Design Data Warehouses For Given Below Products:
Note : While designing any Data Warehouse make sure to cover given below points.
a. Design Fact & Dimension tables
b. Create meaningful Primary & Foreign keys
c. Try to follow Star/SnowFlake Schema Design
d. Try to write few SQL queries to generate insightful business metrics (This is the critical point because you need to understand the Data & Business both)
Que1. Design a Data Warehouse for IPL Cricket Tournament

Ans-1-To design a data warehouse for the IPL Cricket Tournament, we need to identify the key entities, their attributes, and the relationships between them. Based on the requirements, we can design the fact and dimension tables, establish primary and foreign keys, and follow a star schema design

VenueDim: This table stores information about the venues where the matches are played.
Attributes:
venue_id (Primary Key): Unique identifier for each venue.
venue_name: Name of the venue.
city: City where the venue is located.
state: State where the venue is located.
country: Country where the venue is located.

TeamDim: This table stores information about the teams participating in the IPL tournament.
Attributes:
team_id (Primary Key): Unique identifier for each team.
team_name: Name of the team.
city: City where the team is based.
state: State where the team is based.
country: Country where the team is based.

Fact Table:
MatchFact: This table represents individual matches played in the IPL tournament and contains metrics related to each match.
Attributes:
match_id (Primary Key): Unique identifier for each match.
date: Date of the match.
venue_id (Foreign Key): Identifier for the match venue.
team1_id (Foreign Key): Identifier for the first team participating in the match.
team2_id (Foreign Key): Identifier for the second team participating in the match.
winner_id (Foreign Key): Identifier for the winning team.
player_of_the_match_id (Foreign Key): Identifier for the player of the match.
runs_scored: Total runs scored in the match.
wickets_taken: Total wickets taken in the match.
extras: Total extra runs in the match (like wides, no balls).
result: Result of the match (e.g., win, tie, abandoned).

PlayerDim: This table stores information about the players participating in the IPL tournament.
Attributes:
player_id (Primary Key): Unique identifier for each player.
player_name: Name of the player.
team_id (Foreign Key): Identifier for the team the player belongs to.
age: Age of the playe

SQL Queries for Business Metrics:

Total matches played by a team:

sql

Copy code

```
SELECT team_name, COUNT(*) AS total_matches
FROM MatchFact
JOIN TeamDim ON MatchFact.team1_id = TeamDim.team_id OR MatchFact.team2_id = TeamDim.team_id
GROUP BY team_name;
```

Total runs scored by a team in a given season:

sql

Copy code

```
SELECT team_name, SUM(runs_scored) AS total_runs
FROM MatchFact
JOIN TeamDim ON MatchFact.team1_id = TeamDim.team_id OR MatchFact.team2_id = TeamDim.team_id
WHERE season = '2023'
GROUP BY team_name;
```

Player with the most player-of-the-match awards:

```
SELECT player_name, COUNT(*) AS total_awards
FROM MatchFact
JOIN PlayerDim ON MatchFact.player_of_the_match_id = PlayerDim.player_id
GROUP BY player_name
ORDER BY total_awards DESC
LIMIT 1;
```

Average runs scored per match by a team in a given season:

```
SELECT team_name, AVG(runs_scored) AS average_runs
FROM MatchFact
JOIN TeamDim ON MatchFact.team1_id = TeamDim.team_id OR MatchFact.team2_id = TeamDim.team_id
WHERE season = '2023'
GROUP BY team_name;
```

