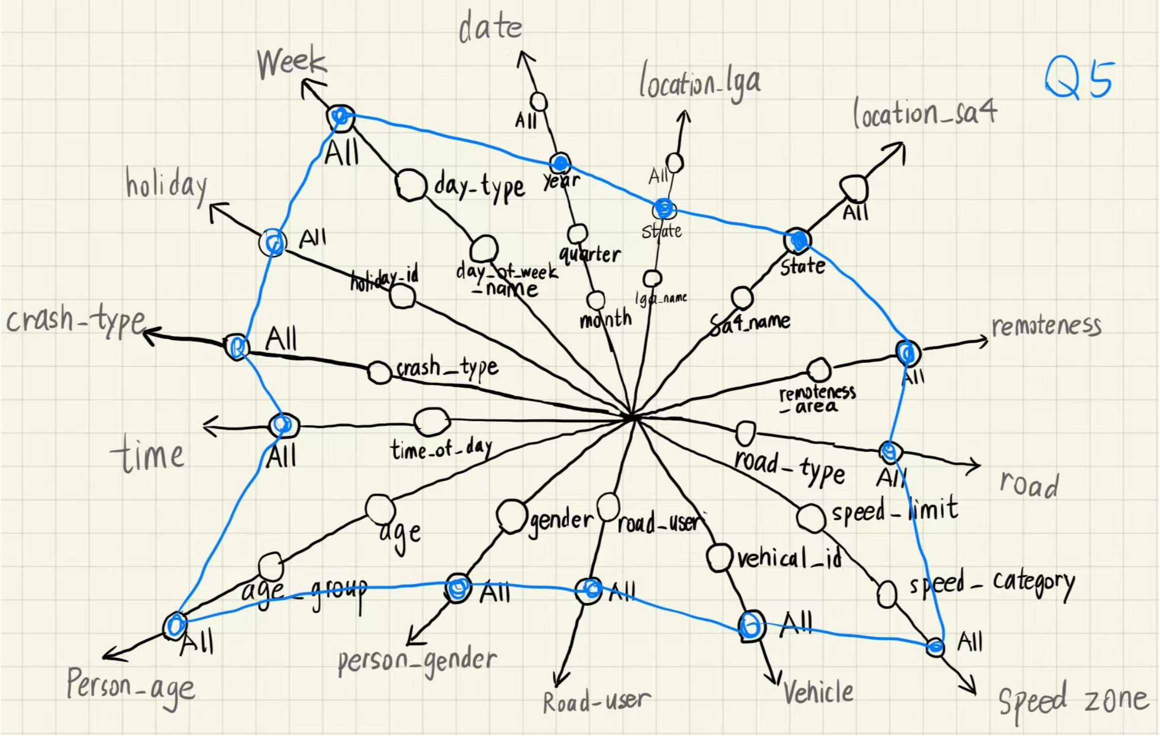


Figure 6: StarNet footprint for Query 5

Query 6: How have fatalities from different types of crashes changed over time (by year) across different speed categories?

This query investigates how fatalities change over time, making `year` from the date dimension the most appropriate node to select. It also includes `speed_category` from the speed zone dimension and `crash_type` from the crash type dimension. All other dimensions are set to "All".

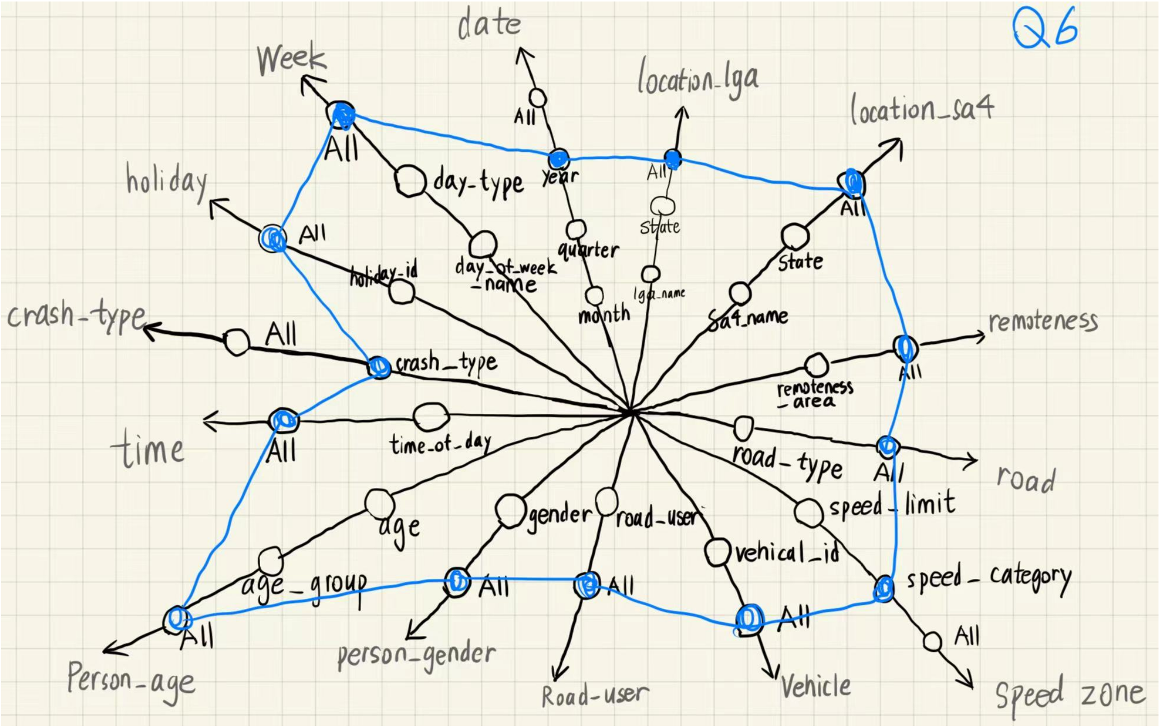


Figure 7: StarNet footprint for Query 6

2.5 Schema Design Rationale

Rather than employing a simple star schema, this project adopts a **Galaxy Schema** (also known as a Fact Constellation Schema) approach. In a galaxy schema, multiple fact tables — in our case, `fact_fatal_crash` and `fact_person_fatality` — share a set of conformed dimension tables such as `dim_location`, `dim_date`, and `dim_vehicle`.

This design supports analysis at multiple levels of granularity: the `fact_fatal_crash` table stores aggregated crash-level metrics (e.g., number of fatalities per crash), while the `fact_person_fatality` table enables individual-level analysis based on demographics, vehicle involvement, and time of day.

Adopting a galaxy schema improves flexibility and query performance in multidimensional analysis, promotes dimension reuse, and avoids data redundancy. According to Kimball and Ross, the galaxy schema is ideal for complex business processes that require multiple fact tables linked to shared dimensions [9].

3 Galaxy Schema and Dimensional Modeling

To accommodate multi-granular analysis of fatal road crashes in Australia, this project adopts a **Galaxy Schema**, also known as a *Fact Constellation Schema*. Unlike a traditional star schema with a single central fact table, the galaxy schema features multiple fact tables that share a set of conformed dimension tables [10]. This architecture is particularly suitable for analytical systems that need to integrate facts at different abstraction levels while maintaining consistency across shared dimensions.

In our implementation, the schema integrates two core fact tables:

- `fact_fatal_crash` — summarizing crash-level data such as number of fatalities and crash types.
- `fact_person_fatality` — representing individual-level fatality records with attributes like age, gender, and road user type.

Both fact tables are linked to a common set of dimension tables, including `dim_location`, `dim_road`, `dim_vehicle`, `dim_holiday`, `dim_date`, and `dim_time`. The consistent dimensional structure allows analysts to drill down from aggregated crash data to individual-level victim profiles or roll up patterns across geography and time.

Compared with a snowflake schema, which emphasizes normalization, or a pure star schema, which may duplicate dimension attributes across facts, the galaxy schema offers a balanced trade-off between flexibility and performance. It facilitates complex analytical use cases such as multi-fact association rule mining, comparative spatial analysis, and temporal risk segmentation without incurring excessive storage or join complexity [11].

The dimensional modeling follows Kimball's four-step methodology: selecting the business process (fatal crashes), determining the grain (crash- and victim-level), identifying conformed dimensions, and choosing measurable facts. This design ensures that the resulting data warehouse is both extensible and aligned with real-world analytical needs.

3.1 Design of Dimension Tables

Dimension tables are designed based on real-world entities and context descriptors. Each dimension captures specific perspectives of a crash or fatality event.

- **Dim_Location:** Captures geographic information such as state, LGA, SA4 region, and remoteness area. This dimension also incorporates population and dwelling data to enable spatial risk analysis.
- **Dim_Date:** Contains calendar attributes including year, month, quarter, and weekday name/type. This supports seasonal and temporal trend analysis.
- **Dim_Holiday:** Contains binary indicators for `christmas_period` and `easter_period`, enabling event-based filtering and holiday effect analysis.
- **Dim_Person:** Stores victim-related demographic information such as age, age group, gender, and role (e.g., pedestrian, driver).
- **Dim_Time:** Represents the classification of crashes based on time of day (e.g., morning, afternoon, night).
- **Dim_Vehicle:** Encodes whether certain heavy vehicles were involved in the crash (e.g., bus, rigid truck, articulated truck).
- **Dim_Road:** Stores contextual road conditions including road type, speed limit, and a derived speed category (e.g., Low, High, Very High).
- **Dim_CrashType:** Classifies the type of crash event (e.g., Single, Multiple).

Table 2: Dimension Table Design

Dimension Table	Attributes (Enumerated / Hierarchical Values)
dim_location	location_id — Primary Key state lga_name sa4_name remoteness_area (e.g., Major Cities, Inner Regional, Remote) population_2023_lga population_2023_remoteness dwelling_records
dim_vehicle	vehicle_id — Primary Key bus_involvement (Yes, No, Unknown) heavy_rigid_truck_involvement (Yes, No, Unknown) articulated_truck_involvement (Yes, No, Unknown)
dim_crash_type	crash_type_id — Primary Key crash_type (e.g., Single, Multiple)
dim_date	date_id — Primary Key year month quarter (1–4) day_of_week_name (e.g., Monday, Tuesday, ...) day_type (e.g., Weekday, Weekend)
dim_holiday	holiday_id — Primary Key christmas_period (True, False) easter_period (True, False)
dim_person	person_id — Primary Key age age_group (0–17, 18–25, 26–40, 41–65, 65+) gender (Male, Female, Unknown) road_user (Driver, Passenger, Pedestrian, Cyclist, Unknown)
dim_time	time_of_day_id — Primary Key time_of_day (e.g., Morning, Afternoon, Evening, Night)
dim_road	road_id — Primary Key road_type (e.g., Urban, Rural, Highway) speed_limit speed_category (Very Low, Low, Medium, High, Very High)

All dimension tables include surrogate keys to uniquely identify each row. Where appropriate, concept hierarchies are included to enable roll-up and drill-down in OLAP analysis.

