AUSTRALIAN ROAD CRASH ANALYSIS: DATAWAREHOUSING AND MINING TECHNIQUES

DHARUN SOMALINGAM

Abstract

In Australia, road accidents are a major concern, causing significant harm to people, infrastructure, and society. Every year, around 1,200 people lose their lives in road incidents, and approximately 40,000 are seriously injured (Australian Government, n.d.). Considering this, it is important to adopt effective measures to identify potential contributing factors and reduce the impact of such incidents.

This study aims to identify key factors that can help mitigate road accidents in Australia (Road Sense Australia, n.d.). To achieve this, we utilise **data warehousing and mining techniques**, following **Kimball's four-step approach** for factor identification and further analysis. In this analysis, we implement **association rule mining** to uncover trends and patterns, predict future combinations, and identify potential threats (Bigham, 2014).

Introduction

Fatal crash analysis involves investigating the key factors contributing to crashes and injuries. In recent times, Global countries are working to mitigate the road fatal crashes, but Australians were still trails behind. In 2023, it recorded 4.8 deaths per 100000 people. more than double Norway 2.2 and Iceland 2.1 the world's lowest rates [BITRE], 2023; International Transport Forum (European Commission, 2024). As of July 2024, 761 lives have been lost on Australian roads, with a 12-month total of 1,327 deaths—up 10% from the previous year. To investigate the essential factors for Accidents, we utilise a data warehouse for systematic analysis. Data warehousing has been phenomenal in modern business for decision-making and analysis, providing a collectively structured working environment (framework). It is subject-oriented, time-variant, integrated, and involves a continuous flow of data that supports the decision-making process **Kimberly Merritt. (2008).**

Data and Processing

Source of Dataset

ARDD: Fatal crashes—December 2024—XLSX and ARDD: Fatalities—December 2024—XLSX data files were extracted from **the Australian Road Deaths Database (ARDD)**. This Database, provided by the **Bureau Of Infrastructure And Transport Research Economics (BITRE)**, contains essential information on road accidents across Australian states. It is managed by the Bureau of Infrastructure and Transport Research Economics (BITRE), which operates under the Analytics, Data and Policy Division of the **Department of Infrastructure, Transport, Regional Development, Communications and the Arts**. Many of BITRE's publications are publicly accessible through their official website. (Bureau of Infrastructure and Transport Research Economics, n.d.).

About Data

The **Australian Road Deaths Database** is divided into two primary datasets: the **Fatalities** dataset and the **Fatal Crashes** dataset. The **Fatalities** dataset contains individual records for each person killed in a crash comprising **51,284 records**, while the **Fatal Crashes** dataset provides information at the crash level, with **56,874 records** one record per fatal incident. Both datasets share several common variables but represent data at different levels of detail. Shared information includes **Crash ID**, **Crash Type**, **Speed Limit**, and **date/time attributes** such as **month**, **year**, **day of the week**, and indicators of **vehicle involvement** (e.g., buses, articulated trucks, and heavy trucks). They also include markers for whether the crash occurred during a **holiday period**, such as **Christmas** or **Easter**. The **Fatal Crashes** dataset additionally includes the **number of fatalities per crash**, whereas the **Fatalities** dataset provides more detailed personal information, including the **age**, **gender**, and **road user type** (e.g., driver, passenger, pedestrian) of each victim.

It's important to note that both datasets contain **missing** typically represented by the value **-9** and the datasets also contain **Undetermined data**. Moreover, the majority of the data has been **pre-processed and cleaned** by **BITRE** prior to publication.

Extract- Transform-Load Process (ETL)

In this Computational World, The **Extract, Transform, and Load (ETL) framework** is a critical step in building a **data warehouse**, facilitating the mapping of data from source systems to the target warehouse. ETL integrates data from various applications or sources, including those from different domains. In the context of data warehousing, ETL selects and adapts the data to meet operational needs through three distinct steps. (Bansal, S. K., & Kagemann, S. (2015))

Extract

The extraction step involves gathering or pulling data from various source systems. This data is typically stored in flat file formats such as CSV (comma-separated values), Excel (XLS) OR txt files. Additionally, the data can be retrieved by the Application Process Interfaces (API's) or RESTful client (this method used to interact with services). The main goal of this step grouping the data together from various sources, regardless of their structure, Format or location.

The **Australian Road Deaths Database (ARDD**) provides fatal crash and fatality data in XLS format. For the extraction process, I use Python with the Pandas library to read these files.

The code above reads the **ARDD: Fatal Crashes—December 2024** Excel file and formats it into a Data Frame named **data_fatal**, using the appropriate column headers. Following a similar process, the **ARDD: Fatalities—December 2024** Excel file is read and formatted into a second Data Frame named **data_fatalities**.

Transform

The Transform phase involves in Data quality checks, Cleansing Data, Identifying the essential Factors and Selecting the target schema. These includes in Normalisation of Data, Interpretation of Outliers, removing the duplicates and Handling missing and null data values and verification of constraint integrity. After Cleaning, the Data is filtered based on

Analytical needs and relevant built in function are applied to prepare the processed Data for loading

The **data cleaning workflow** involves three key stages: **data profiling, error detection, and error correction**. In real-world scenarios, datasets are often messy or "**dirty,**" requiring the modeling of various data aspects—such as **patterns, schema structures, probability distributions, and other metadata**—to ensure accuracy and consistency.
Ilyas, I. F., & Chu, X. (2019). *Data Cleaning* (First edition.)

Once the data is extracted and loaded into the **data_fatal** and **data_fatalities** Data frames, it should undergo preliminary **quality checks and basic customisation**.
This includes verifying column names, checking data types, and counting the number of rows gives a brief idea about the data.

There are some minor issues with column names in the **data_fatal** Data Frame, such as **"Bus \nInvolvement",** which may contain hidden characters or formatting inconsistencies. In addition to renaming such columns for consistency, it's important to identify and address other data quality issues—such as **duplicate records** and **missing or null values**—in both the **data_fatal** and **data_fatalities** datasets. **Data duplication** can occur for various reasons, so it's essential to perform **de-duplication** to ensure data accuracy. Running these checks provides a clearer understanding of the dataset's structure and quality, forming a **strong foundation for reliable analysis**.

The next step involves **merging** the two DataFrames using a **foreign key**—the Crash ID. This creates a comprehensive dataset that combines crash-level details with individual-level fatality information, such as road user roles. This merged dataset is essential for the **transformation process**, enabling a unified view in a single file. However, the presence of **similar or overlapping columns** can aid in **error detection**, ensuring the datasets are matched correctly. This step is also crucial for **defining dimensions** for further analysis in the data warehouse.

During the merging process,**Python pandas** automatically differentiates columns with the same name by appending suffixes such as **_x** and **_y**—where _x refers to columns from **data_fatal** and _y refers to those from **data_fatalities**. To maintain a clean and accurate dataset, it is essential to **review these duplicated columns**, **remove any unnecessary duplicates**, and **rename them appropriately** based on the context of the analysis. This step is crucial for ensuring data integrity and consistency after the merge

After removing the duplicate column from **merged_data**, we iterate through each column name in the DataFrame. During this iteration, we check for unwanted suffixes or prefixes like **"_x", "_y", or "\n"** that **pandas** automatically adds during the merge. We then rename the columns by removing or replacing these default additions, ensuring the column names are cleaner and more readable for further analysis.

During the **error correction** stage, rather than removing missing values (such as **NaN or -9**), replacing them with the string "Unknown" enhances data visibility without compromising its integrity. Columns like "**Articulated Truck Involvement**" and "**Rigid Truck**" contain a considerable amount of missing data. Eliminating these entries without a proper contextual understanding could lead **to inaccurate or biased outcomes**. Therefore, treating missing and null values as "**Unknown**" provides a more practical and consistent solution

Since the data has been merged, it's important to check for any extra spaces or **inconsistencies**. To address this, we are stripping the ends of the strings. Additionally, while inspecting the data file, we noticed that some states are represented in lowercase **(e.g., "Qld", "Vic").** Converting all the data in the **"State"** column **to uppercase** ensures consistency, helping maintain **data integrity** and uniformity, especially when visualising the data.

To enhance our visualisation of road users, we'll convert any missing values or -9 values in the **'road user'** column to **'Others'**. This will be converted into Others instead of "**Unknown**" creates a difference while visualisation

In the **Extract, Transform, Load (ETL**) process, minimising data loss is essential to preserve the integrity and representativeness of the dataset. It is generally recommended not to discard more **than 5% of the total data**, as exceeding this threshold can **introduce bias and compromise** the validity of results by distorting the original data distribution.

In our case, the analysis primarily focuses on **Age** and **Speed Limit**. To ensure accurate results and clear visualisations, we have chosen to remove rows where Speed Limit is unknown. This results in the removal of 1,564 rows from both the Age and Speed Limit columns, which accounts for less than **5% of the total dataset (56,700 rows).** This removal remains within acceptable limits to maintain analytical accuracy.

After completing the **data cleaning process** and gaining a comprehensive understanding of the dataset, we proceeded to construct a **hierarchical dimensional model.** Based on the conceptual hierarchy observed in the data, we designed a data warehouse consisting of **nine dimensional tables**. The columns were selected and organised into appropriate **dimensional and fact tables**. Further details regarding the structure and content of these dimensional and fact tables will be provided in the **Implementation and design section**

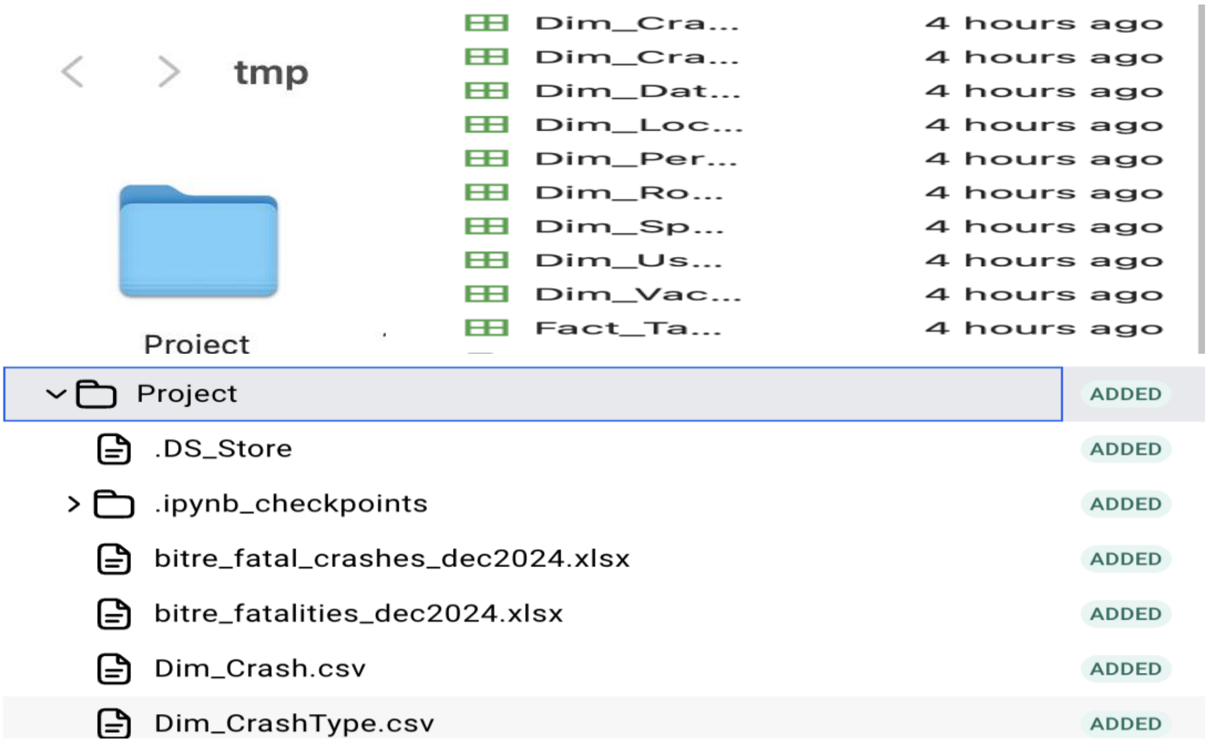
To create the dimensional tables, we selected attributes based on the identified dimensions. For example, the **dim_crash** table includes all crash involvement-related attributes such as **"Bus Involvement", "Heavy Rigid Truck Involvement", and "Articulated Truck Involvement"**. Since we observed a significant amount of repeated data, we removed duplicate records and assigned a unique primary key called **Involvement ID** to each entry. The resulting table was then saved as a **CSV file for further use in the data warehouse**.

We will repeat this process for all other dimensional tables and subsequently merge them into a single consolidated dataset, referred to as **Merged_Data**. This merging is performed using **the unique attributes (primary keys**) from **each dimensional table,** ensuring that the relationships are maintained. The resulting dataset will serve as the foundation for constructing the **fact table**.

After merging the data into a comprehensive dataset containing all **the primary keys (IDs**) from the dimensional tables, we proceed to create the fact table. This involves selecting relevant attributes and including the foreign keys from each dimension. Following the **star schema design**, our fact table consists of **a unique Fact ID, Crash ID, time-related attributes, the number of fatalities, and the foreign keys** referencing the related dimensional tables.

Load

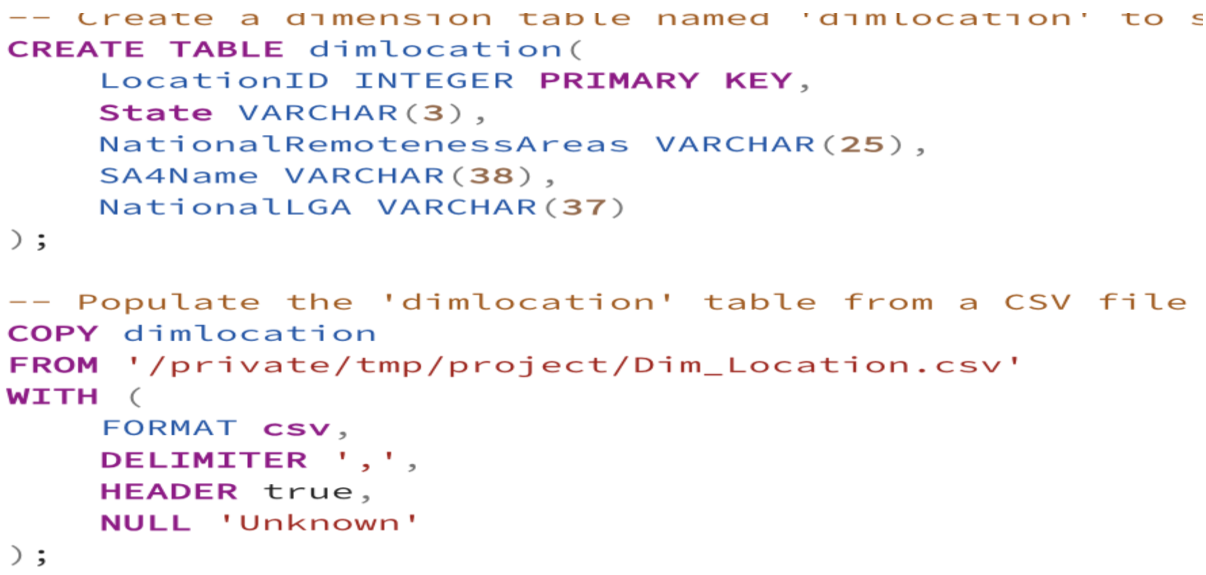
After completing the extraction and transformation processes, the data is cleaned and structured appropriately. The next step involves loading the filtered and processed data into the target operational database or data warehouse, where it will be used for analysis and visualisation purposes.



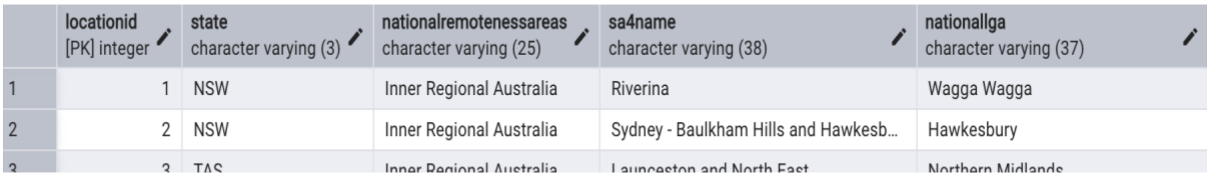
Dim_Cra...	4 hours ago
Dim_Cra...	4 hours ago
Dim_Dat...	4 hours ago
Dim_Loc...	4 hours ago
Dim_Per...	4 hours ago
Dim_Ro...	4 hours ago
Dim_Sp...	4 hours ago
Dim_Us...	4 hours ago
Dim_Vac...	4 hours ago
Fact_Ta...	4 hours ago

Project	ADDED
.DS_Store	ADDED
.ipynb_checkpoints	ADDED
bitre_fatal_crashes_dec2024.xlsx	ADDED
bitre_fatalities_dec2024.xlsx	ADDED
Dim_Crash.csv	ADDED
Dim_CrashType.csv	ADDED

After transferring the files to the **tmp directory** locally and uploading them into the **Docker environment**, we set up a **PostgreSQL database** server. Based on the **dimensional mode**l, we create the necessary tables aligned with our schema design. Once the table structures are verified—ensuring that all required columns are correctly defined—we proceed to load the data into the target database. This process completes **the data population phase, preparing the system for querying and analysis**.



