
THE UNIVERSITY OF WESTERN AUSTRALIA

# CITS5504 - Data Warehousing Project 1 Report

**Student Name: Yulin Yu**

**Student Number: 22743739**

## Table of Contents


- ☐ *Client Identifying* ..... *3*
- ☐ *StarNet design* ..... *4*
- ☐ *Fact table & Dimension tables Design and Dimension tables' hierarchies* *13*
- ☐ *ETL Process* ..... *20*
- ☐ *Multi-dimensional cubes* ..... *26*
- ☐ *Query result visualisation by using atoti* ..... *30*
- ☐ *Association Rule Mining* ..... *40*
- ☐ *Do you agree or disagree with "data cube is an outdated technology"* ... *48*
- ☐ *Reference List* ..... *50*

## ♦ Client Identifying

After reviewing the data set of the project, I have identified two clients as follows:

- Chinese Olympic Games Data Analysis Group: focuses on analysing various aspects of China’s performance and data in the Olympics.
- Australian Olympic Data Analysis Group - focuses on analysing various aspects of Australia’s performance and data in the Olympics.

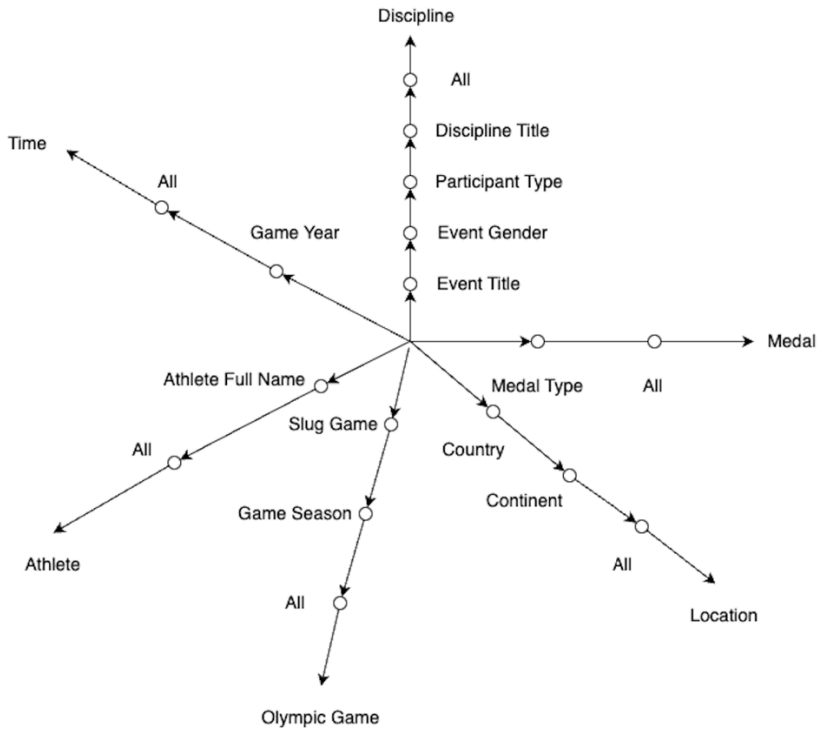
One of the key reasons for me to choose China Olympic Games Data Analysis Group as the first client is that I come from China. Compared with the data of other countries, I have a clearer understanding of China's performance in the Olympic Games, which will help me complete the subsequent work on the project. The second reason is that in recent years, as a developing country with a large population, China's performance in the Olympic Games is relatively outstanding, from a weak country in sports to the current sports power. Therefore, I think it is important and worthwhile to analyse the data of China in various aspects of the Olympic Games.

The reason I chose the Australian Olympic Data Analysis Group as the second client is that I am currently studying in Australia. Therefore, I hope that by analysing the data of Australia, I can learn more about this country. The second reason is that although Australia also shows outstanding performance in the Olympics, it is very different from China in many aspects, such as population size (Australia's population size is much smaller than China's) and country type (Australia is a developed country). Therefore, I think it is also important and of substantial value to analyse the data of Australia in various aspects of the Olympic Games.

## ♦ StarNet design

My data warehouse will import data from two databases. The first database stores data about the performance of various countries in the Olympic Games, and the second database stores data on the mental health of each Olympic host country. Next, I draw two StarNets to determine the dimensions and the conceptual hierarchies of each dimension. All dimensions start at “All” level, which contains the most comprehensive set of data for each dimension. From this top level "All," it is gradually refined to more specific levels to facilitate detailed data analysis.

- **First StarNet for first database:**

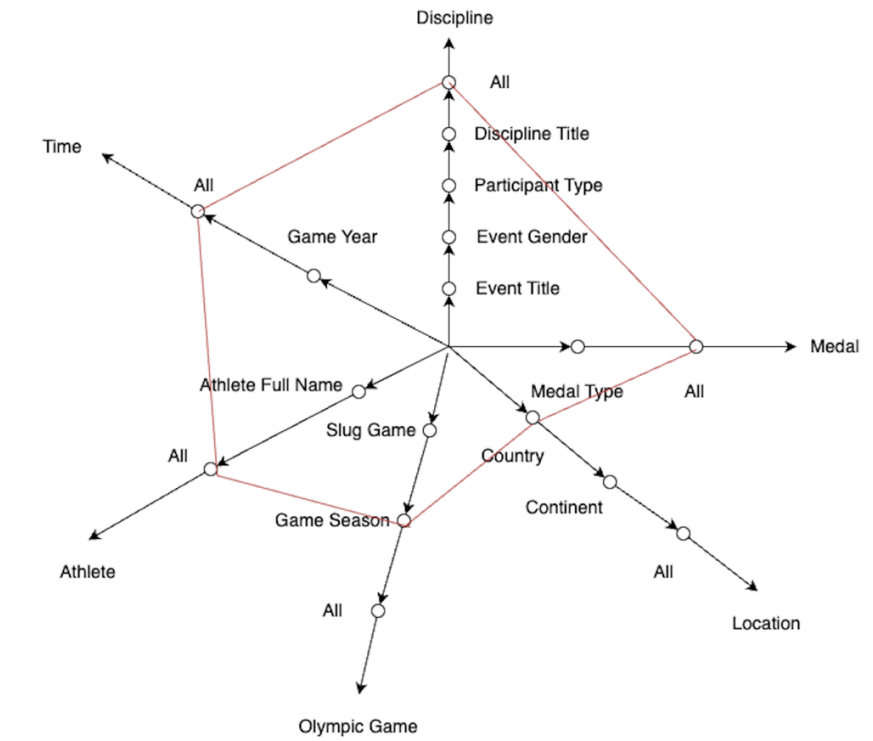



- The conceptual hierarchy of “Time Dimension” are refined from "All" to the specific year in which each Olympic Games was held. This allows clients to analyse from an overall time span and then drill down to a detailed observation and analysis of events in a particular year.
- The conceptual hierarchy of Athlete dimension are refined from "All" to the name of each athlete. This allows clients to start with an overall athlete's perspective and then drill down to individual athlete details, such as participation records and MEDALS won.
- The conceptual hierarchy of the “Olympic Game Dimension” are detailed from "All" to the season in which each Olympic Games was held, and then further to the name of each Olympic Games. This enables clients to categorise Olympic data by season, such as differentiating between Summer and Winter Olympics, and further query and analyse by specific Olympics.

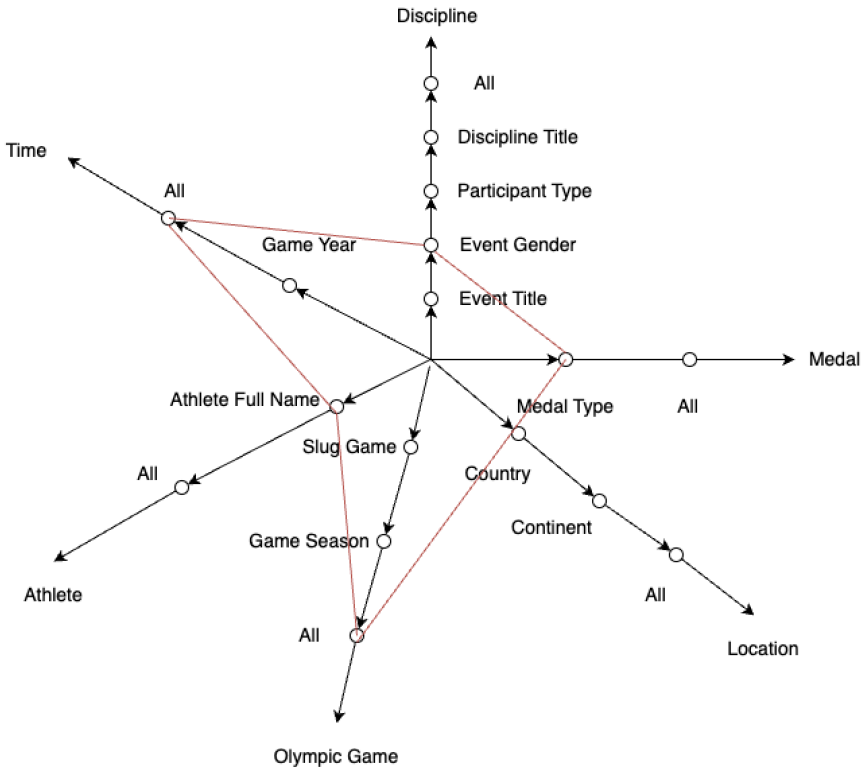
- The conceptual hierarchy of “Discipline Dimension” starts with "All" and is refined various types of disciplines, and further down to the types of participants, the gender classification of events, and the specific name of the event. Such a hierarchy supports in-depth analysis from overall discipline categories to gender-specific or specific event analysis.
- The conceptual hierarchy of “Medal dimensions” are refined from "All" level to specific medal types such as gold, silver, and bronze. This allows clients to see the overall medal distribution and drill down to specific types of medal data.
- The conceptual hierarchy of “Location Dimension” are refined from "All" to individual continents, and then to specific countries. This helps clients analyse the performance of different countries in the Olympic Games.

I list below six examples of business queries that can be answered based on "First StarNet" above (These business query examples are designed for the two clients I mentioned at the beginning of the report). The red path in each StarNet connects to the corresponding StarNet footprint, demonstrating how to answer that business query:

⇒ **Query 1:** What is the trend of the number of medals won by China in the Summer Olympic Games over the various years available in the dataset?

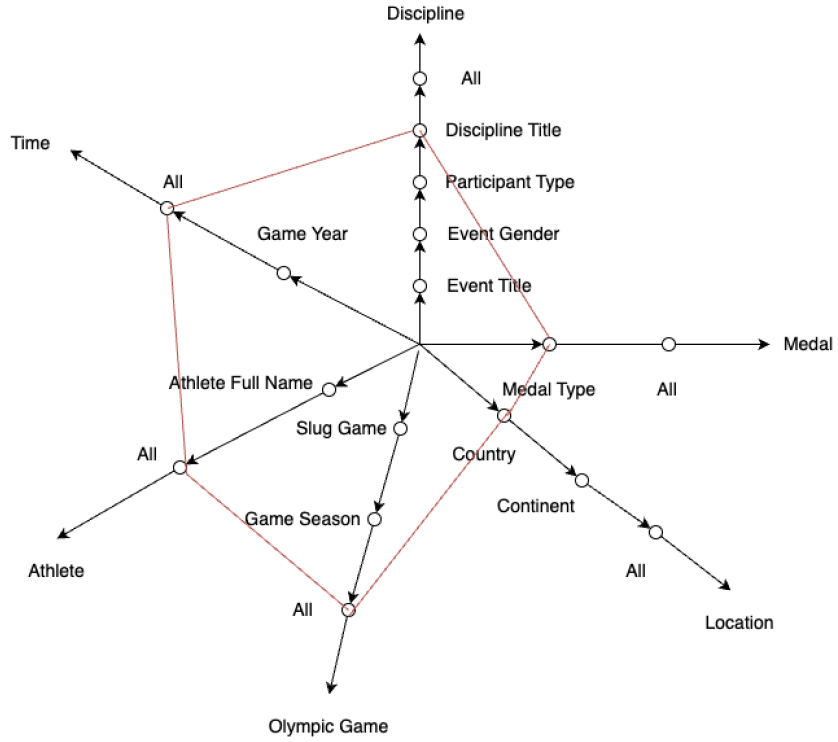



⇒ **Query 2:** Who is the Chinese individual female athlete who has won the most gold medals in shooting discipline?

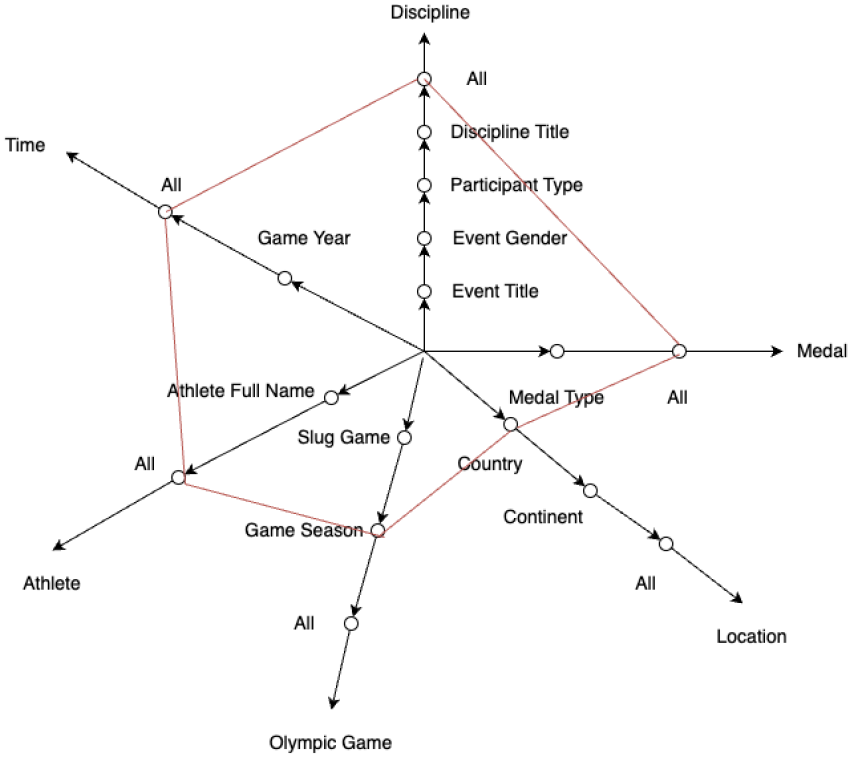



(I selected "Athlete Full Name" instead of "All" in the Athlete dimension of this query is because that in answering this query, I excluded data from Athlete dimension where the athlete's name is "Unknown". I'll explain why I chose to keep these data marked as "Unknown" later in the ETL process.)

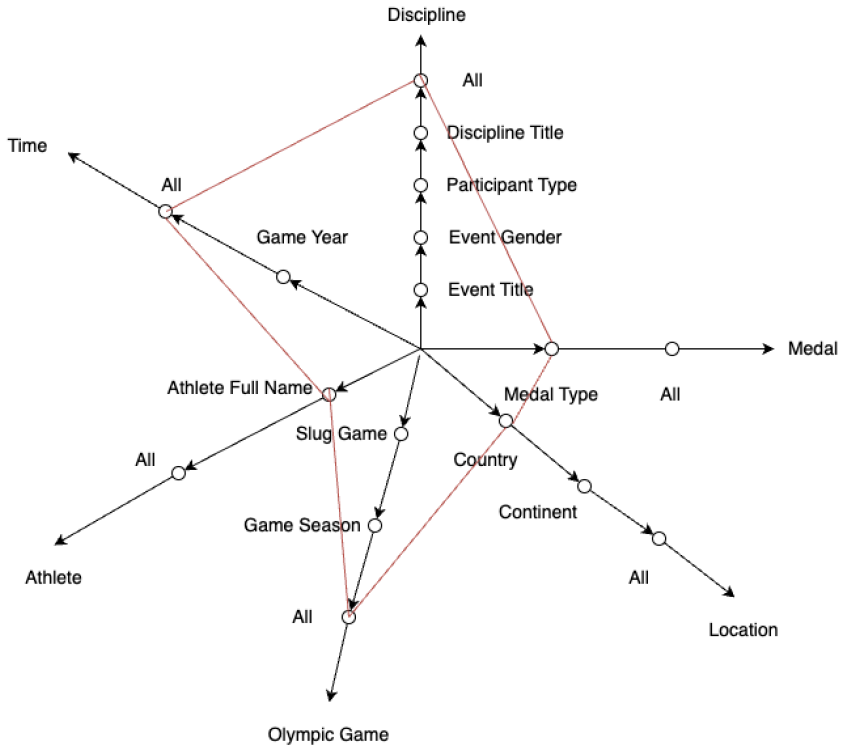
⇒ **Query 3:** What is the discipline category in which China has won the most gold medals in Olympic history?