3.2 Design of Fact Tables

Two fact tables are created to support multiple analytical perspectives across different granularities:

- **Fact_Fatal_Crash:** This table captures aggregated crash-level statistics. Each record corresponds to a unique crash event and references shared dimensions such as date, location, road, vehicle, and crash type. Two key measures are defined:

  - number_fatalities – total number of persons killed in a single crash.
  - crash_count – always equals 1, enabling aggregation and comparison of crash frequencies across dimensions.

- **Fact_Person_Fatality:** This table stores person-level records for each fatality. It links individual demographics and time-of-day attributes to the crash context. The fact table contains:

  - fatality_count – equals 1 per row, enabling disaggregated analysis across population segments (e.g., by age group or gender).

Table 3: Design of fact_fatal_crash Table

Primary Key	fact_crash_id
Foreign Keys	crash_id date_id holiday_id location_id road_id vehicle_id crash_type_id
Measures	number_fatalities crash_count

Table 4: Design of fact_person_fatality Table

Primary Key	fact_person_fatality_id
Foreign Keys	crash_id date_id holiday_id person_id location_id road_id vehicle_id crash_type_id time_of_day_id
Measure	fatality_count

By separating the facts into crash-level and person-level, and allowing them to share common dimensions, the model supports both aggregate and disaggregated insights without redundancy. This also enhances flexibility for OLAP operations such as roll-up, drill-down, and slice-and-dice analysis across various dimensions.

Figure 8 illustrates the galaxy schema structure, where multiple dimension tables are connected to both fact tables via foreign keys. The two green tables represent fact tables, while the orange tables represent dimension tables.

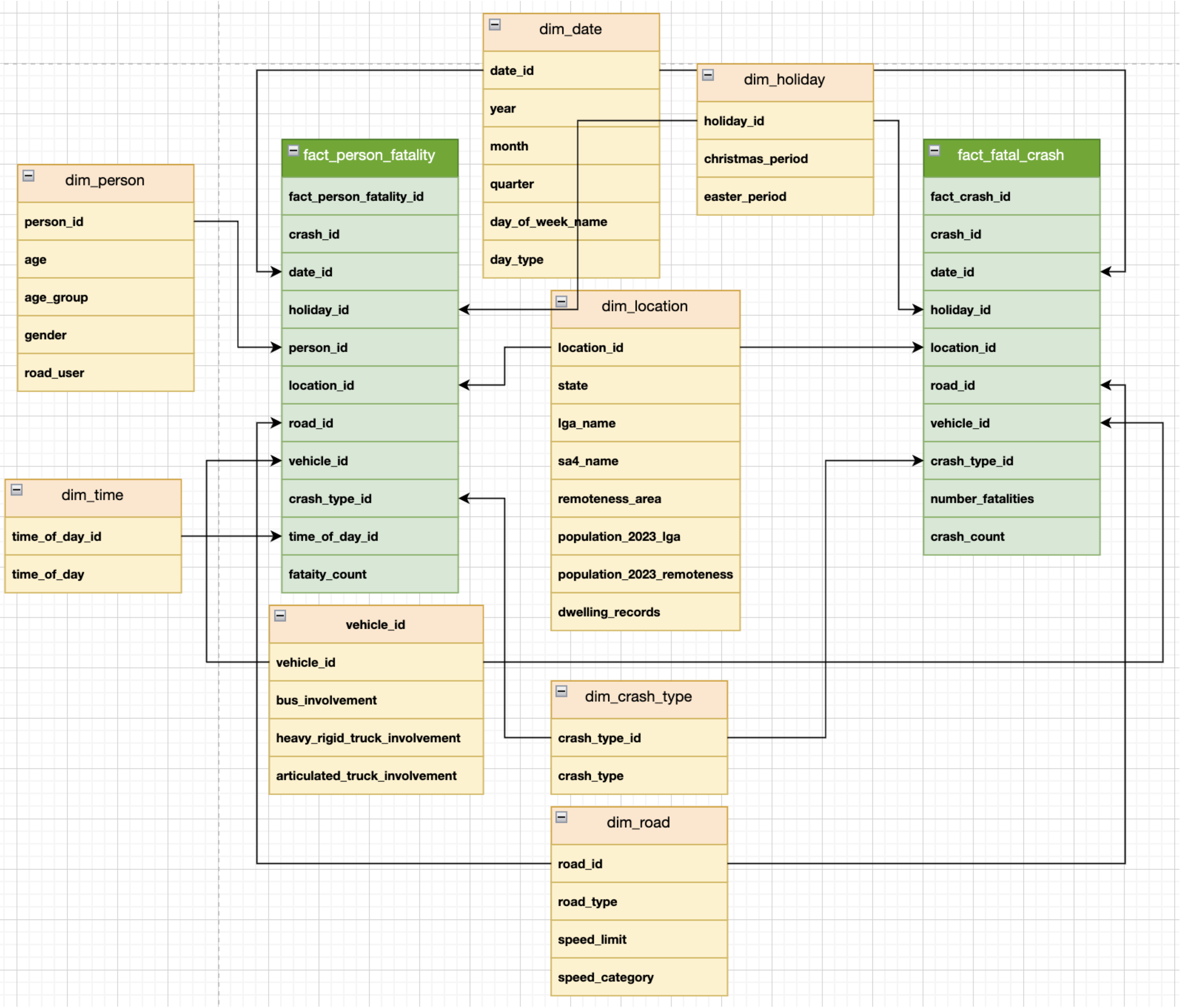


Figure 8: Galaxy Schema Design for Road Fatalities

This dimensional modeling framework provides the foundation for a robust and flexible data warehouse capable of supporting both standard reporting and advanced mining tasks.

4 Extraction, Transform, Load (ETL)

4.1 Data Extraction

Data extraction is the first phase of the ETL process, which involves retrieving raw data from diverse sources such as structured files, databases, or external repositories. It serves as the foundation of data warehousing by collecting accurate and consistent data to support subsequent transformation and analysis stages [9; 11].

In this project, data is sourced from official government datasets and census records relevant to road safety and population distribution in Australia. The primary data sources include:

- `Fatal_Crashes_December_2024.xlsx` – Crash-level information including time, location, and road conditions.
- `bitre_fatalities_dec2024.xlsx` – Individual-level fatality data such as age, gender, and road user type.
- `LGA (count of dwellings).csv` – Census-based dwelling counts by Local Government Area (LGA).
- `Population_estimates.xlsx` – Population estimates from 2001 to 2023 by LGA, remoteness area, and other geographic hierarchies.

The data is extracted using Python, specifically the `pandas` library, which provides powerful data handling capabilities. The `read_csv()` and `read_excel()` functions are utilized to import tabular data with preserved metadata, enabling reproducibility and scalability.

This extraction phase ensures that heterogeneous datasets are standardized and made ready for downstream data cleaning and integration.

4.2 Data Transformation

Data transformation is the second phase of the ETL pipeline, responsible for converting raw, often inconsistent data into a clean and standardized format suitable for downstream dimensional modeling. It plays a critical role in ensuring data quality, semantic consistency, and analytical usability [9].

In this project, transformation is centrally handled via a reusable Python function `common_clean_steps(df)`, which applies a suite of domain-informed preprocessing steps. This strategy is in line with best practices in data warehousing for centralized cleaning and reproducibility [11].

The transformation process consists of several key procedures:

- **Column Standardization:** All column names are converted to lowercase with underscores (_) replacing special characters, ensuring schema consistency across datasets.
- **Missing Value Handling:** Generic invalid values such as `"Unknown"`, `"-9"`, and `"Other/-9"` are treated as missing (`pd.NA`). Critical categorical columns like `gender` and `road_user` are defaulted to `"Unknown"` if no value is available.
- **Boolean Field Normalization:** String fields such as `bus_involvement` are mapped to Python boolean types, supporting binary analysis and dimensional modeling.
- **Speed Field Correction:** Special entries like `"<40"` are normalized to numeric values. Invalid or missing speed limits are dropped from the dataset to maintain referential integrity.
- **Filtering Based on Critical Fields:** Rows lacking essential information such as `age` or `time_of_day` are removed, as they cannot be reliably linked to dimensions.

The excerpt below shows a representative snippet of the cleaning function (see Listing 1):

```
df.columns = (
    df.columns
    .str.replace(r"[^\w]+", "_", regex=True)
    .str.replace(" ", "_")
    .str.strip("_")
    .str.lower()
)
df = df.dropna(axis=1, how='all')

for col in df.columns:
    if df[col].isin(["Unknown", "nan", "-9", -9]).sum() > 0:
        df[col] = df[col].replace(["Unknown", "nan", "-9", -9], pd.NA)
```

Listing 1: Key Cleaning Logic for Column and Missing Value Standardization

This function is applied uniformly to both crash-level and person-level datasets, promoting modularity and data quality. Boolean conversion, value normalization, and semantic filtering follow similar templates. A full implementation is included in the project code.

Tools Used:

- `pandas`: For tabular data manipulation, including renaming, filtering, and imputation.
- `NumPy` and `pd.NA`: For robust missing value handling.

Benefits: This approach ensures consistency, maintainability, and compatibility with the dimensional schema design that follows.

4.3 Dimension Table Generation

Dimension tables are central to star schema modeling, providing descriptive context for the facts stored in the warehouse. They enable users to ”slice and dice” facts by categories such as time, geography, or vehicle type, facilitating analytical exploration and multi-dimensional reporting [9]. Each dimension is constructed using cleaned and preprocessed data outputs from the transformation phase, ensuring referential integrity and semantic consistency [11].

In this project, the generation of dimension tables is implemented through modular Python functions, such as `generate_dim_location()`, `generate_dim_road()`, and others. These functions apply additional transformations where necessary—for example, creating derived attributes like `speed_category` or `age_group`, which serve as semantic abstractions on top of raw numerical values.

The design of each dimension table reflects a combination of domain knowledge and data engineering practices:

- **Dim_Location:** Combines hierarchical geographic attributes (state, LGA, SA4, remoteness area) with population and dwelling data. Derived from crash-level data and enriched with external ABS sources, it enables spatial aggregation and risk profiling across regions.
- **Dim_Road:** Standardizes road types and numeric speed limits, with a derived `speed_category` column to support semantic analysis. Transformation logic includes value correction from prior steps (e.g., handling `"<40"`).

- **Dim_Person:** Derived from fatality records, it includes individual-level features such as age, gender, and road user type. The `age_group` column is generated via binning to enable age-segment analysis.

- **Dim_Date:** Constructs temporal hierarchies including year, month, quarter, weekday, and `day_type` (weekend or weekday). The `day_of_week` field is inferred during transformation based on the `dayweek` input.

- **Dim_Holiday:** Encodes binary holiday flags (`christmas_period`, `easter_period`) extracted and normalized during cleaning.

- **Dim_Vehicle:** Identifies whether the crash involved specific vehicle types (bus, heavy truck, etc.), with values normalized from string labels to boolean flags during transformation.