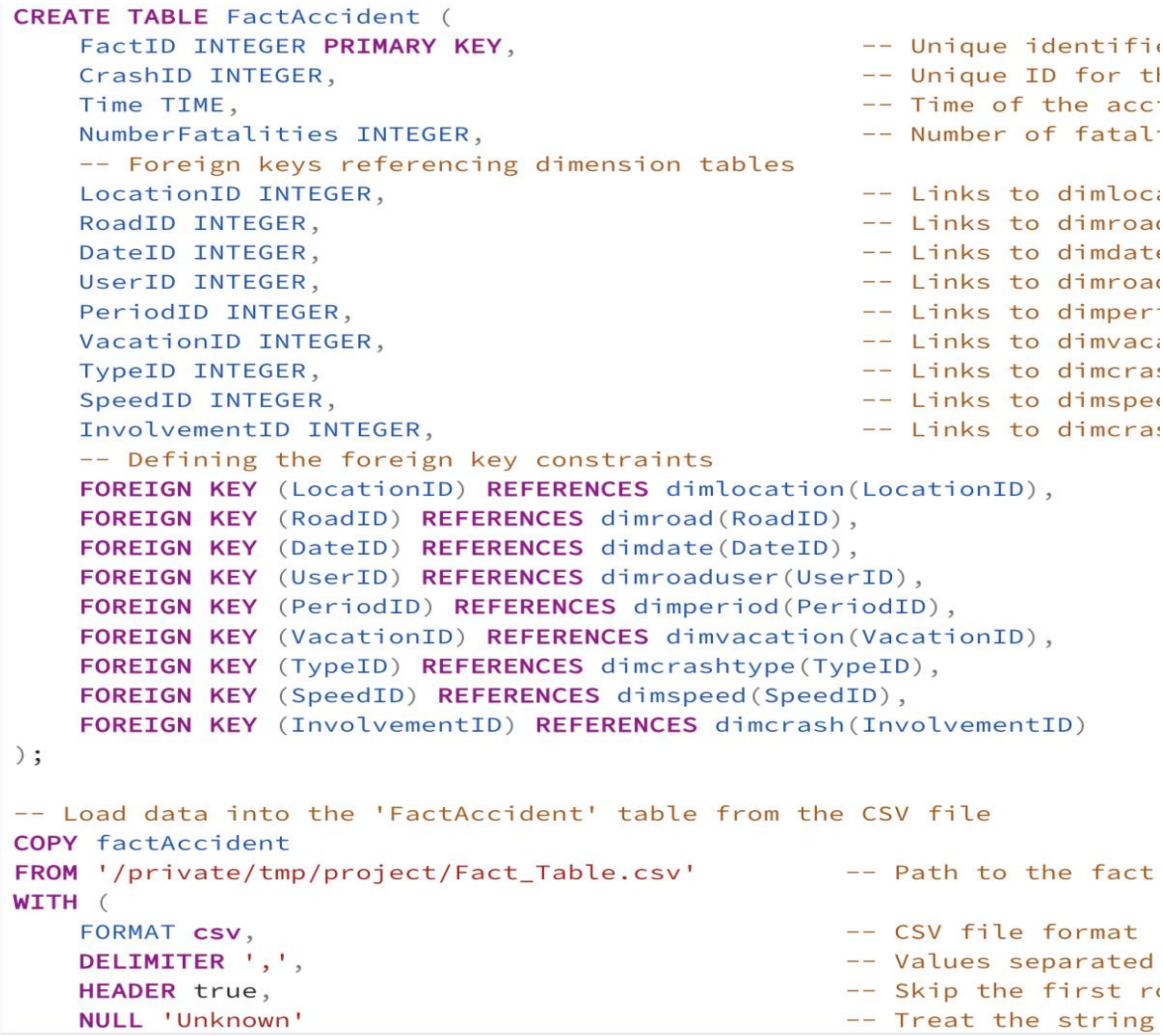
```
-- Create a dimension table named 'dimlocation' to s
CREATE TABLE dimlocation(
    LocationID INTEGER PRIMARY KEY,
    State VARCHAR(3),
    NationalRemotenessAreas VARCHAR(25),
    SA4Name VARCHAR(38),
    NationalLGA VARCHAR(37)
);

-- Populate the 'dimlocation' table from a CSV file
COPY dimlocation
FROM '/private/tmp/project/Dim_Location.csv'
WITH (
    FORMAT csv,
    DELIMITER ',',
    HEADER true,
    NULL 'Unknown'
);
```



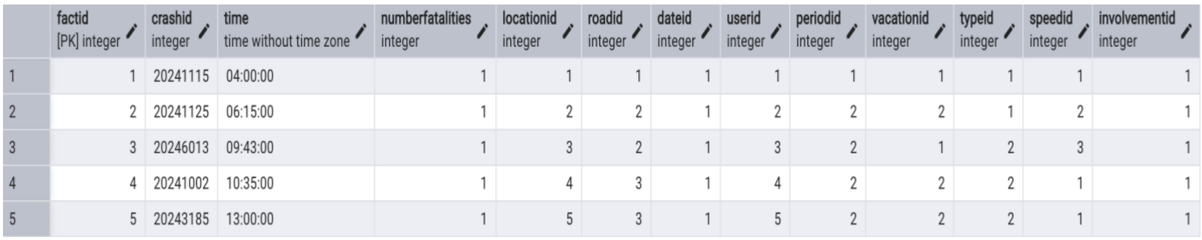
	locationid [PK] integer	state character varying (3)	nationalremotenessareas character varying (25)	sa4name character varying (38)	nationallga character varying (37)
1	1	NSW	Inner Regional Australia	Riverina	Wagga Wagga
2	2	NSW	Inner Regional Australia	Sydney - Baulkham Hills and Hawkesb…	Hawkesbury
3	3	TAS	Inner Regional Australia	Launceston and North East	Northern Midlands

Create the remaining dimension tables and load the corresponding data into the database using the **COPY command**. Once all the dimension tables are populated, create the **fact table**. Ensure that **the primary keys** from each dimension table are included as **foreign keys** in the **fact table** to maintain referential integrity.

Additionally, we replace unknown or invalid values with **NULLs** to ensure compatibility and integrity when loading the data into the database.

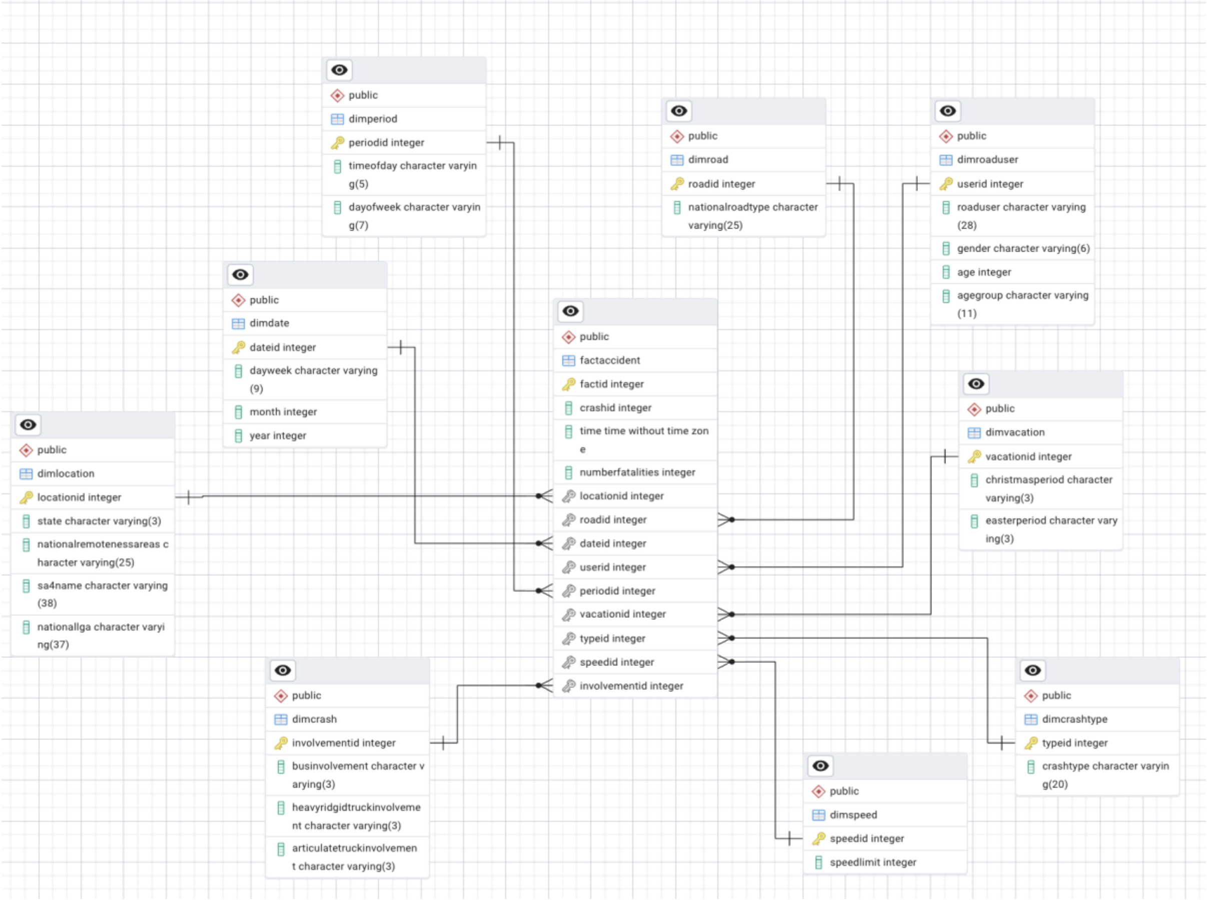


```
CREATE TABLE FactAccident (
    FactID INTEGER PRIMARY KEY,                    -- Unique identifi
    CrashID INTEGER,                               -- Unique ID for th
    Time TIME,                                     -- Time of the acc
    NumberFatalities INTEGER,                      -- Number of fatal
    -- Foreign keys referencing dimension tables
    LocationID INTEGER,                            -- Links to dimloca
    RoadID INTEGER,                                -- Links to dimroad
    DateID INTEGER,                                -- Links to dimdate
    UserID INTEGER,                                -- Links to dimroad
    PeriodID INTEGER,                              -- Links to dimper
    VacationID INTEGER,                            -- Links to dimvaca
    TypeID INTEGER,                                -- Links to dimcras
    SpeedID INTEGER,                               -- Links to dimspee
    InvolvementID INTEGER,                         -- Links to dimcras
    -- Defining the foreign key constraints
    FOREIGN KEY (LocationID) REFERENCES dimlocation(LocationID),
    FOREIGN KEY (RoadID) REFERENCES dimroad(RoadID),
    FOREIGN KEY (DateID) REFERENCES dimdate(DateID),
    FOREIGN KEY (UserID) REFERENCES dimroaduser(UserID),
    FOREIGN KEY (PeriodID) REFERENCES dimperiod(PeriodID),
    FOREIGN KEY (VacationID) REFERENCES dimvacation(VacationID),
    FOREIGN KEY (TypeID) REFERENCES dimcrashtype(TypeID),
    FOREIGN KEY (SpeedID) REFERENCES dimspeed(SpeedID),
    FOREIGN KEY (InvolvementID) REFERENCES dimcrash(InvolvementID)
);

-- Load data into the 'FactAccident' table from the CSV file
COPY factAccident
FROM '/private/tmp/project/Fact_Table.csv'        -- Path to the fact
WITH (
    FORMAT csv,                                   -- CSV file format
    DELIMITER ',',                                -- Values separated
    HEADER true,                                  -- Skip the first ro
    NULL 'Unknown'                                -- Treat the string
```



	factid [PK] integer	crashid integer	time time without time zone	numberfatalities integer	locationid integer	roadid integer	dateid integer	userid integer	periodid integer	vacationid integer	typeid integer	speedid integer	involvementid integer
1	1	20241115	04:00:00	1	1	1	1	1	1	1	1	1	1
2	2	20241125	06:15:00	1	2	2	1	2	2	2	1	2	1
3	3	20246013	09:43:00	1	3	2	1	3	2	1	2	3	1
4	4	20241002	10:35:00	1	4	3	1	4	2	2	2	1	1
5	5	20243185	13:00:00	1	5	3	1	5	2	2	2	1	1

Implementations and Schema


public
dimperiod
periodid integer
timeofday character varying(5)
dayofweek character varying(7)
public
dimroad
roadid integer
nationalroadtype character varying(25)
public
dimroaduser
userid integer
roaduser character varying (28)
gender character varying(6)
age integer
agegroup character varying (11)
public
dimdate
dateid integer
dayweek character varying (9)
month integer
year integer
public
factaccident
factid integer
crashid integer
time time without time zone
numberfatalities integer
locationid integer
roadid integer
dateid integer
userid integer
periodid integer
vacationid integer
typeid integer
speedid integer
involvementid integer
public
dimlocation
locationid integer
state character varying(3)
nationalremotenessareas character varying(25)
sa4name character varying (38)
nationallga character varying(37)
public
dimvacation
vacationid integer
christmasperiod character varying(3)
easterperiod character varying(3)
public
dimcrash
involvementid integer
businvolvement character varying(3)
heavyridgidtruckinvolvement character varying(3)
articulatetruckinvolvement character varying(3)
public
dimcrashtype
typeid integer
crashtype character varying(20)
public
dimspeed
speedid integer
speedlimit integer

Star Schema

In this data warehouse, we designed a **Star Schema** consisting of a central **fact table** connected to multiple **dimension tables**. These dimension tables are linked to the fact table using **foreign keys (surrogate keys)**, which serve as the primary identifiers in each dimension.
The star schema provides a well-organised structure that simplifies both basic and complex data reporting. It enables efficient querying and is commonly used for analytical purposes due to its straightforward layout. However, it's important to note that the star schema focuses on the relational structure for reporting and does not inherently support advanced features like **security or auditing** within its multi-dimensional design—these must be implemented separately on top of the relational tables. (Amin, M. M., Sutrisman, A., & Dwitayanti, Y. (2021).)

Dimensional Tables

Conceptual Hierarchy is the process of organising data within a database into multiple levels of abstraction, either from general to specific or based on logical groupings in a layer or different levels. It structures the data in a meaningful way, making it easier to analyse and interpret. This is a key technique in **data mining**, and is commonly used in **dimensional modeling** and **data warehousing** to support multi-level analysis and decision-making.

Location Dimension

Attributes

- *LocationID.   (Primary key)*
- *State*
- *National remoteness areas*
- *SA4 name*
- *National LGA name*

The **Location Dimension** captures details about where each fatal crash occurred. It includes information such as the **Australian state** where the incident took place and the corresponding **National remoteness area** classification (e.g., Major Cities, Remote). Additionally, it identifies the specific **SA4 region** and the **National Local Government Area (LGA)** that the location falls under. This structure helps in accurately categorising and analysing crash data based on geographic context.

Conceptual Hierarchy, These attributes in the **Location Dimension** define the grain level of the crash region, enabling drill-down from the **National LGA** area to the **SA4** regions, which are part of the **national remoteness areas** and fall within specific states

National LGA name < SA4 regions < National remoteness areas < State

Date Dimension

Attributes

- *DateID (Primary key)*
- *DayWeek*
- *Month*
- *Year*

The **Date Dimension** describes the **day, month,** and **year** of each fatal crash, covering data from 1989 to 2024. It includes all **months and days**, and this dimensional table is essential for visualising road incidents based on their **dates**.

Conceptual Hierarchy, The attributes in the **Date Dimension** define the grain level of the crash timing, allowing for drill-down from the **dayweek** to the month and from the **month** to the **year**. This hierarchy helps us visualise the timing of crashes at various levels of granularity.

Dayweek < Month < Year

Road User Dimensional

Attributes

- *UserID (Primary key)*
- *RoadUser*

- *Gender*
- *Age*
- *AgeGroup*

The **Road User Dimension** explains **demographic information** about fatalities involved in crashes, including the **type of road user** (e.g., driver, passenger, pedestrian), their **gender**, **age, and age group range**. This dimension focuses on providing detailed demographic data about the fatalities, which supports analysis and insights into the characteristics of those involved in fatal crashes.

The **Conceptual Hierarchy** of these attributes explains the grain level of the demographic features of fatalities, starting from the type of road user to their gender, age, and the corresponding age range they fall under. These attributes are organised in a **schema hierarchy** and follow a **partial order**, representing a structured relationship between the different levels of demographic information.

{RoadUser < Gender} < Age < AgeGroup

Road Dimension

Attributes

- *RoadID (Primary key)*
- *NationalRoadType*

The **Road Dimension** stores details about the types of roads where accidents occur, including data such as **arterial roads, local roads**, and **highways**. This dimension features a single attribute and helps visualisation

e the distribution of accidents across **different road types**.

The **Conceptual Hierarchy** of this attribute provides grain-level information about the types of roads where crashes occur. Although it follows a **single-level hierarchy**, it establishes a crucial link between road types and crash occurrences.

Road → ***NationalRoadType***

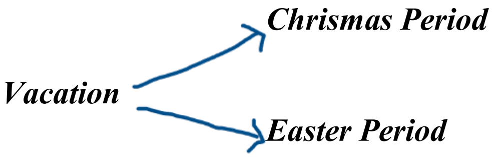
Vacation Dimension

Attributes

- *VacationID (Primary Key)*
- *ChristmasPeriod*
- *EasterPeriod*

The **Vacation Dimension** contains attributes related to the timing of the crash event, specifically whether it occurred during the **Christmas or Easter period**. This dimension helps us understand how **holidays and vacation** periods contribute to the occurrence of major accidents in Australia

The **Concept Hierarchy,** the attributes illustrates the grain level of crash events occurring on significant days, such as **vacation** periods like **Easter or Christmas**, to provide insights into the impact of these holidays on accidents. This hierarchy follows a **set grouping structure**.



Period Dimension

Attributes

- *PeriodID (Primary Key)*
- *TimeofDay*
- *DayofWeek*

The **Period Dimension** includes attributes such as the **Time of Day**, indicating whether the fatal crash occurred during the day or night, and **Day of Week**, which identifies whether the crash happened on a **weekday or weekend**. This dimension provides insights into the timing patterns of crashes.

The **Conceptual Hierarchy** of these attributes explains the grain level of crashes in specific periods, such as whether they occurred during the day or night, and whether it was on a weekday or weekend in a roll up. It follows a **schema hierarchy** with a **total order**, providing insights into the relationship between the period and crash occurrences.