⇒ **Query 4**: What is the trend of the number of medals won by Australia in the Winter Olympic Games over the various years available in the dataset?

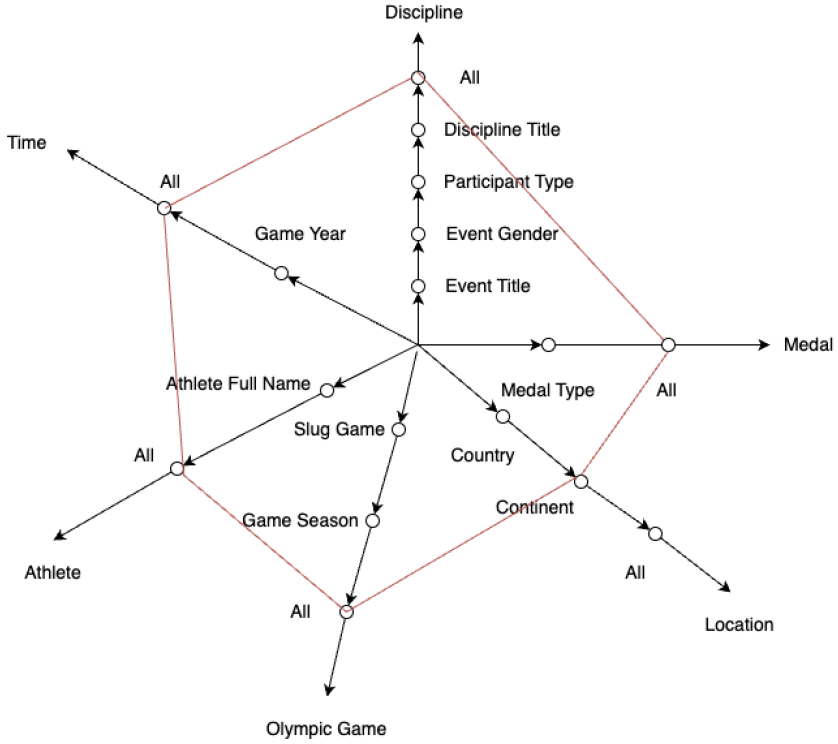



⇒ **Query 5:** Who are the Australian athletes who have won the most gold, silver medals respectively?

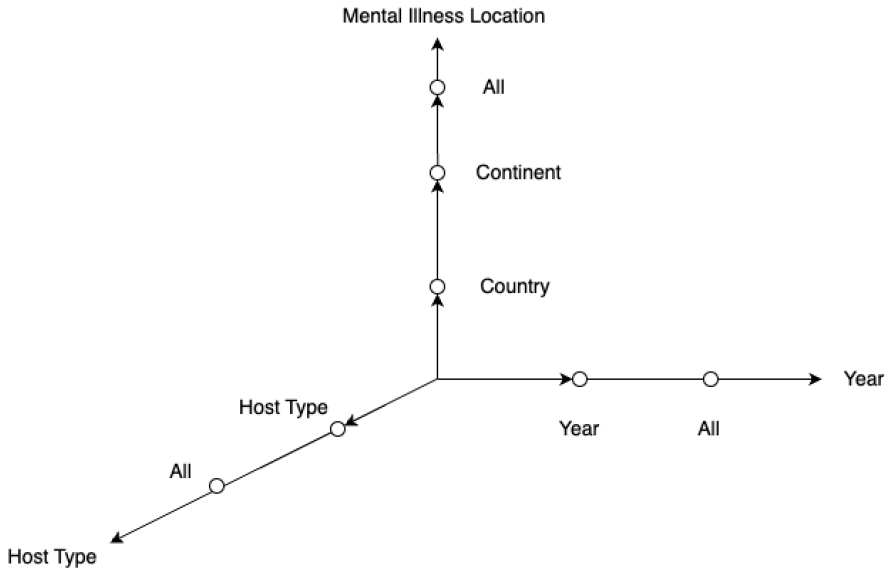



(I selected "Athlete Full Name" instead of "All" in the Athlete dimension of this query is because that in answering this query, I excluded data from Athlete dimension where the athlete's name is "Unknown". I'll explain why I chose to keep these data marked as "Unknown" later in the ETL process.)

⇒ **Query 6**: How many medals have been won by the continent that includes Australia? Is Australia the country with the most medals on that continent?




- **Second StarNet for second database**:

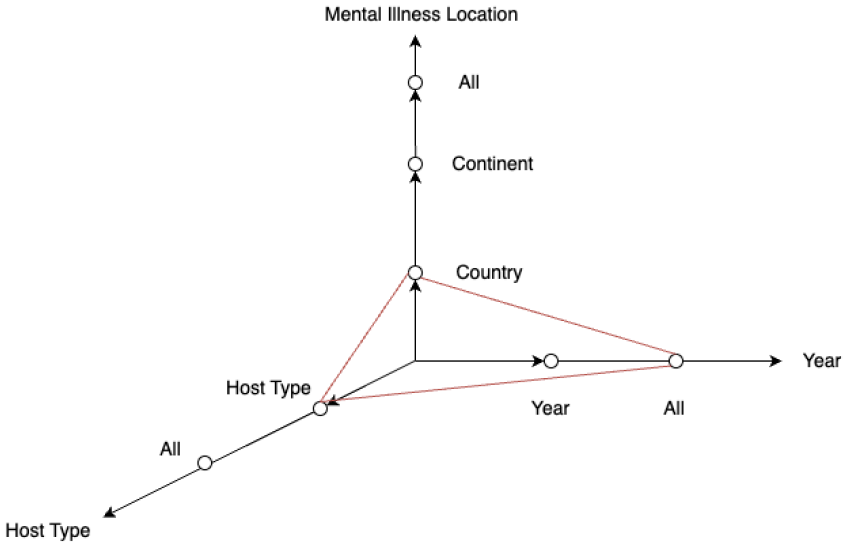



- The conceptual hierarchy of “Year Dimension” is refined from "All" to specific years. This allows clients to analyse from an overall time span and then drill down to a specific years, enabling the analysis of disability-adjusted life years (DALYs) data for mental illness for each year.
- The conceptual hierarchy of “Location Dimension” is refined from "All" to "Continent", then further to individual Olympic host countries. This allows clients to perform data analysis by continent and then further look at data for specific host countries, such as disability-adjusted life years for mental illness for China or Australia.

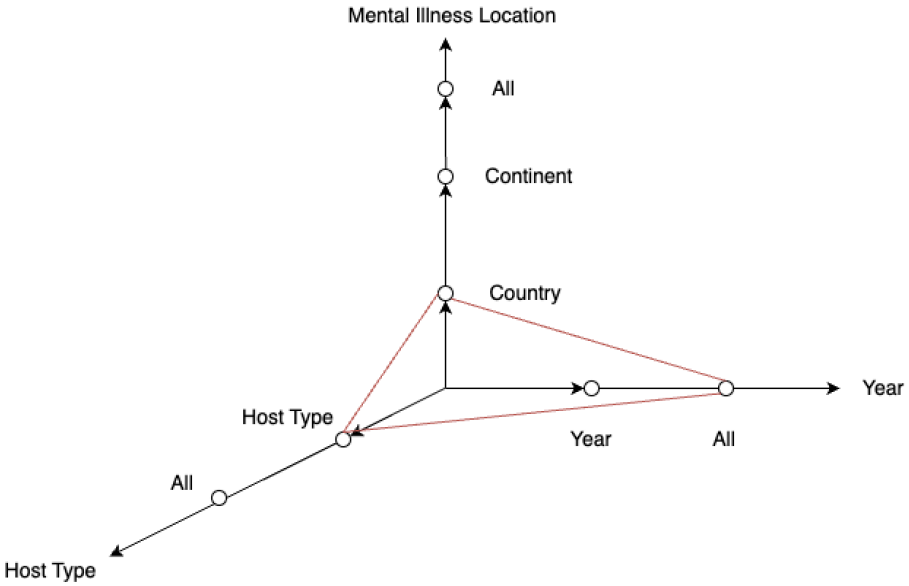
- The conceptual hierarchy of the “Host Type Dimension” is refined from "All" to the specific "host type," which allows clients to analyse and compare disability-adjusted life years data for mental illness when a country as a host versus a non-host country.

  I list below four examples of business queries that can be answered based on "Second StarNet" above (These business query examples are designed for the two clients I mentioned at the beginning of the report). The red path in each StarNet connects to the corresponding StarNet footprint, demonstrating how to answer that business query:

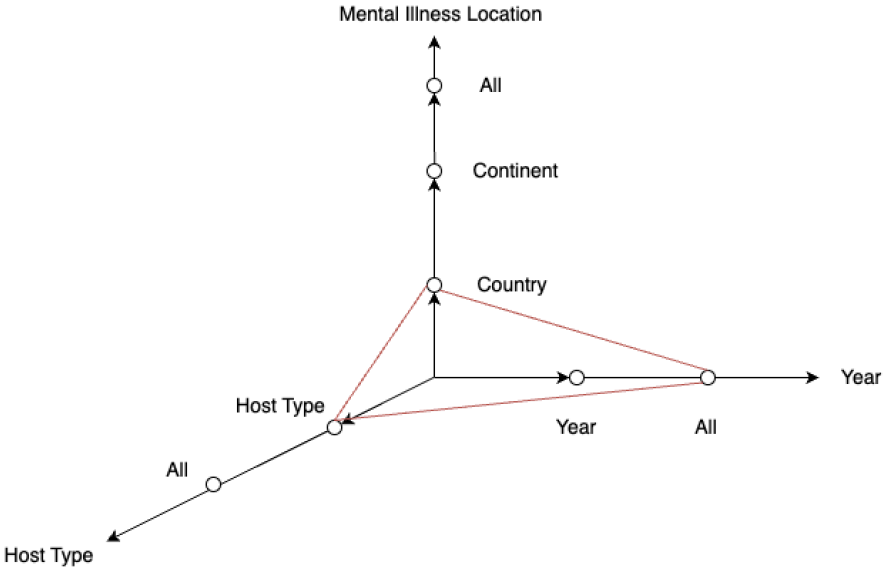
  ⇒ **Query 7**: What is the difference in disability-adjusted life years (DALYs) due to depression disorders in Australia when it is hosting the Olympics compared to when it is not?



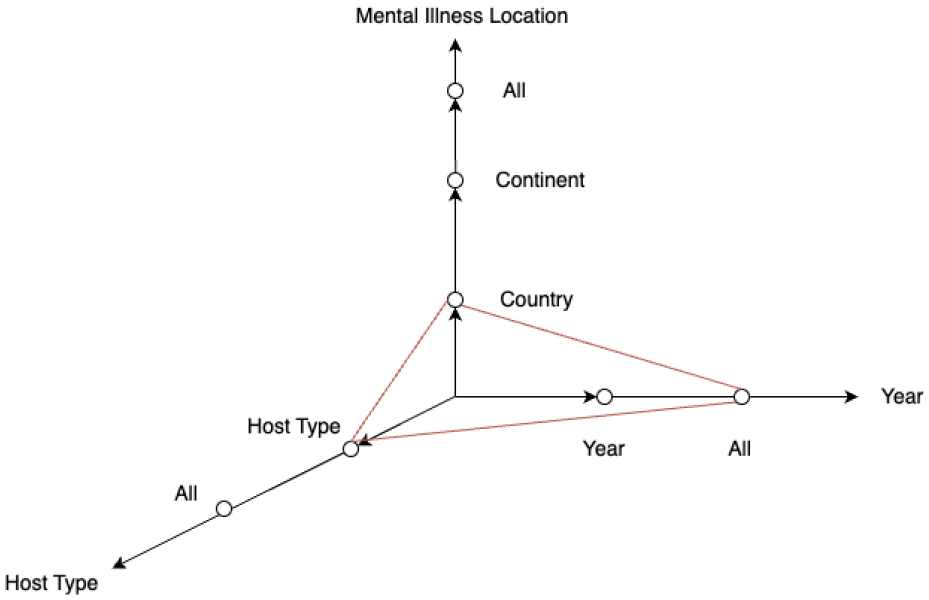

  ⇒ **Query 8**: Which mental disorder results in the lowest disability-adjusted life years (DALYs) in Australia when it is hosting the Olympics?




⇒ **Query 9:** What is the difference in disability-adjusted life years (DALYs) due to depression disorders and eating disorders in China when it is hosting the Olympics?

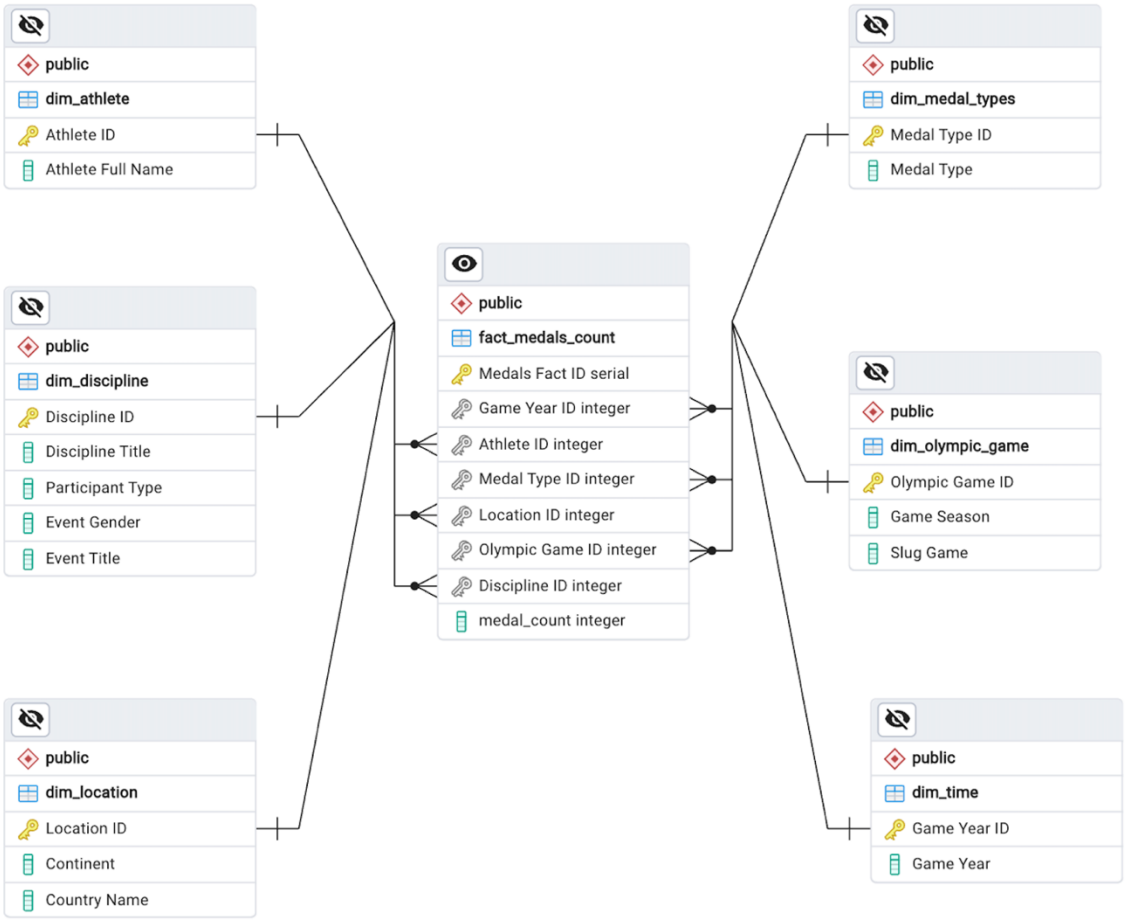



⇒ **Query 10**: Which mental disorder results in the highest disability-adjusted life years (DALYs) in China when it is not hosting the Olympics?