- **Dim_Crash_Type:** Maps textual crash type descriptions (e.g., ”Single”, ”Multiple”) to surrogate keys.

- **Dim_Time:** Encodes time-of-day categories into unique IDs, allowing time-based aggregation.

A simplified illustration of the transformation logic for road dimensions is shown in Listing 2:

```python
def classify_speed_category(speed):
    if speed <= 30:
        return 'Very Low'
    elif speed <= 50:
        return 'Low'
    elif speed <= 70:
        return 'Medium'
    elif speed <= 100:
        return 'High'
    else:
        return 'Very High'

dim_road['speed_category'] = dim_road['speed_limit'].apply(classify_speed_category)
```

Listing 2: Speed Category Derivation in Dim_Road

All dimension tables are created as standalone DataFrames with surrogate primary keys (e.g., `location_id`, `road_id`, `person_id`). This design facilitates referential integrity and optimizes join operations with the fact tables.

In summary, the dimension table generation process builds upon the transformed data by introducing categorization, abstraction, and normalization—all critical steps for high-quality analytical processing in a data warehouse environment.

5 Data Loading

The final stage in the ETL process involves loading the cleaned and transformed datasets into a PostgreSQL relational database. This step marks the transition from temporary in-memory structures to persistent and queryable storage, enabling efficient OLAP-style analytics on structured schemas [11].

The loading process is designed to maintain referential integrity across all dimension and fact tables, ensuring consistency with the star schema developed earlier. It builds upon the outputs from the transformation phase, particularly the standardized column naming, missing value handling, and surrogate key generation [9].

The data loading workflow consists of the following procedural steps:

1. **Drop existing tables (if applicable):** To avoid conflicts from prior runs and enforce schema resets, tables are dropped in reverse dependency order.

2. **Create all tables:** Tables are generated from SQL DDL schemas defined in Python (`schemas.py`). These include both dimension and fact tables with all relevant primary and foreign keys.

3. **Insert cleaned data:** Cleaned CSVs, exported from the transformation phase, are loaded into PostgreSQL via the `insert_dataframe()` function using psycopg2.

4. **Preview or query data:** Tables can be queried for verification or used in further SQL analysis and dashboarding.

The creation of tables is based on an import order defined in `TABLE_IMPORT_ORDER`, which respects foreign key dependencies. An excerpt from this process is shown in Listing 3:

```python
def create_all_tables():
    for table in TABLE_IMPORT_ORDER:
        create_table(table, TABLE_SCHEMAS[table])
```

Listing 3: Creating All Tables Based on Defined Schema Order

The schema definitions for each table are maintained as DDL strings within the `schemas.py` file. A representative snippet is shown below (Listing 4):

```python
TABLE_SCHEMAS = {
  "dim_location": '''
    location_id SERIAL PRIMARY KEY,
    state VARCHAR(20),
    ...
  '''
}
```

Listing 4: Defining DDL for the ‘dim$_{l}$*ocation‘Table*

Once the tables are created, the cleaned CSV files are imported using a utility that first replaces `pd.NA` values with `None`, then formats the SQL insert statement dynamically:

```python
def insert_dataframe(table_name: str, df: pd.DataFrame):
    df = prepare_df_for_postgres(df)  # Replace missing values with None
    ...
    insert_many(insert_sql, data)
```

Listing 5: Importing Cleaned CSVs to PostgreSQL

Finally, a verification utility is provided to preview the loaded tables or export them for external review (Listing 6):

```python
def preview_all_tables(limit: int = None):
    for table_name in TABLE_SCHEMAS:
        ...
        df.to_csv(f"DB_files_export/{table_name}.csv", index=False)
```

Listing 6: Query and Preview PostgreSQL Tables

This approach ensures that the data warehouse is populated in a reproducible, schema-driven manner. Each transformation step prior to loading directly contributes to schema alignment and referential correctness in the PostgreSQL environment.

6 The Value of Data Cube in Contemporary Data Warehousing

6.1 Conceptual Overview and Benefits

The concept of the *data cube* is foundational to modern data warehousing and business intelligence systems. A data cube is a multi-dimensional array of values that allows efficient representation, exploration, and aggregation of data across multiple dimensions [12]. It serves as the computational and structural backbone of OLAP (Online Analytical Processing) systems, enabling users to perform fast and flexible analyses such as slicing, dicing, roll-up, and drill-down operations [13].

In contemporary data warehouse design, especially under the dimensional modeling paradigm popularized by Kimball, data cubes provide a semantic layer on top of the physical star schema. Fact tables store quantitative measures (e.g., number of fatalities, crash counts), and dimension tables provide contextual information (e.g., time, location, road type). The data cube organizes this model into a form that is intuitive for business users and optimized for analytical queries [9].

Key benefits of data cubes in current data warehousing environments include:

- **Multi-dimensional querying:** Users can explore data across multiple dimensions simultaneously, such as analyzing fatal crashes by location, time of day, and road type.
- **Aggregated insights:** Cubes enable pre-computation and caching of summaries, significantly accelerating dashboard performance in tools like Tableau or Power BI.
- **Enhanced interpretability:** The cube structure mirrors real-world business questions (e.g., ”Which road types are most dangerous on weekends?”), making insights more accessible to non-technical users.
- **Support for drill-down analysis:** Analysts can easily navigate from high-level summaries to granular views, which is essential in domains like road safety where root cause investigation is critical.

In this project, the constructed PostgreSQL data warehouse is cube-compatible through its well-designed star schema. The use of date, location, crash type, person demographics, and road features as dimensions enables a wide range of OLAP-style queries. This cube-compatible structure forms the basis for our business queries and Tableau dashboard visualizations.

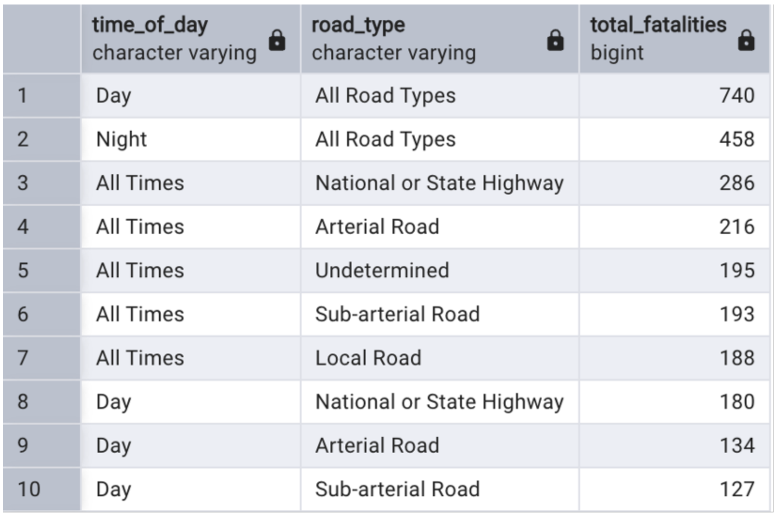
6.2 SQL-Based Cube Queries and Result Interpretation

Query 1: What is the number of fatalities for different combinations of time of day and road type? Analysis of fatalities by combinations of time of day and road type.

```sql
-- Select time of day and road type (with labels for aggregated rows), and compute total
    fatalities
SELECT
  COALESCE(t.time_of_day, 'All Times') AS time_of_day,     -- Replace NULL (from CUBE)
      with 'All Times' for readability
  COALESCE(r.road_type, 'All Road Types') AS road_type,     -- Replace NULL (from CUBE)
      with 'All Road Types'
  SUM(f.fatality_count) AS total_fatalities               -- Aggregate total fatalities
FROM fact_person_fatality f
-- Join with dimension tables to bring in descriptive attributes
JOIN dim_time t ON f.time_of_day_id = t.time_of_day_id
JOIN dim_road r ON f.road_id = r.road_id
JOIN dim_date d ON f.date_id = d.date_id
-- Filter for the year 2024 only
WHERE d.year = 2024
-- Use CUBE to generate all combinations and subtotal groupings
GROUP BY CUBE (t.time_of_day, r.road_type)
-- Exclude the grand total row (i.e., All Times    All Road Types)
HAVING GROUPING(t.time_of_day) + GROUPING(r.road_type) < 2
-- Sort the results in descending order of fatalities
ORDER BY total_fatalities DESC;
```

Listing 7: Query 1 - Time of Day vs Road Type with Fatality Totals

A multi-dimensional **cube** was implemented in PostgreSQL using the `GROUP BY CUBE` operator on the dimensions `time_of_day` and `road_type`. This generates all combinations of the two dimensions, along with subtotals for each individual one. The query joins `fact_person_fatality` with `dim_time`, `dim_road`, and `dim_date`, and filters the data to include only records from the year 2024. **As shown in the partial result table below**, the output includes subtotals for each time period or road type, such as total fatalities across "All Road Types" during the day.



	time_of_day character varying	road_type character varying	total_fatalities bigint
1	Day	All Road Types	740
2	Night	All Road Types	458
3	All Times	National or State Highway	286
4	All Times	Arterial Road	216
5	All Times	Undetermined	195
6	All Times	Sub-arterial Road	193
7	All Times	Local Road	188
8	Day	National or State Highway	180
9	Day	Arterial Road	134
10	Day	Sub-arterial Road	127

Figure 9: Result Table for Query 1

Query 2: Which age groups and road user roles are most associated with fatalities?
Analysis of the contribution of age groups and road user roles to fatalities

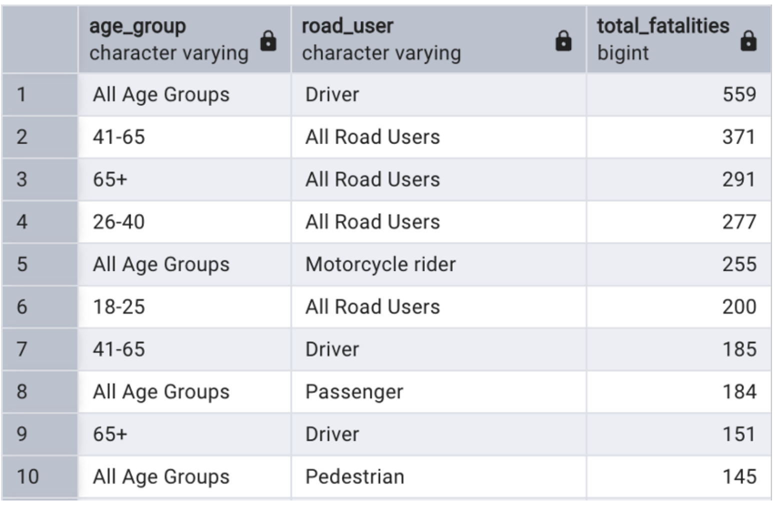
```sql
SELECT
  COALESCE(p.age_group, 'All Age Groups') AS age_group,     -- Replace NULLs for
      readability
  COALESCE(p.road_user, 'All Road Users') AS road_user,
  SUM(f.fatality_count) AS total_fatalities
FROM fact_person_fatality f
JOIN dim_person p ON f.person_id = p.person_id
JOIN dim_date d ON f.date_id = d.date_id
WHERE d.year = 2024
GROUP BY CUBE (p.age_group, p.road_user)
HAVING GROUPING(p.age_group) + GROUPING(p.road_user) < 2  -- Exclude total summary row (
    All    All)
ORDER BY total_fatalities DESC;
```

Listing 8: Query 2 - Age Group vs Road User Role with Fatality Totals

The query joins the fact table `fact_person_fatality` with `dim_person` and filters records for the year 2024. To improve interpretability, the output replaces `NULL` values in the `age_group` and `road_user` fields with descriptive labels such as "All Age Groups" and "All Road Users", ensuring that subtotals are clearly identified.