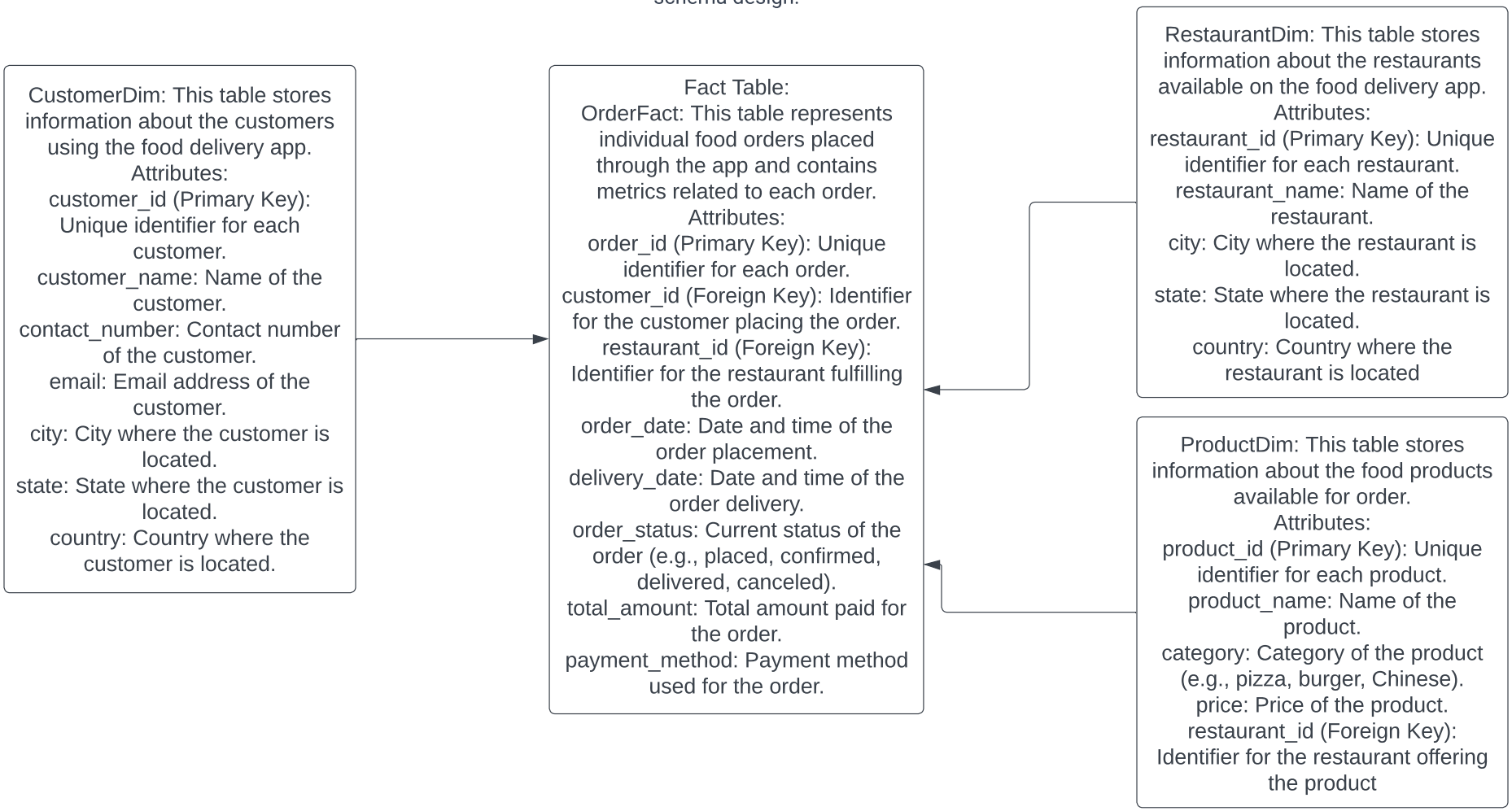
Matches won by a team:

```
SELECT team_name, COUNT(*) AS total_wins
FROM MatchFact
JOIN TeamDim ON MatchFact.winner_id = TeamDim.team_id
GROUP BY team_name;
```

These queries provide insights into various metrics, such as total matches played, total runs scored, player performance, average runs, and total wins, which can be used to analyze the IPL Cricket Tournament data.

Design Data Warehouses For Given Below Products:
Note : While designing any Data Warehouse make sure to cover given below points.
a. Design Fact & Dimension tables
b. Create meaningful Primary & Foreign keys
c. Try to follow Star/SnowFlake Schema Design
d. Try to write few SQL queries to generate insightful business metrics (This is the critical point because you need to understand the Data & Business both).
Que2.Design a Data Warehouse for Food delivery app like Swiggy, Zomato

Ans-2To design a data warehouse for a food delivery app like Swiggy or Zomato, we need to identify the key entities, their attributes, and the relationships between them. Based on the requirements, we can design the fact and dimension tables, establish primary and foreign keys, and follow a star schema design.



