



# **Airline Data Management and Analysis Using Power BI**

**BY- BULBUL SINGH**

## Table of Contents

S. No.	Section Title
1	Introduction
2	Task 1: Data Cleaning and Preparation
3	Task 2: Data Modeling
4	Task 3: Enhanced Data Insights
5	Task 4: Calculations Using DAX
6	Task 5: Visualization and Interactive Features
7	Task 6: Final Dashboard and Power BI Service
8	Conclusion

**Video explanation link-** [https://drive.google.com/file/d/1OljztVJh01-NEqXzcJ4\\_-Y74DYhyZk4G/view?usp=sharing](https://drive.google.com/file/d/1OljztVJh01-NEqXzcJ4_-Y74DYhyZk4G/view?usp=sharing)

## Introduction

This project focuses on analyzing airline operational data using Power BI. The goal is to transform raw data from three different sources — Flight\_Information, Ticket\_Information, and Passenger\_Information — into meaningful insights through data cleaning, modeling, DAX calculations, and interactive dashboards.

By applying various Power BI features such as relationships, conditional columns, measures, slicers, and drillthrough pages, the report delivers a complete view of flight performance, passenger trends, and ticket booking status. The final dashboard provides a powerful tool for decision-making, with enhanced visuals and automated data refresh for real-time insights.

**Problem Statement:** The airline industry operates with numerous complexities, requiring effective data management and insights into flight schedules, passenger details, and ticketing systems. This project aims to analyze airline operations for improving efficiency and customer satisfaction.

## Task 1: Data Cleaning and Preparation

In this task, I cleaned and prepared all three datasets — Flight\_Information, Ticket\_Information, and Passenger\_Information — using **Power BI's Power Query Editor**.

I checked each sheet individually for appropriate data types (text, integers, and dates) and looked for any irregular formatting, such as leading or trailing spaces. This stage aided in getting the dataset ready for precise modelling and computations in the future.

Table: RemoveColumns(#"Removed Duplicates",{"Column6", "Column7", "Column8", "Column9", "Column10", "Column11", "Column12", "Column13",

FlightID	FlightNumber	Airline	Destination	Status
1	1001 FL1102	Airline D	Houston	On Time
2	1002 FL1435	Airline B	Chicago	On Time
3	1003 FL1860	Airline A	New York	Cancelled
4	1004 FL1270	Airline C	Chicago	Delayed
5	1005 FL1106	Airline C	New York	Delayed
6	1006 FL1071	Airline A	Phoenix	On Time
7	1007 FL1700	Airline C	Los Angeles	Cancelled
8	1008 FL1020	Airline C	Los Angeles	Delayed
9	1009 FL1614	Airline A	Los Angeles	Cancelled
10	1010 FL1121	Airline D	Chicago	Cancelled
11	1011 FL1466	Airline A	Phoenix	On Time
12	1012 FL1214	Airline D	New York	Delayed
13	1013 FL1330	Airline C	Houston	On Time
14	1014 FL1458	Airline C	New York	Delayed
15	1015 FL1087	Airline C	Houston	Delayed
16	1016 FL1372	Airline B	New York	Delayed
17	1017 FL1099	Airline D	Phoenix	Delayed
18	1018 FL1871	Airline B	Houston	Delayed
19	1019 FL1663	Airline B	Chicago	Cancelled
20	1020 FL1130	Airline A	New York	On Time
21	1021 FL1661	Airline B	New York	Cancelled
22	1022 FL1308	Airline A	Houston	Delayed
23	1023 FL1769	Airline A	Chicago	On Time
24	1024 FL1343	Airline B	Chicago	Delayed
25	1025 FL1491	Airline D	Phoenix	On Time
26	1026 FL1413	Airline D	Chicago	Cancelled
27	1027 FL1805	Airline D	Chicago	On Time
28	1028 FL1385	Airline D	Chicago	On Time
29	1029 FL1191	Airline D	Los Angeles	On Time
30	1030 FL1955	Airline B	Phoenix	On Time

5 COLUMNS, 200 ROWS Column profiling based on top 1000 rows

PREVIEW DOWNLOADED AT 16:37

*In the Flight Information sheet, Many column contained only null values. Since it had no usable data, I deleted the entire column to maintain a clean and relevant dataset. I also ensured that the columns such as FlightID, Airline, and Status were in the correct data types.*