Additionally, the combination where both dimensions are `NULL` (i.e., the grand total) is excluded using a `HAVING` clause.

This approach allows comparison across both individual categories and their totals, helping identify high-risk demographic segments. **As shown in the top 10 rows of the result table below**, drivers aged 41–65 and 65+ are among the most frequently involved in fatal crashes.



	age_group character varying	road_user character varying	total_fatalities bigint
1	All Age Groups	Driver	559
2	41-65	All Road Users	371
3	65+	All Road Users	291
4	26-40	All Road Users	277
5	All Age Groups	Motorcycle rider	255
6	18-25	All Road Users	200
7	41-65	Driver	185
8	All Age Groups	Passenger	184
9	65+	Driver	151
10	All Age Groups	Pedestrian	145

Figure 10: Result Table for Query 2

Query 3: Which months have the highest number of fatal crashes, and is there a seasonal pattern? Would analyzing time of day enhance this insight?
Analysis of monthly and time-of-day patterns in fatal crash distribution

```sql
SELECT
  d.month,
  COALESCE(t.time_of_day, 'All Times') AS time_of_day,    -- Label nulls in time_of_day
  SUM(f.fatality_count) AS total_fatalities
FROM fact_person_fatality f
JOIN dim_date d ON f.date_id = d.date_id
JOIN dim_time t ON f.time_of_day_id = t.time_of_day_id
WHERE d.year = 2024
GROUP BY ROLLUP (d.month, t.time_of_day)                  -- Month    Time hierarchy
```

```sql
HAVING GROUPING(d.month) + GROUPING(t.time_of_day) < 2   -- Exclude grand total (null-null)
ORDER BY total_fatalities DESC;
```

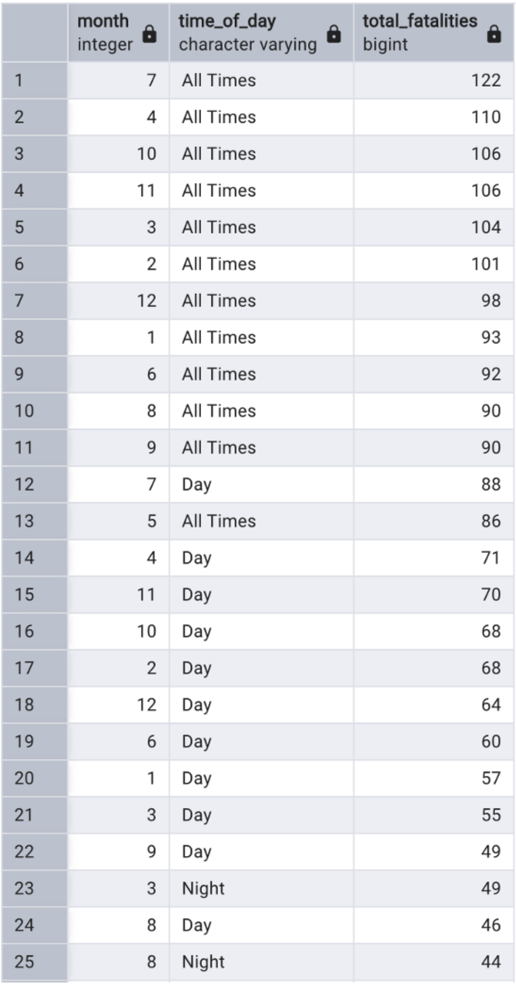
Listing 9: Query 3 - Monthly and Time-of-Day Fatality Analysis

To support Query 3, which explores the monthly pattern of fatal crashes and the potential influence of time of day, a ROLLUP operation was applied across the month and time_of_day dimensions.

The query joins the fact table fact_person_fatality with dim_date and dim_time, and filters records to include only those from the year 2024.

To enhance readability, NULL values in the time_of_day column were replaced with the label "All Times", allowing the results to clearly display monthly subtotals and breakdowns by time of day. The grand total row (where both columns are NULL) was excluded using a HAVING clause.

As shown in the top 25 rows of the result table below, July (i.e., month 7) had the highest number of fatalities overall, with a noticeable peak during the day. The breakdown also reveals a consistent dominance of day-time fatalities across most months.



	month integer	time_of_day character varying	total_fatalities bigint
1	7	All Times	122
2	4	All Times	110
3	10	All Times	106
4	11	All Times	106
5	3	All Times	104
6	2	All Times	101
7	12	All Times	98
8	1	All Times	93
9	6	All Times	92
10	8	All Times	90
11	9	All Times	90
12	7	Day	88
13	5	All Times	86
14	4	Day	71
15	11	Day	70
16	10	Day	68
17	2	Day	68
18	12	Day	64
19	6	Day	60
20	1	Day	57
21	3	Day	55
22	9	Day	49
23	3	Night	49
24	8	Day	46
25	8	Night	44

Figure 11: Result Table for Query 3

Query 4: During holidays such as Christmas and Easter, which road types and time periods see more fatal crashes?

Analysis of fatalities by road type and time of day during holiday periods

```sql
SELECT
```

```
  h.holiday_id,
  COALESCE(r.road_type, 'All Road Types') AS road_type,      -- Label NULLs for clarity
  COALESCE(t.time_of_day, 'All Times') AS time_of_day,       -- Label NULLs for clarity
  SUM(f.fatality_count) AS total_fatalities
FROM fact_person_fatality f
JOIN dim_holiday h ON f.holiday_id = h.holiday_id
JOIN dim_road r ON f.road_id = r.road_id
JOIN dim_time t ON f.time_of_day_id = t.time_of_day_id
JOIN dim_date d ON f.date_id = d.date_id
WHERE d.year = 2024 AND h.holiday_id IN (1, 3)                -- Only include Christmas
    and Easter
GROUP BY CUBE (h.holiday_id, r.road_type, t.time_of_day)
HAVING GROUPING(h.holiday_id) + GROUPING(r.road_type) + GROUPING(t.time_of_day) < 3  --
    Remove full grand total
ORDER BY total_fatalities DESC;
```

Listing 10: Query 4 - Holiday vs Road Type and Time of Day

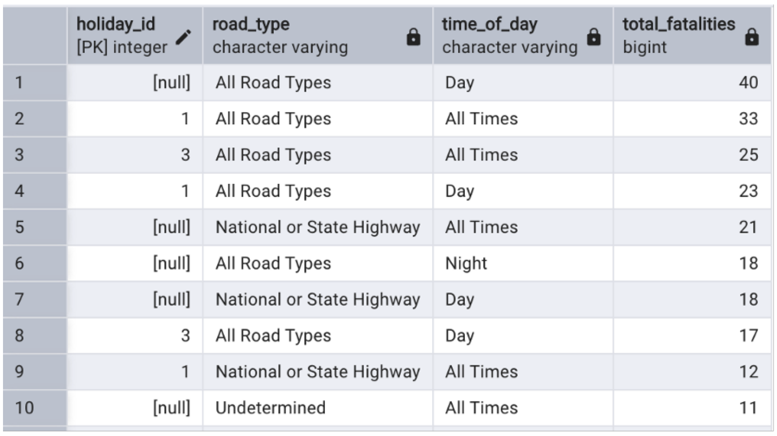
To support Query 4, which explores which road types and time periods experience more fatal crashes during holidays, the fact table `fact_person_fatality` is joined with `dim_holiday`, `dim_road`, `dim_time`, and `dim_date`.

The query filters data to include only records from the year 2024 where `holiday_id` equals 1 (Christmas) or 3 (Easter). Records with `holiday_id = 2` — representing non-holiday periods — are excluded to focus the analysis on holiday-specific crash patterns.

A `GROUP BY CUBE` clause is applied across the three dimensions: holiday, road type, and time of day. This produces all possible combinations and subtotals for each dimension. Many rows in the result will contain `NULL` in the `holiday_id` column — these represent subtotal rows aggregating across all holidays.

As shown in the result table below, the first row with `holiday_id = NULL`, `road_type = All Road Types`, and `time_of_day = Day` indicates there were 40 fatalities during daytime across both Christmas and Easter combined. This type of subtotal enables comparison across holiday periods and time segments.

We observe that Christmas (`holiday_id = 1`) had more fatalities than Easter (`holiday_id = 3`), especially on arterial roads and during daytime. Subtotals with `All Road Types` or `All Times` further highlight which time frames or



	holiday_id [PK] integer	road_type character varying	time_of_day character varying	total_fatalities bigint
1	[null]	All Road Types	Day	40
2	1	All Road Types	All Times	33
3	3	All Road Types	All Times	25
4	1	All Road Types	Day	23
5	[null]	National or State Highway	All Times	21
6	[null]	All Road Types	Night	18
7	[null]	National or State Highway	Day	18
8	3	All Road Types	Day	17
9	1	National or State Highway	All Times	12
10	[null]	Undetermined	All Times	11

Figure 12: Result Table for Query 4

Query 5: Which Australian states had the highest fatality rates in 2023?

To fairly compare road safety across states, we calculated the fatality rate per 100,000 people in each state. Raw fatality counts alone can be misleading due to population differences. A normalized metric like fatality rate allows for more accurate interpretation.

The query aggregates total deaths per state from the `fact_fatal_crash` table and joins it with a population summary derived from the `dim_location` dimension. Duplicated Local Government Area (LGA) records were removed using `DISTINCT`, ensuring each population value is counted once before being summed at the state level.

Importantly, this is the only query where we filter by the year 2023—unlike the others which use 2024—because the population data available in our resources only covers up to 2023. Therefore, fatality rate is calculated accordingly for that year.