TimeofDay < DayofWeek

Speed Dimension

Attributes

- *SpeedID (Primary Key)*
- *SpeedLimit*

The **Speed Dimension** consists of a single attribute called **SpeedLimit**, which records the speed limit at the location of the crash. The **Conceptual Hierarchy** for this dimension is a single-level hierarchy. This attribute establishes the connection between speed limits and crash occurrences.

Speed → ***SpeedLimit***

Crash Dimension

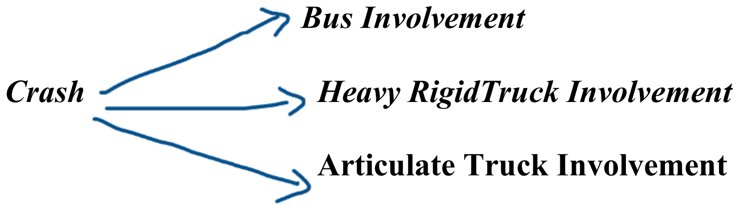
Attributes

- *InvolvementID (Primary Key)*

- *BusInvolvement*
- *HeavyRidigidTruckInvolvement*
- *ArticulateTruckInvolvement*

The **Crash Dimension** captures detailed information about the vehicles involved in a crash. It records the grain level of crash involvement, including attributes such as **Bus Involvement**, **Heavy Rigid Truck Involvement**, and **Articulated Truck Involvement**. These details are essential for analysing the types of vehicles involved in crashes and understanding patterns in vehicle-related incidents.

The **Conceptual Hierarchy** groups these attributes based on the number of crashes, following a **set grouping structure hierarchy**. In this hierarchy, data is organised according to the type of vehicle involved in the crash, allowing for meaningful analysis of **crash patterns** by **vehicle category**.



Crash Type Dimension

Attributes

- *TypeID (Primary Key)*
- *CrashType*

The **Crash Type Dimension** contains information about the number of vehicles involved in a crash, indicating whether it was a **single-vehicle** or **multi-vehicle** incident. The **Conceptual Hierarchy** for this attribute follows a **single-level hierarchy**, highlighting the relationship between the number of vehicles and the nature of the crash.

Crash Type Dimension → ***CrashType***

Fact Measurement Table

Fact Accident

Attributes

- *FactID.   (Primary key)*
- *CrashID*
- *Time*
- *NumberFatalities*
- *LocationID*
- *RoadID*

- *DateID*
- *UserID*
- *PeriodID*
- *VacationID*
- *TypeID*
- *SpeedID*
- *Involvement ID*

The **FactAccidents** table includes a primary key, **Fact ID**, which uniquely identifies each record. It also contains the **Crash ID**, the **time of the crash**, and the **number of fatalities**, extracted from the merged dataset. Additionally, the table incorporates surrogate keys from all related dimension tables: **Location ID**, **Road ID**, **Date ID**, **User ID**, **Period ID**, **Vacation ID**, **Type ID**, **Speed ID**, and **Involvement ID**.
This fact table serves as the foundation for analysis, particularly focusing on the **number of fatalities**. The data warehouse is structured using a **star schema**, allowing efficient querying and multidimensional analysis, details above provided

Kimball's Analysis

There are several approaches to data modeling, and we are using **Kimball's methodology**, which is widely adopted and considered a best practice in the industry. This approach follows a **bottom-up design**, starting with the development of data marts that are later integrated into a complete data warehouse.
Kimball's technique emphasises the use of a **star schema**, which structures data into **fact tables** for quantitative data and **dimension tables** for descriptive context. Although it involves some denormalisation, this structure enhances performance and makes data easier to understand and use.
The star schema is especially valuable because it provides a **consistent and proven method** for designing data warehouses. It supports efficient data analysis, reporting, and business intelligence processes.

Pick a process to model

We are examining the number of fatalities in fatal crashes as the core element for building our data warehouse. Our goal is to understand how factors such as location and time of day impact the number of fatalities, which will help identify the causes of the issue. This approach differs from earlier modelling techniques, like Bill Inmon's, which focused on business entities (e.g., customer model, product model, etc.) as the starting point.

Declaring the Grain

Identifying the levels of detail that can be captured in the fact table is crucial. The grain must be declared before selecting dimensions because the facts must align with the defined grain. The grain refers to the lowest level of data, or the atomic level, and can also include rolled-up summary grains, depending on the analysis requirement. Refer **Implementation and schema**

Identification of Dimensions

Dimensions provide the context for analysis or process events. These tables consist of descriptive attributes used by visualisation tools like Power BI or Tableau to filter, group, and modify data fields. Identifying all possible dimensions is essential for effective analysis. Dimensional tables are often referred to as the *heart* of a data warehouse, as they serve as the entry points and provide the descriptive labels that enable meaningful analysis
Based on the dataset, we derive nine dimensional tables. These dimensions represent key factors such as **Location, Crash Involvement, Fatality Details, Speed, Crash Type, Vacation Period, Date, Road Type, and the Time Period of the accident.**

Determining the Facts

Developing a **fact table** is a core component of building a data warehouse. Fact tables store **measurements or metrics** that result from business process events, and these are typically numerical data. Each row in a fact table is directly linked to relevant **dimension tables**, providing context for the stored facts. Refer **implementation and schema**

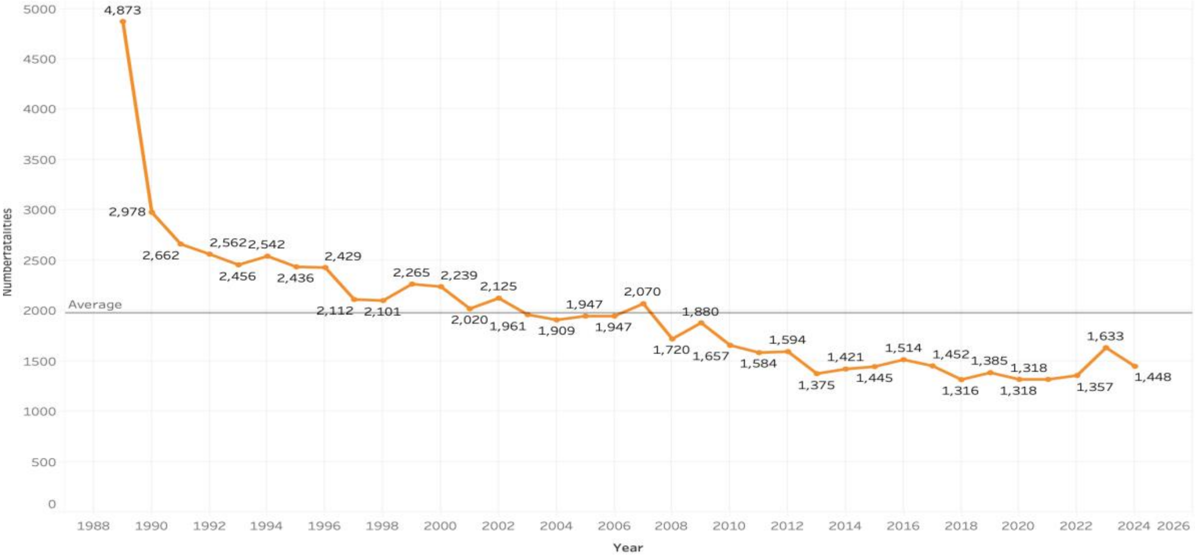
Implementing a fact table improves **data storage efficiency** by organising data into a structured tabular format, which also enhances the performance of analytical queries. Fact tables are essential in **OLAP (Online Analytical Processing)** systems as they support operations like **SUM**, **AVERAGE**, **MIN**, **MAX**, and **COUNT**, enabling powerful and flexible data analysis. Refer **Implementation and schema**

Analysis Queries (Business Queries)

About the relation between the demographics and date

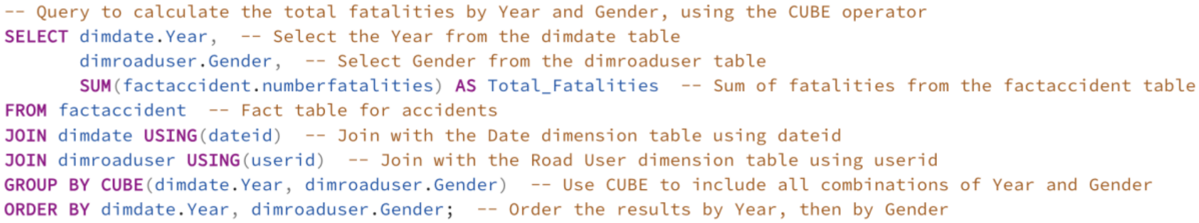
"What is the total number of road fatalities recorded each year, including the overall total across all years?"

```
-- Query to calculate the total fatalities by Year, using the CUBE operator to include all possible combinations
SELECT dimdate.Year,  -- Select the Year from the dimdate table
       SUM(factaccident.numberfatalities) AS Total_Fatalities  -- Sum of fatalities from the factaccident table
FROM factaccident  -- Fact table for accidents
JOIN dimdate USING(dateid)  -- Join with the Date dimension table using dateid
GROUP BY CUBE(dimdate.Year)  -- Use CUBE to include aggregations for each combination of Year (including nulls)
ORDER BY dimdate.Year;  -- Order the results by Year
```

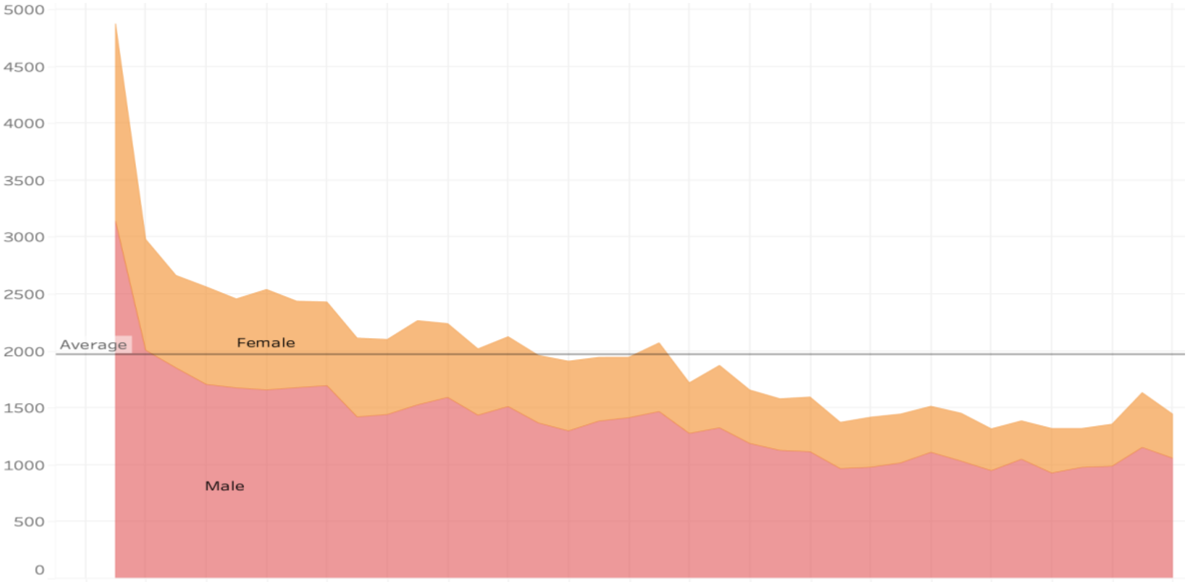


The line graph clearly shows a gradual decline in the number of fatalities from crashes between 1989 and 2020. The numbers then remained relatively stable until 2022, followed by a noticeable increase in 2023 and a decline again in 2024. **The highest number of fatalities occurred in 1989 with 4,873.**

"What is the total number of road fatalities each year, broken down by gender, including yearly totals and overall totals by gender and year?



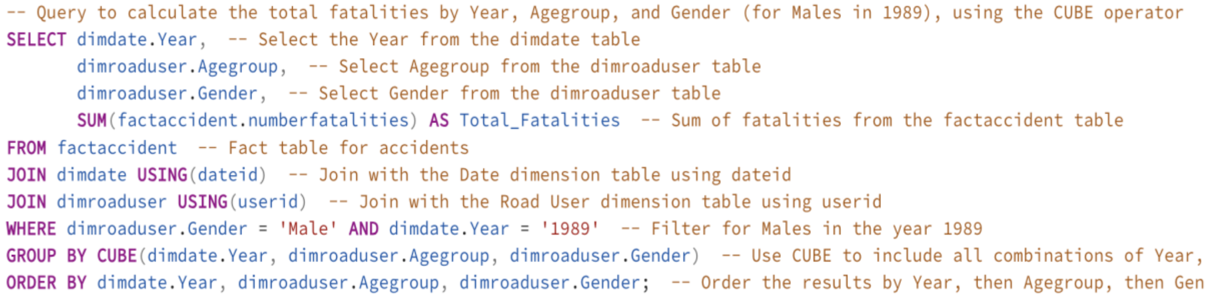
```
-- Query to calculate the total fatalities by Year and Gender, using the CUBE operator
SELECT dimdate.Year,  -- Select the Year from the dimdate table
       dimroaduser.Gender,  -- Select Gender from the dimroaduser table
       SUM(factaccident.numberfatalities) AS Total_Fatalities  -- Sum of fatalities from the factaccident table
FROM factaccident  -- Fact table for accidents
JOIN dimdate USING(dateid)  -- Join with the Date dimension table using dateid
JOIN dimroaduser USING(userid)  -- Join with the Road User dimension table using userid
GROUP BY CUBE(dimdate.Year, dimroaduser.Gender)  -- Use CUBE to include all combinations of Year and Gender
ORDER BY dimdate.Year, dimroaduser.Gender;  -- Order the results by Year, then by Gender
```



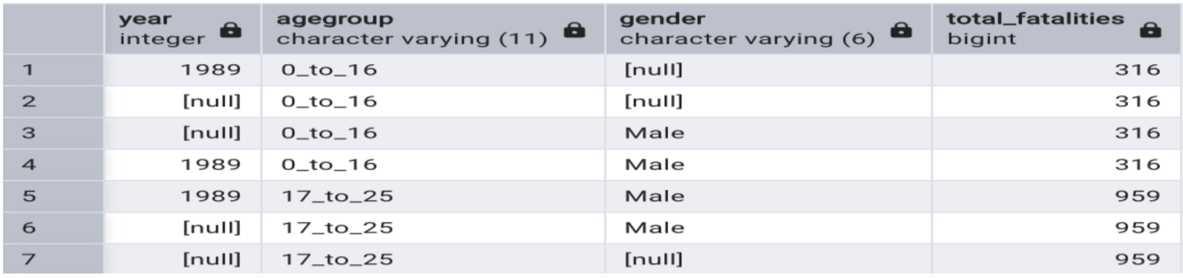
From the area graph, it is evident that **the majority of fatalities are male**, and this trend remains consistent across the years. However, there is a noticeable decline in fatalities involving females, whereas fatalities among males show only a slight downward variation over time.

Based on the above analysis, we will visualise the relationship between male fatalities—who represent the highest number of overall fatalities—and the different age groups.

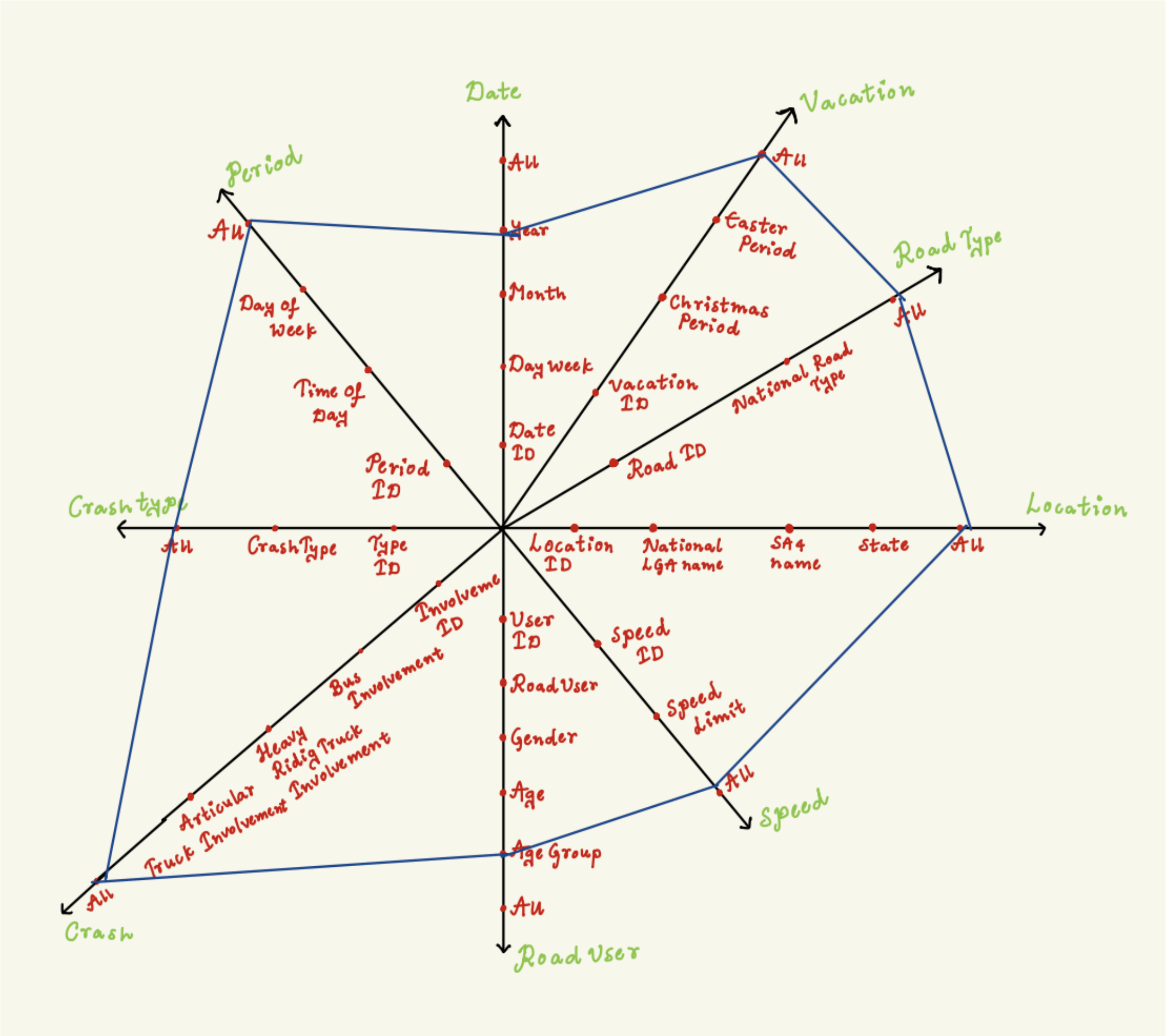
"What is the total number of fatalities among male road users in the year 1989, across different age groups?"



