CSE 606 Data Warehousing
Major Project

Group 16:

Rahul Maheshwari (MT19027)

P. Akshay Kumar (MT19094)

Nikunj Agarwal (MT19093)

a) Decision Problem: Select a city to organize the Football World Cup.

b) Building the DIEM Schema

1. Select a city for organizing Football World Cup:
    a. Uncertainties
        i. Security for spectators and players
        ii. Popularity of the sport in city
        iii. Financial status of the city
    b. Action
        i. Organize Match
        ii. Accommodate teams
        iii. Manage bookings
    c. Objective
        i. Maximize Gate Collections – **MEANS**
        ii. Maximize Sponsorship - **MEANS**
        iii. Minimize Expenditure - **MEANS**
        iv. Maximize Viewership - **CSF**
        v. Maximize Revenue – **ENDS**
    d. Object
        i. O_{Act}:
        OrganizeMatch (Match, Stadium, City, BookedSeats, PromotionCost, MatchExpenses)
        AccommodateTeams (Team, Budget)
        ManageBookings (Match, Stadium, City, Stand, Price, SeatCount)
        ii. O_{Unc}:
        Security (NumberOfGuards, Match, Stadium, SurvillanceSystem)
        Match_Attendance (City, Stadium, Match, Date, Attendance)
        FinancialHealth (City, Budget, Debt)
        iii. O_{obj}:
        Gate Collections
        Viewership
        Sponsorship
        Expenditure
        Revenue

2. Business Intelligence Elicitation Process

The actions are considered one after another:

a. OrganizeMatch (Match, Stadium, City, BookedSeats, PromotionCost, MatchExpenses)

Derived Attribute:
PromotionCost=DigitalAdvCost + OutdoorAdvCost + SocialMediaAdvCost

This yield:

OrganizeMatch (Match, Stadium, City, BookedSeats, DigitalAdvCost, OutdoorAdvCost, SocialMediaAdvCost, MatchExpenses)

b. AccommodateTeams (Team, Budget)

Derived Attribute:
Budget=HospitalityCharges + TransportCost + SecurityCost + MiscCost

This yield:

AccommodateTeams (Team, HospitalityCharges, TransportCost, SecurityCost, MiscCost)

c. ManageBookings (Match, Stadium, Stand, City, Price, SeatCount)

Derived Attribute:
Price=BasePrice + Tax

This yield:
ManageBookings (Match, Stadium, Stand, City, BasePrice, Tax, SeatCount)

3. Choice Elicitation:
The attributes for computing the various O_{obj} are as follows:

a. GateCollections (Match, Stadium, Stand, City, Price, BookedSeat, SeatCount)
b. Viewership (Match, ViewCount, Medium, Region)
c. Sponsorship (Match, Tournament, Sponsor, SponsorshipMoney)
d. Expenditure (Match, Tournament, PromotionCost, MatchExpenses, AccomodationCost)
e. Revenue (Match, Tournament, GateCollections, Viewership, Sponsorship)

c) Instantiating GOM4DW

In this step, we determine the functional dependencies among the attributes of the objects. We present the objects first and then the functional dependencies.

I. For Business Intelligence:

The objects considered are due to the procedure of Step b above as well as the objects from the uncertainties obtained in Step a.

1. OrganizeMatch (Match, Stadium, City, BookedSeats, DigitalAdvCost, OutdoorAdvCost, SocialMediaAdvCost, MatchExpenses)

   Match, Stadium, City ->BookedSeats, DigitalAdvCost, OutdoorAdvCost, SocialMediaAdvCost, MatchExpenses

2. AccommodateTeams (Team, HospitalityCharges, TransportCost, SecurityCost, MiscCost)

   Team-> HospitalityCharges, TransportCost, SecurityCost, MiscCost

3. ManageBookings (Match, Stadium, Stand, City BasePrice, Tax, SeatCount)

   Match, Stadium, Stand, City -> BasePrice, Tax, SeatCount

Uncertainities:

4. Security (NumberOfGuards, Match, Stadium, City, SurvillanceSystem)

   Match, Stadium, City -> NumberOfGuards, SurvillanceSystem

5. Match_Attendance (Match, Stadium, City, Attendance)

   Match, Stadium, City -> Attendance

6. FinancialHealth (City, Budget, Debt)

   City->Budget, Debt

II. For Choice Phase:

1. GateCollections (Match, Stadium, Stand, City, Price, BookedSeat, SeatCount)

   Match, Stadium, Stand, City -> Price, BookedSeat, SeatCount

2. Viewership (Match, ViewCount , Medium, Region)

   Match, Medium, Region -> ViewCount

3. Sponsorship (Match, Tournament, Sponsor, SponsorshipMoney)

   Match, Tournament, Sponsor -> SponsorshipMoney

4. Expenditure (Match, Tournament, PromotionCost, MatchExpenses, AccomodationCost)

   Match, Tournament -> PromotionCost, MatchExpenses, AccomodationCost

5. Revenue (Match, Tournament, GateCollections, Viewership, Sponsorship)

   Match, Tournament -> GateCollections, Viewership, Sponsorship

- **Functional Dependency and Tuple Analysis**

Table I: Data and Category Objects from Functional Dependencies and Tuples

No.	Category Object from FD	Data Objects from FD	Category Objects from Tuples	Data Objects taking tuples into account
1	Match, Stadium, City	BookedSeats, DigitalAdvCost, OutdoorAdvCost, SocialMediaAdvCost, MatchExpenses	-	As in column 2
2	Team	HospitalityCharges, TransportCost, SecurityCost, MiscCost	-	As in column 2
3	Match, Stadium, Stand, City	BasePrice, Tax, SeatCount	-	As in column 2
4	Match Stadium, City	NumberOfGuards, SurvillanceSystem	-	As in column 2
5	Match Stadium, City	Attendance	-	As in column 2
6	City	Budget, Debt	-	As in column 2
7	Match, Stadium, Stand, City	Price, BookedSeat, SeatCount	-	As in column 2
8	Match, Medium, Region	ViewCount	-	As in column 2
9	Match, Tournament, Sponsor	SponsorshipMoney	-	As in column 2
10	Match, Tournament	PromotionCost, MatchExpenses, AccomodationCost	-	As in column 2
11	Match, Tournament	GateCollections, Viewership, Sponsorship	-	As in column 2

- **Obtaining GOM4DW Schema**

Attributes are not shown multiple times.