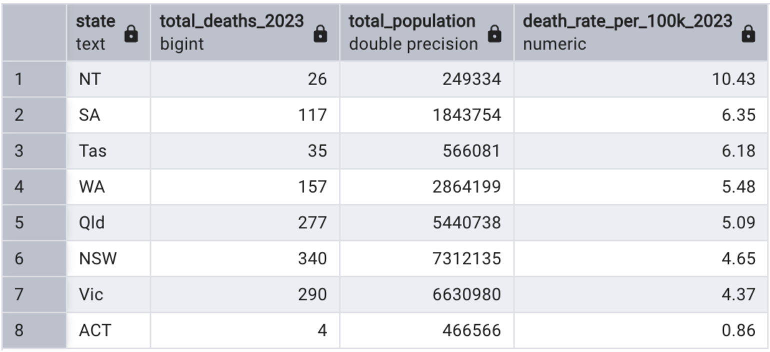
This query does not apply `CUBE` or `ROLLUP`, as population is a non-additive metric that cannot be meaningfully aggregated at higher levels. A simple `GROUP BY state` ensures correct and interpretable results.

```sql
-- Step 1: Get unique LGA-level population per state
WITH unique_lga_population AS (
    SELECT DISTINCT state, lga_name, population_2023_lga
    FROM dim_location
),

-- Step 2: Aggregate to state-level population
state_population AS (
    SELECT
        state,
        SUM(population_2023_lga) AS total_population
    FROM unique_lga_population
    GROUP BY state
)

-- Step 3: Join with fatal crash data and calculate death rate per 100k
SELECT
    l.state,
    SUM(f.number_fatalities) AS total_deaths_2023,
    p.total_population,
    ROUND(
        (SUM(f.number_fatalities) * 100000.0 / NULLIF(p.total_population, 0))::numeric, 2
    ) AS death_rate_per_100k_2023
FROM fact_fatal_crash f
JOIN dim_location l ON f.location_id = l.location_id
JOIN dim_date d ON f.date_id = d.date_id
JOIN state_population p ON l.state = p.state
WHERE d.year = 2023
GROUP BY l.state, p.total_population
ORDER BY death_rate_per_100k_2023 DESC;
```

Listing 11: Calculate fatality rate per 100k by state (2023)



	state text	total_deaths_2023 bigint	total_population double precision	death_rate_per_100k_2023 numeric
1	NT	26	249334	10.43
2	SA	117	1843754	6.35
3	Tas	35	566081	6.18
4	WA	157	2864199	5.48
5	Qld	277	5440738	5.09
6	NSW	340	7312135	4.65
7	Vic	290	6630980	4.37
8	ACT	4	466566	0.86

Figure 13: Result Table for Query 5

Query 6: How have fatalities from different types of crashes changed over time (by year) across different speed categories?

To address this query, which investigates how different crash types contribute to fatalities **over time across different speed zones**, a `GROUP BY CUBE` query was implemented in PostgreSQL. This approach generates all combinations between `year`, `crash_type`, and `speed_category`, including subtotals for each dimension (e.g., total fatalities per crash type regardless of speed zone, and vice versa).

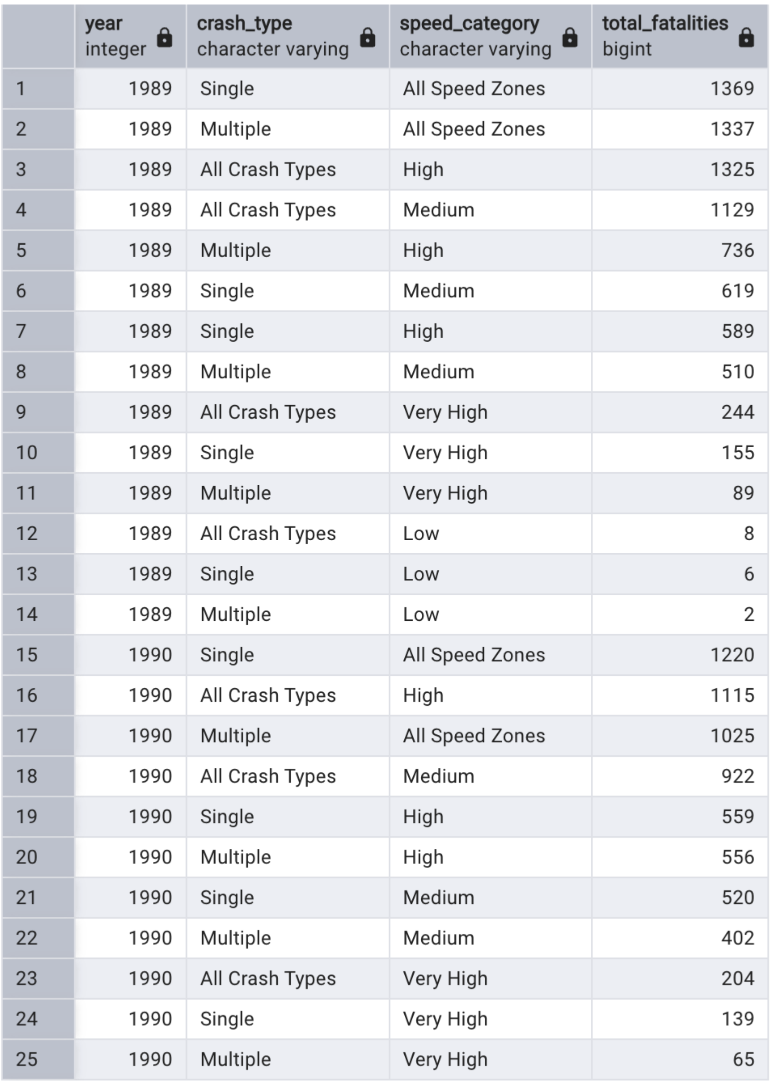
The query joins the fact table `fact_person_fatality` with `dim_crash_type`, `dim_road`, and `dim_date`. To improve readability, `NULL` values produced by the `CUBE` operator are replaced using `COALESCE` with meaningful labels such as "All Crash Types" and "All Speed Zones." The `HAVING` clause is used to exclude grand total rows where both fields are aggregated.

The result table includes year-by-year fatality counts broken down by crash type and speed category. For example, in 1989, single-vehicle crashes in high-speed zones led to 589 fatalities, while multiple-vehicle crashes in the same zones resulted in 736 deaths. These figures suggest that crash severity tends to be higher in high-speed environments. Overall, the data supports targeting speed management and crash-type-specific interventions in road safety policy.

```
SELECT
  d.year AS year,  -- Extract year from the date dimension
  COALESCE(c.crash_type, ’All Crash Types’) AS crash_type,  -- Replace NULLs from CUBE
      with label for subtotal
  COALESCE(r.speed_category, ’All Speed Zones’) AS speed_category,  -- Replace NULLs from
       CUBE with label for subtotal
  SUM(f.fatality_count) AS total_fatalities  -- Compute total number of fatalities for
      each group
FROM fact_person_fatality f
JOIN dim_crash_type c ON f.crash_type_id = c.crash_type_id  -- Join with crash type
    dimension
JOIN dim_road r ON f.road_id = r.road_id  -- Join with road dimension to get speed
    category
JOIN dim_date d ON f.date_id = d.date_id  -- Join with date dimension to access year
GROUP BY CUBE (d.year, c.crash_type, r.speed_category)  -- Group using CUBE to get all
    combinations and subtotals
HAVING GROUPING(c.crash_type) + GROUPING(r.speed_category) < 2
-- This filters out the total-total rows (e.g., All Crash Types + All Speed Zones),
-- keeping only full details and one-level subtotals
ORDER BY d.year ASC, total_fatalities DESC;  -- Sort by year, then by fatalities in
```

```
    descending order
```

Listing 12: Fatalities by crash type and speed zone



	year integer	crash_type character varying	speed_category character varying	total_fatalities bigint
1	1989	Single	All Speed Zones	1369
2	1989	Multiple	All Speed Zones	1337
3	1989	All Crash Types	High	1325
4	1989	All Crash Types	Medium	1129
5	1989	Multiple	High	736
6	1989	Single	Medium	619
7	1989	Single	High	589
8	1989	Multiple	Medium	510
9	1989	All Crash Types	Very High	244
10	1989	Single	Very High	155
11	1989	Multiple	Very High	89
12	1989	All Crash Types	Low	8
13	1989	Single	Low	6
14	1989	Multiple	Low	2
15	1990	Single	All Speed Zones	1220
16	1990	All Crash Types	High	1115
17	1990	Multiple	All Speed Zones	1025
18	1990	All Crash Types	Medium	922
19	1990	Single	High	559
20	1990	Multiple	High	556
21	1990	Single	Medium	520
22	1990	Multiple	Medium	402
23	1990	All Crash Types	Very High	204
24	1990	Single	Very High	139
25	1990	Multiple	Very High	65

Figure 14: Result Table for Query 6

7 Visualization of Business Queries

Query 1: What is the number of fatalities for different combinations of time of day and road type?

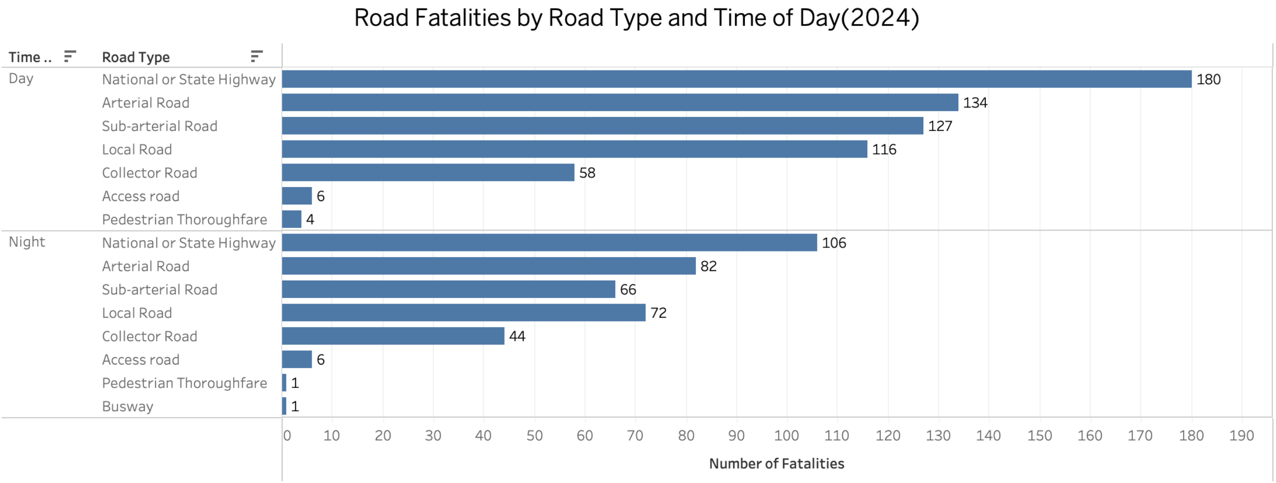


Figure 15: Visualization for Query 1

The bar chart illustrates the distribution of road fatalities across different road types and time periods in 2024. Fatal crashes occurred more frequently during the **daytime** across almost all road types. **National or State Highways** accounted for the highest number of fatalities both during the day (180) and at night (106), followed by **Arterial Roads** and **Sub-arterial Roads**. Pedestrian Thoroughfares and Access Roads saw relatively few fatalities, especially at night. This suggests that **high-speed and high-traffic roads pose greater risks**, particularly during daylight hours when road usage is heavier.

Query 2: Which age groups and road user roles (e.g., pedestrian, driver) are most associated with fatalities?