SQL Queries for Business Metrics:

Total revenue generated by the food delivery app:

```
SELECT SUM(total_amount) AS total_revenue
FROM OrderFact;
```

Top-selling products by quantity:

```
SELECT product_name, SUM(quantity) AS total_quantity
FROM OrderFact
JOIN OrderItemFact ON OrderFact.order_id = OrderItemFact.order_id
JOIN ProductDim ON OrderItemFact.product_id = ProductDim.product_id
GROUP BY product_name
ORDER BY total_quantity DESC
LIMIT 5;
```

Total orders placed by a customer:

```
SELECT customer_name, COUNT(*) AS total_orders
FROM OrderFact
JOIN CustomerDim ON OrderFact.customer_id = CustomerDim.customer_id
GROUP BY customer_name;
```

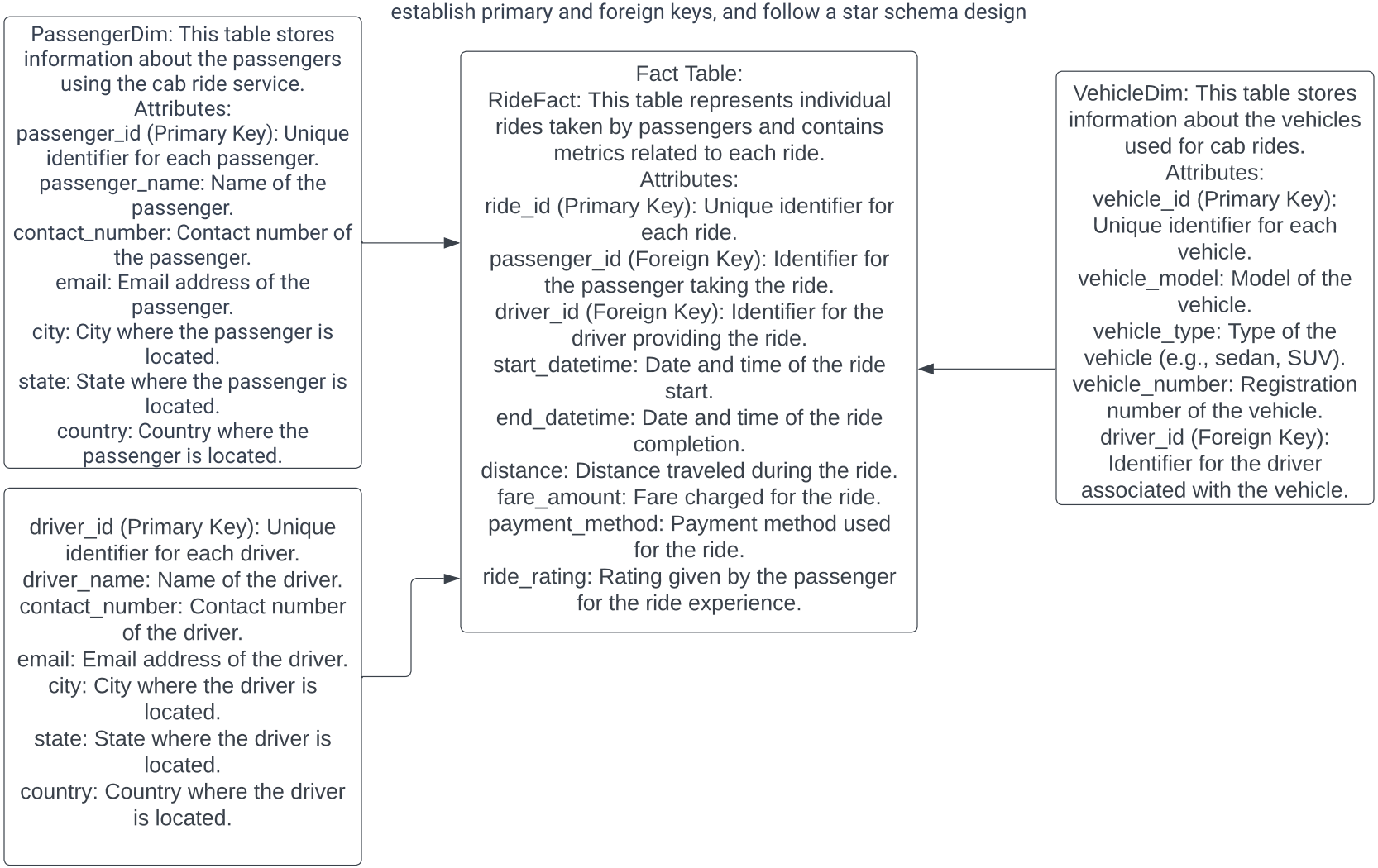
Average order value by city:

```
SELECT city, AVG(total_amount) AS average_order_value
FROM OrderFact
JOIN CustomerDim ON OrderFact.customer_id = CustomerDim.customer_id
GROUP BY city;
```