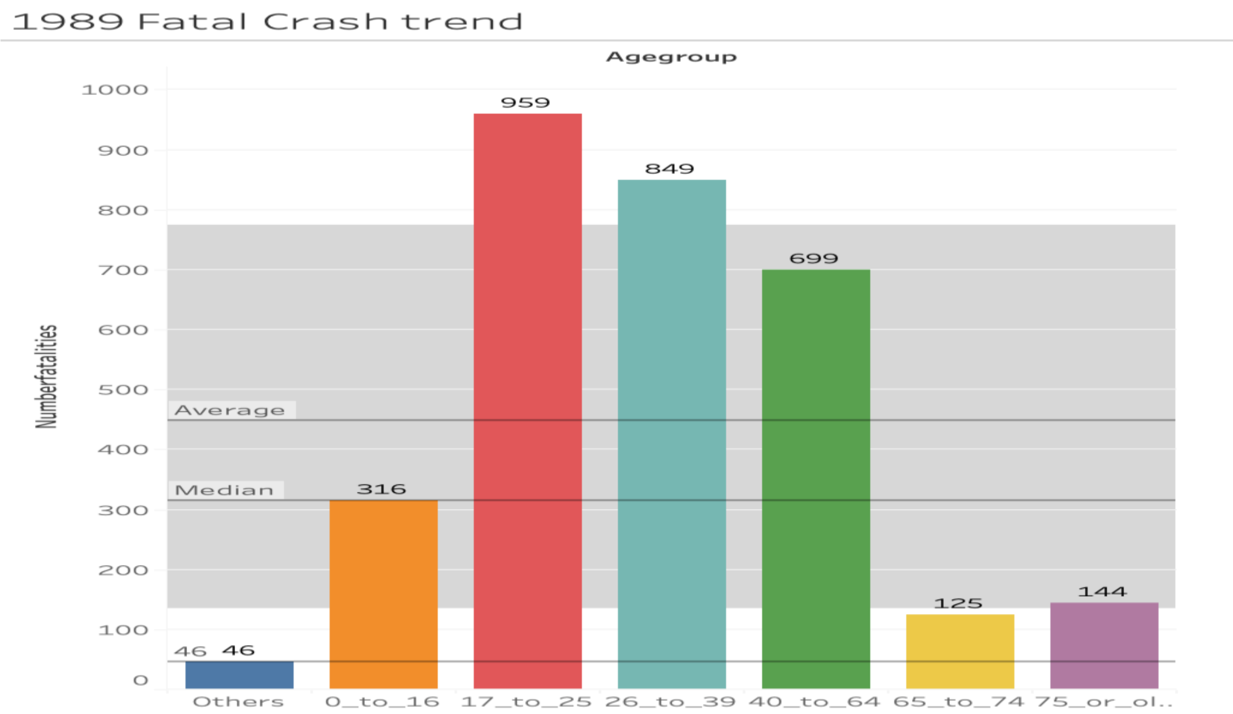
```
-- Query to calculate the total fatalities by Year, Agegroup, and Gender (for Males in 1989), using the CUBE operator
SELECT dimdate.Year,  -- Select the Year from the dimdate table
       dimroaduser.Agegroup,  -- Select Agegroup from the dimroaduser table
       dimroaduser.Gender,  -- Select Gender from the dimroaduser table
       SUM(factaccident.numberfatalities) AS Total_Fatalities  -- Sum of fatalities from the factaccident table
FROM factaccident  -- Fact table for accidents
JOIN dimdate USING(dateid)  -- Join with the Date dimension table using dateid
JOIN dimroaduser USING(userid)  -- Join with the Road User dimension table using userid
WHERE dimroaduser.Gender = 'Male' AND dimdate.Year = '1989'  -- Filter for Males in the year 1989
GROUP BY CUBE(dimdate.Year, dimroaduser.Agegroup, dimroaduser.Gender)  -- Use CUBE to include all combinations of Year,
ORDER BY dimdate.Year, dimroaduser.Agegroup, dimroaduser.Gender;  -- Order the results by Year, then Agegroup, then Gen
```



	year integer	agegroup character varying (11)	gender character varying (6)	total_fatalities bigint
1	1989	0_to_16	[null]	316
2	[null]	0_to_16	[null]	316
3	[null]	0_to_16	Male	316
4	1989	0_to_16	Male	316
5	1989	17_to_25	Male	959
6	[null]	17_to_25	Male	959
7	[null]	17_to_25	[null]	959

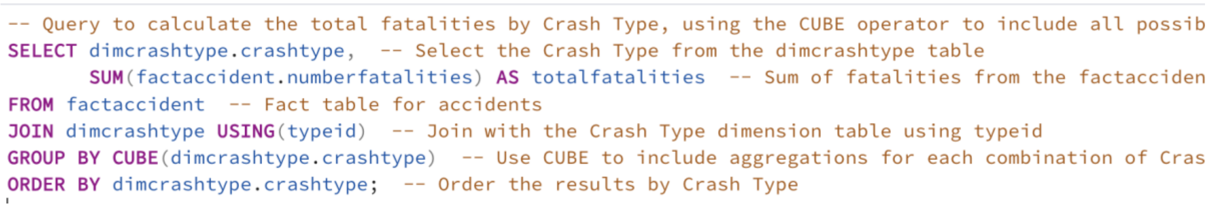


In the bar graph above, males aged between **17 and 25 recorded the highest number of fatalities, with 959 deaths**. This is closely followed by the 26 to 39 age group, which had 849 fatalities. The 40 to 64 age group accounted for nearly 700 deaths. Children under the age of 16 had a median fatality count of 316. **For older age groups, there were 125 fatalities among those aged 65 to 74**, and 144 fatalities for those aged 75 and above. Additionally, there were 46 cases where the age of the male victims was unknown



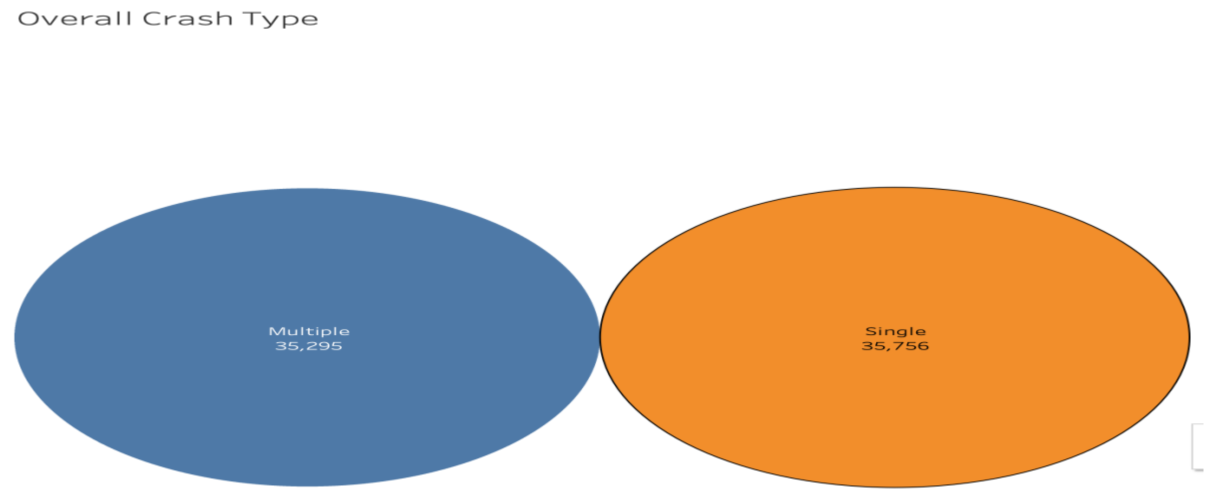
Relation between the road and involvement

What is the total number of fatalities for each crash type, including the overall total across all crash types?

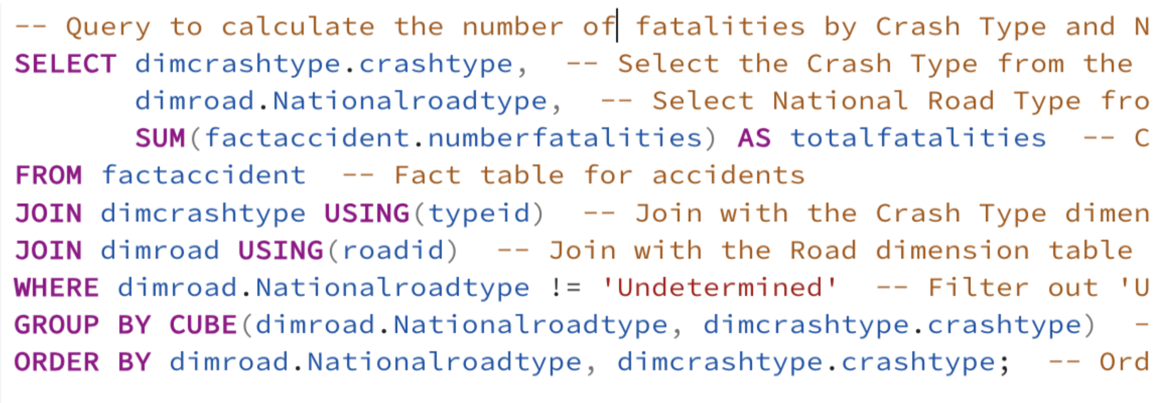


```
-- Query to calculate the total fatalities by Crash Type, using the CUBE operator to include all possib
SELECT dimcrashtype.crashtype,  -- Select the Crash Type from the dimcrashtype table
       SUM(factaccident.numberfatalities) AS totalfatalities  -- Sum of fatalities from the factacciden
FROM factaccident  -- Fact table for accidents
JOIN dimcrashtype USING(typeid)  -- Join with the Crash Type dimension table using typeid
GROUP BY CUBE(dimcrashtype.crashtype)  -- Use CUBE to include aggregations for each combination of Cras
ORDER BY dimcrashtype.crashtype;  -- Order the results by Crash Type
```

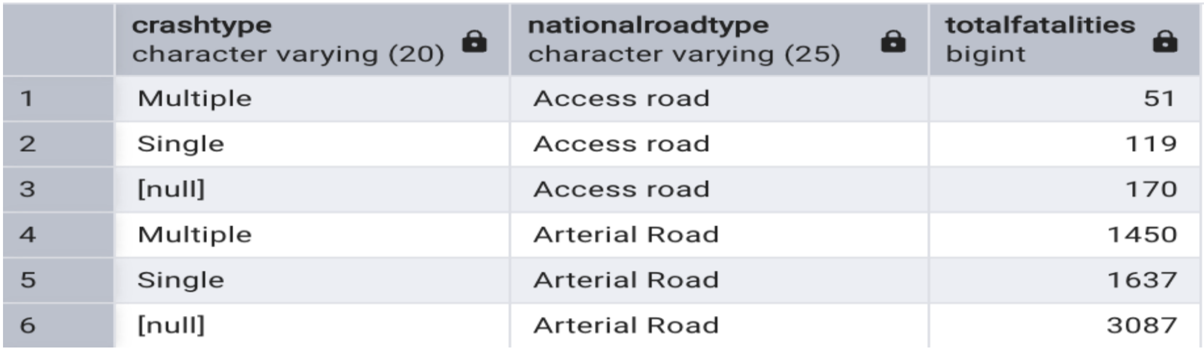
The difference in the number of fatalities caused by single-vehicle and multiple-vehicle crashes was 35,756 and 35,295, respectively, which is almost negligible.



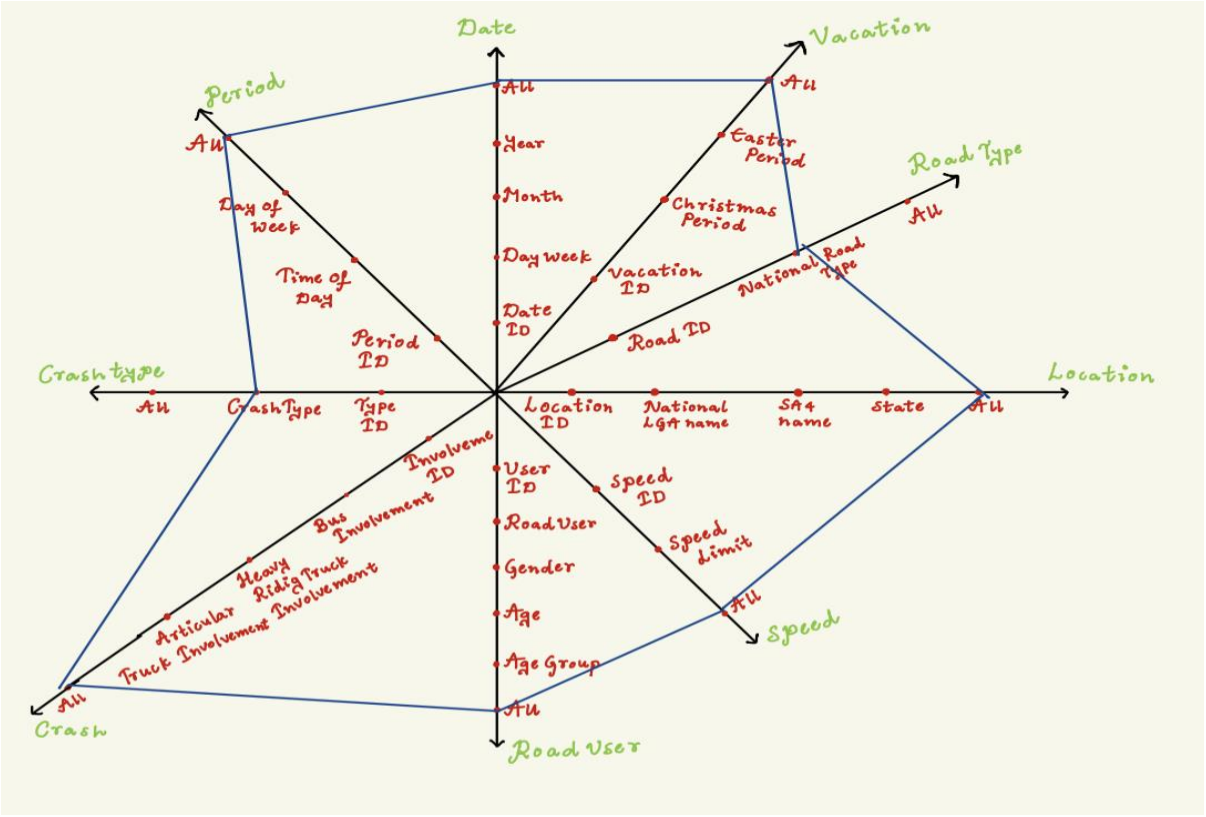
What is the total number of fatalities for each crash type, categorised by national road type, excluding 'Undetermined' road types?