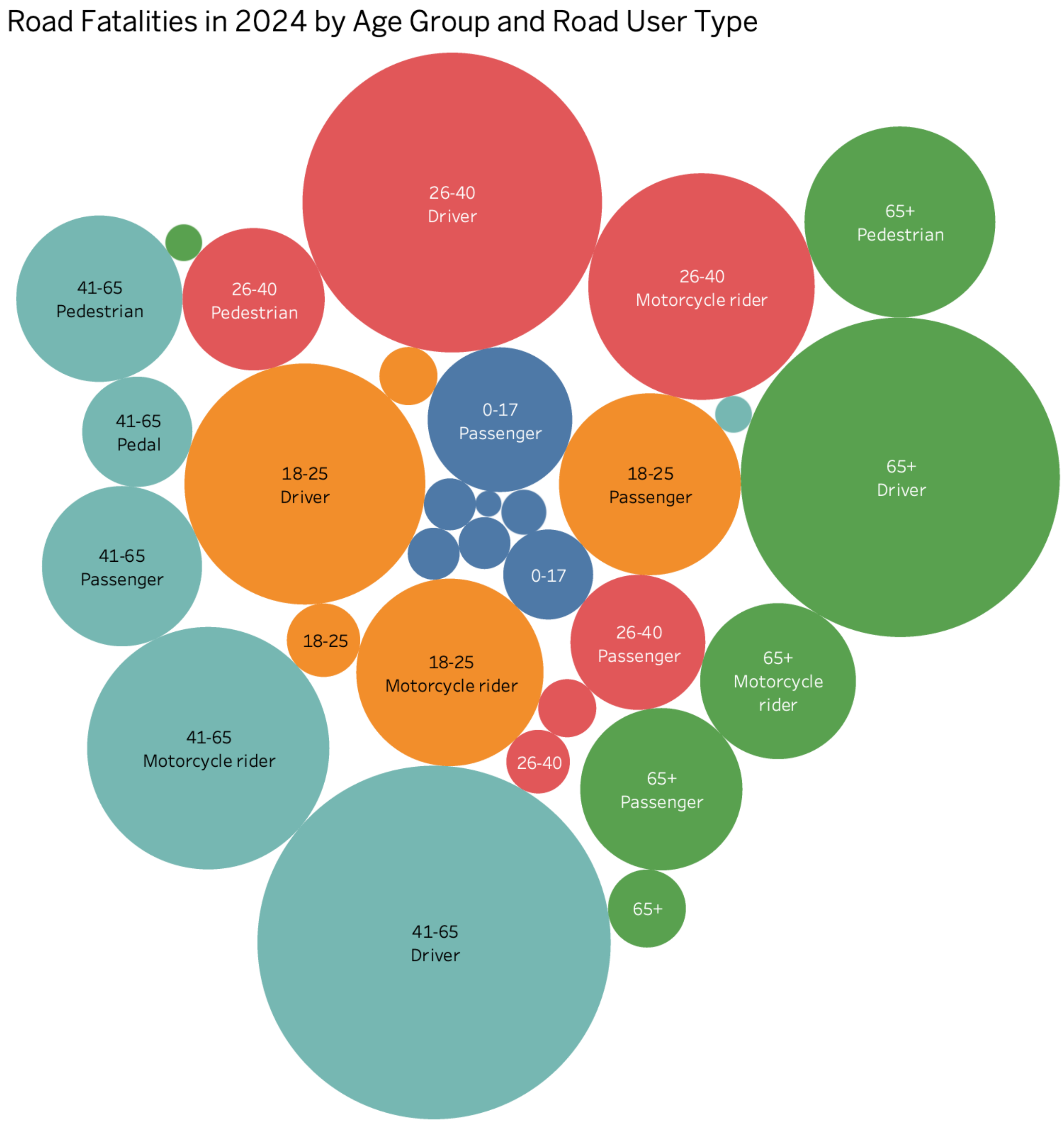


Figure 16: Visualization for Query 2

The bubble chart presents the number of road fatalities in 2024 by age group and road user type. The largest fatality counts are observed among **drivers aged 26–40 and 41–65**, as well as **elderly drivers aged 65+**. Significant fatalities also occurred among **motorcycle riders** in the 18–25 and 41–65 age brackets. Vulnerable road users such as **pedestrians aged 65+** and **passengers aged 0–17** also show noticeable fatality numbers. These findings suggest that **middle-aged drivers and older road users are among the most at-risk groups**.

Query 3: Which months have the highest number of fatal crashes, and is there a seasonal pattern? Would analyzing time of day enhance this insight?

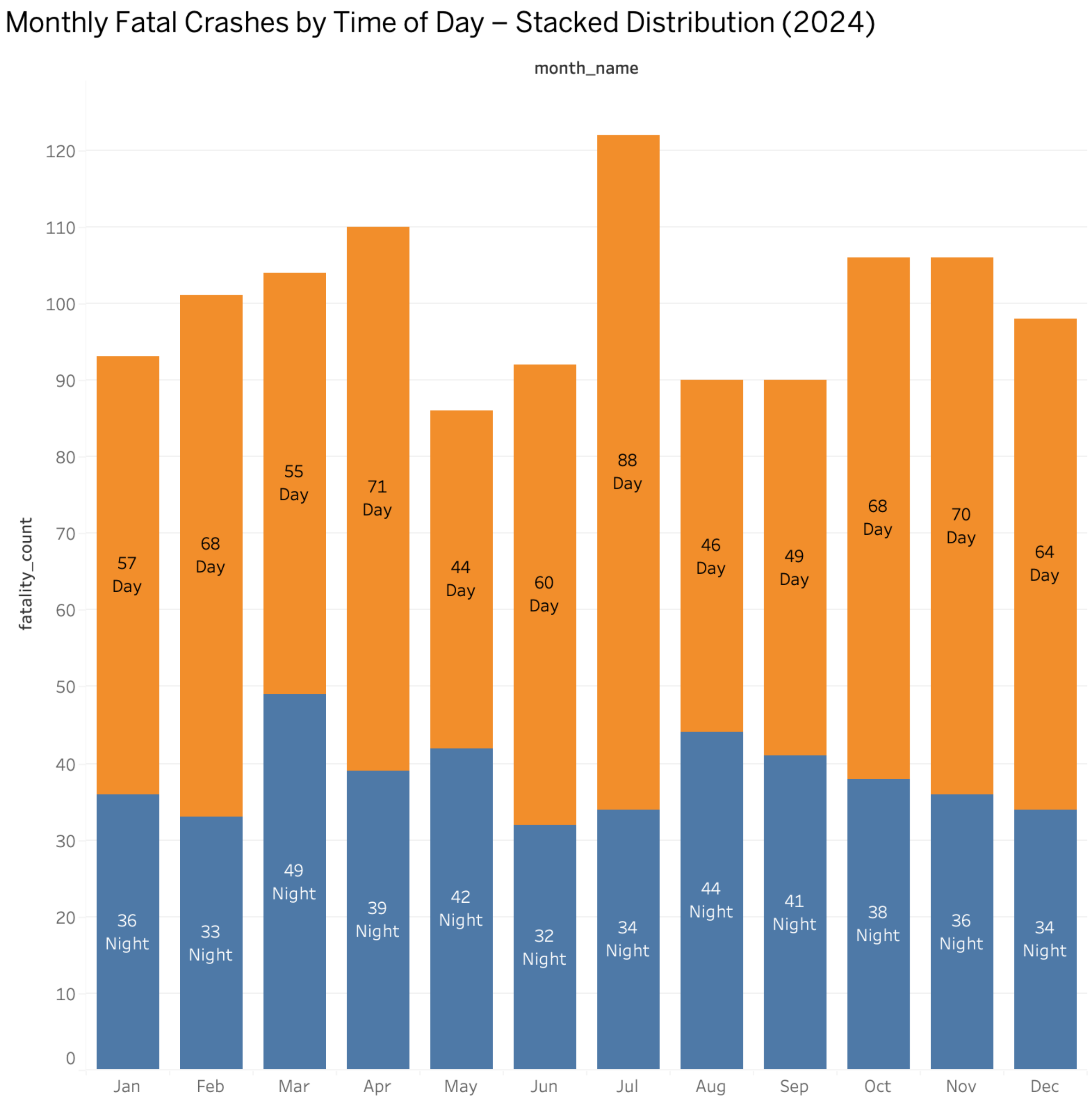


Figure 17: Visualization for Query 3

A stacked bar chart was used to visualize monthly fatal crash counts by time of day, allowing comparison of both **seasonal patterns** and **day/night distributions** in a single view. The chart reveals that **July recorded the highest number of fatalities**, followed by April and November. Fatal crashes are consistently higher during the **daytime** across all months. These findings suggest that certain periods of the year—such as **mid-winter and early spring**—may warrant additional safety measures or public awareness efforts.

Query 4: During holidays such as Christmas and Easter, which road types and time periods see more fatal crashes?

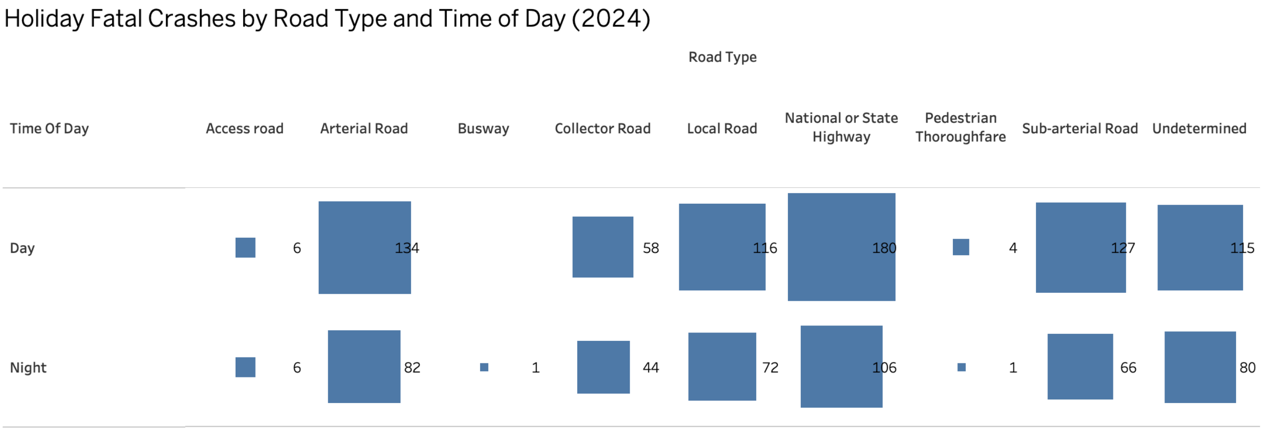


Figure 18: Visualization for Query 4

A square area chart was selected to represent fatal crashes during holiday periods by **road type** and **time of day**. This format allows for immediate visual comparison of categorical intersections. The data reveals that **National or State Highways** and **Arterial Roads** experienced the highest number of fatalities, especially during the **daytime**. The concentration of crashes on major roads suggests increased long-distance travel and higher exposure during holidays, underscoring the need for enhanced enforcement or awareness campaigns around these high-risk areas.