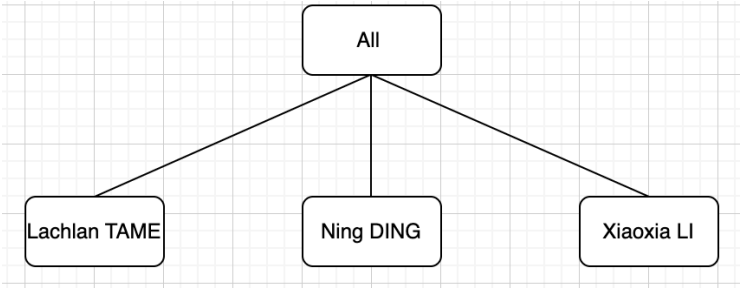

Mental Illness Location
All
Continent
Country
Year
Host Type
Year
All
All
Host Type

## ♦ Fact table & Dimension tables Design and Dimension tables' hierarchies

- **Entity Relationship Diagram for first database:**

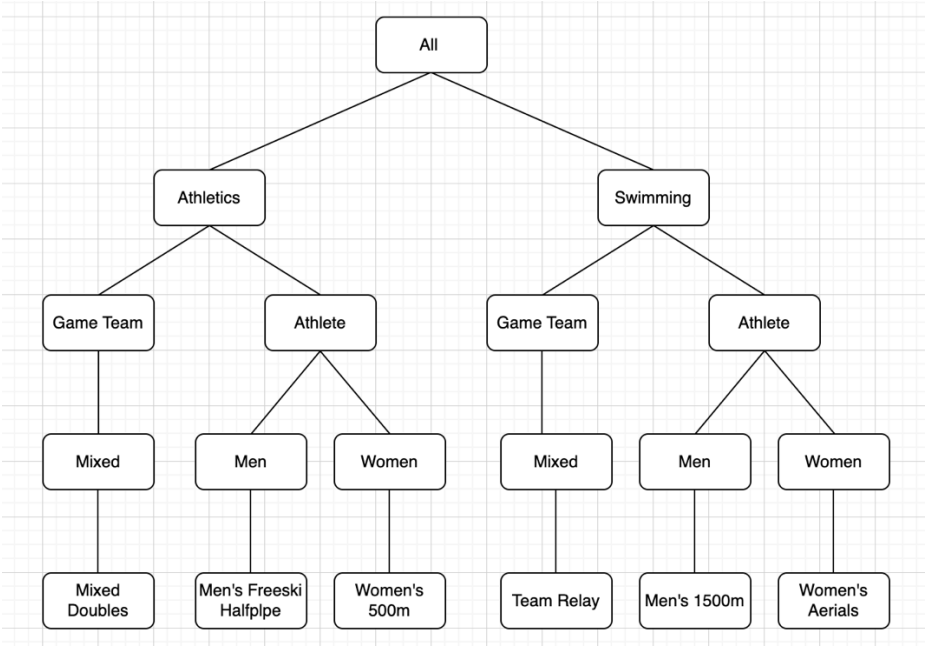



- **dim_athlete:**
  - ⇒ **"Athlete ID"** is the primary key of this dimension table, which is a unique identifier for each athlete.
  - ⇒ **"Athlete Full Name"** is the full name of each athlete.
  - ⇒ **Concept Hierarchy:**




  - ⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the number of medals associated with a particular athlete's name.

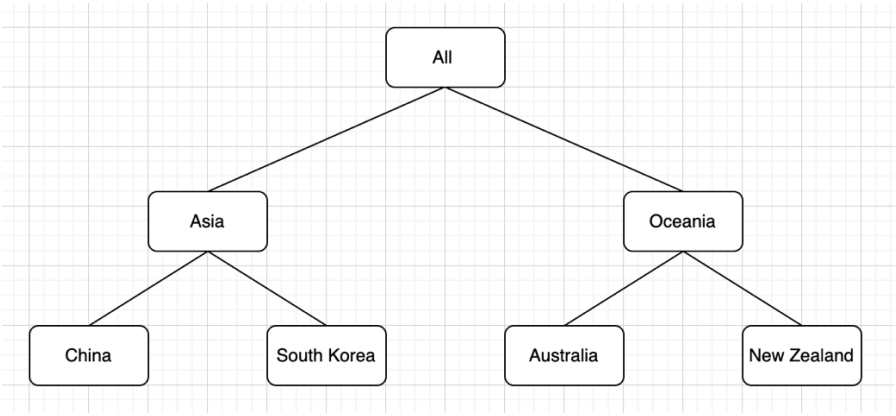
- **dim_discipline:**
  - ⇒ **“Discipline ID”** is the primary key of this dimension table, which is a unique identifier for each specific sport event.
  - ⇒ **“Discipline Title”** is the name of the category of disciplines, such as Athletics, Swimming, etc.
  - ⇒ **“Participant Type”** indicates a participant type, including Game Team and Athlete.
  - ⇒ **“Event Gender”** is the gender of the participants, including Mixed, Men and Women.
  - ⇒ **“Event Title”**is the name of a specific sport event, such as Men's Freeski Halfpipe, women's 500m, etc.
  - ⇒ **Concept Hierarchy:**




  - ⇒ **Granularity:** This dimension table enables the data warehouse queries to be refined down to the number of medals associated with a particular event.

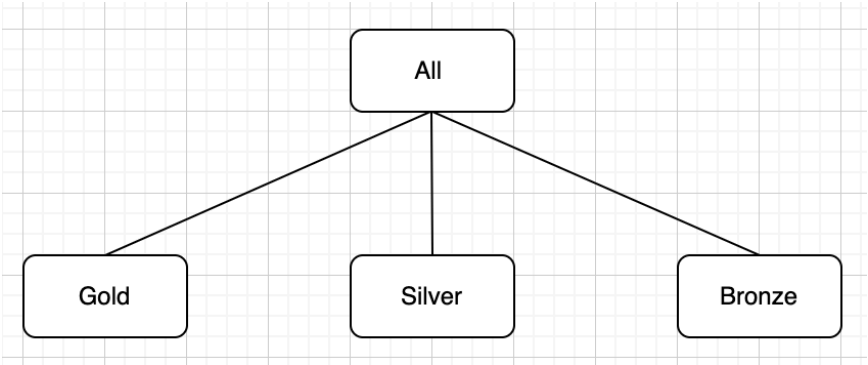
- **dim_location:**
  - ⇒ **“Location ID”** is the primary key for this dimension table, which is a unique identifier for each country.
  - ⇒ **“Continent”** is the name of the continent, such as Asia, Oceania, etc.
  - ⇒ **“Country”** is the name of a country, such as Australia, China, etc.

⇒ **Concept Hierarchy:**




⇒ **Granularity:** This dimension table allows data warehouse queries to be refined down to the number of medals associated with a specific country.

- **dim_medal_types:**
  - ⇒ **“Medal Type ID”** is the primary key for this dimension table and is a unique identifier for each medal type.
  - ⇒ **“Medal Type”** is the name of each medal type, including Gold, Silver, and Bronze.
  - ⇒ **Concept Hierarchy:**

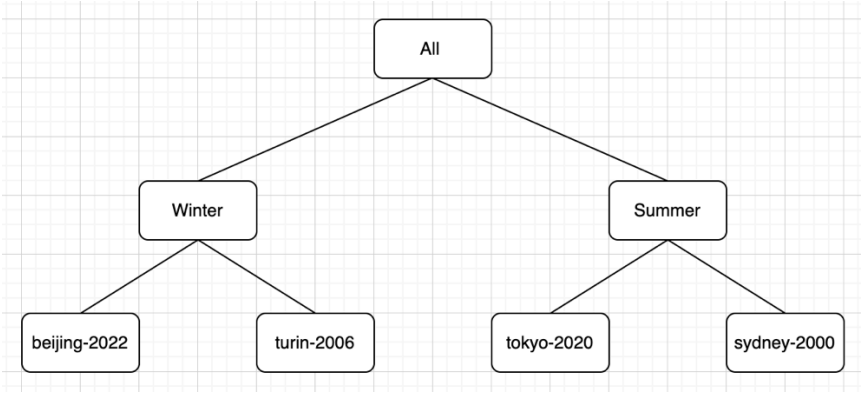



  - ⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the number of medals associated with a specific type of medal.

- **dim_olympic_game:**
  - ⇒ **“Olympic Game ID”** is the primary key for this dimension table, which is a unique identifier for each Olympic game.
  - ⇒ **“Game Season**” indicates the season in which the Olympics game is held, including Winter and Summer.

⇒ **“Slug Game”** is the name of every Olympic Games, such as beijing-2022.

⇒ **Concept Hierarchy:**



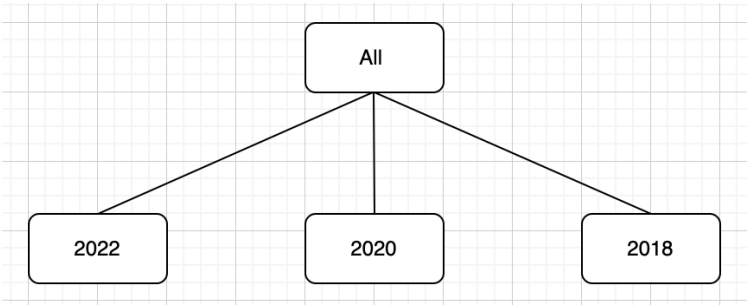

⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the number of medals associated with a specific Olympic Games.

- **dim_time:**

⇒ **“Game Year ID”** is the primary key for this dimension table, which is a unique identifier for each Olympic year.

⇒ **“Game Year”** is the year in which each Olympic Games is held.

⇒ **Concept Hierarchy:**




⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the number of medals associated with a specific Olympic Games year.

- **fact_medals_count:**

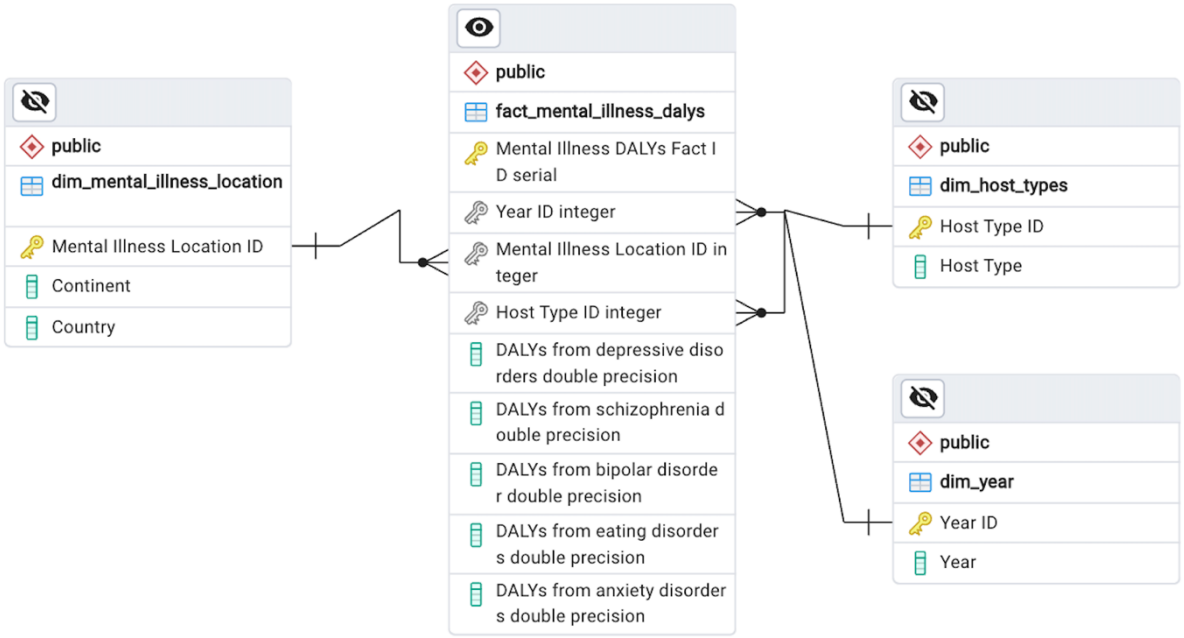
⇒ **Medals Fact ID**: is the primary key for the fact table, which is the Unique identifier for each result entry.

⇒ **Game Year ID, Athlete ID, Medal Type ID, Location ID, Olympic Game ID, Discipline ID:** This fact table associates with the other six dimension tables by referring to their primary keys as Foreign keys.

⇒ **Medal_count:** The aggregate measure used to store the total number of medals.

⇒ **Granularity**: Each row of this fact table represents a particular type of medal count won by a particular country on a particular continent in a particular sport event at a particular Olympics in a particular season in a particular year.

- **Entity Relationship Diagram for second database:**



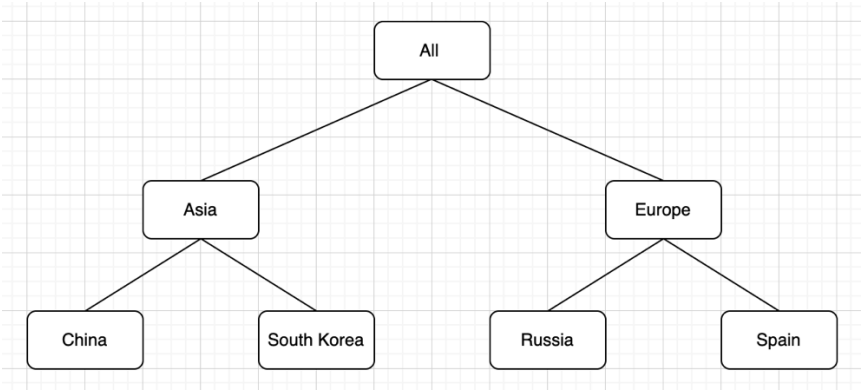

- **dim_mental_illness_location:**

⇒ **“Mental Illness Location ID”**: is the primary key for this dimension table, which is a unique identifier for each Olympic host country.

⇒ **“Continent”:** is the name of the continent to which the Olympic host country belongs, such as Asia, Oceania, etc.

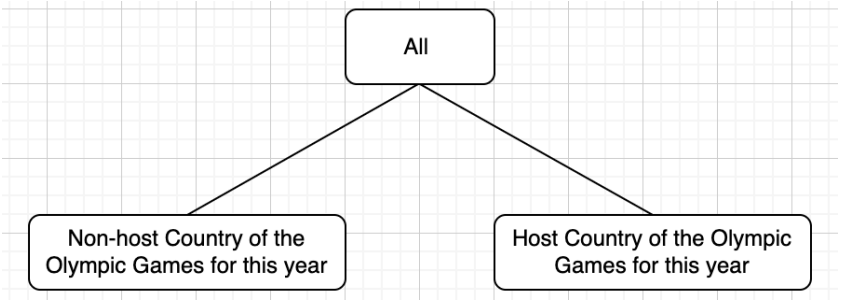
⇒ **“Country”:** is the name of the host country, such as Australia, China, etc.

⇒ **Concept Hierarchy:**



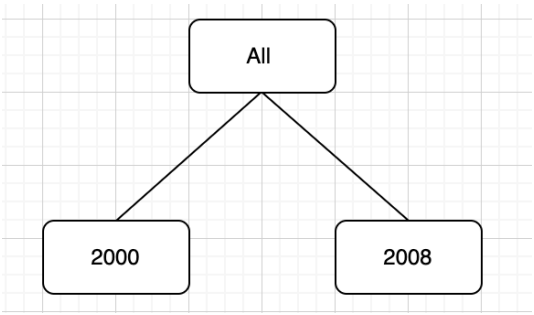

⇒ **Granularity**: This dimension table enables data warehouse queries to be refined down to the Disability-Adjusted Life Years (DALYs) for mental illnesses associated with a specific Olympic host country.

- **df_host_types:**
  - ⇒ **“Host Type ID**”: is the primary key for this dimension table and is a unique identifier for each host type.
  - ⇒ **“Host Type”:** indicates the host type, which can be non-host or host.
  - ⇒ **Concept Hierarchy:**




  - ⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the Disability-Adjusted Life Years (DALYs) for mental illnesses associated with a specific host type. This allows clients to analyse and compare disability-adjusted life year data for mental illness when a country is a host versus a non-host country.

- **dim_year:**
  - ⇒ **“Year ID”**: is the primary key for this dimension table, which is a unique identifier for the year in which the Olympic host country held the Olympic Games.
  - ⇒ **“Year”:** is the year in which the Olympic Games are held in each host country.
  - ⇒ **Concept Hierarchy:**




⇒ **Granularity:** This dimension table enables data warehouse queries to be refined down to the Disability-Adjusted Life Years (DALYs) for mental illnesses associated with a specific year.