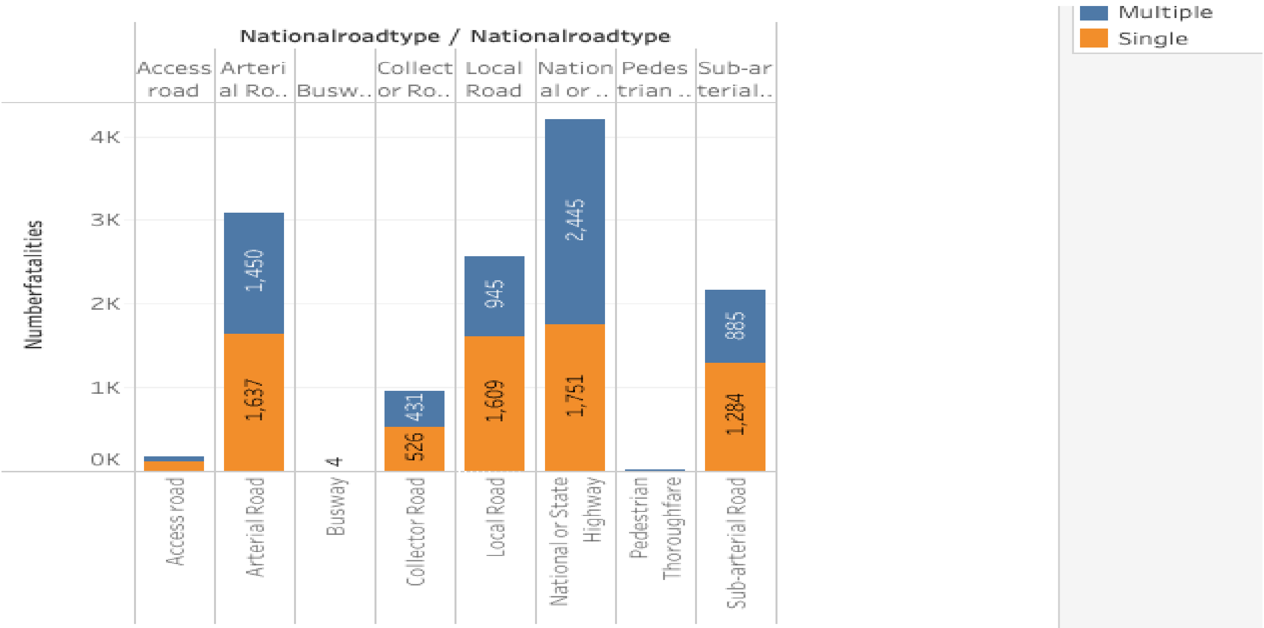


```
-- Query to calculate the number of fatalities by Crash Type and N
SELECT dimcrashtype.crashtype,   -- Select the Crash Type from the
       dimroad.Nationalroadtype,   -- Select National Road Type fro
       SUM(factaccident.numberfatalities) AS totalfatalities   -- C
FROM factaccident   -- Fact table for accidents
JOIN dimcrashtype USING(typeid)   -- Join with the Crash Type dimen
JOIN dimroad USING(roadid)   -- Join with the Road dimension table
WHERE dimroad.Nationalroadtype != 'Undetermined'   -- Filter out 'U
GROUP BY CUBE(dimroad.Nationalroadtype, dimcrashtype.crashtype)   -
ORDER BY dimroad.Nationalroadtype, dimcrashtype.crashtype;   -- Ord
```



	crashtype character varying (20)	nationalroadtype character varying (25)	totalfatalities bigint
1	Multiple	Access road	51
2	Single	Access road	119
3	[null]	Access road	170
4	Multiple	Arterial Road	1450
5	Single	Arterial Road	1637
6	[null]	Arterial Road	3087

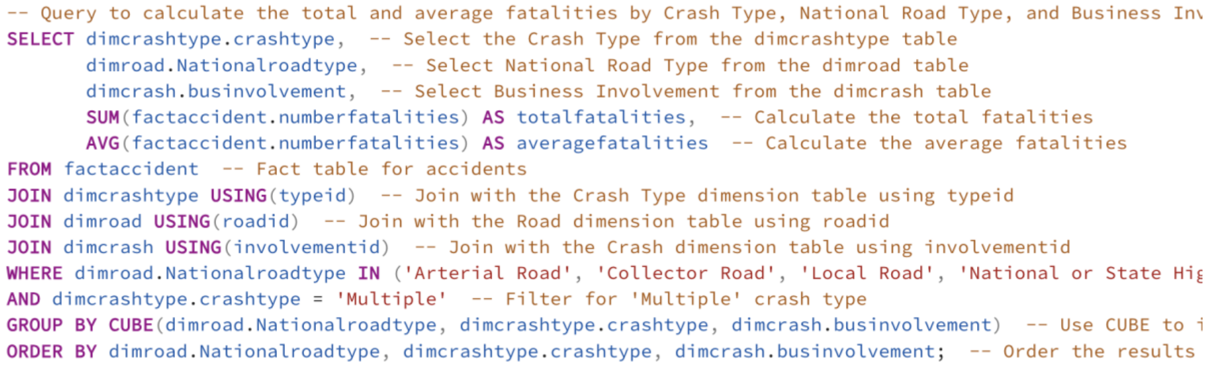




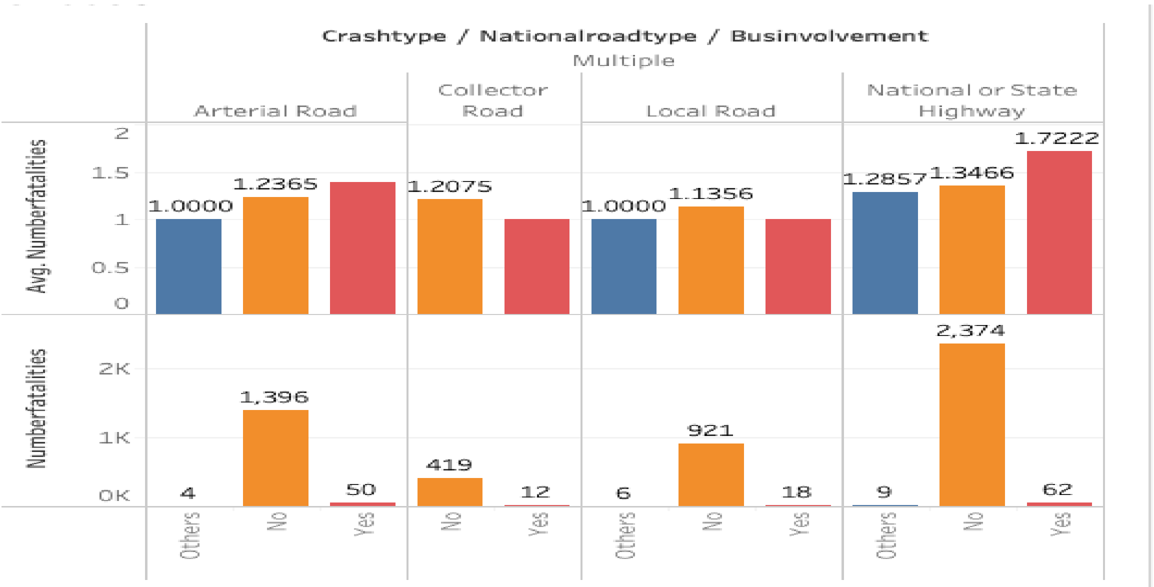
The highest total number of fatalities occurred on highways, with 4,196 fatalities, of which 1,751 were from single-vehicle crashes, and the rest resulted from multiple-vehicle crashes. Arterial roads saw just over 3,000 fatalities, with a relatively balanced distribution: 1,637 from single-vehicle crashes and 1,450 from multiple-vehicle crashes. Local roads and sub-arterial roads had fatalities ranging between 2,000 and 2,500, with most of these attributed to single-vehicle crashes. Access roads had around 200 fatalities, while busways and pedestrian thoroughfares recorded almost negligible fatalities, with only 4 people.

Based on the above analysis, we will focus on identifying the most dangerous roads and examine incidents involving multiple crashes with bus involvement.

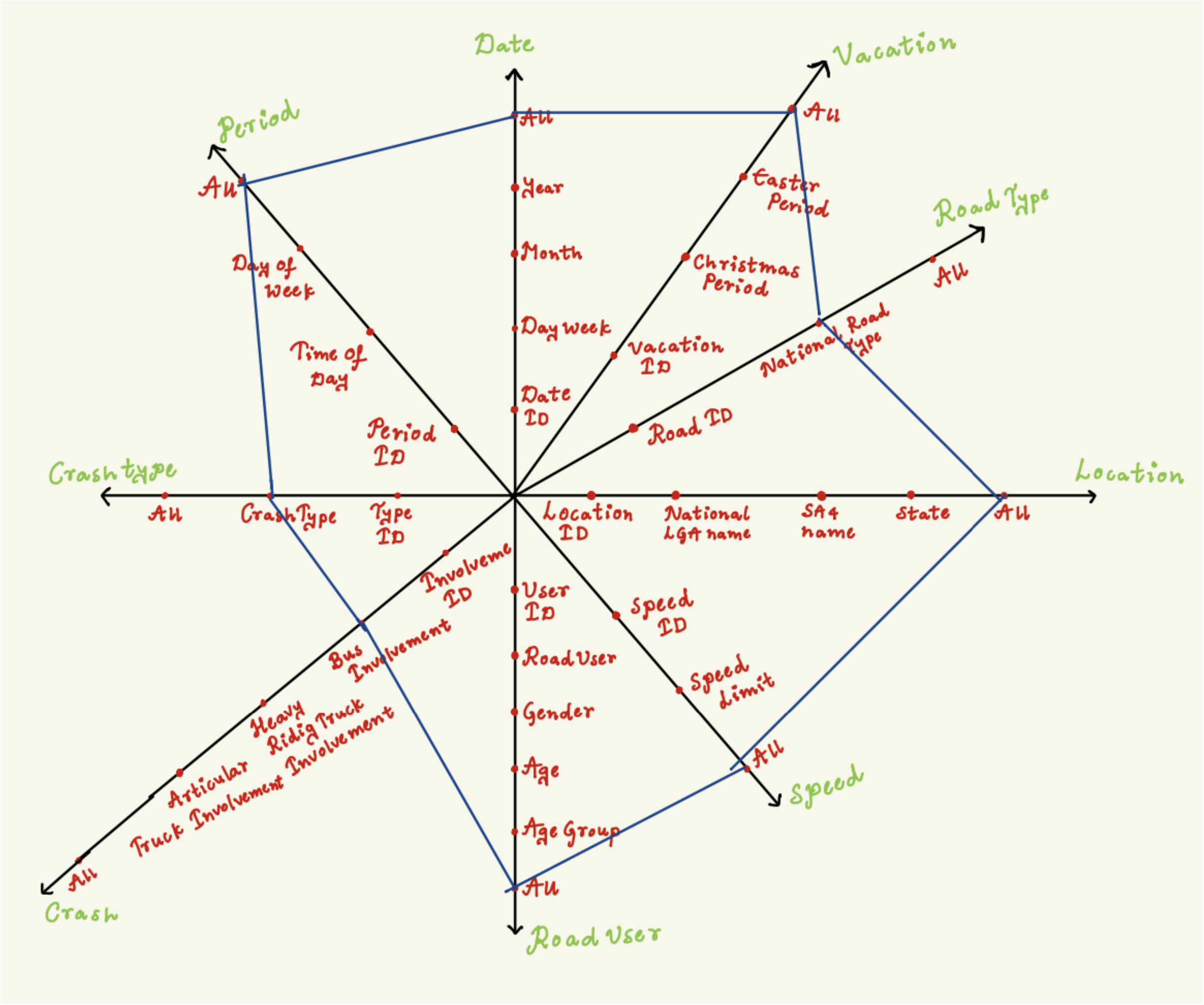
What is the total and average number of fatalities for 'Multiple' crash types, based on national road type and business involvement, for roads such as Arterial, Collector, Local, and National or State Highways?



```
-- Query to calculate the total and average fatalities by Crash Type, National Road Type, and Business Inv
SELECT dimcrashtype.crashtype,   -- Select the Crash Type from the dimcrashtype table
       dimroad.Nationalroadtype,  -- Select National Road Type from the dimroad table
       dimcrash.businvolvement,  -- Select Business Involvement from the dimcrash table
       SUM(factaccident.numberfatalities) AS totalfatalities,  -- Calculate the total fatalities
       AVG(factaccident.numberfatalities) AS averagefatalities  -- Calculate the average fatalities
FROM factaccident  -- Fact table for accidents
JOIN dimcrashtype USING(typeid)  -- Join with the Crash Type dimension table using typeid
JOIN dimroad USING(roadid)  -- Join with the Road dimension table using roadid
JOIN dimcrash USING(involvementid)  -- Join with the Crash dimension table using involvementid
WHERE dimroad.Nationalroadtype IN ('Arterial Road', 'Collector Road', 'Local Road', 'National or State Hig
AND dimcrashtype.crashtype = 'Multiple'  -- Filter for 'Multiple' crash type
GROUP BY CUBE(dimroad.Nationalroadtype, dimcrashtype.crashtype, dimcrash.businvolvement)  -- Use CUBE to i
ORDER BY dimroad.Nationalroadtype, dimcrashtype.crashtype, dimcrash.businvolvement;  -- Order the results
```

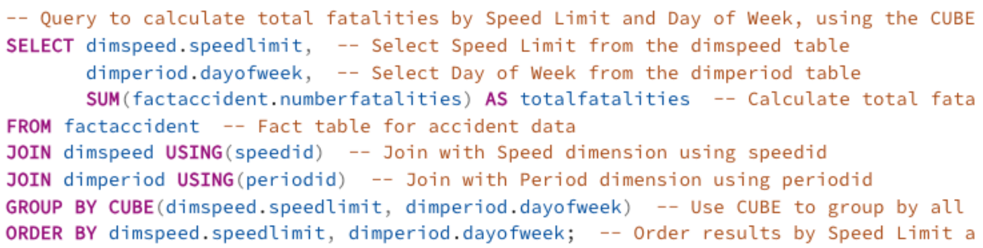


The chart compares crash statistics across various road types, including arterial roads, collector roads, local roads, and national or state highways, displaying both average weight factors (top row) and the number of fatalities (bottom row). The data is organised by crash type, national road type, and brain involvement classifications. National or state highways show **the highest weight factor (1.72)**, while the "Other" categories report a notable total of 25 fatalities across all road types.



Relation Between the Speed and Period

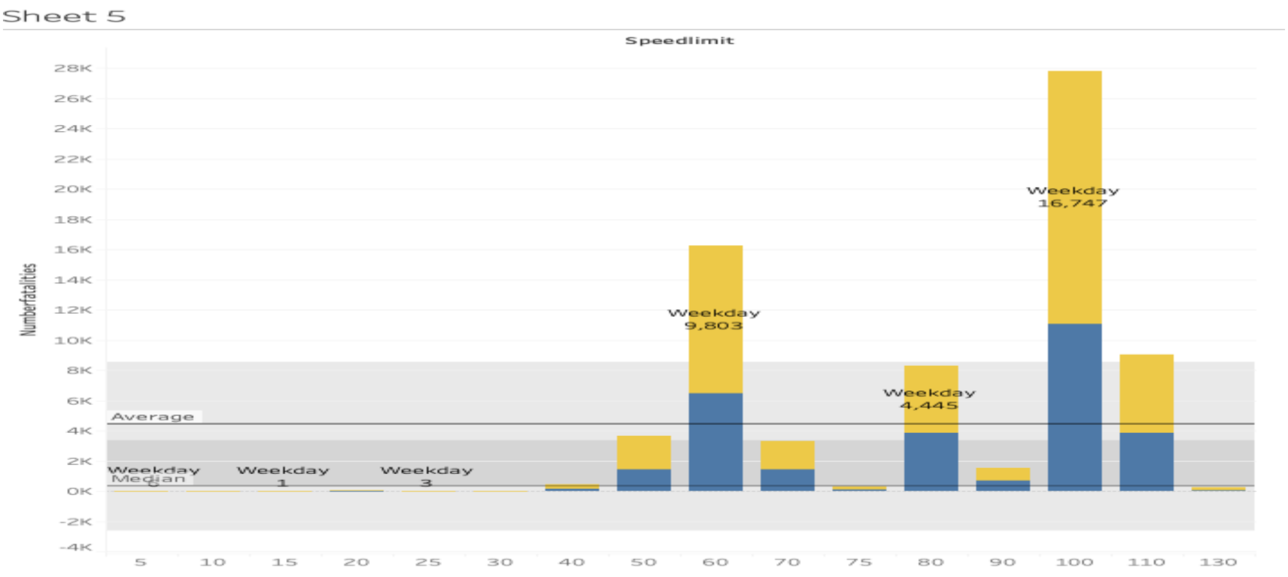
What is the total number of fatalities across different speed limits and days of the week?



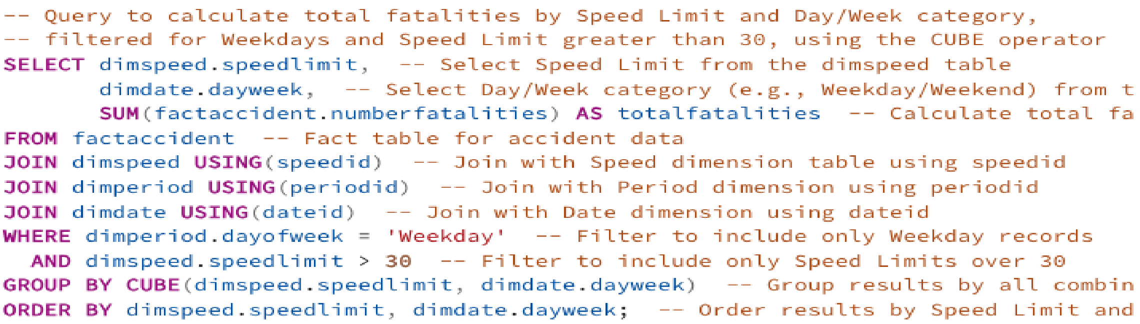
```
-- Query to calculate total fatalities by Speed Limit and Day of Week, using the CUBE
SELECT dimspeed.speedlimit,  -- Select Speed Limit from the dimspeed table
       dimperiod.dayofweek,  -- Select Day of Week from the dimperiod table
       SUM(factaccident.numberfatalities) AS totalfatalities  -- Calculate total fata
FROM factaccident  -- Fact table for accident data
JOIN dimspeed USING(speedid)  -- Join with Speed dimension using speedid
JOIN dimperiod USING(periodid)  -- Join with Period dimension using periodid
GROUP BY CUBE(dimspeed.speedlimit, dimperiod.dayofweek)  -- Use CUBE to group by all
ORDER BY dimspeed.speedlimit, dimperiod.dayofweek;  -- Order results by Speed Limit a
```

The number of fatalities in fatal crashes is significantly **lower at very low speed limits** but begins to increase gradually as the **speed limit exceeds 40 km/h**. There's a noticeable fluctuation in the trend, with the highest number of fatalities occurring at a speed limit of 100 km/h — accounting for nearly 28,000 deaths, most of which happened on weekdays (16,747). Similarly, speed zones of 60 km/h also recorded a high number of fatalities, around 16,000. Interestingly, despite weekends generally having less traffic, a considerable number of crashes and fatalities still occurred on Saturdays and Sundays, highlighting that weekend travel is not necessarily safer.

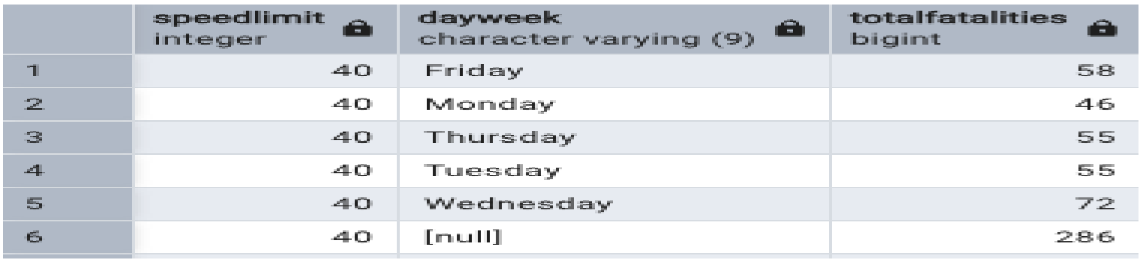
We are now visualising the distribution of fatalities across weekdays to identify which days have higher fatality counts and explore how these patterns correlate with other factors.