Query 5: Are there significant differences in road safety levels across Australian states?

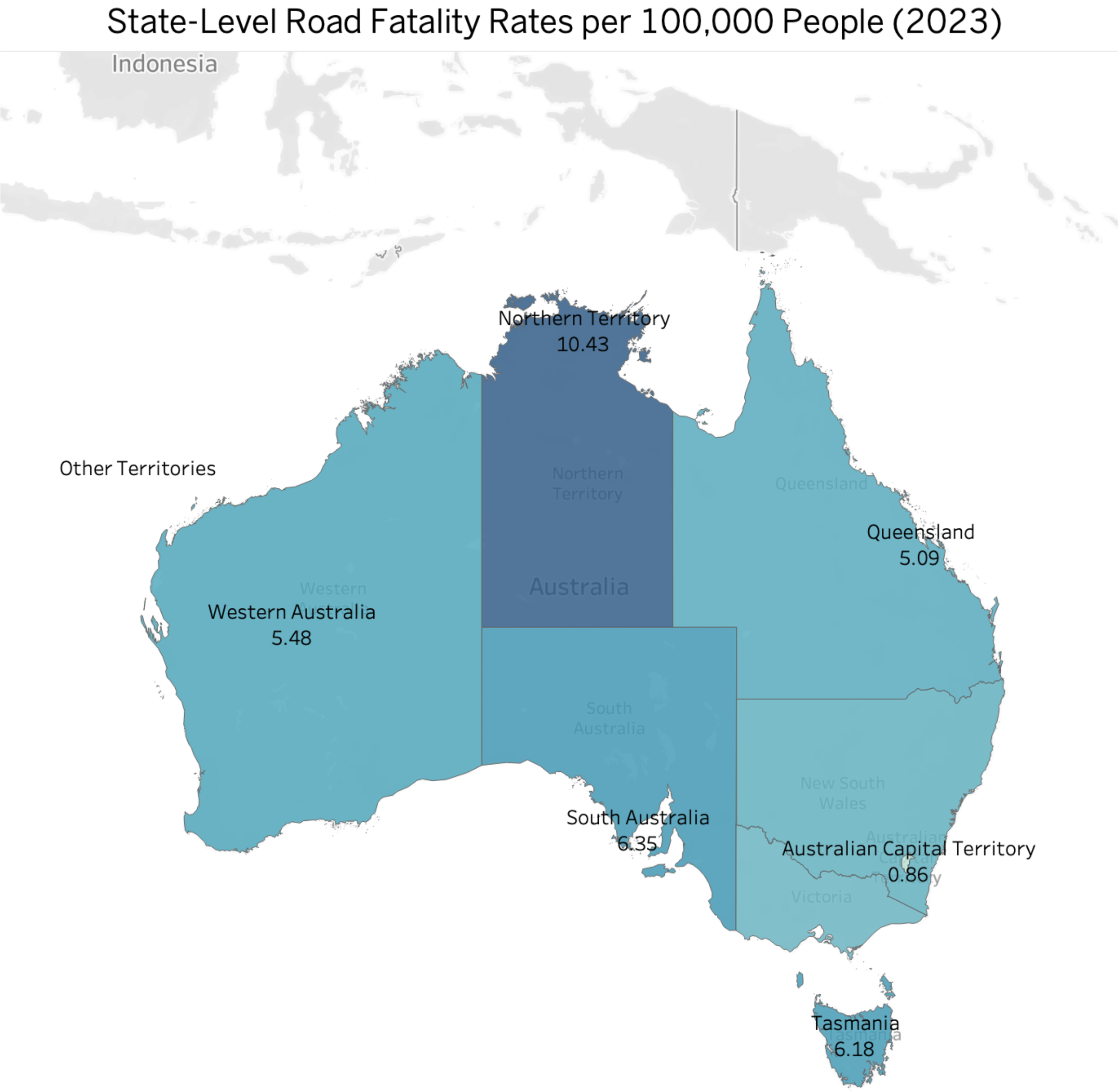


Figure 19: Visualization for Query 5

To visualize road fatality rates across Australian states, we used the `STE_2021_AUST_GDA2020` GeoJSON file from the project's introduction page, which contains the spatial boundaries of Australian states. When imported into Tableau, the GeoJSON file appears as a spatial table with geometry fields and full state names.

To support this spatial visualization, we first created a summary table using the SQL query described in Section 6.2. This query calculates the fatality rate per 100,000 people for each state in the year 2023, based on total fatalities and state-level population aggregated from unique Local Government Area (LGA) records.

The resulting summary table includes *state abbreviation*, *total fatalities*, *total population*, and the calculated *fatality rate per 100,000 people*. We named this table `death_rate_per_100k_2023`, and exported it from pgAdmin 4 as a CSV file.

This CSV file was then joined in Tableau with the GeoJSON spatial data using the full state name as the linking key. This join enables a geographic visualization of road fatality rates across states. Since the most recent population statistics available are for 2023, this map reflects fatality rates only for that year and should not be compared with 2024 data used in previous queries. The choropleth map illustrates the fatality rate per 100,000 people across Australian states for the year 2023. The Northern Territory exhibits the highest fatality rate of **10.43**, while the Australian Capital Territory has the lowest at **0.86**. Other states like South Australia (6.35), Tasmania (6.18), and Western Australia (5.48) also show elevated rates.

Interestingly, these figures appear higher than the national average of **4.8 deaths per 100,000 people**, as reported on the project's introduction page. One plausible reason is that the population data used in this analysis, extracted from the `dim_location` table, aggregates to only about **25 million** people. In contrast, the official 2023 population of Australia was approximately **26.6 million** [14]. This discrepancy in population denominators results in overestimated fatality rates, especially in areas with incomplete or duplicated LGA-level records.

Query 6: How do different types of crashes contribute to fatalities across various speed limit categories?

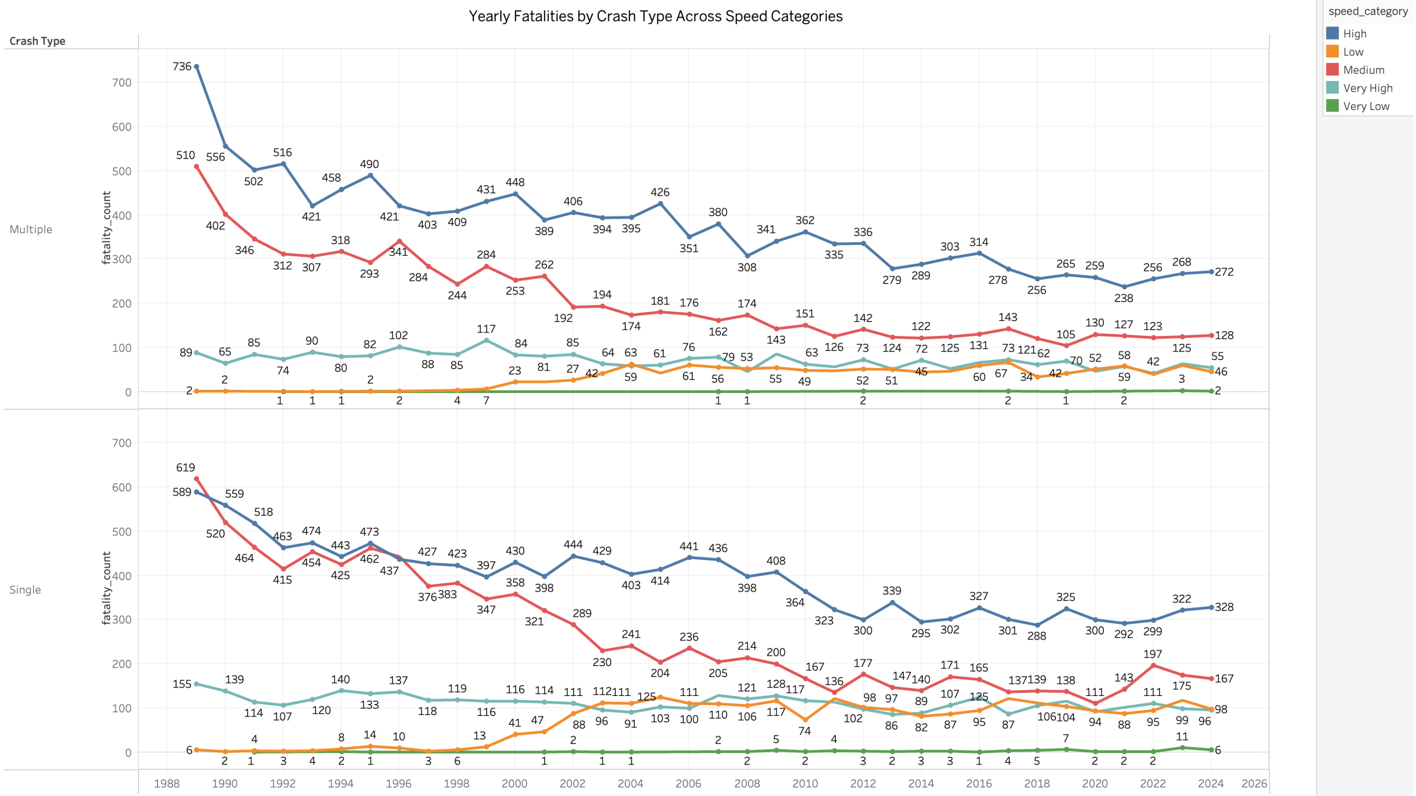


Figure 20: Visualization for Query 6

This line chart visualizes the relationship between crash types and speed zones over time, addressing how different types of crashes contribute to fatalities across various speed limit categories. The `speed_category` field, derived from posted speed limits, provides a meaningful ordinal dimension to evaluate severity trends, while the `crash_type` dimension separates fatalities into single-vehicle and multiple-vehicle crash types.

This visualization was chosen to highlight both absolute and relative trends across different speed environments. A clear pattern emerges: for both crash types, fatality counts are consistently highest in "High" and "Very High" speed zones, confirming the strong correlation between speed and crash severity.

Conversely, “Low” and “Very Low” speed zones contribute only marginally to overall fatalities.

In addition to the speed-related differences, a general downward trend in total fatalities is observed across nearly all categories from 1989 to 2024. This suggests that broader road safety measures—such as improvements in vehicle design, road infrastructure, and enforcement—may have contributed to long-term reductions in crash-related deaths.

However, while the overall trend is declining, the persistent dominance of high-speed zones in fatal crash counts indicates that speed management remains a critical priority for policymakers. The chart supports targeting high-risk speed environments for enhanced enforcement, traffic calming, and public awareness campaigns.

8 Association Rule Mining

Association Rule Mining (ARM) is a rule-based machine learning technique commonly used to uncover interesting patterns, correlations, or causal structures among items in large transactional datasets. It has been widely applied in market basket analysis, healthcare, and transportation safety research [15]. In this project, ARM is used to discover interpretable patterns associated with fatal crashes in Australia, particularly focusing on rules where the consequent involves the `road_user` type.