Table II: GOM4DW Schema with Abstracted/Separated Objects

No.	Category Objects	Simple Data Objects	Aggregate Objects	Categories over which aggregated	History
1	Match (Match#, Team A, Team B, Datetime) , Stadium (StadiumName, Capacity), City (CityName, FinancialStability, PastTournaments, Popularity)	OrganizeMatch(BookedSeats , DigitalAdvCost, OutdoorAdvCost, SocialMediaAdvCost, MatchExpenses)			
2	Team (TeamName, GroupName, HeadCount, Allocated Hotel, City)	AccomodateTeams(HospitalityCharges, TransportCost, SecurityCost, MiscCost)			
3	Match, Stadium, Stand (Stand Name, Stadium Name), City	ManageBookings (BasePrice, Tax, SeatCount)			
4	Match, Stadium, City	Security (NumberOfGuards, SurvillanceSystem)			
5	Match, Stadium, City	Match_Attendance(Attendance)			
6	City	FinancialHealth(Budget, Debt)			
4	Match, Stadium, Stand, City	GateCollections(Price, BookedSeat, SeatCount)	TotalGateCollections	Stadium, Stand, City	Per City, 4 years
5	Match, Medium (Medium Type), Region (Region Name)	Viewership (ViewCount)	TotalViews	Medium, Region	Per Region, 4 years Per Medium, 4 years
6	Match, Tournament (TournamentName, StartDate, EndDate, TeamCount, MatchCount, City), Sponsor	Sponsorship (SponsorshipMoney)			Per Tournament, 4 years

	(SponsorName, Contract)				
7	Match, Tournament	Expenditure (PromotionCost, MatchExpenses, AccomodationCost)	TotalExpen tidure	Tournam ent	Per Tournament, 4 years
8	Match, Tournament	Revenue (GateCollections, Viewership, Sponsorship)	TotalReven ue	Tournam ent	Per Tournament, 4 years

d) Implementation of Conversion algorithm - GOM4DW to Star Schema Conversion.

Database connection

Conversion algorithm was implemented in Python and the script gets triggered onclick of "Proceed to OLAP" button on the "createNewInfo" screen on the GUI. Pyodbc object was used to connect to the MSSQL database.

Handling Data Objects

Each data object was stored in a 'fact' list and its associated attributes were stored in "facts_attributes_dict" with key as the 'data object' (fact). If the data object had other data objects as attributes, then the data object was added as a dimension along with its attributes to "dimensions_attributes_dict". For every dimension, an attribute with name "dimension_id" was added to the dimension attributes list which will acts as the primary key for dimension table when the ROLAP schema is built. This is also added to "foreign_key_list" which will be later added to the fact table since this will act as the foreign key constraint link between the fact table and the dimension table.

Handling category objects

Each category associated with the data object that is not present in the list of dimensions, is added to the dimensions list and its associated attributes are linked with the dimension. Also, if the change type of a category attribute is "no_change" and timestamp is NULL, then "timestamp" is added as an attribute to the dimension dict. Each dimension is then linked to the data object which is initially fetched from the database. If there are subcategories then a subcategory dictionary is created with subcategory as key and its attributes as values. This is then later linked with the associated dimension.

Handling the time attributes

All the date/day attributes associated with the "Potato Adhati vyapari" usecase are replaced with a single "Time Dimension" with attribute as (date,day,week,month,quarter,year) since most of the ROLAP operations that occurs with respect to this use case is operated on daily basis to retrieve useful information and maintaining history information in the system.

Output of Conversion Algorithm:

Facts:

Fact Table ['tournament_id', 'match_id', 'sponsorship_id', 'time_id', 'sponsor_id', 'medium_id', 'viewership_id', 'stadium_id', 'city_id', 'region_id', 'stand_id', 'team_id', 'gate_collections_id', 'seat_count', 'digital_adv_cost', 'booked_seats', 'debt', 'view_count', 'transport_cost', 'attendance', 'social_media_adv_cost', 'outdoor_adv_cost', 'viewership', 'survillance_system', 'sponsorship', 'hospitality_charges', 'security_cost', 'tax', 'price', 'budget', 'promotion_cost', 'gate_collections', 'base_price', 'accomodation_cost', 'match_expenses', 'misc_cost', 'number_of_guards', 'sponsorship_money']

Dimensions:

Time Dimension Table ['time_id', 'date', 'day', 'week', 'month', 'quarter', 'year']

team Dimension Table: ['team_id', 'team_name', 'group_name', 'city', 'head_count', 'allocated_hotel']

tournament Dimension Table: ['tournament_id', 'team_count', 'start_date', 'city_id', 'match_count', 'tournament_name', 'end_date']

city Dimension Table: ['city_name', 'past_tournaments', 'financial_stability', 'city_id', 'popularity']

match Dimension Table: ['datetime', 'match_number', 'team_a', 'match_id', 'team_b']

stand Dimension Table: ['stand_name', 'stand_id', 'stadium_name']

stadium Dimension Table: ['capacity', 'stadium_name', 'stadium_id']

gate collections Dimension Table: ['price', 'seat_count', 'gate_collections_id', 'booked_seats']

viewership Dimension Table: ['viewership_id', 'view_count']

sponsorship Dimension Table: ['sponsorship_id', 'sponsorship_money']

sponsor Dimension Table: ['contract', 'sponsor_name', 'sponsor_id']

region Dimension Table: ['region_name', 'region_id']

medium Dimension Table: ['medium_id', 'medium_type']

Create SQLs for FACT and DIMENSION tables:

CREATE TABLE DBO.team_TABLE (team_id int IDENTITY (1,1) PRIMARY KEY,team_name text,group_name text,city text,head_count int,allocated_hotel text);

CREATE TABLE DBO.city_TABLE (city_name text,past_tournaments bit,financial_stability text,city_id int IDENTITY(1,1) PRIMARY KEY,popularity int);

CREATE TABLE DBO.match_TABLE (datetime timestamp,match_number int,team_a text,match_id int IDENTITY(1,1) PRIMARY KEY,team_b text);

CREATE TABLE DBO.stand_TABLE (stand_name text,stand_id int IDENTITY(1,1) PRIMARY KEY,stadium_name text);

CREATE TABLE DBO.stadium_TABLE (capacity int,stadium_name text,stadium_id int IDENTITY(1,1) PRIMARY KEY);

CREATE TABLE DBO.gate_collections_TABLE (price float(4),seat_count int,gate_collections_id int IDENTITY(1,1) PRIMARY KEY,booked_seats int);