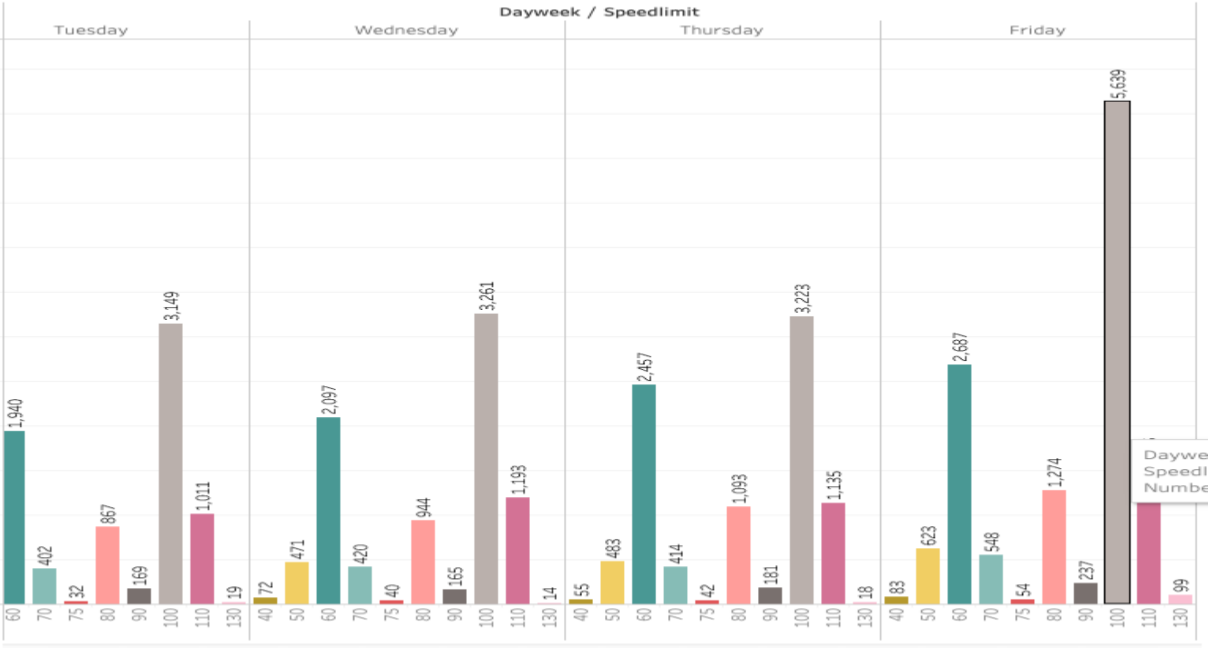
What is the total number of fatalities on weekdays for roads with speed limits greater than 30, based by speed limit and day/week classification?

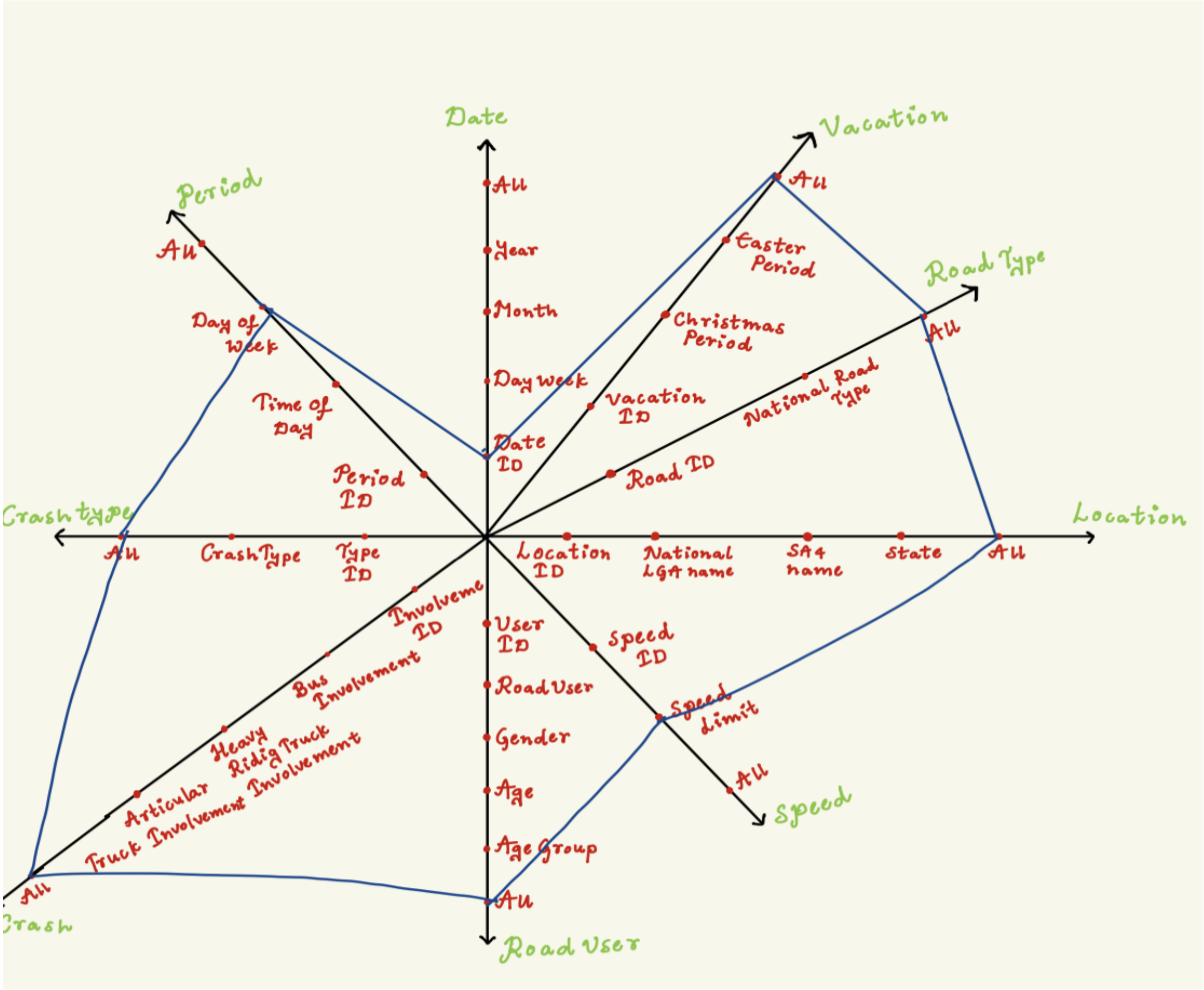


```
-- Query to calculate total fatalities by Speed Limit and Day/Week category,
-- filtered for Weekdays and Speed Limit greater than 30, using the CUBE operator
SELECT dimspeed.speedlimit,  -- Select Speed Limit from the dimspeed table
       dimdate.dayweek,  -- Select Day/Week category (e.g., Weekday/Weekend) from t
       SUM(factaccident.numberfatalities) AS totalfatalities  -- Calculate total fa
FROM factaccident  -- Fact table for accident data
JOIN dimspeed USING(speedid)  -- Join with Speed dimension table using speedid
JOIN dimperiod USING(periodid)  -- Join with Period dimension using periodid
JOIN dimdate USING(dateid)  -- Join with Date dimension using dateid
WHERE dimperiod.dayofweek = 'Weekday'  -- Filter to include only Weekday records
  AND dimspeed.speedlimit > 30  -- Filter to include only Speed Limits over 30
GROUP BY CUBE(dimspeed.speedlimit, dimdate.dayweek)  -- Group results by all combin
ORDER BY dimspeed.speedlimit, dimdate.dayweek;  -- Order results by Speed Limit and
```



	speedlimit integer	dayweek character varying (9)	totalfatalities bigint
1	40	Friday	58
2	40	Monday	46
3	40	Thursday	55
4	40	Tuesday	55
5	40	Wednesday	72
6	40	[null]	286

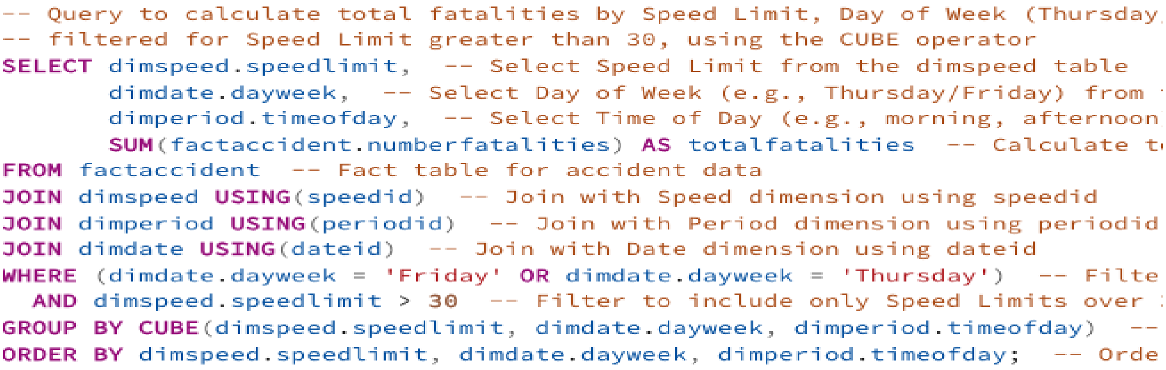




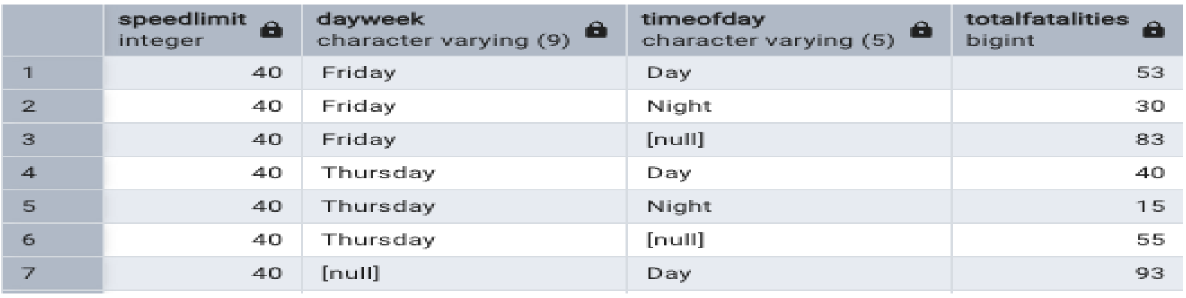
The graph shows that fatalities are highest on Fridays, particularly in areas with a 100 km/h speed limit. A similar trend is observed on other weekdays, with higher speed zones consistently recording more fatalities like Thursday. Lower speed zones, such as 30–60 km/h, have significantly fewer fatal incidents. This suggests a strong correlation between higher speed limits and fatal crashes, especially towards the end of the workweek

Based on this, we can visualize the time period with the highest number of fatalities, which occurs on Thursdays or Fridays.

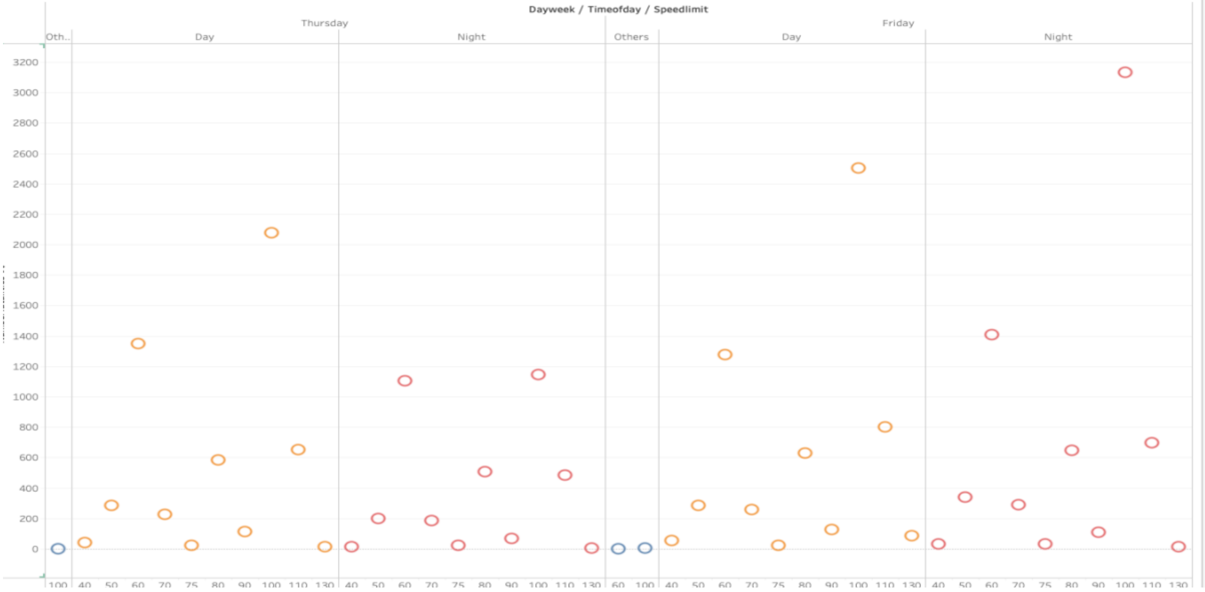
What is the total number of fatalities on Thursday and Friday, for roads with speed limits greater than 30, across by speed limit, day of the week, and time of day

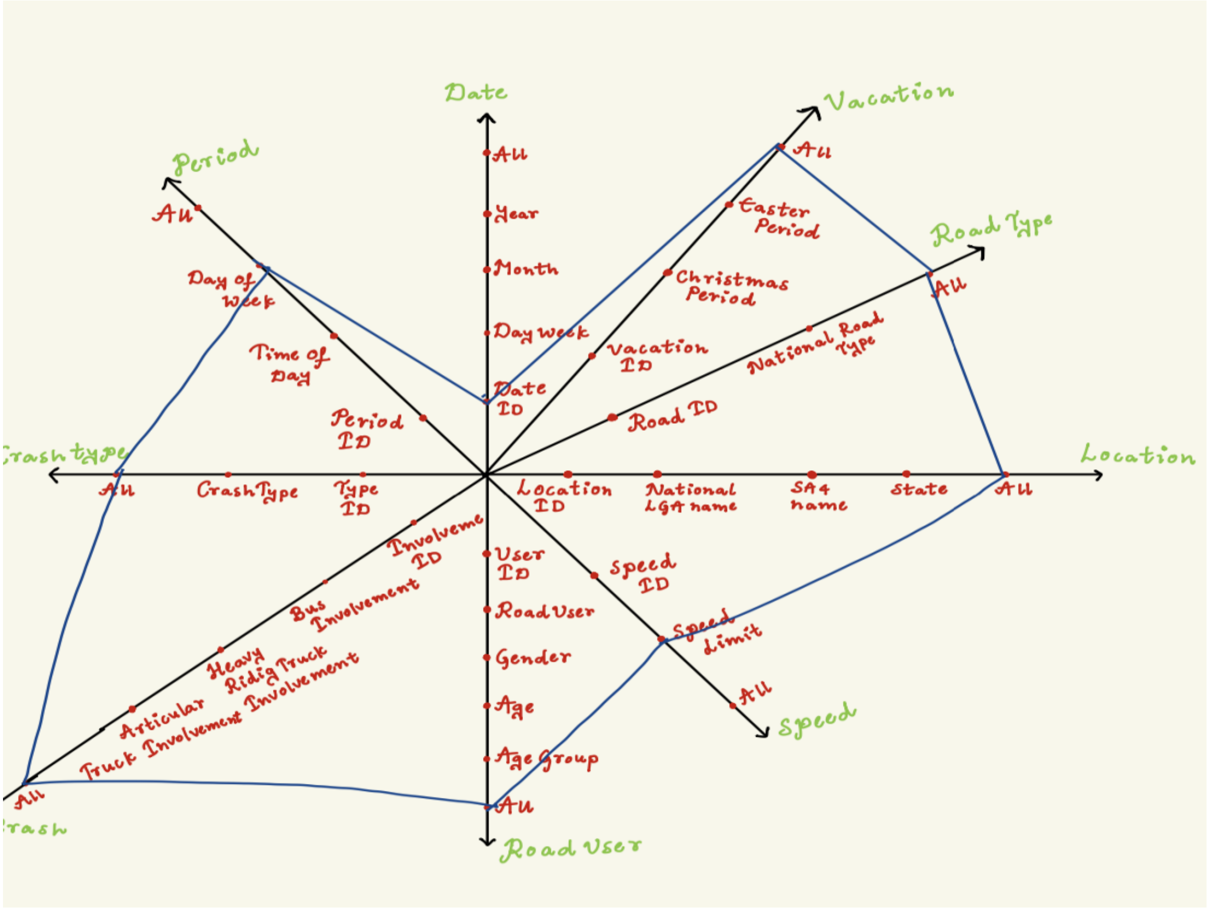


```
-- Query to calculate total fatalities by Speed Limit, Day of Week (Thursday
-- filtered for Speed Limit greater than 30, using the CUBE operator
SELECT dimspeed.speedlimit,   -- Select Speed Limit from the dimspeed table
       dimdate.dayweek,   -- Select Day of Week (e.g., Thursday/Friday) from 
       dimperiod.timeofday,   -- Select Time of Day (e.g., morning, afternoon
       SUM(factaccident.numberfatalities) AS totalfatalities   -- Calculate t
FROM factaccident   -- Fact table for accident data
JOIN dimspeed USING(speedid)   -- Join with Speed dimension using speedid
JOIN dimperiod USING(periodid)   -- Join with Period dimension using periodid
JOIN dimdate USING(dateid)   -- Join with Date dimension using dateid
WHERE (dimdate.dayweek = 'Friday' OR dimdate.dayweek = 'Thursday')   -- Filte
  AND dimspeed.speedlimit > 30   -- Filter to include only Speed Limits over 
GROUP BY CUBE(dimspeed.speedlimit, dimdate.dayweek, dimperiod.timeofday)   --
ORDER BY dimspeed.speedlimit, dimdate.dayweek, dimperiod.timeofday;   -- Orde
```