- **fact_mental_illness_dalys:**
  - ⇒ **Mental Illness DALYs Fact ID:** is the primary key of the fact table, which is a unique identifier for each result entry.
  - ⇒ **Year ID, Mental Illness Location ID, Host Type ID:** This fact table associates with the other three dimension tables by referring to their primary keys as Foreign keys.
  - ⇒ **DALYs from depressive disorders, DALYs from schizophrenia, DALYs from bipolar disorder, DALYs from eating disorders, DALYs from anxiety disorders:** These are measures used to store the number of Disability-Adjusted Life Years (DALYs) for each type of mental illness.
  - ⇒ **Granularity:** Each row of this fact table represents the Disability-Adjusted Life Years for specific types of mental illnesses in a specific country/region of a particular continent, in a specific year, and whether it is an Olympic host country or not.

## ♦ ETL Process

I use python for the ETL process. As mentioned in the previous part of the report, my plan is to separate the data I need into two databases. The first database stores data about the performance of various countries in the Olympic Games, and the second database stores data on the mental health of each Olympic host country. Here I only briefly explain the ETL process for the dataset concerning the performance of each country in the Olympic Games. For the complete process and detailed explanation, please refer to Project_1_ETL_part.ipynb and Project_1_atoti_part.ipynb.

- **The ETL process for the dataset concerning the performance of each country in the Olympic Games:**

- **Extract:**
  Extract data from csv files (source systems) by using python and pandas library. For the dataset concerning the performance of each country in the Olympic Games, the csv files I need are *olympic_medals.csv, list-of-countries_areas- by-continent-2024.csv,* and olympic_hosts.csv. So I use pd.read_csv() function to load the data from these three csv files:

```
import pandas as pd
df_olympic_medals = pd.read_csv("olympic_medals.csv")
df_continent = pd.read_csv("list-of-countries_areas-by-continent-2024.csv")
df_olympic_hosts = pd.read_csv("olympic_hosts.csv")
```

  I then check for columns that contain empty values, and I will appropriately handle them in subsequent steps:

Check the columns that contain null values for the above four dataframes:

```
# check the columns that contain null value for df_olympic_medals. If the columns with null values are needed later on, then these null values would be processed accordingly."
null_counts_olympic_medals = df_olympic_medals[["discipline_title", "slug_game", "event_title", "event_gender", "medal_type", "participant_type", "participant_title", "athlete_
null_columns_olympic_medals = null_counts_olympic_medals[null_counts > 0]
print(null_columns_olympic_medals)
```

```
participant_title    15113
athlete_url           4670
athlete_full_name     3624
country_code          1502
dtype: int64
```

```
# check the columns that contain null value for df_continent. If the columns with null values are needed later on, then these null values would be processed accordingly."
null_counts_continent = df_continent[["country", "region"]].isnull().sum()
null_columns_continent = null_counts_continent[null_counts_continent > 0]
print(null_columns_continent)
```

```
Series([], dtype: int64)
```

```
# check the columns that contain null value for df_olympic_m. If the columns with null values are needed later on, then these null values would be processed accordingly."
null_counts_olympic_hosts = df_olympic_hosts[["game_slug", "game_end_date", "game_start_date", "game_location", "game_name", "game_season", "game_year"]].isnull().sum()
null_columns_olympic_hosts = null_counts_olympic_hosts[null_counts_olympic_hosts > 0]
print(null_columns_olympic_hosts)
```

```
Series([], dtype: int64)
```

- **Transform:**
  In this stage, I performed data pre-processing operations – data merging, data cleaning, data reordering, data mapping, data reduction, so that the data can be analysed correctly in the future. For the complete process and detailed explanation, please refer to "Project_1_ETL_part.ipynb". Here, I briefly provide one example for each type of data pre-processing operation:
  ⇒ Data merging: Merging dataframe df_olympic_medals and df_olympic_hosts.

⇒ Data cleaning: Filling missing values in the “athlete_full_name” column with “Unknown”( I didn't drop the rows with missing values of athlete name because I think it's important to keep the information of other columns), and removing rows with invalid values “#NAME?” in the “athlete_full_name” column (I drop these rows because I checked that the rows with #NAME? are actually have the same information with other rows, so these kinds of rows are actually duplicates).

```
# Fistly, Merging the df_olympic_medals and df_olympic_hosts dataframes and storing the result in a new dataframe:
merged_medals_hosts = pd.merge(df_olympic_medals, df_olympic_hosts, how="inner", left_on="slug_game", right_on="game_slug")

''' Since there are some missing values in the athlete_full_name column, so I filled them with 'Unknown'
(I didn't drop the rows because I think it's important to keep the information of other columns):'''
merged_medals_hosts['athlete_full_name'] = merged_medals_hosts['athlete_full_name'].fillna('Unknown')

''' There are #NAME? in the athlete_full_name column, so I dropped them.
(I drop these rows because I checked that the rows with #NAME? are actually have the same information with other rows,
so these kind of rows are actually duplicates):'''
merged_medals_hosts = merged_medals_hosts[merged_medals_hosts['athlete_full_name'] != "#NAME?"]
```

⇒ Data reordering: Reordering the sequence of columns.

⇒ Data reduction: Dropping columns I do not need while reordering them.

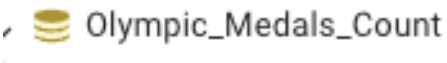
```
# Reorder the columns and keep only the columns I need:
df_merged_medals_hosts = merged_medals_hosts[['country_name', 'slug_game', 'discipline_title','event_title', 'event_gender',
                                              'medal_type', 'participant_type', 'athlete_full_name', 'game_year', 'game_season']]
```

⇒ Data mapping: Creating a dictionary to map the old country names to the new country names. Then standardise and map country names in the “country_name” column of ‘df_merged_medals_hosts’ dataframe.

```
''' Before merging, I checked the country names in the df_merged_data_medals_hosts dataframe and the country names in the df_continent dataframe,
    and found that there are some inconsistent country names, so I created a dictionary to map the inconsistent country names to the more common country names:'''
country_name_mapping = {
    "People's Republic of China": "China",
    "Great Britain": "United Kingdom",
    "United States": "United States of America",
    "Soviet Union": "Russia",
    "Republic of Korea": "South Korea",
    "Democratic People's Republic of Korea": "North Korea",
    "Islamic Republic of": "Iran",
    "Viet Nam": "Vietnam",
    "Czechoslovakia": "Czech Republic",
    # Later can add more mappings here
}
df_merged_medals_hosts['country_name'] = df_merged_medals_hosts['country_name'].map(country_name_mapping).fillna(df_merged_medals_hosts['country_name'])
```

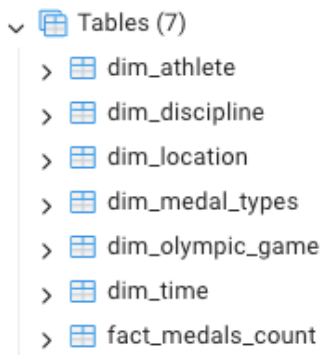
- **Load:**

Before Loading data to the data warehousing, I need to organise a data staging area. I used the PostgreSQL database as the data staging area. I created a database named "Olympic_Medals_Count" and then created the required staging tables in it:



```sql
CREATE TABLE dim_athlete (
    "Athlete ID" INT PRIMARY KEY,
    "Athlete Full Name" VARCHAR(255) NOT NULL
);

CREATE TABLE dim_medal_types (
    "Medal Type ID" INT PRIMARY KEY,
    "Medal Type" VARCHAR(50) NOT NULL
);

CREATE TABLE dim_time (
    "Game Year ID" INT PRIMARY KEY,
    "Game Year" VARCHAR(50) NOT NULL
);

CREATE TABLE dim_discipline (
    "Discipline ID" INT PRIMARY KEY,
    "Discipline Title" VARCHAR(255) NOT NULL,
    "Participant Type" VARCHAR(255) NOT NULL,
    "Event Gender" VARCHAR(50) NOT NULL,
    "Event Title" VARCHAR(255) NOT NULL
);

CREATE TABLE dim_olympic_game (
    "Olympic Game ID" INT PRIMARY KEY,
    "Game Season" VARCHAR(50) NOT NULL,
    "Slug Game" VARCHAR(255) NOT NULL
);

CREATE TABLE dim_location (
    "Location ID" INT PRIMARY KEY,
    "Continent" VARCHAR(50) NOT NULL,
    "Country Name" VARCHAR(255) NOT NULL
);
```

```sql
CREATE TABLE fact_medals_count (
    "Medals Fact ID" SERIAL PRIMARY KEY,
    "Game Year ID" INT NOT NULL,
    "Athlete ID" INT NOT NULL,
    "Medal Type ID" INT NOT NULL,
    "Location ID" INT NOT NULL,
    "Olympic Game ID" INT NOT NULL,
    "Discipline ID" INT NOT NULL,
    medal_count INT DEFAULT 1,
    FOREIGN KEY ("Game Year ID") REFERENCES dim_time("Game Year ID"),
    FOREIGN KEY ("Athlete ID") REFERENCES dim_athlete("Athlete ID"),
    FOREIGN KEY ("Medal Type ID") REFERENCES dim_medal_types("Medal Type ID"),
    FOREIGN KEY ("Location ID") REFERENCES dim_location("Location ID"),
    FOREIGN KEY ("Olympic Game ID") REFERENCES dim_olympic_game("Olympic Game ID"),
    FOREIGN KEY ("Discipline ID") REFERENCES dim_discipline("Discipline ID")
);
```



Then, I used to_sql() function to load the processed data from the previous stage into the staging tables:

3(1-1). "Olympic_Medals_Count" PostgreSQL database:

```
# Create a database connection engine to connect the "Olympic_Medals_Count" PostgreSQL database:
from sqlalchemy import create_engine
engine = create_engine('postgresql://postgres:postgres@pgdb:5432/Olympic_Medals_Count')
```

```
# dump the data in df_athlete_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_athlete_dimension.to_sql('dim_athlete', con = engine, if_exists = 'append', index = False)
```

```
888
```

```
# dump the data in df_medal_type_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_medal_type_dimension.to_sql('dim_medal_types', con = engine, if_exists = 'append', index = False)
```

```
3
```

```
# dump the data in df_time_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_time_dimension.to_sql('dim_time', con = engine, if_exists = 'append', index = False)
```

```
37
```

```
# dump the data in df_discipline_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_discipline_dimension.to_sql('dim_discipline', con = engine, if_exists = 'append', index = True)
```

```
589
```

```
# dump the data in df_olympic_game_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_olympic_game_dimension.to_sql('dim_olympic_game', con = engine, if_exists = 'append', index = True)
```

```
53
```

```
# dump the data in df_location_dimension to "Olympic_Medals_Count" PostgreSQL database:
df_location_dimension.to_sql('dim_location', con = engine, if_exists = 'append', index = True)
```

```
153
```

```
# dump the data in df_medals_fact to "Olympic_Medals_Count" PostgreSQL database:
df_medals_fact.to_sql('fact_medals_count', con = engine, if_exists = 'append', index = False)
```

```
680
```

Next, we need to connect to a PostgreSQL database using Atoti to load the data in it into atoti to design data warehouse. Here I used 'atoti' to connect to the 'Olympic_Medals_Count' database (see "Project_1_atoti_part.ipynb" for details):

```
import atoti as tt
```

```
Welcome to Atoti 0.8.10!

By using this community edition, you agree with the license available at https://docs.atoti.io/latest/eula.html.
Browse the official documentation at https://docs.atoti.io.
Join the community at https://www.atoti.io/register.

Atoti collects telemetry data, which is used to help understand how to improve the product.
If you don't wish to send usage data, you can request a trial license at https://www.atoti.io/evaluation-license-request.

You can hide this message by setting the `ATOTI_HIDE_EULA_MESSAGE` environment variable to True.
```

**Create an Atoti session:**

```
session = tt.Session(
    user_content_storage=".content",
    port=9092,
    java_options=["-Xms1G", "-Xmx10G"]
)
```

**1(1). Connect to "Olympic_Medals_Count" PostgreSQL Database:**

```
db_name = "Olympic_Medals_Count"
db_user = "postgres"
db_password = "postgres"
db_host = "pgdb"
db_port = "5432"

jdbc_url = f"jdbc:postgresql://{db_host}:{db_port}/{db_name}?user={db_user}&password={db_password}"
```

Then, in order to design my data warehouse with Atoti, I load data from "Olympic_Medals_Count" database to atoti:

```
medals_count_fact_table = session.read_sql(
    "SELECT * FROM fact_medals_count",
    keys=["Medals Fact ID"],
    table_name="fact_medals_count",
    url=jdbc_url,
)
```

```
time_dimension_table = session.read_sql(
    "SELECT * FROM dim_time",
    keys=["Game Year ID"],
    table_name="dim_time",
    url=jdbc_url,
)
```

```
olympic_game_dimension_table = session.read_sql(
    "SELECT * FROM dim_olympic_game",
    keys=["Olympic Game ID"],
    table_name="dim_olympic_game",
    url=jdbc_url,
)
```

```
medal_type_dimension_table = session.read_sql(
    "SELECT * FROM dim_medal_types",
    keys=["Medal Type ID"],
    table_name="dim_medal_types",
    url=jdbc_url,
)
```

```
location_dimension_table = session.read_sql(
    "SELECT * FROM dim_location",
    keys=["Location ID"],
    table_name="dim_location",
    url=jdbc_url,
)
```

```
discipline_dimension_table = session.read_sql(
    "SELECT * FROM dim_discipline",
    keys=["Discipline ID"],
    table_name="dim_discipline",
    url=jdbc_url,
)
```

```
athlete_dimension_table = session.read_sql(
    "SELECT * FROM dim_athlete",
    keys=["Athlete ID"],
    table_name="dim_athlete",
    url=jdbc_url,
)
```

Then, I connected those seven tables above to implement a star schema, and drew a star schema by using method ".tables.schema":

**1(3). Implement and draw a Star Schema:**

```
medals_count_fact_table.join(time_dimension_table, medals_count_fact_table["Game Year ID"] == time_dimension_table["Game Year ID"])
```

```
medals_count_fact_table.join(olympic_game_dimension_table, medals_count_fact_table["Olympic Game ID"] == olympic_game_dimension_table["Olympic Game ID"])
```

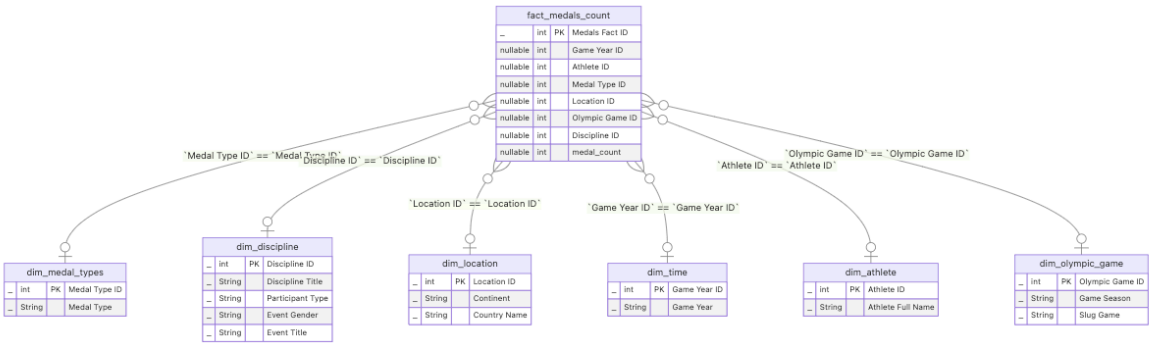
```
medals_count_fact_table.join(medal_type_dimension_table, medals_count_fact_table["Medal Type ID"] == medal_type_dimension_table["Medal Type ID"])
```

```
medals_count_fact_table.join(location_dimension_table, medals_count_fact_table["Location ID"] == location_dimension_table["Location ID"])
```

```
medals_count_fact_table.join(discipline_dimension_table, medals_count_fact_table["Discipline ID"] == discipline_dimension_table["Discipline ID"])
```

```
medals_count_fact_table.join(athlete_dimension_table, medals_count_fact_table["Athlete ID"] == athlete_dimension_table["Athlete ID"])
```

```
session.tables.schema
```



## ♦ Multi-dimensional cubes

Next I used the “.create_cube()” method to create a multi-dimensional cube to do some OLAP analysis such as roll up and drill down. Same as the above section, I only briefly explained the creation of the first multi-dimensional cube. For the complete and detailed process, please refer to "Project_1_atoti_part.ipynb" :