CREATE TABLE DBO.viewership_TABLE (viewership_id int IDENTITY(1,1) PRIMARY KEY,view_count int);

CREATE TABLE DBO.sponsorship_TABLE (sponsorship_id int IDENTITY(1,1) PRIMARY KEY,sponsorship_money float(4));

CREATE TABLE DBO.sponsor_TABLE (contract int,sponsor_name text,sponsor_id int IDENTITY(1,1) PRIMARY KEY);

CREATE TABLE DBO.region_TABLE (region_name text,region_id int IDENTITY(1,1) PRIMARY KEY);

CREATE TABLE DBO.medium_TABLE (medium_id int IDENTITY(1,1) PRIMARY KEY,medium_type text);

CREATE TABLE DBO.tournament_TABLE (tournament_id int IDENTITY(1,1) PRIMARY KEY,team_count int,start_date date,city_id int REFERENCES city_TABLE(city_id),match_count int,tournament_name text,end_date date);

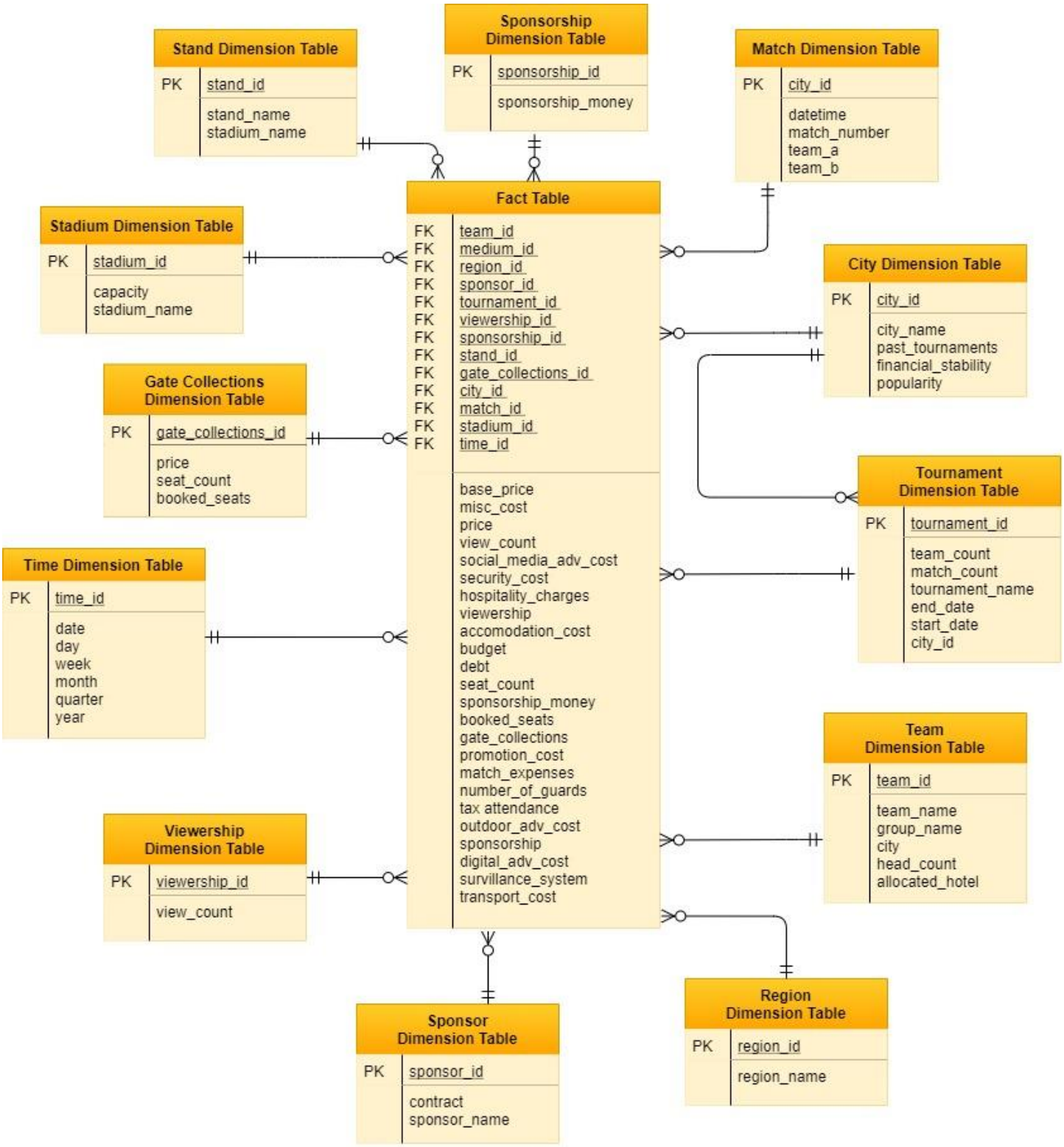
CREATE TABLE DBO.TIME_TABLE (time_id int,date date,day int,week int,month int,quarter int,year int);

CREATE TABLE DBO.FACT_TABLE (tournament_id int REFERENCES tournament_TABLE(tournament_id),match_id int REFERENCES match_TABLE(match_id),sponsorship_id int REFERENCES sponsorship_TABLE(sponsorship_id),time_id int REFERENCES TIME_TABLE(time_id),sponsor_id int REFERENCES sponsor_TABLE(sponsor_id),medium_id int REFERENCES medium_TABLE(medium_id),viewership_id int REFERENCES viewership_TABLE(viewership_id),stadium_id int REFERENCES stadium_TABLE(stadium_id),city_id int REFERENCES city_TABLE(city_id),region_id int REFERENCES region_TABLE(region_id),stand_id int REFERENCES stand_TABLE(stand_id),team_id int REFERENCES team_TABLE(team_id),gate_collections_id int REFERENCES gate_collections_TABLE(gate_collections_id),seat_count int,digital_adv_cost float(4),booked_seats int,debt float(4),view_count int,transport_cost float(4),attendance int,social_media_adv_cost float(4),outdoor_adv_cost float(4),survillance_system text,hospitality_charges float(4),security_cost float(4),tax float(4),price float(4),budget float(4),promotion_cost float(4),base_price float(4),accomodation_cost float(4),match_expenses float(4),misc_cost float(4),number_of_guards int,sponsorship_money float(4));

* Refer to the output.txt file for detailed output

e) Designing the Star Schema

Following is the star schema developed by code:



Type of fact table: Incident fact table - Fact table per match

Type of dimension tables:

DIMENSION	TYPE
Time Dimension Table	Date Dimension
team Dimension Table	Casual Dimension
tournament Dimension Table	Casual Dimension
city Dimension Table	Casual Dimension
match Dimension Table	Casual Dimension
stand Dimension Table	Casual Dimension
stadium Dimension Table	Casual Dimension
gate collections Dimension Table	Casual Dimension
viewership Dimension Table	Casual Dimension
sponsorship Dimension Table	Casual Dimension
sponsor Dimension Table	Casual Dimension
region Dimension Table	Casual Dimension
medium Dimension Table	Casual Dimension

f) Screenshot for tables created in SQL Server:

SQLQuery1.sql - D...CAMJHE\aniku (52))*

```
select * from INFORMATION_SCHEMA.TABLES where TABLE_NAME like '%_TABLE';
```

100 %

Dimension Tables created

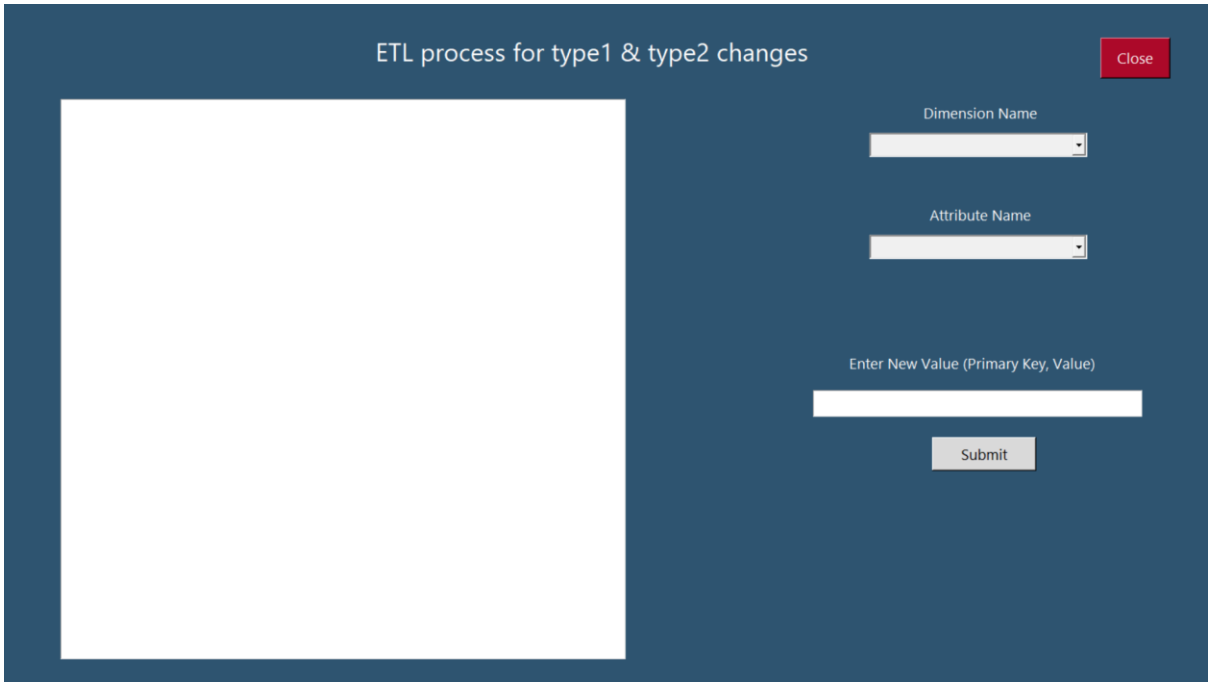
Results | Messages

	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	TABLE_TYPE
1	p4	dbo	team_TABLE	BASE TABLE
2	p4	dbo	match_TABLE	BASE TABLE
3	p4	dbo	city_TABLE	BASE TABLE
4	p4	dbo	stadium_TABLE	BASE TABLE
5	p4	dbo	stand_TABLE	BASE TABLE
6	p4	dbo	viewership_TABLE	BASE TABLE
7	p4	dbo	gate_collections_TABLE	BASE TABLE
8	p4	dbo	sponsorship_TABLE	BASE TABLE
9	p4	dbo	sponsor_TABLE	BASE TABLE
10	p4	dbo	medium_TABLE	BASE TABLE
11	p4	dbo	region_TABLE	BASE TABLE
12	p4	dbo	TIME_TABLE	BASE TABLE
13	p4	dbo	tournament_TABLE	BASE TABLE
14	p4	dbo	FACT_TABLE	BASE TABLE

Fact Table created

g) The ETL process including for Type I and II changes

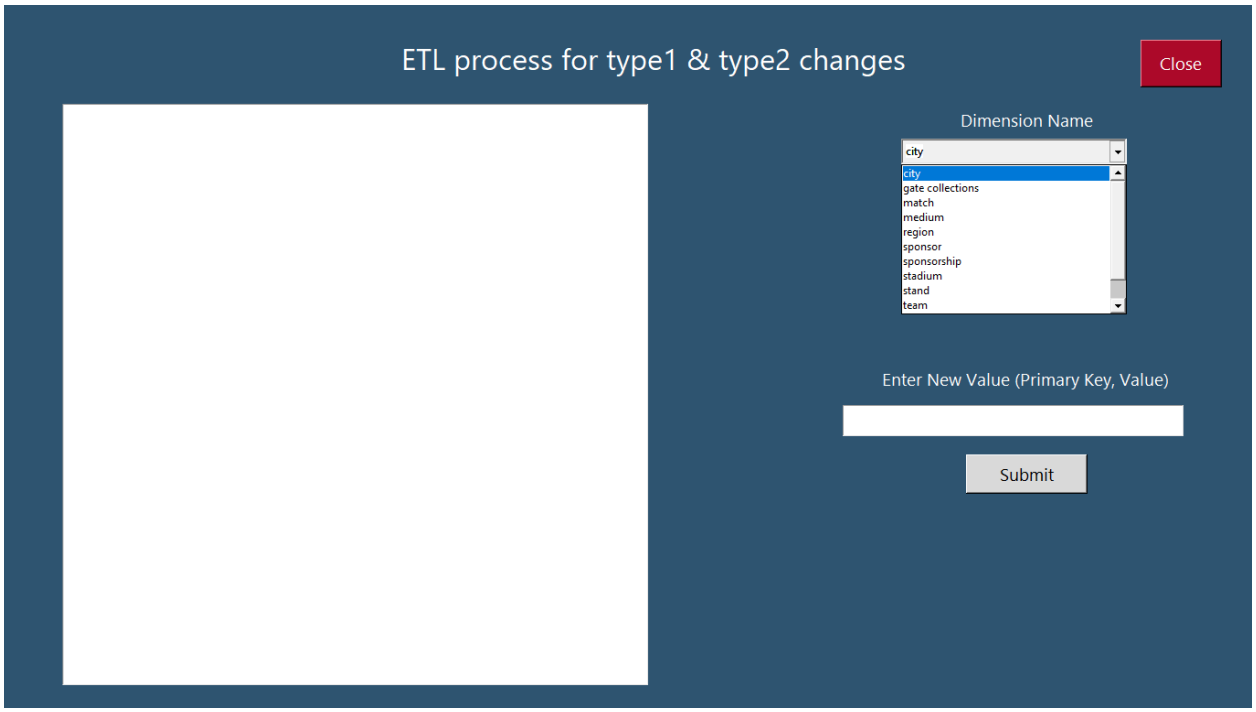
For the ETL (Extract, Transform, Load) process, a GUI (Graphical User Interface) is created for easy and convenient operations. It is created to handle the changes for Type-1 and Type-2 attributes that are present in schema.The GUI is implemented using Python with the help of tkinter library.



