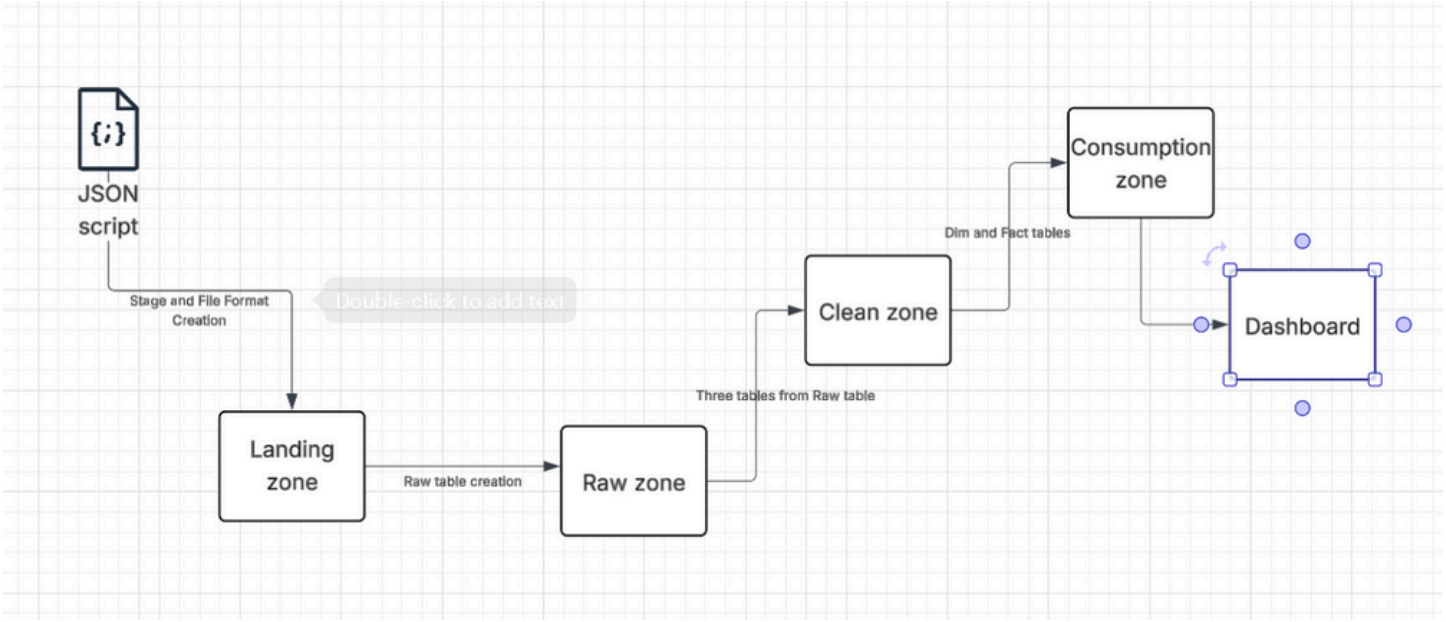
Cricket Data Lakehouse

Cricket Data Lakehouse (Medallion architecture) – Snowflake

Git: [AWS-Projects/Snowflake Cricket Datalake at main · diljotrandhawaa/AWS-Projects]

Workflow:

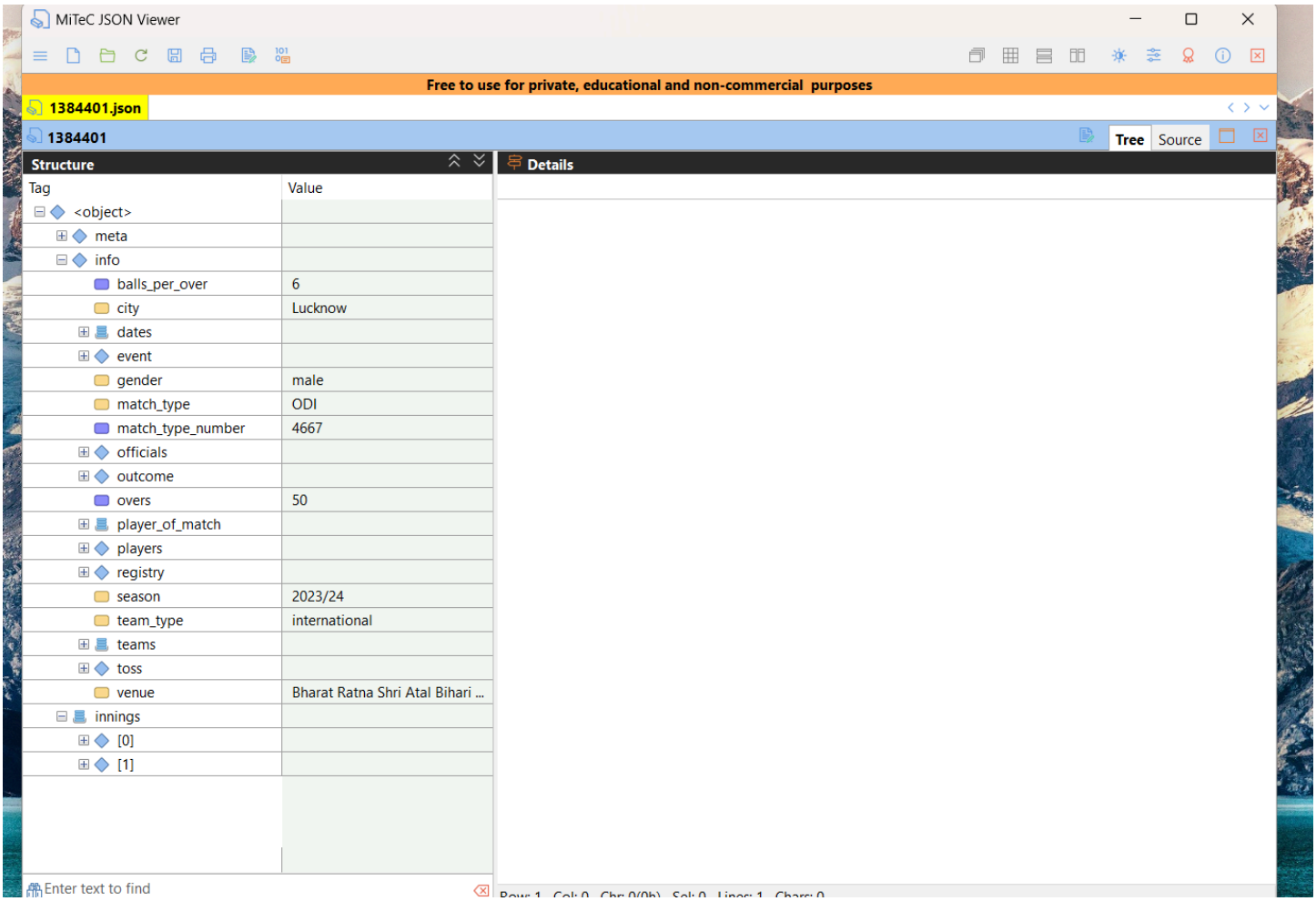


The entire Project is done in Snowflake's Web Interface. The blocks "Landing zone", "Raw zone", "Clean zone" and "Consumption zone" are four schemas created to imitate the layers of Medallion Architecture.

Tech used:
Snowflake
Snowflake Web Interface
Json

Steps:

Step 1: Data Insights
Raw data is present on-premises. The data is in the format of JSON files. To view and get the idea of data structure, a third-party tool can be used, for example "MiTech JSON Viewer". Below is the structure of each JSON file of our Cricket data:



Data Insights:

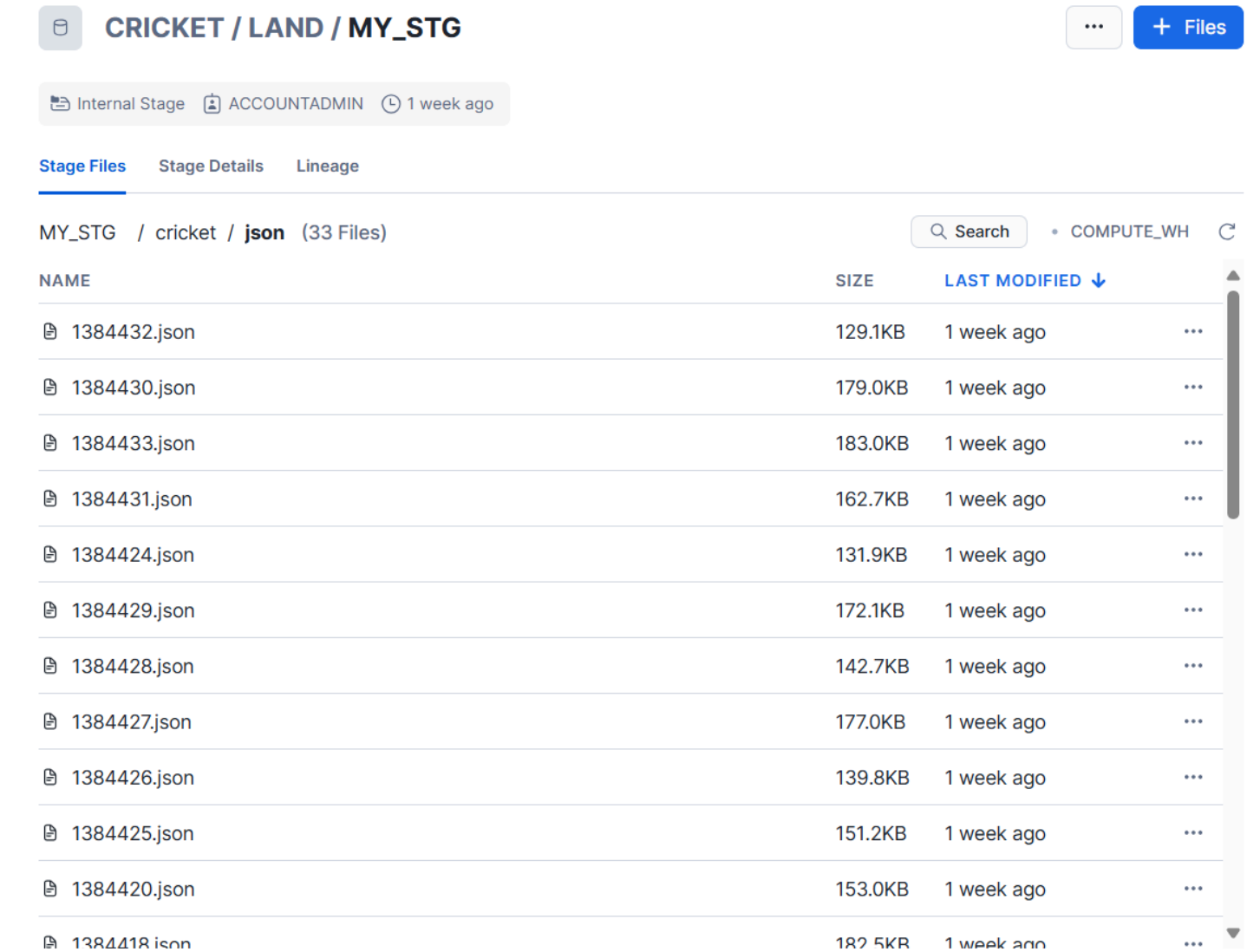
1. Each file corresponds to one Cricket match between two teams.
2. “info” contains information about the location of the match, officials like Referees and Players. It also has information about the teams playing in the match and who won the toss, etc.
3. “Innings” has information about the match itself. It has data for each over played in the match by both teams, for example, how many runs were scored in each ball of the over, how many wickets were taken, etc.

Step 2: Schema Creation:

First, we create the database and four different schemas for Landing, Raw, Clean and Consumption layers.

Then in Landing schema, we create the stage and file format for our json files.

Then we upload the Json files (33 in total) to the stage.



Step 3: First Raw table creation:
When the data is loaded, In Raw zone schema, we create a raw table combining data from all Json files using copy data function

```
copy into cricket.raw.match_raw_tbl from
    (
    select
        t.$1:meta::object as meta,
        t.$1:info::variant as info,
        t.$1:innings::array as innings,
        --
        metadata$filename,
        metadata$file_row_number,
        metadata$file_content_key,
        metadata$file_last_modified
    from @cricket.land.my_stg/cricket/json (file_format => 'cricket.land.my_json_format') t
    )
    on error = continue;
```

Step 4:
In Clean zone schema, we create threetables out of the raw table. The four tables are for

Match, Player and Delivery.
Every table is created with a Primary key and the relationships are created between the tables using foreign keys.

Step 5:
In Consumption zone, we create the Dimension and Fact tables. The dimension tables for Referee, Team, Player, Venue, Match type and Date are created. The fact tables for Match and Delivery are created. and the data is inserted using copy data function.
For example, below is the structure of delivery_fact table

```
CREATE or replace TABLE delivery_fact (
    match_id INT ,
    team_id INT,
    bowler_id INT,
    batter_id INT,
    non_striker_id INT,
    over INT,
    runs INT,
    extra_runs INT,
    extra_type VARCHAR(255),
    player_out VARCHAR(255),
    player_out_kind VARCHAR(255),

    CONSTRAINT fk_del_match_id FOREIGN KEY (match_id) REFERENCES match_fact (match_id),
    CONSTRAINT fk_del_team FOREIGN KEY (team_id) REFERENCES team_dim (team_id),
    CONSTRAINT fk_bowler FOREIGN KEY (bowler_id) REFERENCES player_dim (player_id),
    CONSTRAINT fk_batter FOREIGN KEY (batter_id) REFERENCES player_dim (player_id),
    CONSTRAINT fk_stricker FOREIGN KEY (non_striker_id) REFERENCES player_dim (player_id)
);
```

Step 6: Create Dashboard
After the Dimension and Fact tables are created and stored. We use the data to create dashboards.
First Dashboard: Highest Run Scorers (Players)

Dashboard 2: Team Rankings in the tournament (by matches won)

2025-04-21 8:26pm

10 rows

Rank	Team Name	Total Wins
1	South Africa	6
2	Australia	6
3	India	6
4	Afghanistan	4
5	New Zealand	3
6	Netherlands	2
7	Sri Lanka	2
8	Pakistan	2
9	England	1
10	Bangladesh	1