**1(4). Create a cube:**

```
cube = session.create_cube(medals_count_fact_table)
```

```
cube
```

▾ **fact_medals_count**
  ▸ **Dimensions**
  ▸ **Measures**

Then I created the hierarchies contained in the cube, and checked the hierarchies, levels, and measures we created:

**1(5). Create hierarchies:**

```
hierarchies, levels, measures = cube.hierarchies, cube.levels, cube.measures
```

**1(6). Check hierarchies:**

```
hierarchies
```

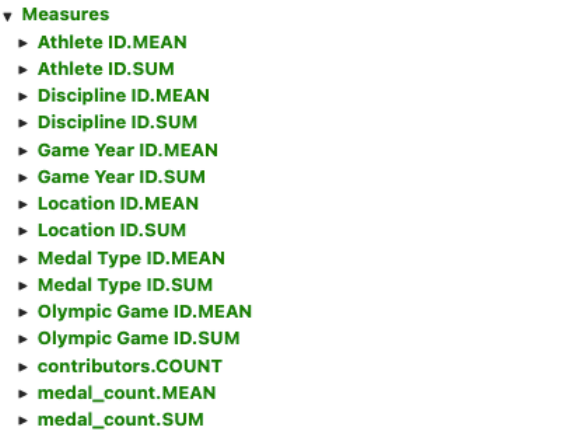
▾ **Dimensions**
  ▾ **dim_athlete**
    ▾ **Athlete Full Name** *[] 1 item*
      **0** "Athlete Full Name"
  ▾ **dim_discipline**
    ▾ **Discipline Title** *[] 1 item*
      **0** "Discipline Title"
    ▾ **Event Gender** *[] 1 item*
      **0** "Event Gender"
    ▾ **Event Title** *[] 1 item*
      **0** "Event Title"
    ▾ **Participant Type** *[] 1 item*
      **0** "Participant Type"
  ▾ **dim_location**
    ▾ **Continent** *[] 1 item*
      **0** "Continent"
    ▾ **Country Name** *[] 1 item*
      **0** "Country Name"

**1(7). Check levels:**

```
levels
```


**1(8). Check measures:**

```
measures
```




After checking the cube's hierarchies, I found some errors, so next I fixed the hierarchies to the correct order:

**1(9). Fix errors that exist in the current hierarchies:**

```
# dimensions
# dim_athlete: Athlete Full Name
# dim_discipline: Discipline Title, Participant Type, Event Gender, Event Title
# dim_location: Continent, Country Name
# dim_medal_types: Medal Type
# dim_olympic_game: Game Season, Slug Game
# dim_time: Game Year
```

```
hierarchies["dim_athlete"] = [levels["Athlete Full Name"]]
```

```
hierarchies["dim_discipline"] = [levels["Discipline Title"], levels["Participant Type"], levels["Event Gender"], levels["Event Title"]]
```

```
hierarchies["dim_location"] = [levels["Continent"], levels["Country Name"]]
```

```
hierarchies["dim_medal_types"] = [levels["Medal Type"]]
```

```
hierarchies["dim_olympic_game"] = [levels["Game Season"], levels["Slug Game"]]
```

```
hierarchies["dim_time"] = [levels["Game Year"]]
```

```
hierarchies
```

Then, I checked the hierarchies again, I found some redundant hierarchies, so I further cleaned them up:

**1(10). Clean hierarchies that we don't need:**

```
# clean dim_athlete
del hierarchies[('dim_athlete', 'Athlete Full Name')]
```

```
# clean dim_discipline
del hierarchies[('dim_discipline', 'Discipline Title')]
del hierarchies[('dim_discipline', 'Event Gender')]
del hierarchies[('dim_discipline', 'Event Title')]
del hierarchies[('dim_discipline', 'Participant Type')]
```

```
# clean dim_location
del hierarchies[('dim_location', 'Continent')]
del hierarchies[('dim_location', 'Country Name')]
```

```
# clean dim_medal_types
del hierarchies[('dim_medal_types', 'Medal Type')]
```

```
# clean dim_olympic_game
del hierarchies[('dim_olympic_game', 'Game Season')]
del hierarchies[('dim_olympic_game', 'Slug Game')]
```

```
# clean dim_time
del hierarchies[('dim_time', 'Game Year')]
```

```
# clean fact_medals_count
del hierarchies[('fact_medals_count', 'Medals Fact ID')]
```

With the correct hierarchies, we then can analyse data at different granularities through roll-up and drill-down. For example, for `dim_location`, clients can roll up to the entire `dim_location` level to view the medal counts for all countries. They can also drill down to the medal counts for a specific continent, and further drill down to the medal counts for a specific country within that continent:

```
hierarchies
```


▾ **Dimensions**
  ▾ **dim_athlete**
    ▾ **dim_athlete** *[] 1 item*
      **0** "Athlete Full Name"
  ▾ **dim_discipline**
    ▾ **dim_discipline** *[] 4 items*
      **0** "Discipline Title"
      **1** "Participant Type"
      **2** "Event Gender"
      **3** "Event Title"
  ▾ **dim_location**
    ▾ **dim_location** *[] 2 items*
      **0** "Continent"
      **1** "Country Name"
  ▾ **dim_medal_types**
    ▾ **dim_medal_types** *[] 1 item*
      **0** "Medal Type"
  ▾ **dim_olympic_game**
    ▾ **dim_olympic_game** *[] 2 items*
      **0** "Game Season"
      **1** "Slug Game"
  ▾ **dim_time**
    ▾ **dim_time** *[] 1 item*
      **0** "Game Year"

Measures also need to be cleaned up, as there are a lot of meaningless measures, such as summing and averaging the IDs in dimension tables. Therefore, I cleaned up all the meaningless measures and kept only the meaningful ones, in this case “medal_count.SUM”. “medal_count.SUM” will be used to calculate the sum of the number of medals:

**1(11). Clean meaningless measures:**

```
# Clean measures
del measures["Athlete ID.MEAN"]
del measures["Athlete ID.SUM"]
del measures["Discipline ID.MEAN"]
del measures["Discipline ID.SUM"]
del measures["Game Year ID.MEAN"]
del measures["Game Year ID.SUM"]
del measures["Location ID.MEAN"]
del measures["Location ID.SUM"]
del measures["Medal Type ID.MEAN"]
del measures["Medal Type ID.SUM"]
del measures["Olympic Game ID.MEAN"]
del measures["Olympic Game ID.SUM"]
del measures["contributors.COUNT"]
del measures["medal_count.MEAN"]
```

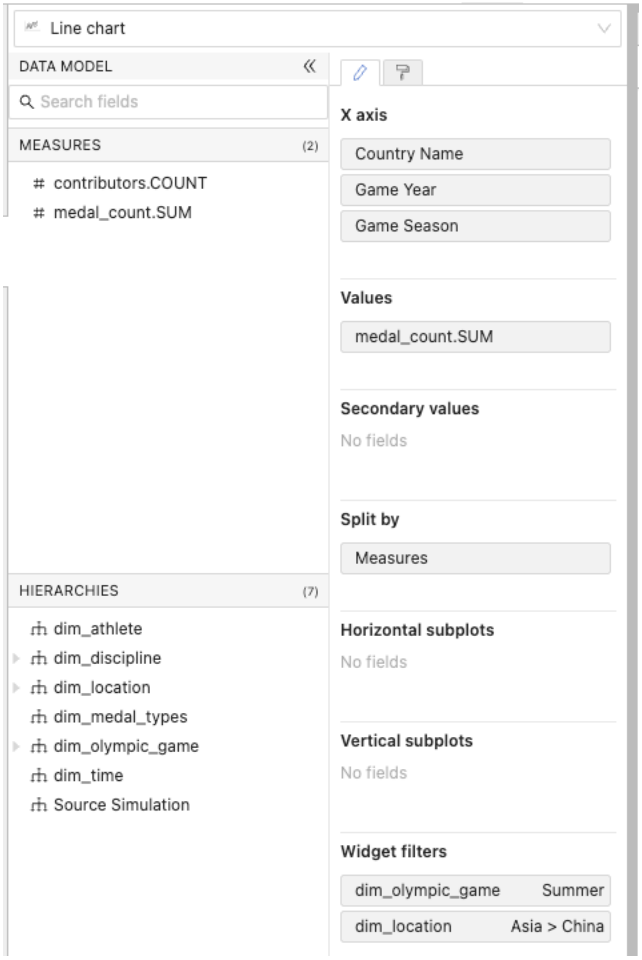
```
measures
```

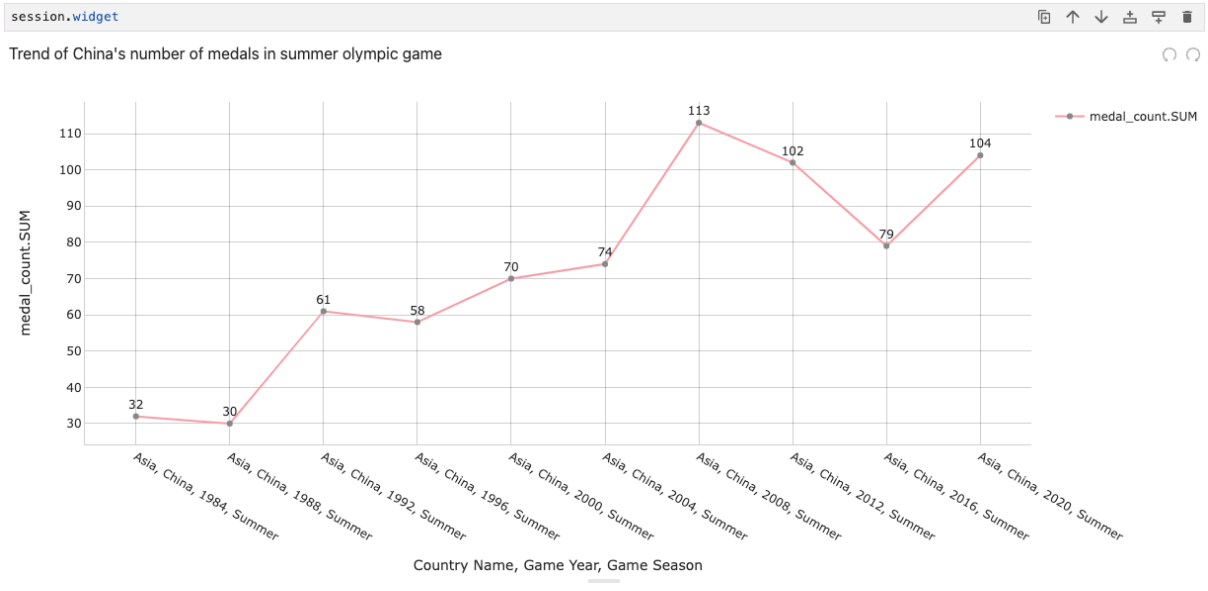
▾ **Measures**
  ▸ **medal_count.SUM**

## ♦ Query result visualisation by using atoti

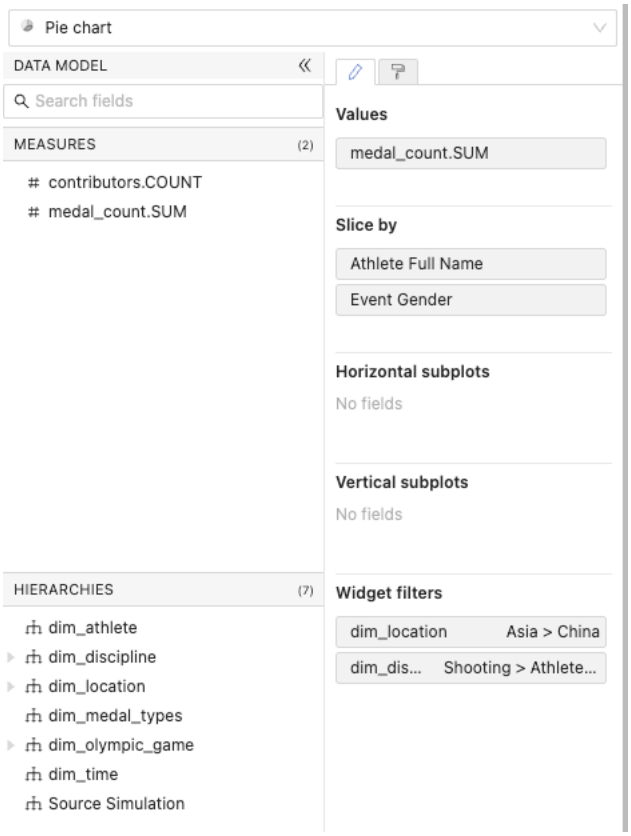
In this section, I visualized the results of all ten business queries by using atoti's web interface. For different queries, I chose appropriate charts to display the results. For instance, for queries that answer about trends, I chose line chart to show the results.

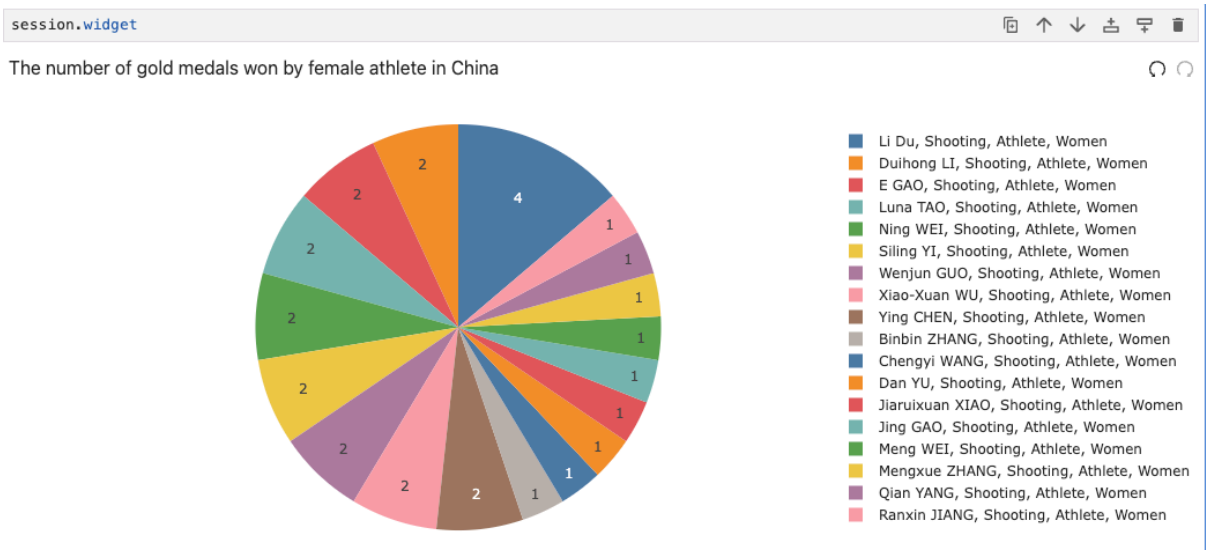
- **Query 1. What is the trend of the number of medals won by China in the Summer Olympic Games over the various years available in the dataset:**