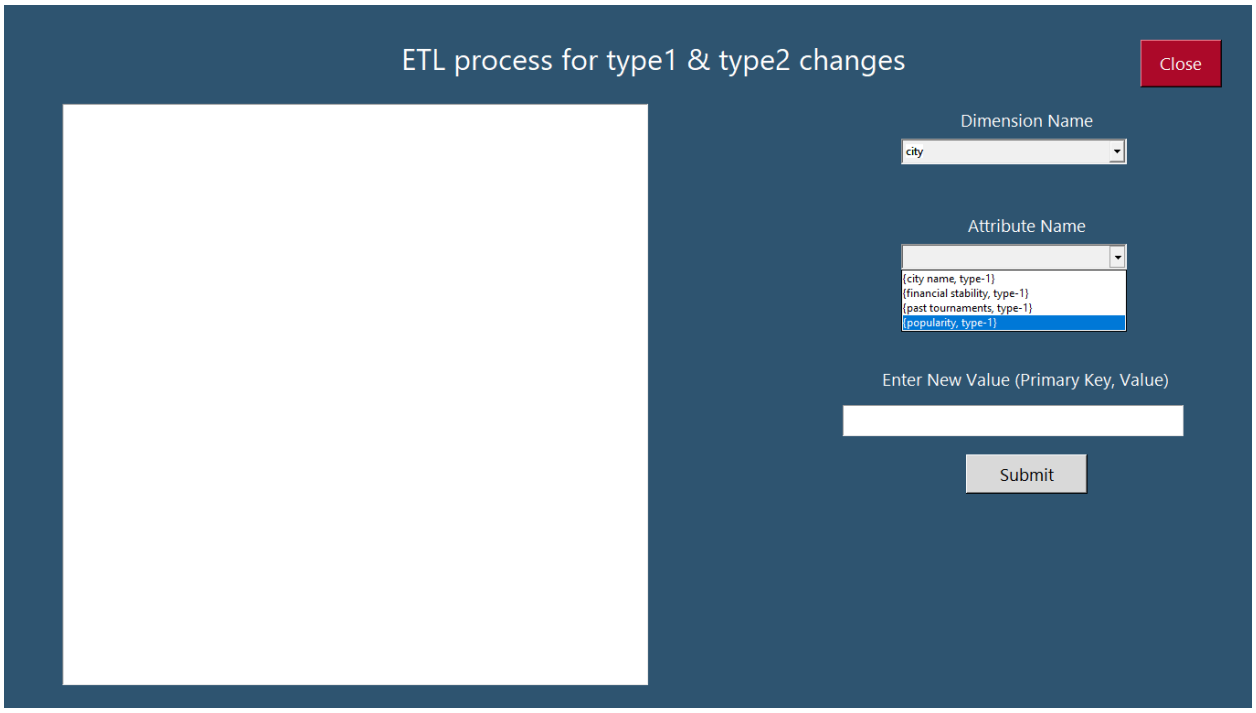
ETL Process GUI

Steps to perform Type-1 and Type-2 changes:

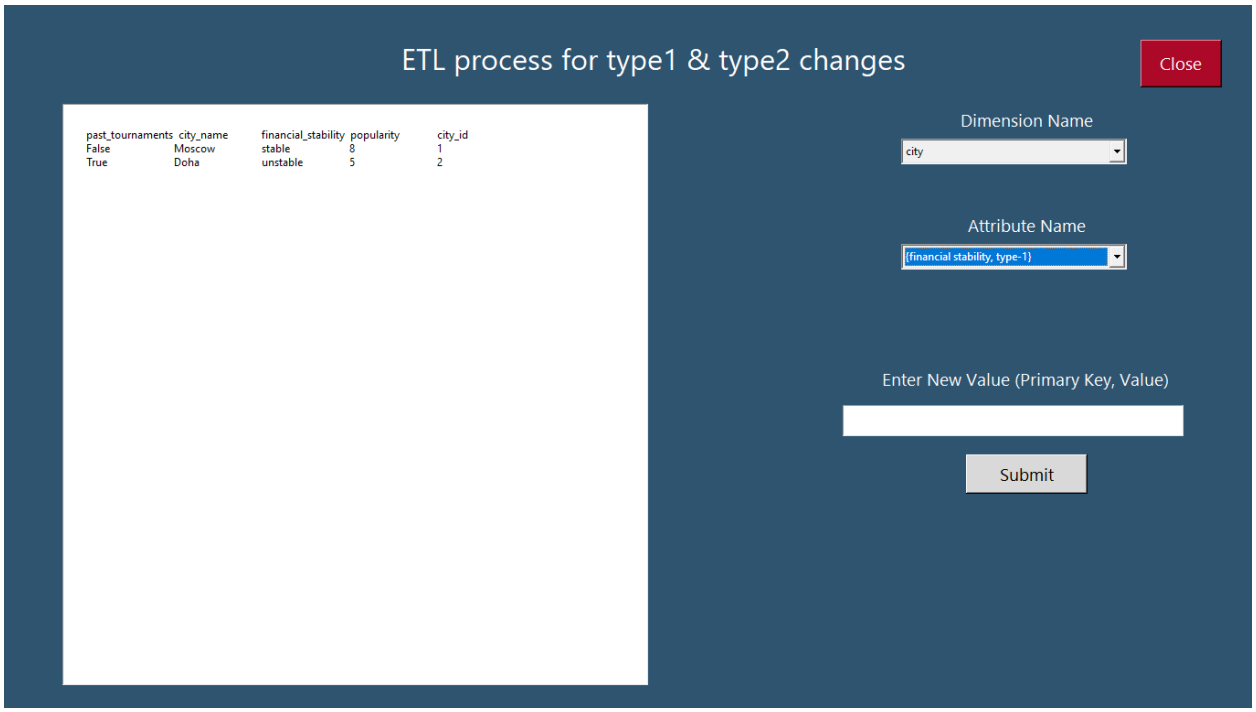
1) Select the dimension from the **Dimension Name** dropdown:



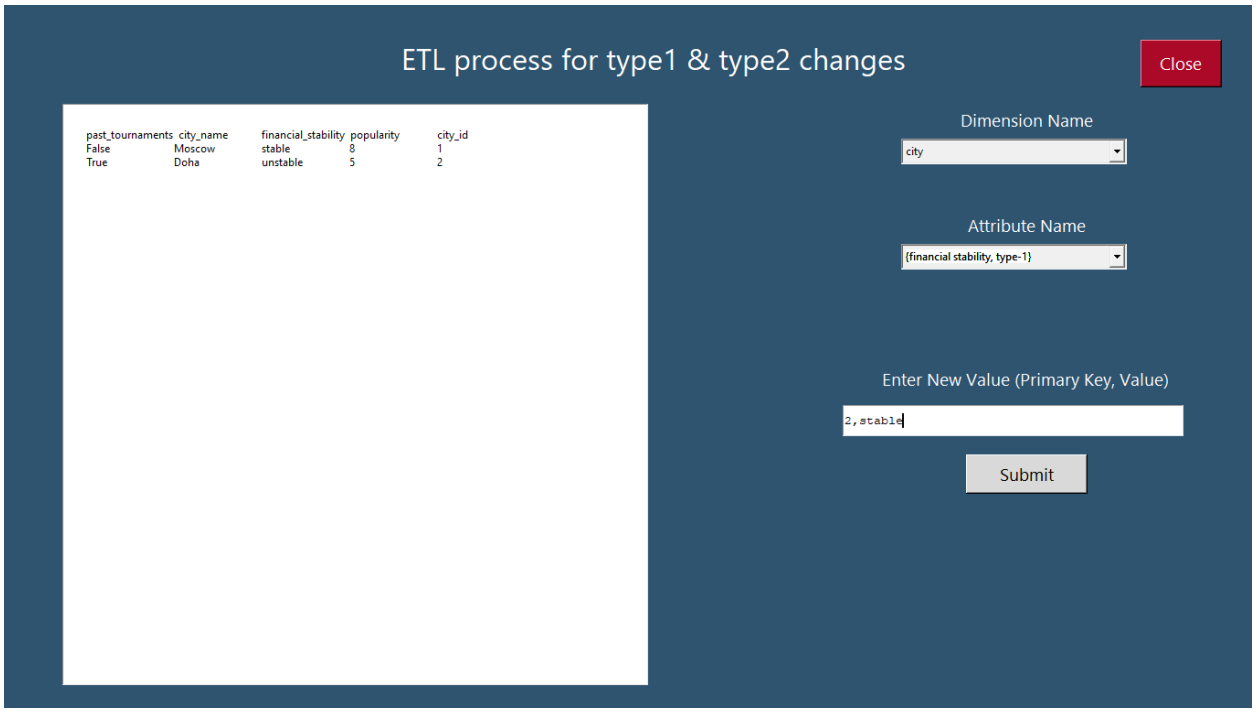
2) Select the attribute of the selected dimension which populates in dropdown **Attribute Name**. It also shows the type of each attribute (Type-1 or Type-2).



3) As soon as attribute is selected as mentioned in Step 2, the dimension table populates at the space in left-hand side of the screen, showing all the contents of the table.

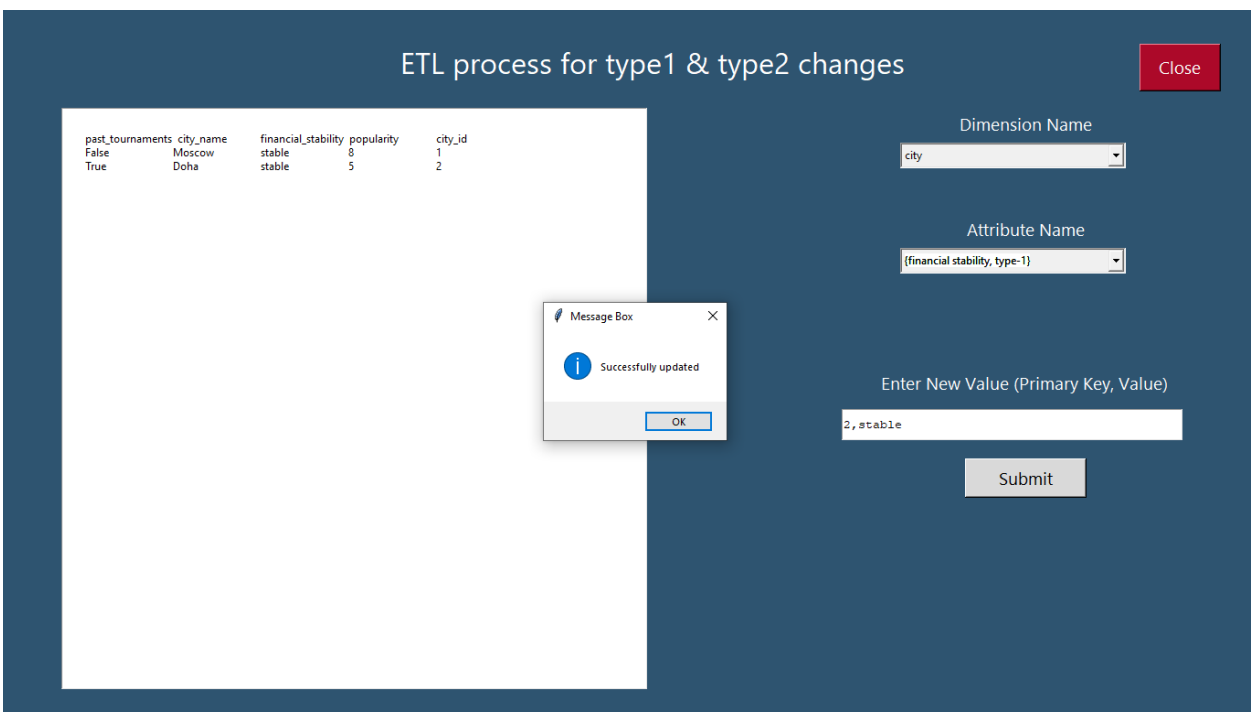


4) User can enter the value that he/she wants to change. Also, system requires the primary key value of the row that user wants to change. Desired format for making an entry is (**id_value, new_value**).