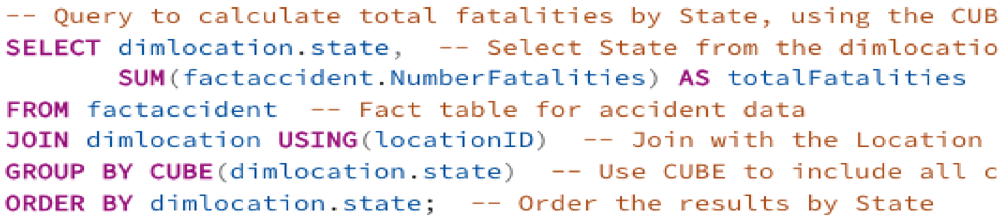
	speedlimit integer	dayweek character varying (9)	timeofday character varying (5)	totalfatalities bigint
1	40	Friday	Day	53
2	40	Friday	Night	30
3	40	Friday	[null]	83
4	40	Thursday	Day	40
5	40	Thursday	Night	15
6	40	Thursday	[null]	55
7	40	[null]	Day	93



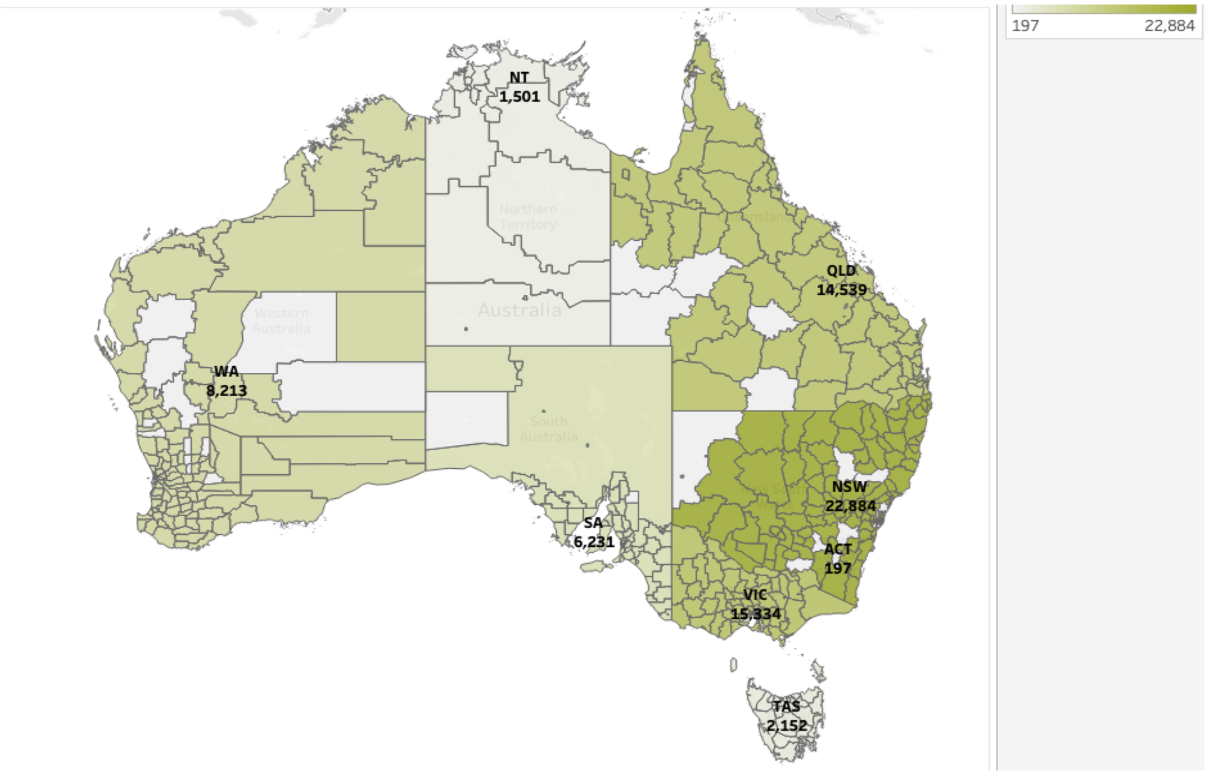


Relation between Location and Fatalities

What is the total number of fatalities for each state, including the overall total across all states?

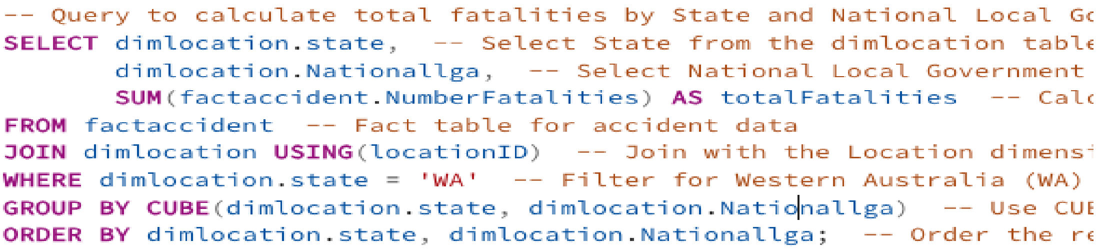


```
-- Query to calculate total fatalities by State, using the CUB
SELECT dimlocation.state,   -- Select State from the dimlocatio
       SUM(factaccident.NumberFatalities) AS totalFatalities
FROM factaccident   -- Fact table for accident data
JOIN dimlocation USING(locationID)   -- Join with the Location
GROUP BY CUBE(dimlocation.state)   -- Use CUBE to include all c
ORDER BY dimlocation.state;   -- Order the results by State
```

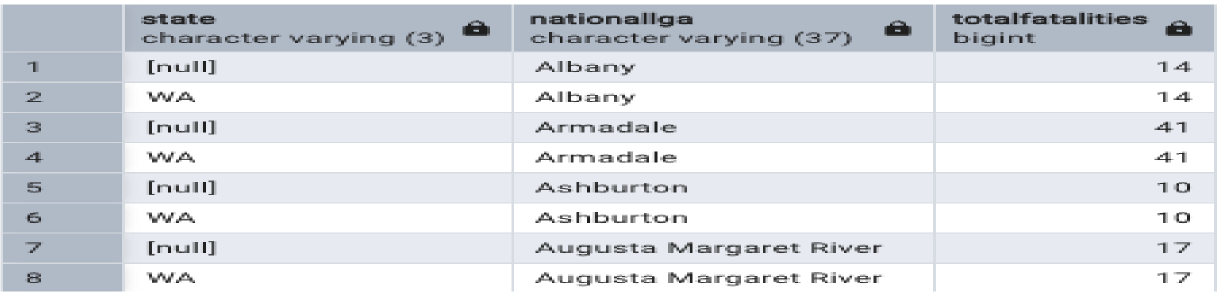


In our visualisation, **New South Wales (NSW)** holds the **highest position with 22,884** fatalities, followed by **Victoria with 15,334 and Queensland (QLD) at 14,539** fatalities. **Western Australia (WA) has significantly fewer fatalities respective to its area** compare, totalling 8,213. **South Australia** records 6,321 fatalities, while **Tasmania** has a much lower number at 215. The **Northern Territory** reports the least number of road accidents across all region

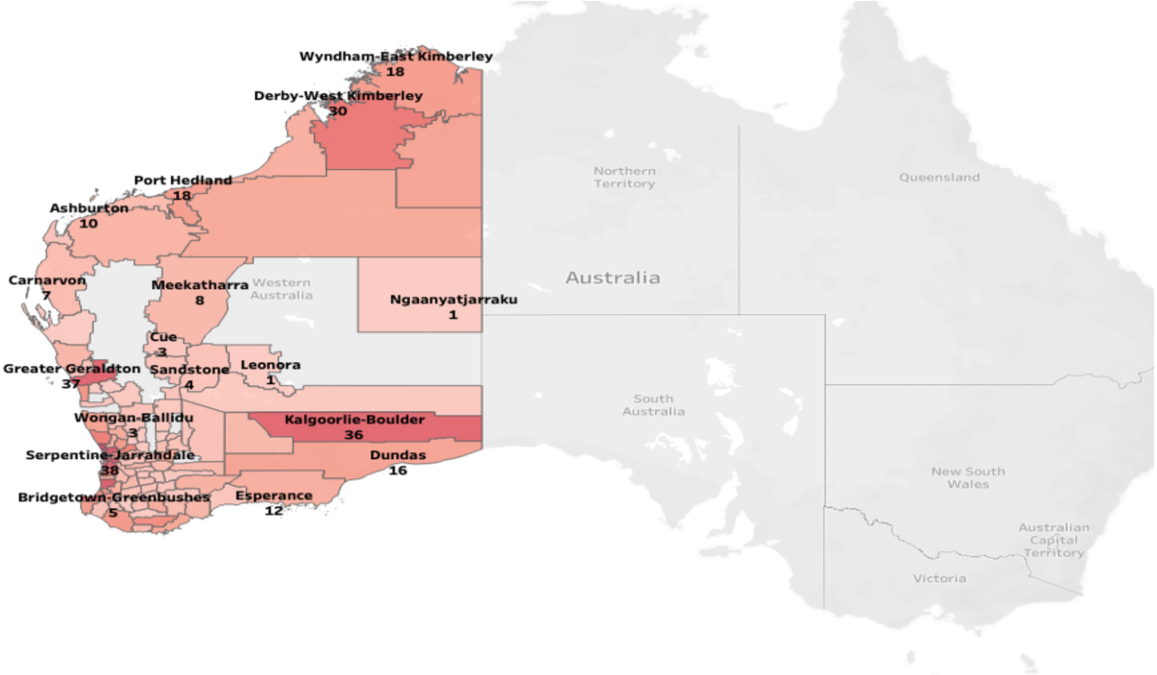
What is the total number of fatalities in Western Australia, based on their national local government area (LGA)?

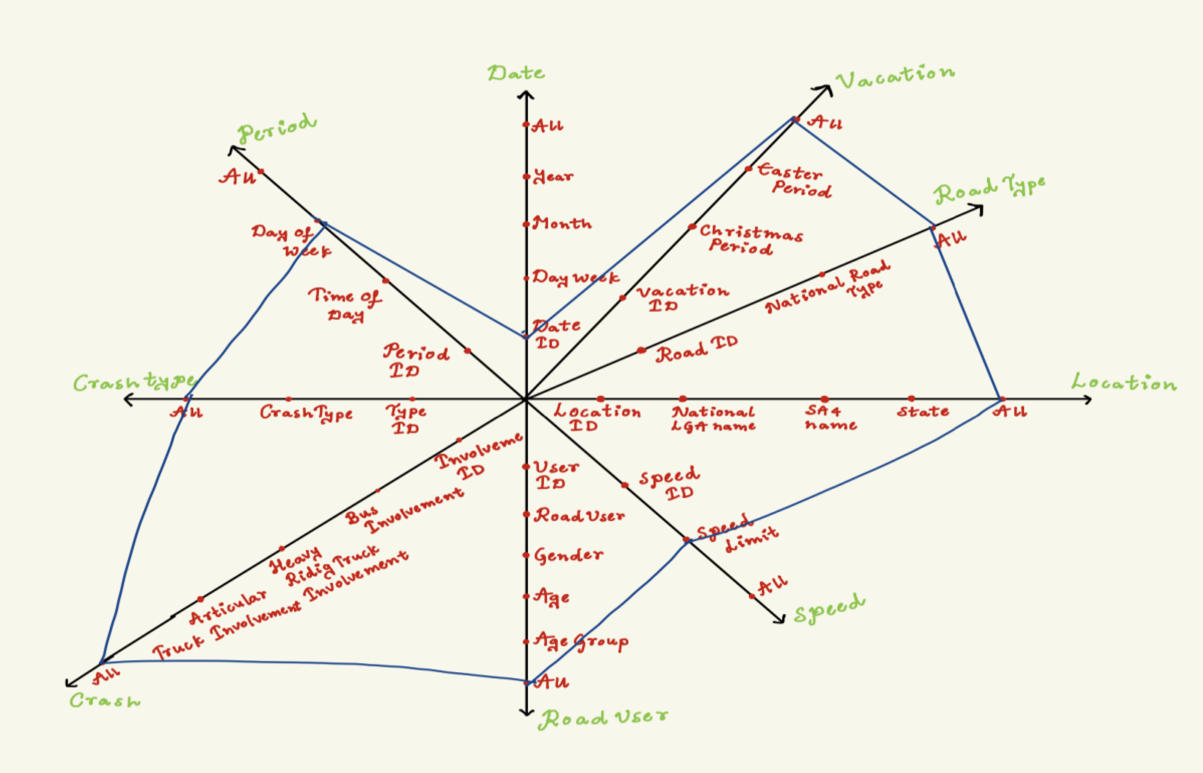


```
-- Query to calculate total fatalities by State and National Local Gc
SELECT dimlocation.state,   -- Select State from the dimlocation table
       dimlocation.Nationallga,   -- Select National Local Government
       SUM(factaccident.NumberFatalities) AS totalFatalities   -- Calc
FROM factaccident   -- Fact table for accident data
JOIN dimlocation USING(locationID)   -- Join with the Location dimensi
WHERE dimlocation.state = 'WA'   -- Filter for Western Australia (WA)
GROUP BY CUBE(dimlocation.state, dimlocation.Nationallga)   -- Use CUE
ORDER BY dimlocation.state, dimlocation.Nationallga;   -- Order the re
```

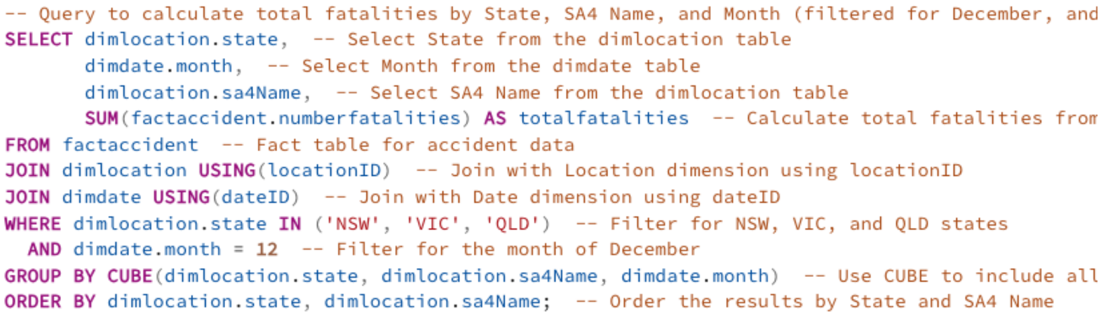


	state character varying (3)	nationallga character varying (37)	totalfatalities bigint
1	[null]	Albany	14
2	WA	Albany	14
3	[null]	Armadale	41
4	WA	Armadale	41
5	[null]	Ashburton	10
6	WA	Ashburton	10
7	[null]	Augusta Margaret River	17
8	WA	Augusta Margaret River	17

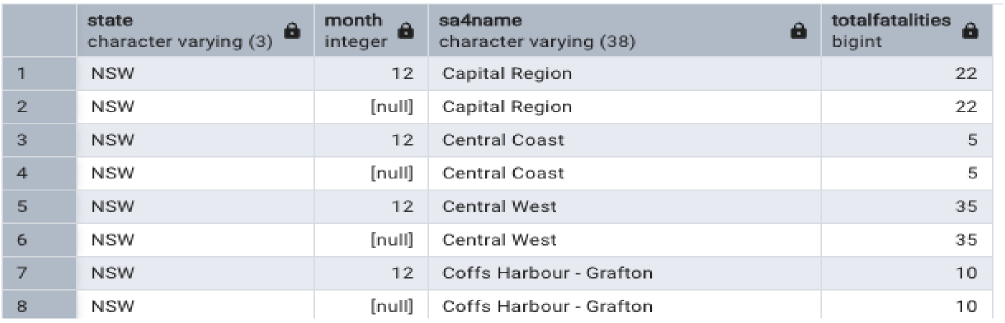




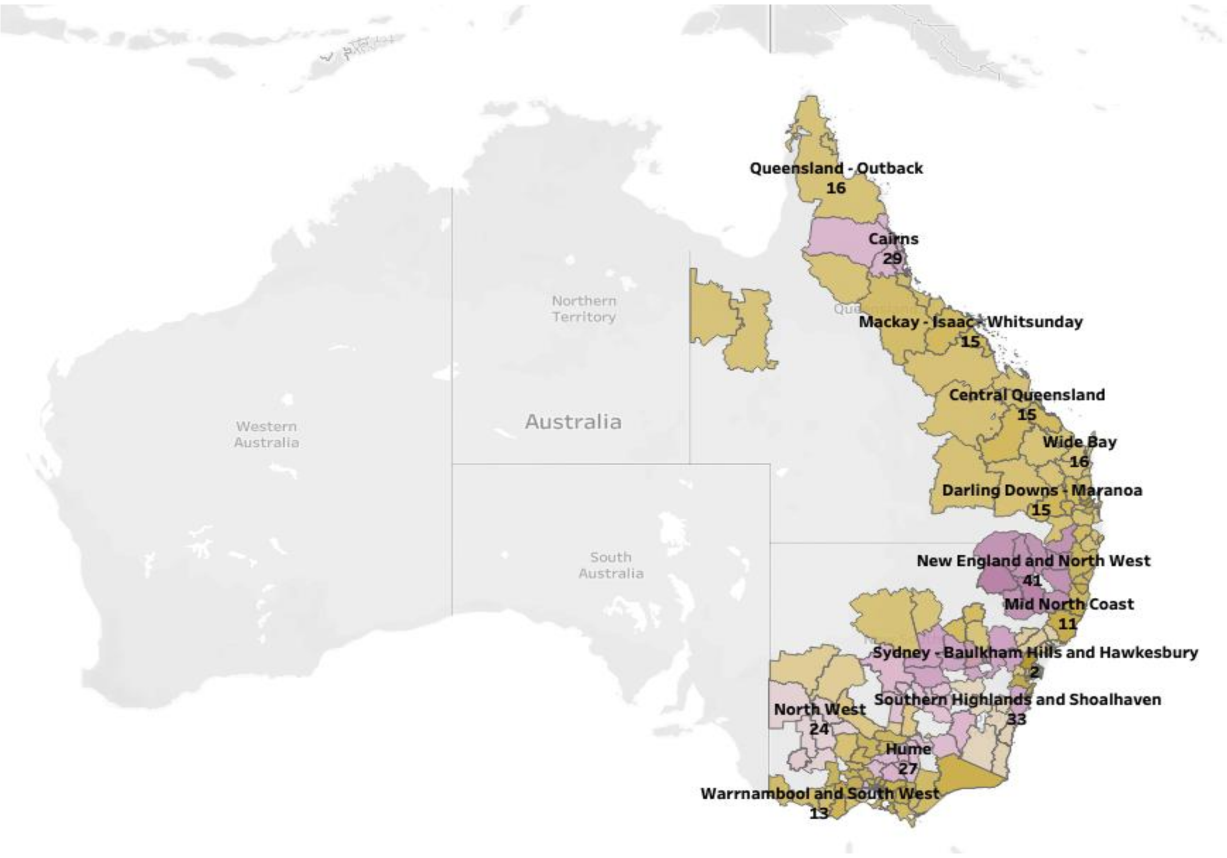
What is the total number of fatalities in December, categorised by state, SA4 name, and month, filtered for NSW, VIC, and QLD?