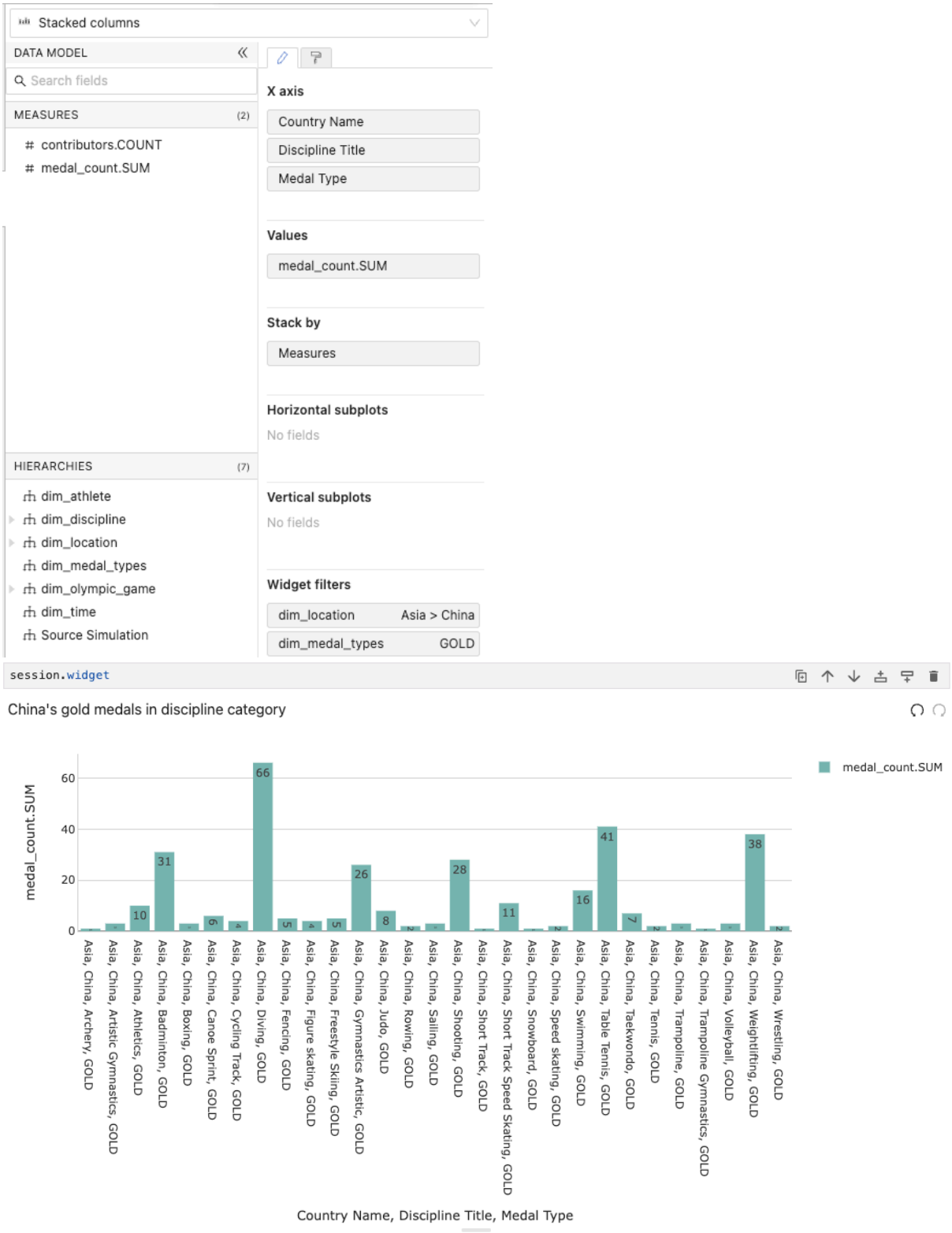



- **Query 2: Who is the Chinese individual female athlete who has won the most gold medals in shooting discipline?**

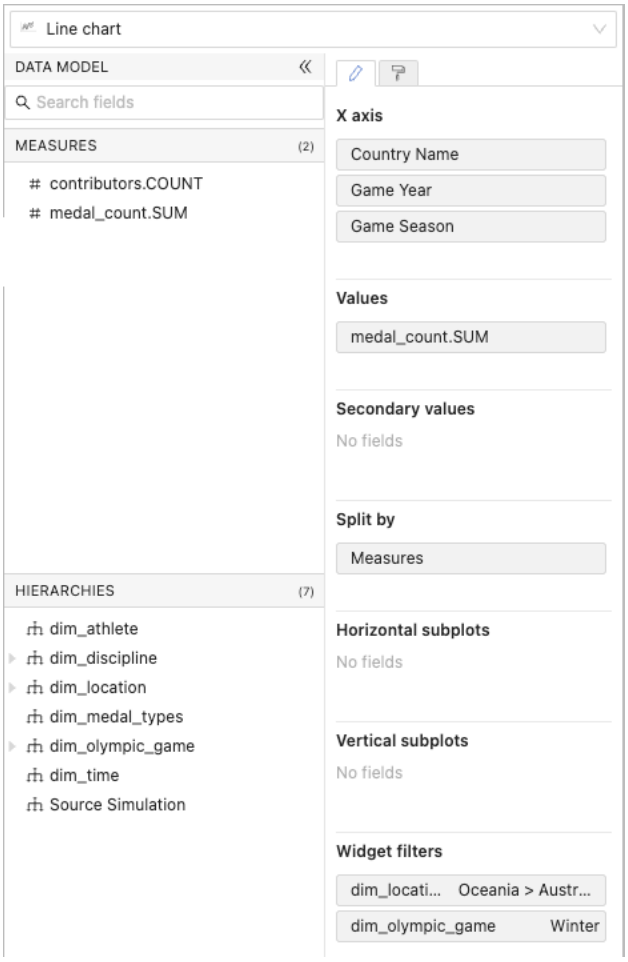


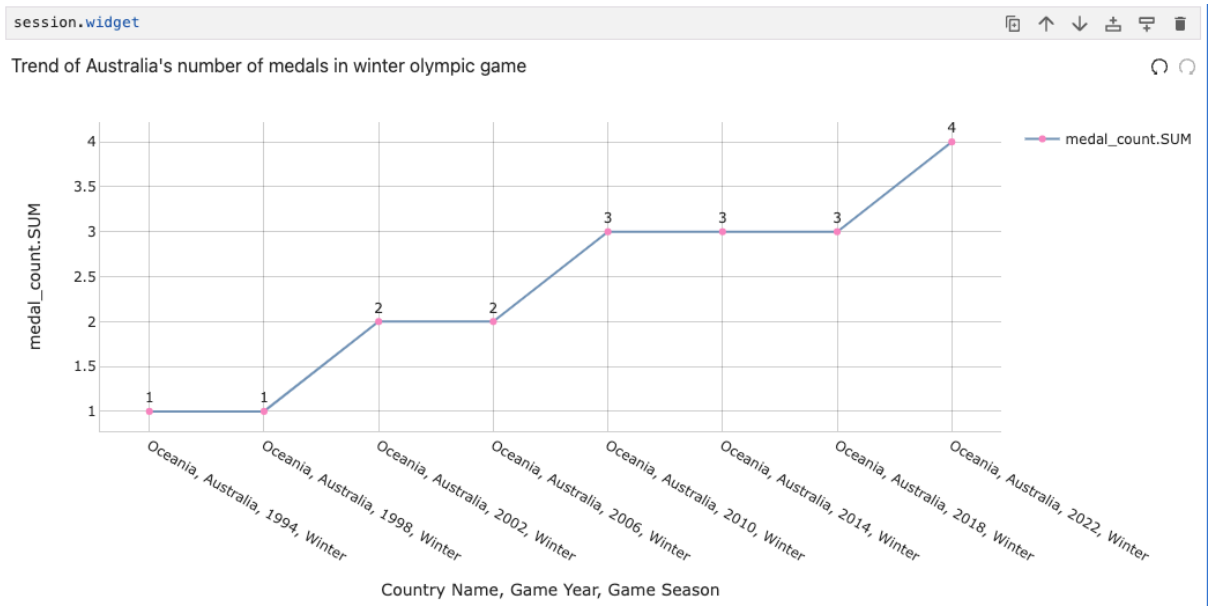




- **Query 3:  What is the discipline category in which China has won the most gold medals in Olympic history?**

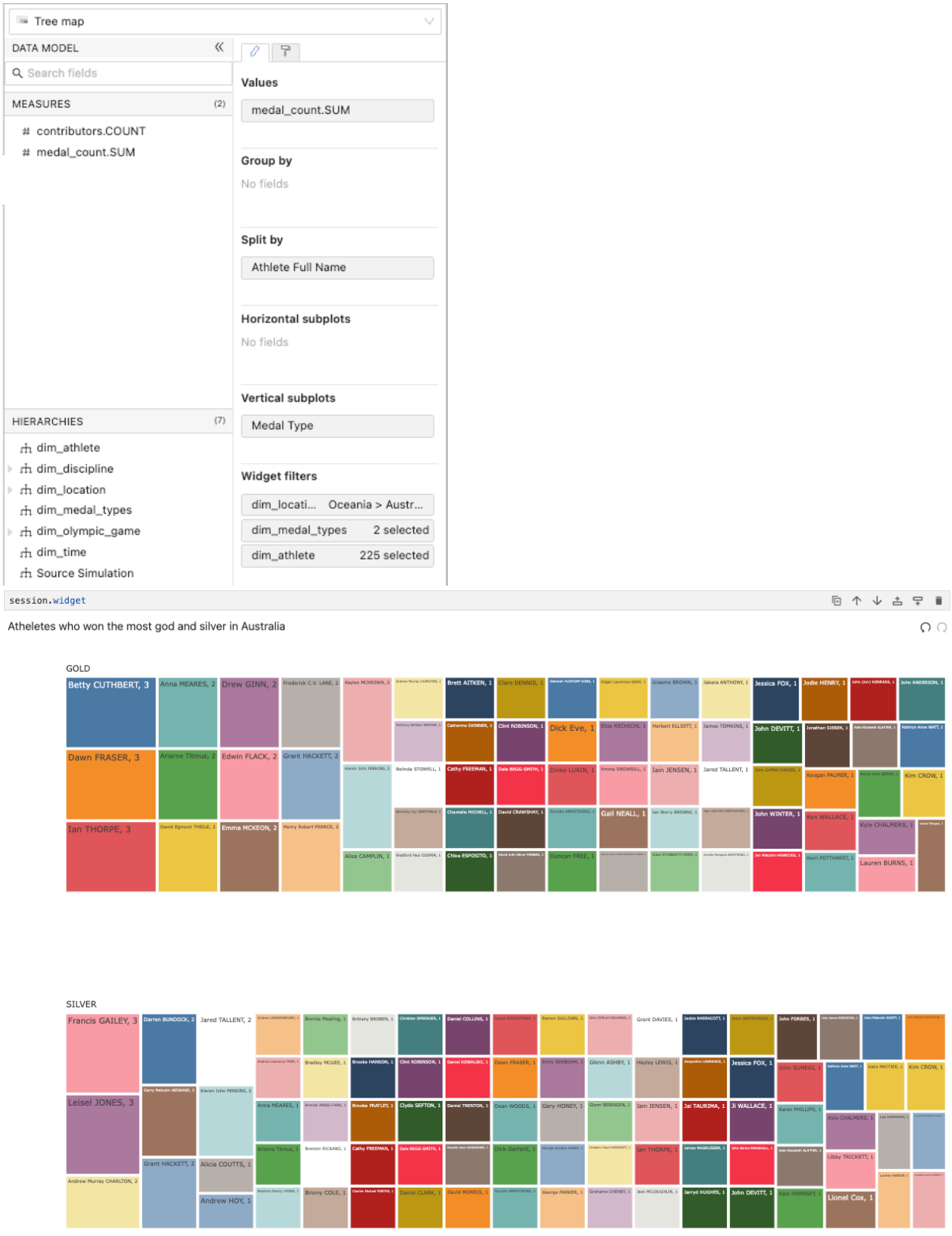



- **Query 4: What is the trend of the number of medals won by Australia in the Winter Olympic Games over the various years available in the dataset?**

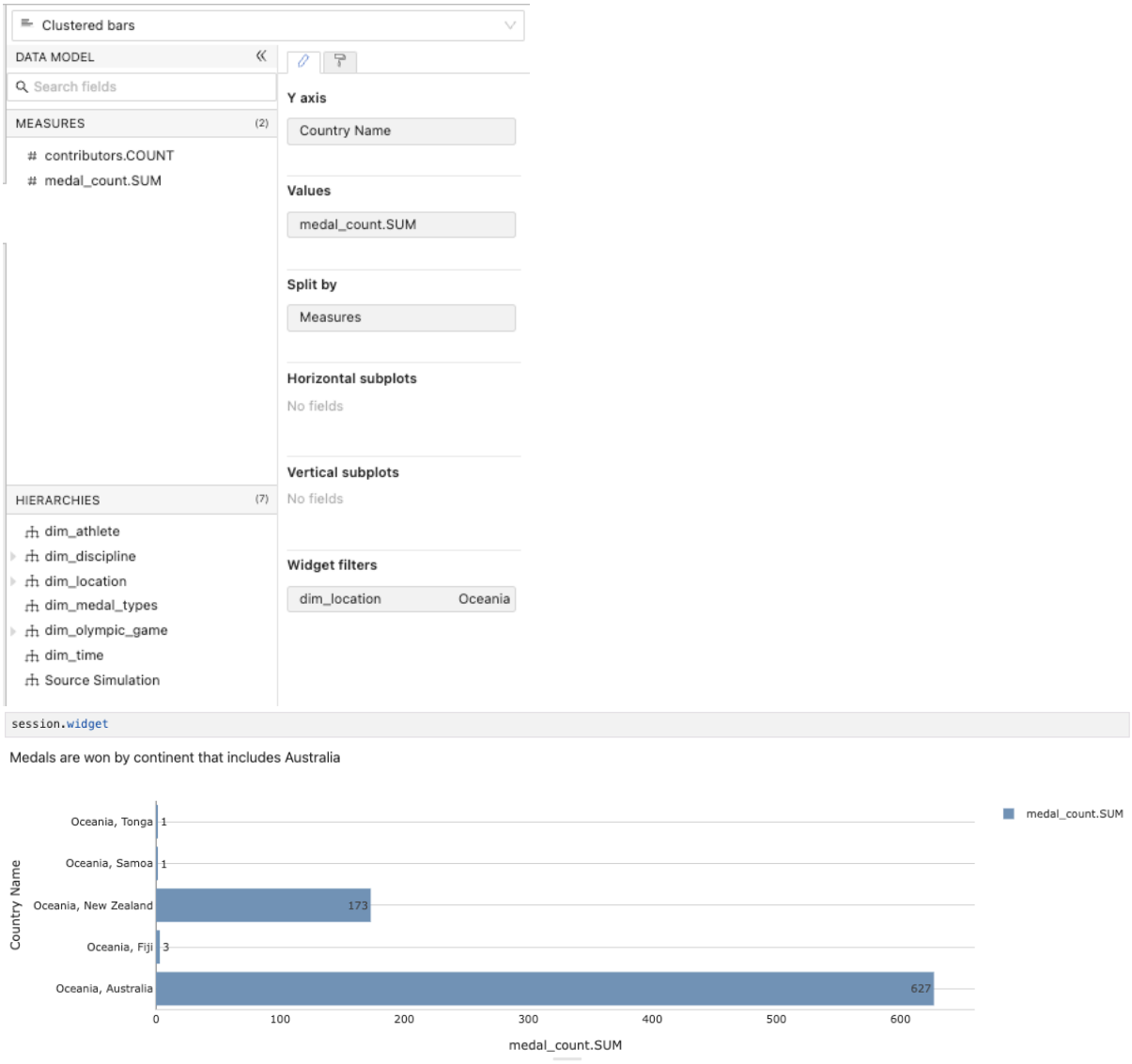


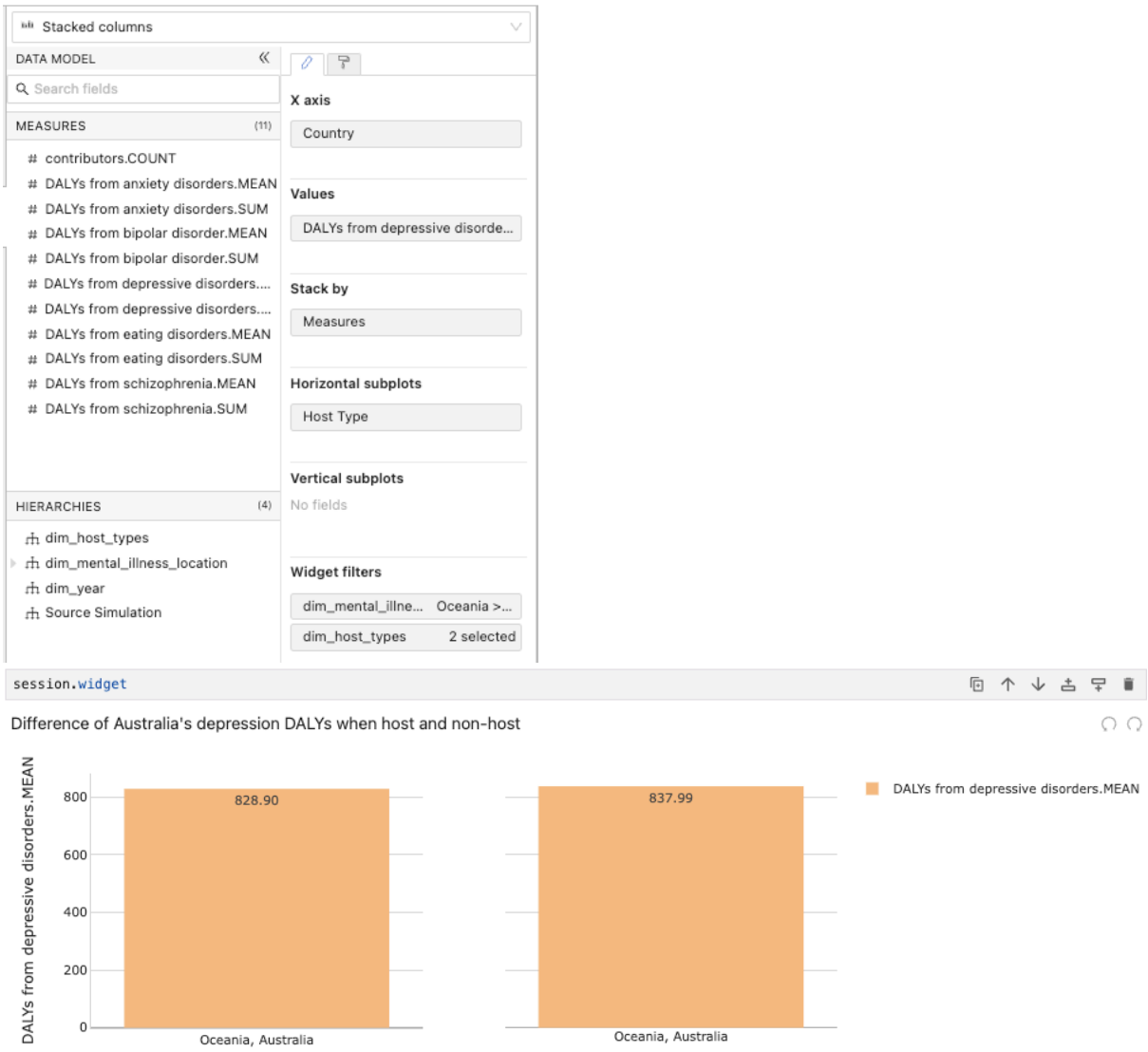

- **Query 5: Who are the Australian athletes who have won the most gold, silver medals respectively?**