These queries provide insights into various metrics, such as total revenue, top-selling products, customer behavior, and average order value, which can be used to analyze the performance of the food delivery app.

Design Data Warehouses For Given Below Products:
Note : While designing any Data Warehouse make sure to cover given below points.
a. Design Fact & Dimension tables
b. Create meaningful Primary & Foreign keys
c. Try to follow Star/SnowFlake Schema Design
d. Try to write few SQL queries to generate insightful business metrics (This is the critical point because you need to understand the Data & Business both).
Que3-Design a Data Warehouse for cab ride service like Uber, Lyft

Ans3-To design a data warehouse for a cab ride service like Uber or Lyft, we need to identify the key entities, their attributes, and the relationships between them. Based on the requirements, we can design the fact and dimension tables, establish primary and foreign keys, and follow a star schema design



SQL Queries for Business Metrics:

Total rides completed:

```
SELECT COUNT(*) AS total_rides
FROM RideFact;
```

Average fare amount per ride:

```
SELECT AVG(fare_amount) AS average_fare_amount
FROM RideFact;
```

Top-rated drivers by average rating:

```
SELECT driver_name, AVG(ride_rating) AS average_rating
FROM RideFact
JOIN DriverDim ON RideFact.driver_id = DriverDim.driver_id
GROUP BY driver_name
ORDER BY average_rating DESC
LIMIT 5;
```

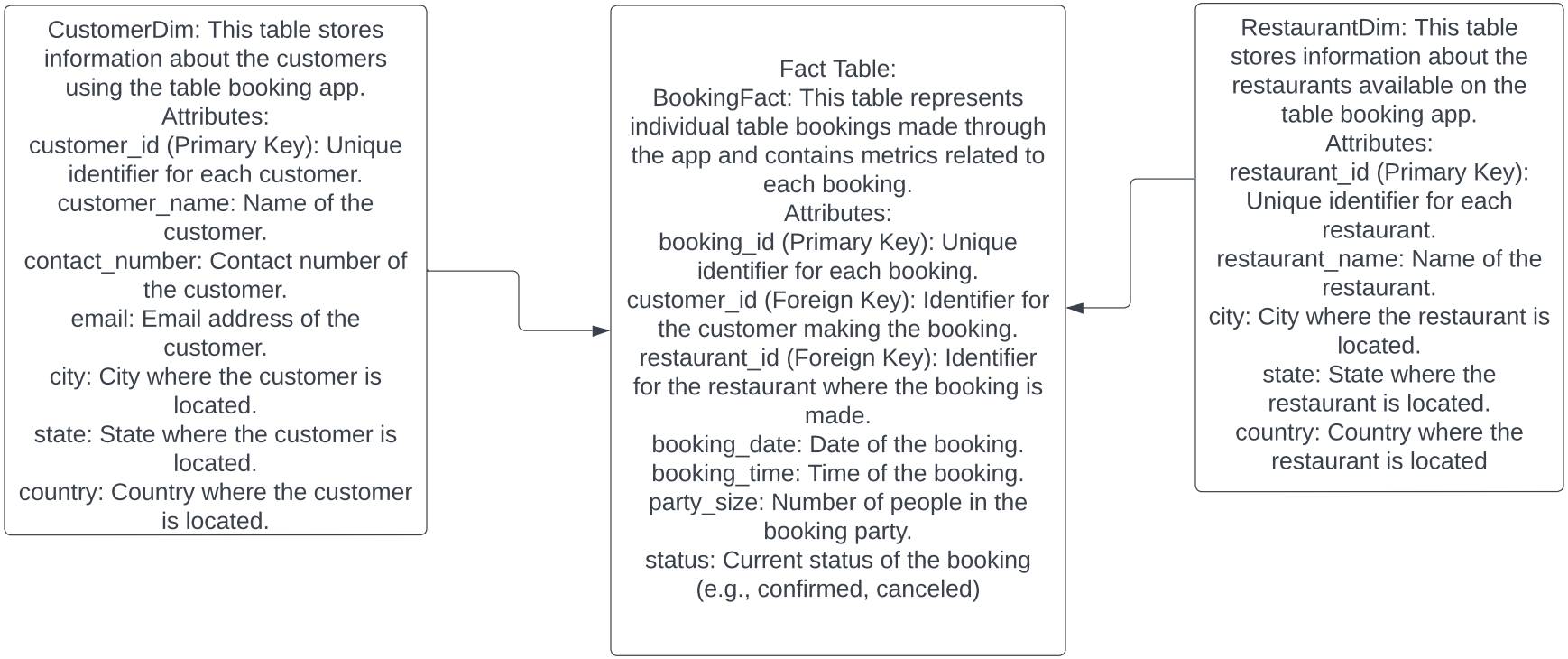
Total rides taken by a passenger:

```
SELECT passenger_name, COUNT(*) AS total_rides
FROM RideFact
JOIN PassengerDim ON RideFact.passenger_id = PassengerDim.passenger_id
GROUP BY passenger_name;
```