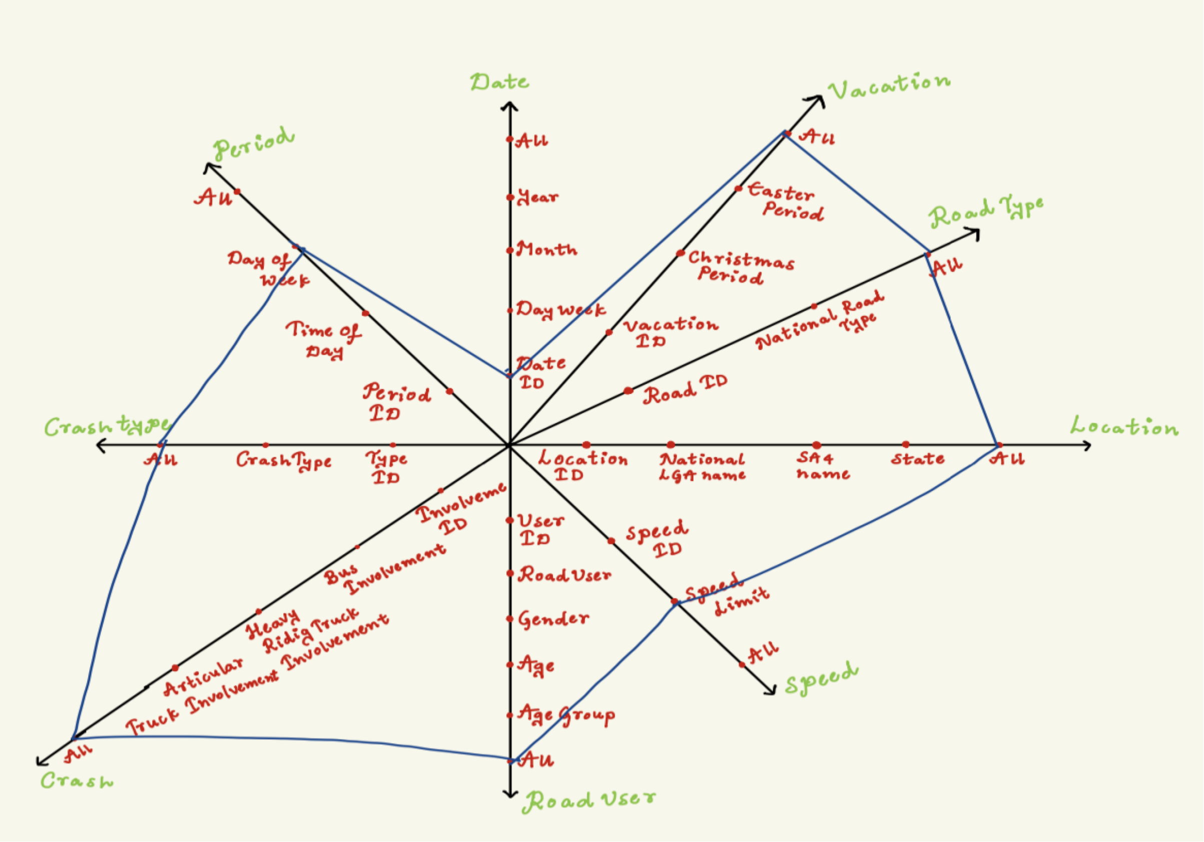


```
-- Query to calculate total fatalities by State, SA4 Name, and Month (filtered for December, and
SELECT dimlocation.state,  -- Select State from the dimlocation table
       dimdate.month,  -- Select Month from the dimdate table
       dimlocation.sa4Name,  -- Select SA4 Name from the dimlocation table
       SUM(factaccident.numberfatalities) AS totalfatalities  -- Calculate total fatalities from
FROM factaccident  -- Fact table for accident data
JOIN dimlocation USING(locationID)  -- Join with Location dimension using locationID
JOIN dimdate USING(dateID)  -- Join with Date dimension using dateID
WHERE dimlocation.state IN ('NSW', 'VIC', 'QLD')  -- Filter for NSW, VIC, and QLD states
  AND dimdate.month = 12  -- Filter for the month of December
GROUP BY CUBE(dimlocation.state, dimlocation.sa4Name, dimdate.month)  -- Use CUBE to include all
ORDER BY dimlocation.state, dimlocation.sa4Name;  -- Order the results by State and SA4 Name
```



	state character varying (3)	month integer	sa4name character varying (38)	totalfatalities bigint
1	NSW	12	Capital Region	22
2	NSW	[null]	Capital Region	22
3	NSW	12	Central Coast	5
4	NSW	[null]	Central Coast	5
5	NSW	12	Central West	35
6	NSW	[null]	Central West	35
7	NSW	12	Coffs Harbour - Grafton	10
8	NSW	[null]	Coffs Harbour - Grafton	10





Association Rules Mining

Association Rule Mining is a technique used to uncover relationships or associations between attributes and their values in a dataset. It focuses on identifying patterns rather than solving classification or prediction problems. This method is widely used to explore attribute associations and analyse their trends.

Key Steps in the Process:

1. Data Preprocessing:
Quantitative attribute values are discretised and mapped to a sequence of consecutive integers starting from 1. This reduces computational complexity. Additionally, attribute values are normalised to ensure consistency.

2. Identification of Strong Association Areas:
The algorithm identifies regions in the dataset where strong associations between antecedents and consequents are likely to exist. By focusing only on these regions, the algorithm reduces the search space, improving efficiency.

3. Decomposition of Strong Association Areas:
Each identified region is broken down into a set of atomic conditions (triplets) involving antecedents or consequents. These conditions are then combined to form a set of candidate rules.

4. Verification of Candidate Rules:
Each candidate rule is evaluated against defined thresholds for quality and interestingness.

Only the rules that meet or exceed these thresholds are retained as valid and meaningful associations. (Ahirwal, D. (2011))

Aprirori Algorithm

The **Apriori algorithm** is one of the most common associated rule mining algorithmn. It operates by identifying frequent itemsets within a dataset and then uses these itemsets to generate association rules in the form of *"If item X is present, then item Y is also likely to be present."* The algorithm follows a **bottom-up approach**, starting with individual items and progressively combining them into more complex itemsets. (Ali, 2023)

The Apriori algorithm was used for association rule discovery to identify patterns in the dataset, highlighting correlations between crash-related features such as Crash Type, Speed Limit, Gender-Age Group, and Time of Day with Road User. The strength of each generated rule was evaluated using support and confidence metrics. The Strength of Each generated rule was measured using support and confidence

a) Mininum support = Support (0.01)
b) Mininum Confidence = Confidence (0.07)

TOP K RULES :

Top 7 RULES,
Rule 334

Antecedent	Consequent	Confidence	Support	Lift
(single, 75_or_older,60)	*Pedestrian*	0.8321	0.01942	5.39983

If: A single vehicle involved, 75 or older person, with speed limit of 60; **Then** Pedestrain;

Confidence : 83.21% high likelihood

Lift : 5.39 Strong Correlation

This rule suggests that pedestrian-involved crashes were involved by single-vehicle accidents involving were aged 75 or older, particularly when the vehicle was traveling at a speed of 60 km/h.

Rule 516

Antecedent	Consequent	Confidence	Support	Lift
(Male, Single, 75_or_older,60)	*Pedestrian*	0.8274	0.0118	5.3689

If: A Male Person, Single vehicle involved Crash, 75 or older person, with a speed limit of vehicle 60; **Then** Pedestrain

Confidence: 82.7% very high likelihood

Lift: 5.36 Strong Correlation

The rule illustrates that males aged 75 or older are more likely to be involved in pedestrian crashes caused by a single vehicle traveling at a speed of 60 km/h.

Rule 513

Antecedent	Consequent	Confidence	Support	Lift
(Single, Day,60,75_or_older)	*Pedestrain*	0.81212	0.014582	5.2697

IF: A single vehicle involved Crash, During the Day , with a speed limit of vehicle 60 and the fatality age is 75 or older **Then** Pedestrain

Confidence: 81.21% very high likelihood

Lift: 5.26 Strong Correlation

This rule implies that crashes involving a single vehicle traveling at 60 km/h during the daytime are more likely to involve pedestrians aged 75 or older

Rule 328

Antecedent	Consequent	Confidence	Support	Lift
(65_to_74,Single, 60)	*Pedestrain*	0.79477	0.01102	5.15

IF: A 65 to 75 years old, A single vehicle Involve Crash, with a speed limit of vehicle 60 **Then** Pedestrain

Confidence: 79.4% high Likelihood

Lift: 5.15 high Correlation

This rule shows that crashes involving a single vehicle traveling at 60 km/h are more likely to result in fatalities for pedestrians.

Rule 154

Antecedent	Consequent	Confidence	Support	Lift
(0_to_16, 100,Day)	*Passenger*	0.7623	0.0125	3.429

IF: A 0 to 16 years old, with a vehicle speed limit of 100 and During Day **Then** Passenger

Confidence: 76.23% high Likelihood

Lift: 3.429 Moderate Correlation

This rule indicates that accidents involving vehicles traveling at 100 km/h during the day, fatalities with individuals aged 0 to 16, most commonly involved passengers

Rule 20

Antecedent	Consequent	Confidence	Support	Lift
(0_to_16,100)	*Passenger*	0.7612	0.019	3.4241

IF: A 0 to 16 years old, with a vehicle speed limit of 100 **Then** Passengers

Confidence: 76.1% high Likelihood

Lift: 3.42 Moderate Correlation

This rule suggests that in accidents where vehicles were traveling at a speed of 100 km/h, fatalities among individuals aged 0 to 16 were most likely passengers.

Rule 185

Antecedent	Consequent	Confidence	Support	Lift
(100, Male, 75_or_older)	*Driver*	0.7588	0.010	1.667

IF: A vehicle of 100 speed limit, Male Person, 75 or older **Then** Driver

Confidence: 75.8% high Likelihood

Lift: 1.667 low Correlation

This rule indicates that in crashes where the vehicle speed was around 100 km/h, male fatalities aged 75 or older were most likely to be drivers.

Insights and Recommendations

Association rules 334, 516, and 513 highlight a strong connection between these factors, showing that pedestrian fatalities are much more likely when the crash involves a single vehicle, **the victim is 75 or older, and the speed is around 60 km/h**. The risk increases further when the victim is male, **with Rule 516** showing a **confidence of 82.7%.** The lift **values, all around 5.3**, indicate that these conditions make pedestrian fatalities over five times more likely than random chance, underscoring the vulnerability of older pedestrians in such scenarios. **Elderly individuals aged 75 and above are at significantly higher risk of fatal pedestrian accidents**, particularly in single-vehicle crashes at moderate speeds like 60 km/h. Due to **age-related factors** such as reduced reaction time and difficulty in navigating roads, even relatively safe driving speeds can prove deadly.

Children aged 0–16 are most likely to be passengers in high-speed crashes, particularly on **roads with speed limits around 100 km/h** and during the daytime. **Rules 154 and 20 support this with high confidence (~76%) and moderate lift (~3.4)**, indicating a strong association. These patterns often reflect settings like school runs, family travel, or holiday trips, where children are typically passengers rather than drivers or pedestrians. Factors such as rushing to school, **overspeeding**, and inconsistent seatbelt or child restraint use can increase risks. Additionally, due to their **young age and physical vulnerability**, children are more susceptible to severe outcomes in accidents, making safety measures and awareness critically important.

Recommendations for Government to Reduce Road Fatalities:

1. **Improve and Maintain Pedestrian Infrastructure**
    - Many pedestrian accidents occur in areas where vehicles and pedestrians share the same space.
    - Expanding sidewalks, adding pedestrian-only paths, and clearly separating walking zones from vehicle lanes can significantly reduce these risks.
2. **Increase Traffic Signals and Road Signage, Especially During Daytime**
    - A higher number of traffic lights, pedestrian crossings, and warning signs—particularly in high-traffic daytime areas—can help both drivers and pedestrians stay alert and make safer decisions.
    - Daytime enforcement is essential, as data suggests many fatal crashes occur during this period.
3. **Implement Routine Alcohol and Drug Testing During the Day**
    - Random sobriety checks shouldn't be limited to night-time.
    - Conducting these checks during the day can deter impaired driving, especially during school hours and work commutes.
4. **Expand School Zones and Reduce Speed Limits in Sensitive Areas**
    - Create more designated school zones and ensure that speed limits are strictly enforced around them.
    - Use flashing lights, road markings, and crossing guards to enhance child safety during school start and end times.
5. **Mandate and Enforce Seat Belt Usage for All Passengers**
    - Seat belts significantly reduce the risk of injury and death during crashes.
    - Strong enforcement and public awareness campaigns can ensure higher compliance, especially for rear-seat passengers and children.
6. **Enhance Road Quality and Maintenance**
    - Poorly maintained roads with potholes, faded lane markings, or uneven surfaces can contribute to crashes.
    - Regular maintenance and investment in safer road designs (e.g., roundabouts, clear signage, anti-skid surfaces) are critical to improving safety.

LIMITATIONS

- Nine dimensions were considered due to the separation of the date dimension into a period dimension, leading to higher storage usage when creating the dimensional tables.
- Unknown and invalid data in the age and road user dimensions were removed, as the primary analysis focuses on these attributes.
- Invalid data was filtered out to ensure accurate results.
- Top K rules were prioritized based on confidence and lift values.

REFERENCES

1. **Australian Government. (n.d.). *National Road Safety*. https://www.roadsafety.gov.au/**
2. **Road Sense Australia. (n.d.). *Understanding road statistics in Australia*. https://roadsense.org.au/understanding-road-statistics-aus/?gad_source=1&gclid=CjwKCAjwwLO_BhB2EiwAx2e-33BJn9C8QPCp2bEcd8j2wZB5-jm7fCNXoXuRcMuyKgdEFgMeCG_c4hoC92IQAvD_BwE**
3. **Bigham, B. (2014). Road accident data analysis: A data mining approach. *Indian Journal of Scientific Research, 3*(1), 437–443.**
4. **Kimberly Merritt. (2008). USER SATISFACTION IN DATAWAREHOUSING: AN EMPIRICAL INVESTIGATION OF SALIENT VARIABLES. *Issues in Information Systems*, *9*(2), 500–508. https://doi.org/10.48009/2_iis_2008_500-508**
5. **European Commission. (2024, March 8). *2023 figures show stalling progress in reducing road fatalities in too many countries*. European Commission. https://transport.ec.europa.eu/news-events/news/2023-figures-show-stalling-progress-reducing-road-fatalities-too-many-countries-2024-03-08_en**
6. **Bureau of Infrastructure and Transport Research Economics. (n.d.). *Fatal road crash database*. Australian Government. https://www.bitre.gov.au/statistics/safety/fatal_road_crash_database**
7. Bansal, S. K., & Kagemann, S. (2015). Integrating Big Data: A Semantic Extract-Transform-Load Framework. *Computer (Long Beach, Calif.)*, *48*(3), 42–50. https://doi.org/10.1109/MC.2015.76
8. Ilyas, I. F., & Chu, X. (2019). *Data Cleaning* (First edition.). Association for Computing Machinery.
9. Amin, M. M., Sutrisman, A., & Dwitayanti, Y. (2021). Development of Star-Schema Model for Lecturer Performance in Research Activities. *International Journal of Advanced Computer Science & Applications*, *12*(9), 74–80. https://doi.org/10.14569/IJACSA.2021.0120909
10. Kimball, R., & Ross, M. (2013). *The data warehouse toolkit : the definitive guide to dimensional modeling* (Third edition). Wiley.
11. Ahirwal, D. (2011). Efficient Data Mining Technique Using Associate Rule. International Journal of Advanced Research in Computer Science, 2(1).
12. Ali, M. (2023, January). *Association Rule Mining in Python Tutorial.* datacamp.
13. Used CHATGPT 3.5 for Understanding of Dimension Tables and Factables, Kimbells Analysis, Visualisation and Association Rule Mining and commenting python, sql codes