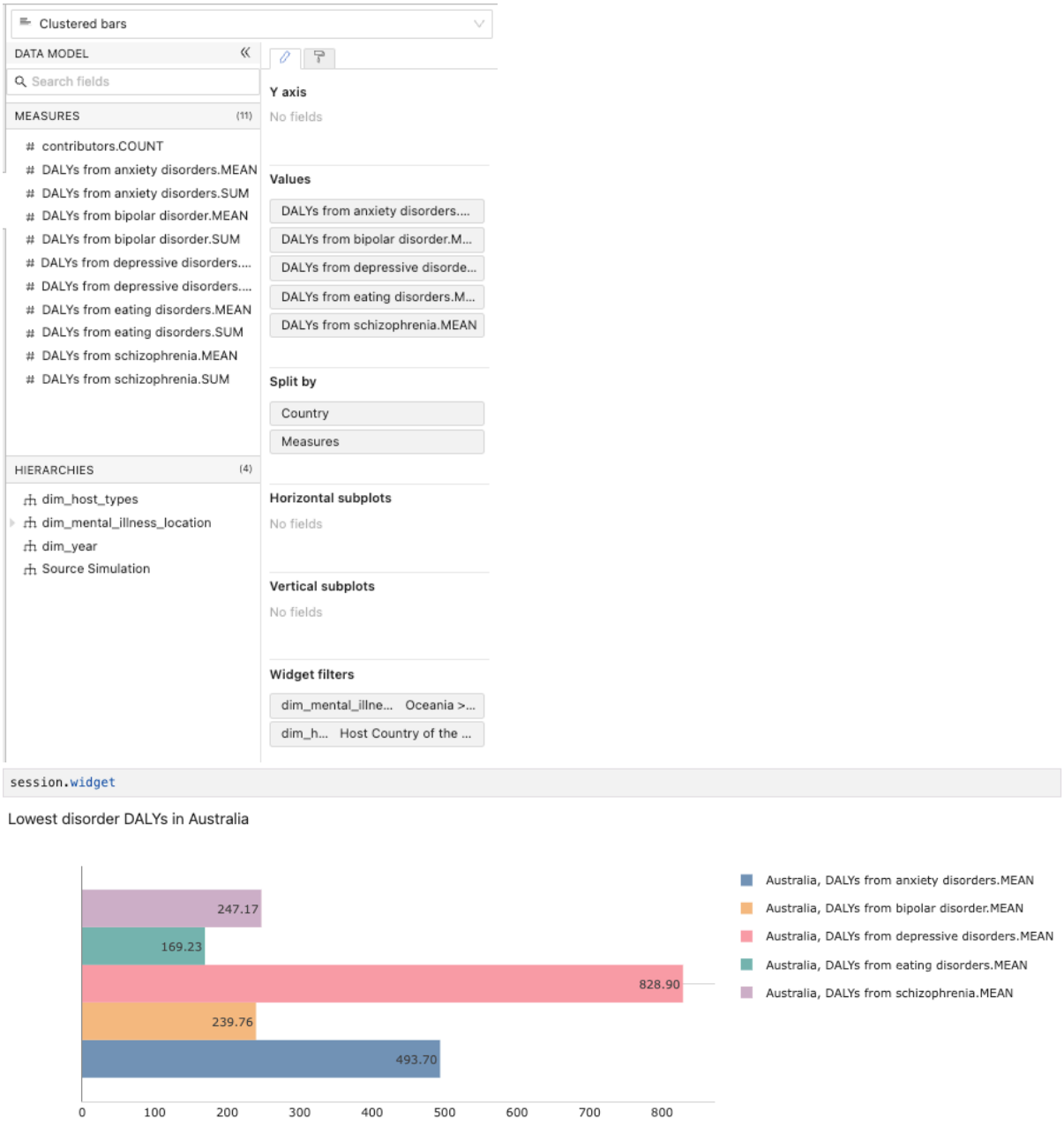

- **Query 6: How many medals have been won by the continent that includes Australia? Is Australia the country with the most medals on that continent?**




- **Query 7: What is the difference in disability-adjusted life years (DALYs) due to depression disorders in Australia when it is hosting the Olympics compared to when it is not?**



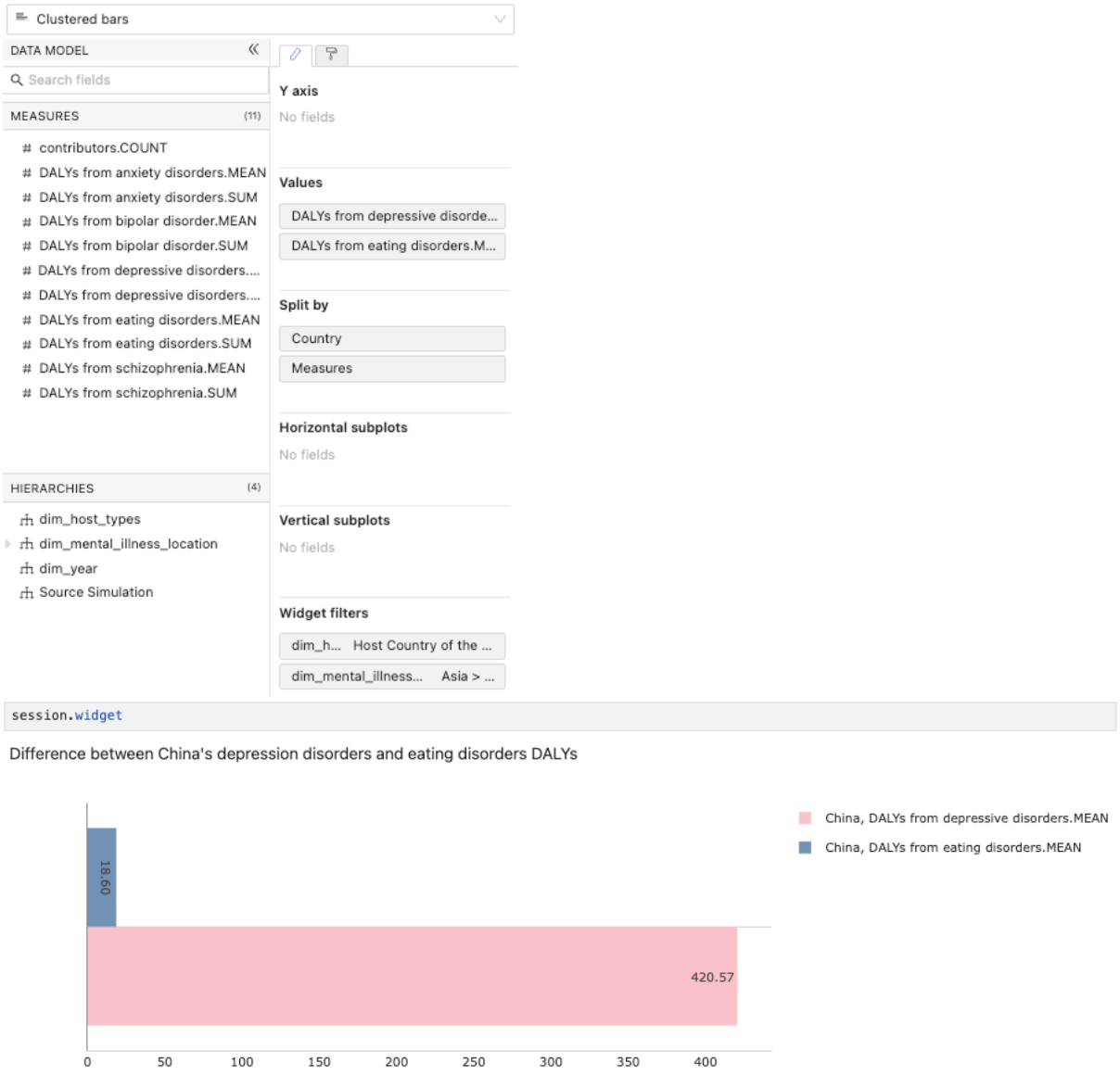

- **Query 8: Which mental disorder results in the lowest disability-adjusted life years (DALYs) in Australia when it is hosting the Olympics?**



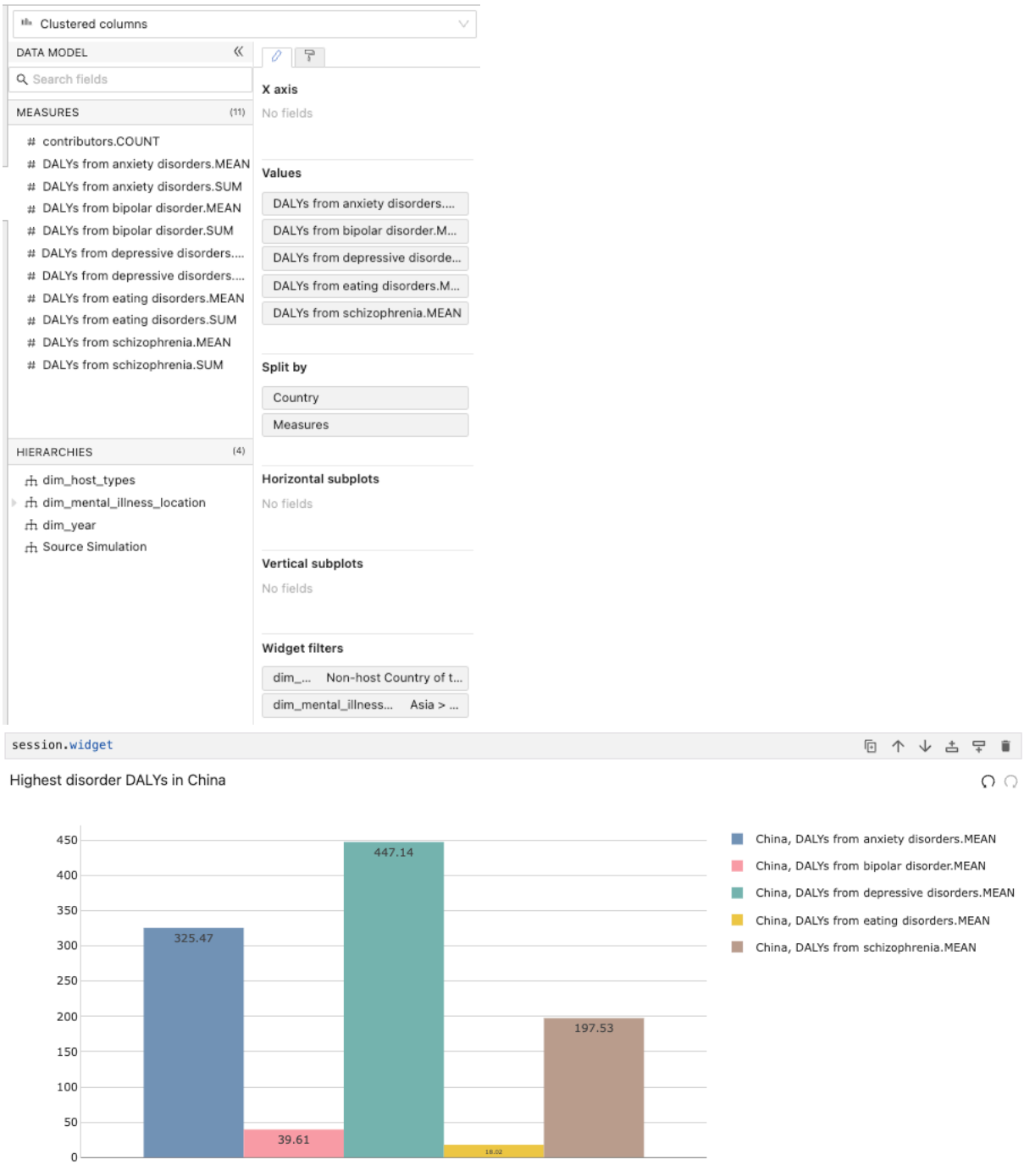

Lowest disorder DALYs in Australia




- **Query 9: What is the difference in disability-adjusted life years (DALYs) due to depression disorders and eating disorders in China when it is hosting the Olympics?**




- **Query 10: Which mental disorder results in the highest disability-adjusted life years (DALYs) in China when it is not hosting the Olympics?**


Clustered columns
DATA MODEL
Search fields
MEASURES (11)
# contributors.COUNT
# DALYs from anxiety disorders.MEAN
# DALYs from anxiety disorders.SUM
# DALYs from bipolar disorder.MEAN
# DALYs from bipolar disorder.SUM
# DALYs from depressive disorders....
# DALYs from depressive disorders....
# DALYs from eating disorders.MEAN
# DALYs from eating disorders.SUM
# DALYs from schizophrenia.MEAN
# DALYs from schizophrenia.SUM
HIERARCHIES (4)
dim_host_types
dim_mental_illness_location
dim_year
Source Simulation
X axis
No fields
Values
DALYs from anxiety disorders....
DALYs from bipolar disorder.M...
DALYs from depressive disorde...
DALYs from eating disorders.M...
DALYs from schizophrenia.MEAN
Split by
Country
Measures
Horizontal subplots
No fields
Vertical subplots
No fields
Widget filters
dim_... Non-host Country of t...
dim_mental_illness... Asia > ...
session.widget
Highest disorder DALYs in China
450
400
350
300
250
200
150
100
50
0
325.47
39.61
447.14
18.02
197.53
China, DALYs from anxiety disorders.MEAN
China, DALYs from bipolar disorder.MEAN
China, DALYs from depressive disorders.MEAN
China, DALYs from eating disorders.MEAN
China, DALYs from schizophrenia.MEAN

## ♦ Association Rule Mining

For my two clients, "China Olympic Data Analysis Team" and "Australia Olympic Data Analysis Team," I will use association rule mining to explore the potential relationships between different Olympic events, the gender of the athletes, and the types of medals for both China and Australia. Then, based on insights derived from the mining results, I will provide business recommendations for each client.