This section documents the methodology, thresholds selection, top-k rules, and policy implications derived from the mining results.

8.1 Apriori Algorithm

The Apriori algorithm is employed to generate frequent itemsets and derive association rules. It is based on the anti-monotonic property: if an itemset is not frequent, all its supersets are also not frequent [15]. We use the implementation provided in the `mlxtend` Python library, which supports boolean matrix encoding, minimum support pruning, and lift-based rule filtering.

The mining process includes the following steps:

- Merge the `fact_person_fatality` table with all relevant dimension tables via foreign keys.
- Select columns of interest based on configurable parameters: `speed_limit` vs. `speed_category`, `easter_period` or `christmas_period`, and types of vehicle involvement (bus, heavy truck, articulated).
- Convert each row to a transaction with `attribute=value` tokens.
- Apply `TransactionEncoder` to produce a binary dataframe suitable for Apriori mining.

A key code snippet that prepares transaction-format data is shown in Listing 13:

```
df = df[selected_cols]
df = df.apply(lambda col: col.map(lambda x: f"{col.name}={x}" if pd.notnull(x) else pd.NA
    ))
transactions = df.apply(lambda row: row.dropna().tolist(), axis=1)

te = TransactionEncoder()
df_encoded = te.fit(transactions).transform(transactions)
```

Listing 13: Converting Dimension-Joined Rows to Transactions

8.2 Selection of Appropriate Thresholds

To ensure statistically meaningful and interpretable results, the following parameters were set based on experimentation and literature guidance [16]:

- **Minimum Support:** 0.02 — to retain only patterns that appear in at least 2% of the records.
- **Minimum Confidence:** 0.60 — to guarantee reasonably reliable conditional probability.
- **Minimum Lift:** 1.0 — to remove rules that are no better than random co-occurrence.
- **Top-K Rules:** For each parameter setting, only the top 50 rules were retained based on lift and confidence.

The selection of threshold parameters for support, confidence, and lift is critical to ensure that the discovered association rules are both statistically significant and interpretable. The minimum support was set to **0.02** to filter out rare itemsets that may not generalize well across the dataset. This value aligns with common practices in association rule mining where a 1–5% support is used as a lower bound to maintain statistical robustness while retaining meaningful patterns [16].

The minimum confidence was configured as **0.60** to ensure that the rules have a reasonable level of conditional reliability—that is, given the antecedent, the consequent is likely to occur in at least 60% of cases. According to Tan et al. [16], this helps eliminate rules that may be frequent but weakly associated.

A lift threshold of **1.0** was used to discard rules that are no better than random co-occurrence. Lift greater than 1 implies a positive correlation between antecedent and consequent, which is essential for highlighting interesting or surprising associations.

Finally, to improve the interpretability and policy relevance of the output, we retained only the **top-50 rules** sorted by lift and confidence, and focused our analysis on rules whose consequents match the prefix `road user=`. This constraint aligns with our goal of identifying high-risk road user groups for targeted traffic safety interventions.

Filtering to rules with a single consequent matching the prefix `road_user=` was done to focus analysis on user-specific risks.

8.3 Insights From Mining Results

The top-ranking association rules—those with the highest lift values—consistently identify `road_user=Pedestrian` as the consequent. As shown in Table 5, these rules highlight a recurring pattern involving a highly vulnerable subgroup: elderly individuals aged 65 and over, involved in weekday, single-vehicle crashes on undetermined road types with speed limits around 60 km/h, and with no involvement of heavy vehicles such as buses or trucks.

> *This pattern suggests that pedestrian fatalities are disproportionately associated with low-speed, low-traffic environments that may appear safe but pose hidden risks, especially for older road users.*

Table 5: Top 9 High-Lift Association Rules for `road_user=Pedestrian`

Antecedents	**Consequent**	**Support**	**Confidence**	**Lift**
age_group=65+, christmas_period=False, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.022	0.860	5.592
age_group=65+, bus_involvement=False, christmas_period=False, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.021	0.860	5.591
age_group=65+, articulated_truck_involvement=False, christmas_period=False, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.021	0.860	5.588
age_group=65+, crash_type=Single, day_type=Weekday, easter_period=False, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.022	0.860	5.586
age_group=65+, bus_involvement=False, crash_type=Single, day_type=Weekday, easter_period=False, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.021	0.859	5.584
age_group=65+, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.022	0.859	5.583
age_group=65+, articulated_truck_involvement=False, crash_type=Single, day_type=Weekday, easter_period=False, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.022	0.859	5.582
age_group=65+, bus_involvement=False, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.021	0.859	5.581
age_group=65+, articulated_truck_involvement=False, crash_type=Single, day_type=Weekday, road_type=Undetermined, speed_limit=60	road_user=Pedestrian	0.022	0.859	5.579

The rules presented in Table 5 exhibit high support (approximately 2.1–2.2%), strong confidence (around 85.9–86.0%), and exceptionally high lift values (above 5.57), indicating that the co-occurrence of these features with pedestrian fatalities is far from random.

This finding suggests strong latent interactions between demographic, temporal, and environmental factors in fatal pedestrian crashes. Such patterns would be difficult to uncover through traditional descriptive statistics or SQL-based analysis alone, highlighting the value of association rule mining in uncovering meaningful and interpretable relationships in traffic safety data.

These insights form the analytical basis for the policy recommendations proposed in the following subsection.

8.4 Suggestions

Based on the extracted rules, we provide the following data-driven recommendations for public authorities and policy makers:

1. **Enhance pedestrian safety infrastructure** in areas with undetermined or poorly labeled roads, especially in regions with speed limits around 60 km/h.
2. **Implement targeted education or visibility campaigns** for senior pedestrians, focusing on weekday behaviors and low-traffic zones.
3. **Prioritize road audit programs** in locations where pedestrian crashes occur without the presence of heavy vehicles — suggesting potential design or signage issues rather than vehicular size.

The results demonstrate the value of association rule mining in uncovering interpretable, non-trivial relationships that can inform proactive safety measures.

9 Limitations

While this project demonstrates the effectiveness of data warehousing and association rule mining in uncovering insights from road crash data, several limitations should be acknowledged.

- **Data Quality and Completeness:** The source datasets, though authoritative, contain missing, ambiguous, or categorical placeholders (e.g., "Unknown" or "-9"). Even after transformation, residual inconsistencies may impact the fidelity of analysis [17].
- **Temporal and Spatial Granularity:** The available data is aggregated monthly and regionally (e.g., at the LGA level), limiting finer-grained temporal (hour-level) and geospatial (GPS or road segment level) analyses. This reduces the ability to capture localized risk factors such as specific intersections or weather events [18].
- **Association Rule Mining Limitations:** While Apriori-based rule mining provides interpretable patterns, it is inherently limited to pairwise correlations and does not capture causal relationships or temporal sequences. Additionally, mining rules with many antecedents may lead to overfitting or spurious associations [16].
- **Bias Toward Frequent Events:** The ARM algorithm favors rules that meet minimum support and confidence thresholds. Consequently, rare but important events (e.g., fatalities involving children or during extreme weather) may be excluded from the rule set.
- **Tooling Constraints:** The project prohibits the use of tools like R or Weka and mandates the use of Python. While this constraint ensures reproducibility and transparency, it limits the ability to explore alternative models or more scalable mining algorithms (e.g., FP-Growth, ECLAT) supported in other platforms [19].

- **Static Visualization:** Although Tableau provides an interactive layer on top of the data warehouse, the dashboard cannot reflect real-time updates or user-defined query creation, which limits its utility for dynamic policy simulation or ongoing monitoring [20].
- **Runtime and Resource Constraints:** Due to the need to explore multiple parameter combinations and perform high-dimensional transaction encoding, the mining process may take several minutes to complete. This reflects a deliberate trade-off between computational cost and the breadth of pattern discovery.

Recognizing these limitations provides guidance for future work, such as integrating real-time data streams, adopting causal inference models, and enhancing user-driven visualization interfaces.

Bibliography

[1] European Commission, "2023 figures show stalling progress in reducing road fatalities in too many countries," 2024. Accessed: Feb. 24, 2025.

[2] S. Guthrie, "The country with the safest roads in the world," 2024. Accessed: Feb. 24, 2025.

[3] Australian Bureau of Statistics, "Remoteness Structure - ASGS Edition 3, 2021," 2021. Accessed: 2025-04-05.

[4] Australian Bureau of Statistics, "Statistical Area Level 4 (SA4) - ASGS Edition 3," 2021. Accessed: 2025-04-05.

[5] Australian Bureau of Statistics, "Local Government Areas (LGAs) - ASGS Edition 3, 2021," 2021. Accessed: 2025-04-05.

[6] Australian Bureau of Statistics, "Census of Population and Housing: Household and Dwelling Characteristics, 2021," 2021. Accessed: 2025-04-05.

[7] Australian Bureau of Statistics, "Regional Population by Local Government Area and Remoteness Area, 2023," 2023. Accessed: 2025-04-05.

[8] "Speed limits in australia." `https://en.wikipedia.org/wiki/Speed_limits_in_Australia`. Accessed: 2025-04-10.

[9] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. John Wiley & Sons, 2013.

[10] R. Kimball and M. Ross, *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Indianapolis, IN: Wiley, 3rd ed., 2013.

[11] W. Inmon, *Building the Data Warehouse*. Wiley, 2005.

[12] J. Gray, S. Chaudhuri, A. Bosworth, A. Layman, D. Reichart, M. Venkatrao, F. Pellow, and H. Pirahesh, "Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals," *Data Mining and Knowledge Discovery*, vol. 1, no. 1, pp. 29–53, 1997.

[13] S. Chaudhuri and U. Dayal, "An overview of data warehousing and olap technology," *ACM SIGMOD Record*, vol. 26, no. 1, pp. 65–74, 1997.

[14] Australian Institute of Health and Welfare, "Profile of australia's population." `https://www.aihw.gov.au/reports/australias-health/profile-of-australias-population`, 2023. Accessed April 10, 2025.

[15] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases (VLDB)*, pp. 487–499, 1994.

[16] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining.* Boston, MA, USA: Pearson/Addison Wesley, 2005.

[17] T. C. Redman, "The impact of poor data quality on the typical enterprise," *Communications of the ACM*, vol. 41, no. 2, pp. 79–82, 1998.

[18] M. Batty, K. Axhausen, F. Giannotti, A. Pozdnoukhov, A. Bazzani, M. Wachowicz, G. Ouzounis, and Y. Portugali, "Smart cities of the future," *The European Physical Journal Special Topics*, vol. 214, no. 1, pp. 481–518, 2012.

[19] C. Borgelt, "Efficient implementations of apriori and eclat," in *Workshop of Frequent Item Set Mining Implementations (FIMI'03)*, 2005.

[20] B. Shneiderman, "The eyes have it: A task by data type taxonomy for information visualizations," *IEEE Symposium on Visual Languages*, pp. 336–343, 1996.