These queries provide insights into various metrics, such as total rides, average fare amount, driver performance, and passenger behavior, which can be used to analyze the cab ride service and make data-driven decisions.

Design Data Warehouses For Given Below Products:
Note : While designing any Data Warehouse make sure to cover given below points.
a. Design Fact & Dimension tables
b. Create meaningful Primary & Foreign keys
c. Try to follow Star/Snowflake Schema Design
d. Try to write few SQL queries to generate insightful business metrics (This is the critical point because you need to understand the Data & Business both)
Que4- Design a Data Warehouse for Restaurant table booking app like Dine out

Ans4-To design a data warehouse for a restaurant table booking app like Dine out, we need to identify the key entities, their attributes, and the relationships between them. Based on the requirements, we can design the fact and dimension tables, establish primary and foreign keys, and follow a star schema design.



SQL Queries for Business Metrics:

Total bookings made:

```
SELECT COUNT(*) AS total_bookings
FROM BookingFact;
```

Popular restaurants by total bookings:

```
SELECT restaurant_name, COUNT(*) AS total_bookings
FROM BookingFact
JOIN RestaurantDim ON BookingFact.restaurant_id = RestaurantDim.restaurant_id
GROUP BY restaurant_name
ORDER BY total_bookings DESC
LIMIT 5;
```

Total bookings by customers

```
SELECT customer_name, COUNT(*) AS total_bookings
FROM BookingFact
JOIN CustomerDim ON BookingFact.customer_id = CustomerDim.customer_id
GROUP BY customer_name;
```