Before association rule mining, I need to install mlxtend library, and import necessary libraries:

```
pip install mlxtend
```

```
Collecting mlxtend
  Downloading mlxtend-0.23.1-py3-none-any.whl.metadata (7.3 kB)
```

```
# Import necessary libraries:
import pandas as pd
import numpy as np
import mlxtend
from mlxtend.preprocessing import TransactionEncoder
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
```

Also, import the dataset I need for association rule mining:

```
# Import dataset:
df_merged_medals_hosts_continent = pd.read_csv("merged_medals_hosts_continent.csv")
```

- **Association rule mining for China:**

- **Filter the dataset:**
  First, I dropped the data of all countries except China, and then kept only the columns I have selected for analysis. Then, I got the datafram I need, “df_association_rule_China”:

```
# Only keep China's related data:
df_association_rule_China = df_merged_medals_hosts_continent[df_merged_medals_hosts_continent['Country Name'].isin(['China'])].copy()
```

```
# Drop the columns that I don't need for this time's association rule mining:
df_association_rule_China.drop(['Country Name', 'Continent', 'Game Season', 'Slug Game',
                                'Participant Type', 'Event Title', 'Game Year', 'Athlete Full Name'], axis=1, inplace=True)
```

```
df_association_rule_China
```

| | Discipline Title | Event Gender | Medal Type |
|---|---|---|---|
| **25** | Freestyle Skiing | Women | GOLD |
| **31** | Freestyle Skiing | Mixed | SILVER |
| **37** | Freestyle Skiing | Men | GOLD |
| **40** | Freestyle Skiing | Women | GOLD |
| **43** | Freestyle Skiing | Women | GOLD |
| **...** | ... | ... | ... |
| **12177** | Diving | Women | GOLD |
| **12185** | Diving | Men | BRONZE |
| **12187** | Diving | Men | SILVER |
| **12241** | Archery | Women | SILVER |
| **12251** | Handball | Women | BRONZE |

807 rows × 3 columns

- **Data Transformation:**
  Since Apriori needs to be used to generate rules later, it's necessary to first use the “Transaction Encoder” to transform the data, thereby adapting it for the Apriori algorithm requirements in the mlxtend library. Before using the “Transaction Encoder”, all data in the dataframe must be converted into string type, as the “Transaction Encoder” requires the input data to be in string type. Then, convert the dataframe into a list of lists. Next, the “Transaction Encoder” can be used to transform this list into a boolean numpy array that fits the requirements of the Apriori algorithm. Since the apriori function can only handle dataframes, then, I convert the boolean numpy array back into a dataframe:

```
# Convert the data in new_df_association_rule_China to string type since TransactionEncoder() function only can handle string type:
new_df_association_rule_China = df_association_rule_China.astype(str)
```

```
# Convert the data from dataframe new_df_association_rule_China into a list:
list_China = new_df_association_rule_China.values.tolist()
```

```
# Covert the list to one-hot encoded boolean numpy array.
# Apriori function allows boolean data type only, such as 1 and 0, or FALSE and TRUE.
te_China = TransactionEncoder()
array_te_China = te_China.fit(list_China).transform(list_China)
```

```
# check the array:
array_te_China
```

```
array([[False, False, False, ..., False,  True, False],
       [False, False, False, ..., False, False, False],
       [False, False, False, ..., False, False, False],
       ...,
       [False, False, False, ..., False, False, False],
       [False,  True, False, ..., False,  True, False],
       [False, False, False, ..., False,  True, False]])
```

```
# check the columns:
te_China.columns_
```

```
['3x3 Basketball',
 'Archery',
 'Artistic Gymnastics',
 'Artistic Swimming',
 'Athletics',
 'BRONZE',
```

```
# Covert the array to a dataframe since Apriori function can handle dataframe only:
arm_df_China = pd.DataFrame(array_te_China, columns = te_China.columns_)
```

- **Association Rules Mining with mlxtend Library**
  I used the Apriori algorithm to find frequent itemsets with a support of no less than 5% from the encoded data. Here, I set the minimum support to a very low value of 0.05, because after trying several higher values, I found that higher minimum supports either generated no or very few frequent itemsets. Therefore, I set the minimum support to 0.05 to capture itemsets that appear in at least 5% of all transactions.

  I then generated association rules from the frequent itemsets based on confidence and lift. Here, I set the minimum confidence to 0.5 and the lift threshold to 1. I chose a minimum confidence of 0.5 because after trying higher values, I either obtained no rules or very few. Therefore, after consideration, I decided on 0.5 as the minimum confidence because I think that rules with a probability of at least 50% are relatively valuable for reference.

  **- Association Rule Generation:**

```
#Find the frequent itemsets
frequent_itemsets_China = apriori(arm_df_China,min_support=0.05,use_colnames =True)
```

```
#Check the length of rules
frequent_itemsets_China['length']=frequent_itemsets_China['itemsets'].apply(lambda x: len(x))
```

```
# Use confidence to filter out association rules that are not strong enough:
# Assume the min confidence is 0.5
rules_con_China = association_rules(frequent_itemsets_China, metric="confidence",min_threshold=0.5)
```

```
# Use lift to filter out association rules
# Assume the min lift is 1
rules_lift_China = association_rules(frequent_itemsets_China, metric="lift",min_threshold=1)
```

- **Result:**
  Finally, I got association rules table that included antecedents, consequents, support, confidence, and lift.

- Result:

```
# Based on min confidence (=0.5),
# output antecedents, consequents, support, confidence and lift.
result_arm_China = rules_con_China[["antecedents","consequents","support","confidence","lift"]]
```

```
result_arm_China
```

| | antecedents | consequents | support | confidence | lift |
|---|---|---|---|---|---|
| 0 | (BRONZE) | (Women) | 0.149938 | 0.565421 | 1.009501 |
| 1 | (Diving) | (GOLD) | 0.081784 | 0.634615 | 1.537942 |
| 2 | (Diving) | (Women) | 0.065675 | 0.509615 | 0.909866 |
| 3 | (Table Tennis) | (GOLD) | 0.050805 | 0.532468 | 1.290394 |
| 4 | (GOLD) | (Women) | 0.223048 | 0.540541 | 0.965080 |
| 5 | (Weightlifting) | (Men) | 0.053284 | 0.693548 | 1.788158 |
| 6 | (SILVER) | (Women) | 0.187113 | 0.580769 | 1.036904 |
| 7 | (Table Tennis) | (Women) | 0.050805 | 0.532468 | 0.950667 |

According to the results above, let's interpret the potential relationships between China's performance in various Olympic events, the gender of participating athletes, and the types of medals:

Firstly, we need to understand what the top-k rules are. It's an algorithm used to discover the top-k association rules that appear in a transaction database. In this context, k rules refer to the top k most important association rules ranked by evaluation metrics such as support, confidence, and lift.

So, when k>=1, in the association rule table for China, we look at the top k important association rules based on support, confidence, and lift. Let's assume k = 3 here:

**(1) Weightlifting -> Men:**

This rule has a support of 0.053284, a confidence of 0.693548, and a lift of 1.788158. Both confidence and lift are the highest among all rules.

- ⇒ **support:** Weightlifting and men occur together in 5.3284% of all observations, indicating that men participated in weightlifting events in 5.3284% of all observations.
- ⇒ **Confidence**: Given weightlifting as the antecedent, the probability of male athletes participating is 69.3548%. This means that when the sport is weightlifting, there's a 69.3548% chance that male athletes are involved.
- ⇒ **Lift**: The lift value here is 1.788158, indicating that the probability of male participation in weightlifting is 1.79 times higher than expected by random chance. This value also suggests a positive correlation between weightlifting and male athletes' participation.
- ⇒ **In summary**, this rule (Weightlifting) -> (Men) indicates that weightlifting is dominated by male athletes, with their participation is

not only widespread but also much higher than the expected random participation.

**(2) Diving -> Gold:**

This rule has a support of 0.081784, a confidence of 0.634615, and a lift of 1.537942. Both confidence and lift are the second highest among all rules.

- ⇒ **support:** Diving and gold occur together in 8.1784% of all observations, indicating that 8.1784% of the time, the diving event wins a gold medal.
- ⇒ **Confidence**: Given diving as the antecedent, the probability of winning a gold medal is 63.4615%. This means that when the sport is diving, there's a 63.4615% chance of winning a gold medal.
- ⇒ **Lift:** The lift value here is 1.537942, indicating that the probability of winning a gold medal in diving is 1.54 times higher than expected by random chance. This value also suggests a positive correlation between diving and winning gold medals.
- ⇒ **In summary**, the rule (Diving) -> (GOLD) demonstrates that diving is a highly successful sport, with a significantly higher likelihood of winning gold medals than usual. This indicates outstanding performances by diving athletes.

**(3) Table Tennis -> Gold:**

This rule has a support of 0.050805, a confidence of 0.532468, and a lift of 1.290394. Confidence ranks fifth among all rules, while lift ranks third.

- ⇒ **support:** Table tennis and gold occur together in 5.0805% of all observations, meaning that 5.0805% of the time, the table tennis event wins a gold medal.
- ⇒ **Confidence**: Given table tennis as the antecedent, the probability of winning a gold medal is 53.2468%. This suggests that when the sport is table tennis, there's a 53.2468% chance of winning a gold medal.
- ⇒ **Lift:** The lift value here is 1.290394, indicating that the probability of winning a gold medal in table tennis is 1.29 times higher than expected by random chance. This value also suggests a positive correlation between table tennis and winning gold medals.
- ⇒ **In summary**, the rule (Table Tennis) -> (GOLD) demonstrates that table tennis is a highly successful sport, with a significantly higher likelihood of winning gold medals than usual. This indicates outstanding performances by table tennis athletes.

- **Suggestions for Chinese Olympic Data Analysis Group:**

  Based on the analysis above, my suggestions to the Chinese Olympic Committee's data analysis group are:

1) Increase investment in diving and table tennis, which are strong events for winning gold medals, to further expand China's dominance in these two sports.
2) For weightlifting, where men dominate the sport, while focusing on the performance of male athletes, it's also important to enhance the development of female athletes.
3) Diving, table tennis, and weightlifting are China's strengths. This indicates that athletes in these sports may face high-pressure environments. Therefore, it's important to strengthen psychological training and support for athletes.

- **Association rule mining for Australia:**
  All the previous steps are the same as China's Association rule mining, so I will start directly from the Result section here:

**- Result:**

```
#Based on min confidence (=0.5),
#output antecedents, consequents, support, confidence and lift.
result_arm_Australia = rules_con_Australia[["antecedents","consequents","support","confidence","lift"]]
```

```
result_arm_Australia
```

| | antecedents | consequents | support | confidence | lift |
|---|---|---|---|---|---|
| 0 | (Athletics) | (Women) | 0.063796 | 0.526316 | 1.352459 |
| 1 | (BRONZE) | (Men) | 0.205742 | 0.535270 | 0.975622 |
| 2 | (Cycling Track) | (Men) | 0.062201 | 0.750000 | 1.367006 |
| 3 | (GOLD) | (Men) | 0.165869 | 0.547368 | 0.997674 |
| 4 | (Rowing) | (Men) | 0.065391 | 0.719298 | 1.311047 |
| 5 | (SILVER) | (Men) | 0.177033 | 0.566327 | 1.032229 |
| 6 | (Swimming) | (Men) | 0.185008 | 0.547170 | 0.997312 |
| 7 | (Swimming, BRONZE) | (Men) | 0.068581 | 0.589041 | 1.073630 |
| 8 | (GOLD, Swimming) | (Women) | 0.055821 | 0.507246 | 1.303457 |
| 9 | (Swimming, SILVER) | (Men) | 0.062201 | 0.557143 | 1.015490 |

So, when k>=1, in the association rule table for Australia, we look at the top k important association rules based on support, confidence, and lift. Let's assume k = 3 here:

**(1) Cycling Track -> Men:**
This rule has a support of 0.062201, a confidence of 0.75, and a lift of 1.367006. Both confidence and lift are the highest among all rules.

⇒ **support:** In all observations, 6.2201% involve cycling track and men simultaneously. indicating that men participate in the cycling track event in 6.2201% of all observations.

⇒ **Confidence:** Given the occurrence of cycling track, there is a 75% probability of male athletes participating. This implies that when the sports event is cycling track, there is a 75% chance of male participation.

⇒ **Lift:** The lift value is 1.367706, indicating that the probability of male participation in cycling track is 1.37 times higher than expected under random conditions. This value also suggests a positive correlation between cycling track and male athletes' participation.

⇒ **In summary,** the rule "Cycling Track -> Men" indicates that cycling track is dominated by male athletes, with their participation being not only widespread but also much higher than the expected random participation.

**(2) Rowing -> Men:**

This rule has a support of 0.065391, a confidence of 0.719298, and a lift of 1.311047. Confidence is the second highest among all rules, and lift is the third highest.

⇒ **support:** In all observations, 6.5391% involve rowing and men simultaneously. indicating that men participate in the rowing event in 6.5391% of the time all observations.

⇒ **Confidence:** Given the occurrence of rowing, there is a 71.9298% probability of male athletes participating. This implies that when the sports event is rowing, there is a 71.9298% chance of male participation.

⇒ **Lift:** The lift value is 1.311047, indicating that the probability of male participation in rowing is 1.31 times higher than expected under random conditions. This value also suggests a positive correlation between rowing and male athletes' participation.

⇒ In summary, the rule "Rowing -> Men" indicates that rowing is dominated by male athletes, with their participation being not only widespread but also much higher than expected random participation.

**(3) BRONZE, Swimming -> Men:**

This rule has a support of 0.068581, a confidence of 0.589041, and a lift of 1.07363. Confidence is the third highest among all rules, and lift is the fifth highest.

⇒ **support:** In all observations, 6.8581% involve men winning bronze medals in swimming events. This means that in all observations, 6.8581% of the time, men win bronze medals in swimming.

⇒ **Confidence:** Given the occurrence of bronze medals in swimming, there is a 58.9041% probability of male athletes winning them. This implies that when bronze medals are won in swimming events, there is a 58.9041% chance of them being won by male athletes.

⇒ **Lift:** The lift value is 1.07363, indicating that the probability of male participation in winning bronze medals in swimming events is 1.07

times higher than expected under random conditions. This value also suggests a positive correlation between winning bronze medals in swimming events and male athletes.

⇒ **In summary**, the rule "(BRONZE, Swimming) -> (Men)" shows a relatively high proportion of male athletes winning bronze medals in swimming events.

- **Suggestions for Australian Olympic Data Analysis Group:**
  Based on the analysis above, my suggestions to the Australian Olympic Committee's data analysis group are:
  1) Increase investment in Cycling Track, Rowing, and Swimming, which are strong events for Australia, to further expand Australia's advantage in these areas.
  2) For Cycling Track and Rowing, where male athletes dominate, while focusing on male athletes' performance, it's also important to enhance the development of female athletes.
  3) Cycling Track, Rowing, and Swimming are Australia's strengths. This indicates that athletes in these sports may face high-pressure environments. Therefore, it's important to strengthen psychological training and support for athletes.

- **Do you agree or disagree with “data cube is an outdated technology”**

  The data cube was popular in the early days of data warehousing, but even today, it still has irreplaceable advantages in certain aspects, so I don't agree with the idea that data cube is an outdated technology.

  Firstly, one of the main advantages of data cube is that they are fast and efficient. The data cube precomputes all possible aggregates, which greatly simplifies complex query operations and provides very high response speed for complex queries. That is, we don't need to query the database every time we need information. This kind of predictive data can be very flexible to handle a variety of sudden queries, which is particularly important in business decisions, because decision makers often need to obtain a variety of data support in a short time.

  Moreover, the data cube is very suitable for multi-dimensional data analysis and information visualization. Users can slice, dice, pivot, and rotate data along each available axis in the multidimensional structure without compromising performance. And data visualization tools can effectively generate charts to help users analyse data from multiple dimensions. We can create many different types of charts in each dimension, and even introduce additional dimensions to display more information about the data. For example, when I analyse the number of medals won by Australia in different disciplines, these data can be visually displayed through data visualization tool. Additionally, I can also add another dimension to show the number of medals won by Australia in various disciplines in different years, so as to compare and analyse the performance of Australian athletes in different time periods.

  The third advantage of data cube is that they are user friendly. It lowers the threshold of data analysis and can be well understood and used by users without technical background, because it provides a relatively simple way to query and analyse data. Users can explore data through a graphical interface without the need to write complex SQL queries. There is also no need to understand the complex structure of the back-end databases.

  With the development of The Times, many good technologies have emerged, but the scenarios and needs adapted to each technology are different. For example, Stream Processing can handle real-time data streams and perform fast analysis, but for scenarios requiring highly optimized query performance and complex multidimensional analysis, data cubes still have an advantage. The choice of technology should be based on the scenario and requirements, not just on the age of technology.

(In this section, I actually asked chat gpt first, and gpt gave me an opinion agreeing that "data cube is an outdated technology." But I didn't take its advice because I didn't agree with it.)

## ♦ Reference List


https://www.philippe-fournier-viger.com/spmf/TopKAssociationRules.php

https://www.kyvosinsights.com/glossary/data-cube/

https://staragile.com/blog/data-cube

https://www.techtarget.com/searchdatamanagement/definition/stream-processing

https://chat.openai.com/