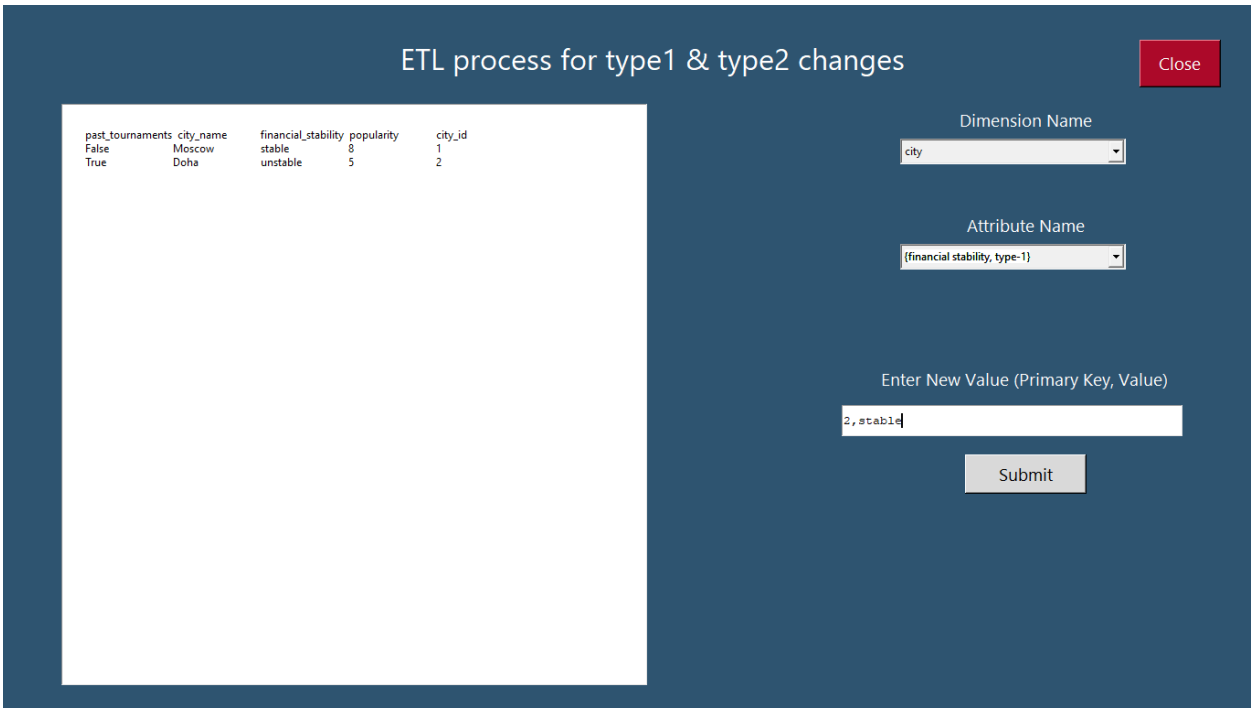
5) Click on **Submit** button to proceed with change. The changes will be shown in the left hand side for verification along with a dialog box, reporting the success message.

Type 1 Change Example – As you can see 'unstable' at row 2 is updated to 'stable'.

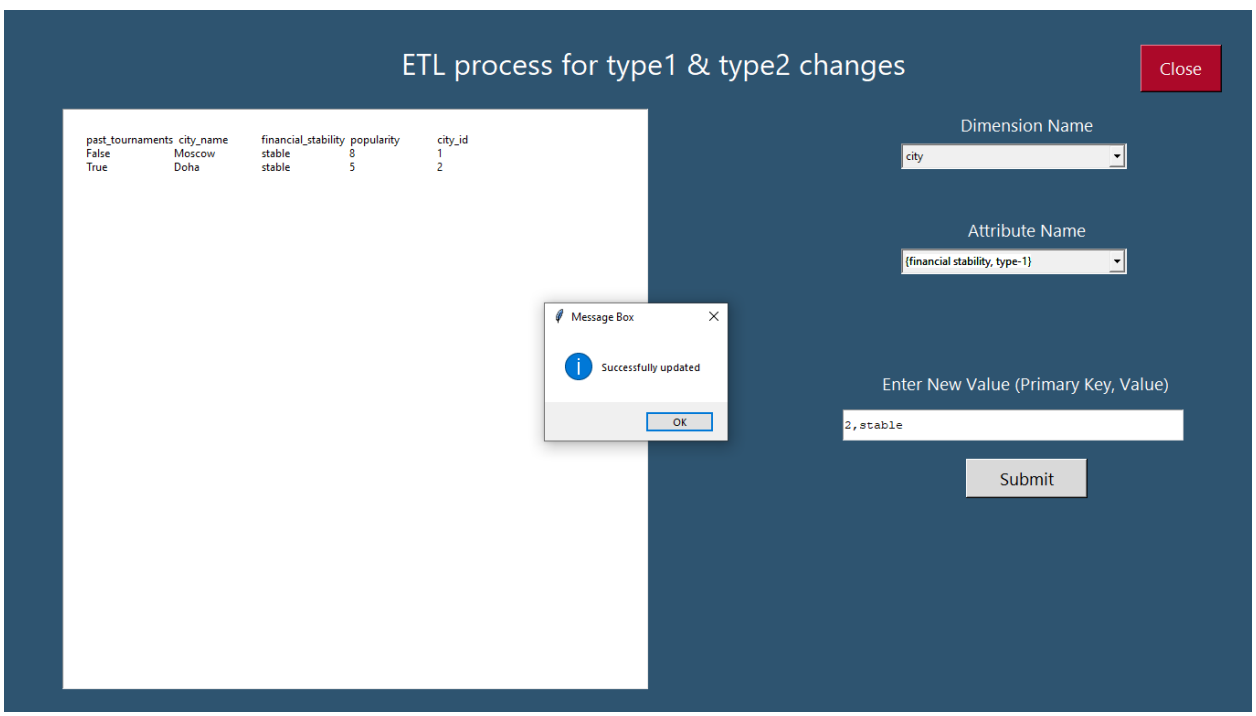


- **Type-1 Change:**

Before making change:



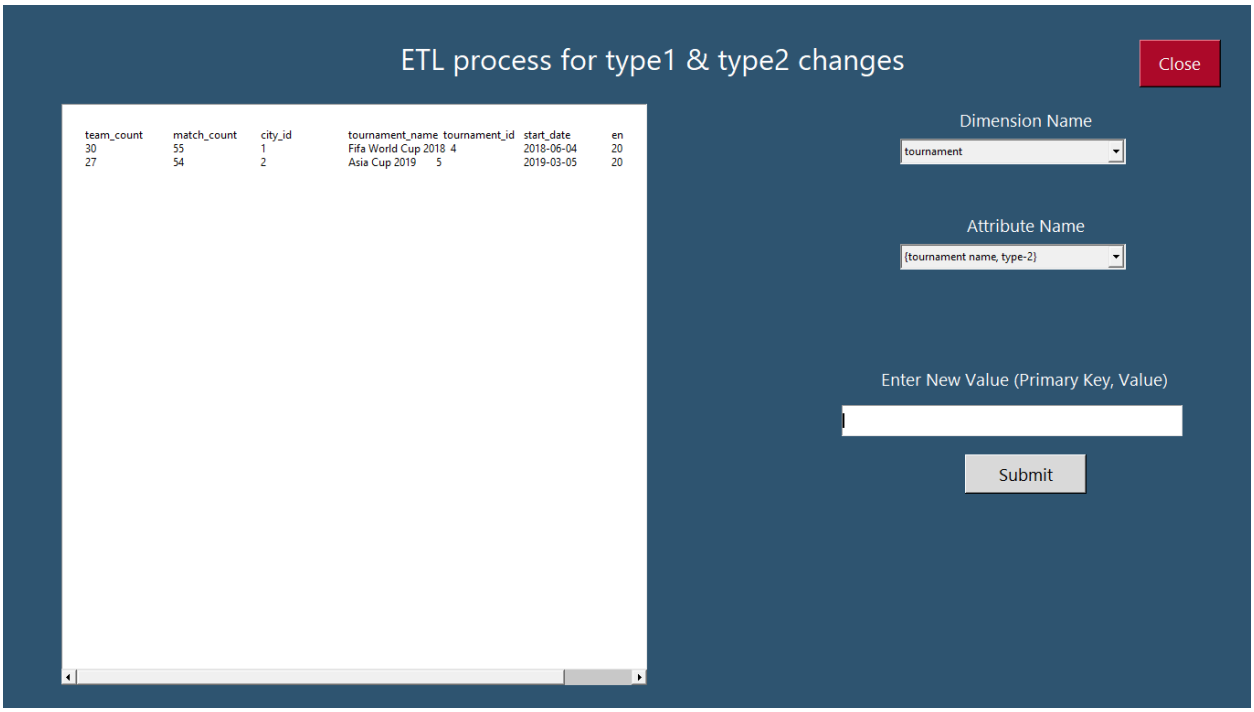
After making change:



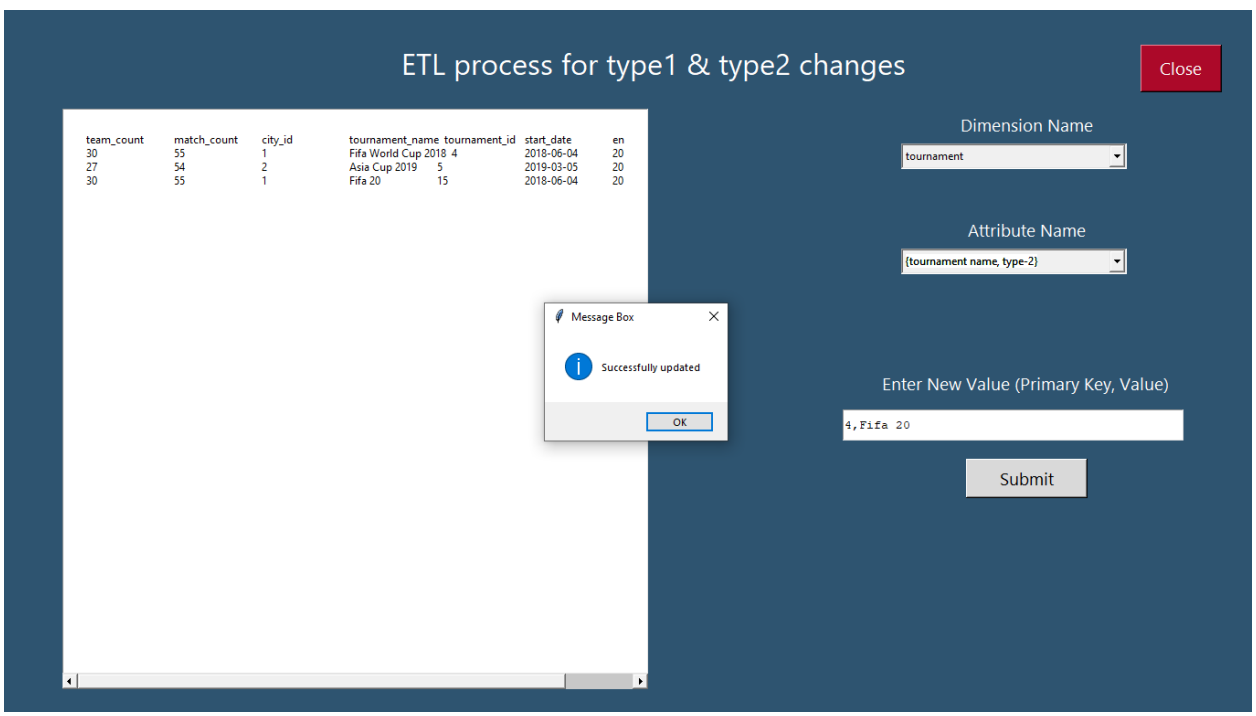
As you can see 'unstable' at row 2 is updated to 'stable'.

- **Type-2 Change:**

Before making change:



After making change:



New entry is added to the table with updated value (Fifa 20)