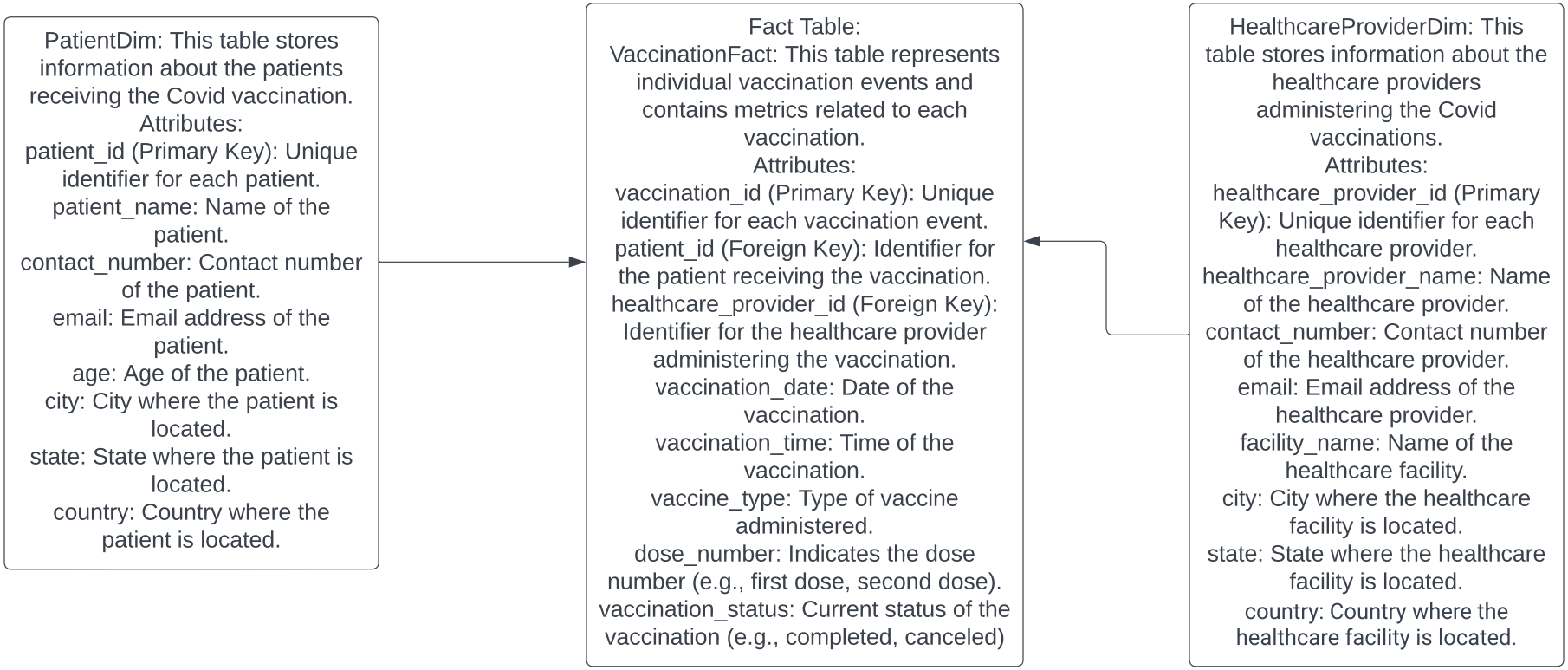
Average party size for bookings:

```
SELECT AVG(party_size) AS average_party_size
FROM BookingFact;
```

These queries provide insights into various metrics, such as total bookings, popular restaurants, customer behavior, and average party size, which can be used to analyze the performance of the table booking app and improve the user experience.

Design Data Warehouses For Given Below Products:
Note : While designing any Data Warehouse make sure to cover given below points.
a. Design Fact & Dimension tables
b. Create meaningful Primary & Foreign keys
c. Try to follow star/Snowflake Schema Design
d. Try to write few SQL queries to generate insightful business metrics (This is the critical point because you need to understand the Data & Business both)
Que5-Design a Data Warehouse for Covid Vaccination Application

Ans-5 To design a data warehouse for a Covid Vaccination Application, we need to identify the key entities, their attributes, and the relationships between them. Based on the requirements, we can design the fact and dimension tables, establish primary and foreign keys, and follow a star schema design. Here's a suggested design:



SQL Queries for Business Metrics:

Total vaccinations administered:

```
SELECT COUNT(*) AS total_vaccinations
FROM VaccinationFact;
```

Popular vaccine types by total vaccinations:

```
SELECT vaccine_type, COUNT(*) AS total_vaccinations
FROM VaccinationFact
GROUP BY vaccine_type
ORDER BY total_vaccinations DESC
LIMIT 5;
```

Total vaccinations by patient:

```
SELECT patient_name, COUNT(*) AS total_vaccinations
FROM VaccinationFact
JOIN PatientDim ON VaccinationFact.patient_id = PatientDim.patient_id
GROUP BY patient_name;
```

Vaccination completion rate by healthcare provider:

```
SELECT healthcare_provider_name,
COUNT(CASE WHEN vaccination_status = 'completed' THEN 1 END) AS completed_vaccinations,
COUNT(*) AS total_vaccinations
FROM VaccinationFact
JOIN HealthcareProviderDim ON VaccinationFact.healthcare_provider_id = HealthcareProviderDim.healthcare_provider_id
GROUP BY healthcare_provider_name;
```

These queries provide insights into various metrics, such as total vaccinations, popular vaccine types, patient vaccination statistics, and completion rates by healthcare providers, which can be used to monitor the progress and effectiveness of the Covid vaccination program.