Query: Table.RemoveColumns(#"Changed Type",{"Column4", "Column5", "Column6", "Column7", "Column8", "Column9", "Column10", "Column11",

	PassengerID	FlightID	SeatNumber
1	1	1161	38A
2	2	1157	24D
3	3	1141	30B
4	4	1046	17E
5	5	1035	29D
6	6	1134	10A
7	7	1082	10A
8	8	1115	20E
9	9	1197	34E
10	10	1047	2E
11	11	1153	43C
12	12	1194	48C
13	13	1010	47A
14	14	1056	23C
15	15	1030	16D
16	16	1109	40D
17	17	1005	25C
18	18	1119	32C
19	19	1033	27E
20	20	1118	32B
21	21	1065	19E
22	22	1146	5B
23	23	1177	28B
24	24	1011	22E
25	25	1085	6A
26	26	1026	5A
27	27	1063	12B
28	28	1086	46B
29	29	1059	49B
30	30	1027	45C

3 COLUMNS, 100 ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 16:37

The Passenger\_Information sheet also had a column filled with nulls, which I removed to streamline the dataset. I also confirmed that each column had the correct data type (e.g., PassengerID as whole number and Name as text) and no extra whitespace or format issues.

Query: Table.RemoveColumns(#"Changed Type",{"Column4", "Column5", "Column6", "Column7", "Column8", "Column9", "Column10", "Column11",

	TicketID	FlightID	BookingStatus
1	5001	1178	Pending
2	5002	1078	Confirmed
3	5003	1117	Cancelled
4	5004	1120	Cancelled
5	5005	1137	Cancelled
6	5006	1162	Pending
7	5007	1076	Pending
8	5008	1035	Cancelled
9	5009	1001	Cancelled
10	5010	1040	Cancelled
11	5011	1064	Pending
12	5012	1150	Cancelled
13	5013	1060	Cancelled
14	5014	1064	Confirmed
15	5015	1093	Confirmed
16	5016	1072	Pending
17	5017	1011	Cancelled
18	5018	1105	Cancelled
19	5019	1014	Confirmed
20	5020	1060	Pending
21	5021	1030	Confirmed
22	5022	1035	Confirmed
23	5023	1165	Confirmed
24	5024	1005	Confirmed
25	5025	1083	Cancelled
26	5026	1123	Cancelled
27	5027	1078	Confirmed
28	5028	1154	Pending
29	5029	1062	Pending
30	5030	1132	Pending

3 COLUMNS, 50 ROWS Column profiling based on top 1000 rows PREVIEW DOWNLOADED AT 16:37

This screenshot shows the Ticket\_Information table where I identified and deleted a column containing only null values. Additionally, I ensured proper formatting of columns like TicketID, FlightID, and Status, and replaced nulls in numeric fields with suitable placeholders where needed.

## Task 2: Data Modeling

For the data model, I created relationships between the three main tables: Flight\_Information, Passenger\_Information, and Ticket\_Information. The common key used was FlightID, which allowed me to establish proper one-to-many relationships.

In Power BI, **cardinality** refers to the relationship between tables based on how many matching rows exist in each table. There are three main types of cardinality:

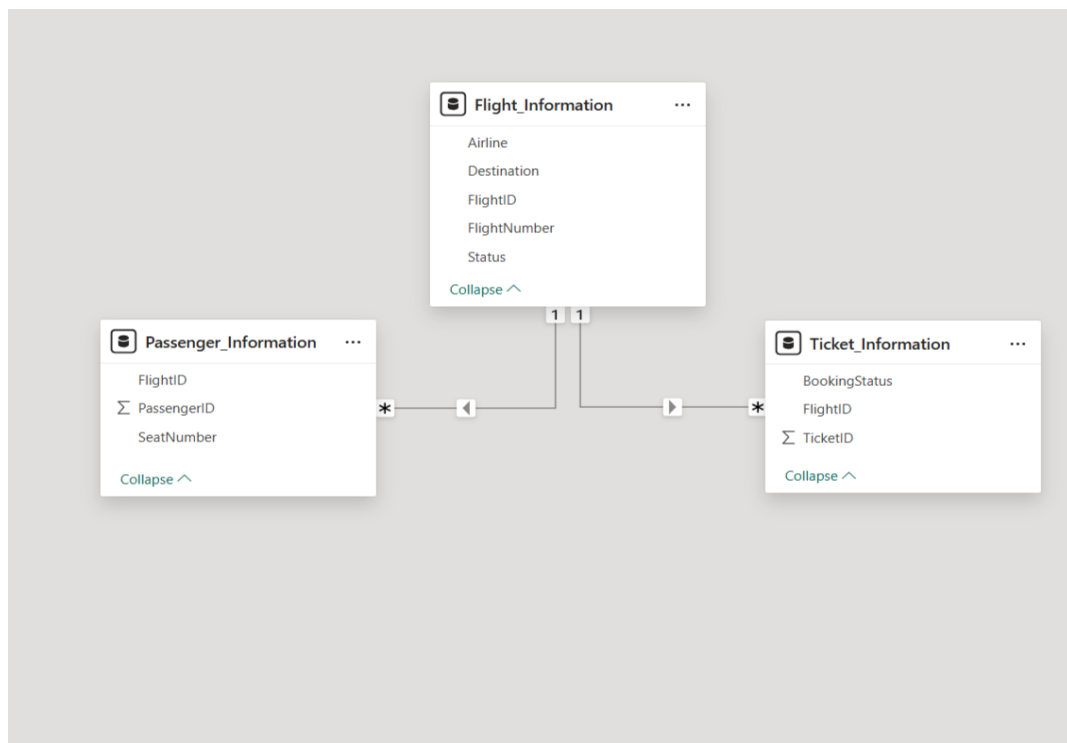
- **One-to-many (1:\*)**
- **Many-to-one (\*:1)**
- **Many-to-many (:)**

In my project, I understood how the datasets relate to one another and used **One-to-Many (1:\*) relationships** for proper data modeling. Here's how I applied cardinality:

- One flight in the Flight\_Information table can have **many passengers** (in Passenger\_Information) and **many tickets** (in Ticket\_Information).
- Therefore, I made Flight\_Information[FlightID] the **"One"** side, and both Ticket\_Information[FlightID] and Passenger\_Information[FlightID] the **"Many"** side.

I ensured that the **relationship direction** was configured properly (usually single-direction filtering from the "One" to "Many" table), which allowed the filters to work correctly in visuals and calculations.

This step was crucial for creating accurate insights in my dashboard, such as calculating passenger counts by airline or filtering ticket data based on flight status.



*This diagram shows the relationships created between the tables using the FlightID key. It also displays the cardinality (one-to-many) and direction of relationships.*

## Task 3: Enhanced Data Insights

I created a conditional column that classified flights as either “Best” or “To Be Improved”, based on the Status field in the Flight\_Information table. If the flight status was “On Time”, it was marked as "Best", otherwise "To Be Improved".

Additionally, I used the Column From Example feature in Power Query to extract the flight number from a compound FlightNumber field. Instead of splitting by delimiter, I typed examples like "A123" and Power BI automatically detected the pattern, helping extract the flight number part efficiently.

	FlightNumber	Airline	Destination	Status	Flight_status_Review	Flight_Number
1	1001 FL1102	Airline D	Houston	On Time	Best	1102
2	1002 FL1435	Airline B	Chicago	On Time	Best	1435
3	1003 FL1860	Airline A	New York	Cancelled	To Be Improved	1860
4	1004 FL1270	Airline C	Chicago	Delayed	To Be Improved	1270
5	1005 FL1106	Airline C	New York	Delayed	To Be Improved	1106
6	1006 FL1071	Airline A	Phoenix	On Time	Best	1071
7	1007 FL1700	Airline C	Los Angeles	Cancelled	To Be Improved	1700
8	1008 FL1020	Airline C	Los Angeles	Delayed	To Be Improved	1020
9	1009 FL1614	Airline A	Los Angeles	Cancelled	To Be Improved	1614
10	1010 FL1121	Airline D	Chicago	Cancelled	To Be Improved	1121
11	1011 FL1466	Airline A	Phoenix	On Time	Best	1466
12	1012 FL1214	Airline D	New York	Delayed	To Be Improved	1214
13	1013 FL1330	Airline C	Houston	On Time	Best	1330
14	1014 FL1458	Airline C	New York	Delayed	To Be Improved	1458
15	1015 FL1087	Airline C	Houston	Delayed	To Be Improved	1087
16	1016 FL1372	Airline B	New York	Delayed	To Be Improved	1372
17	1017 FL1099	Airline D	Phoenix	Delayed	To Be Improved	1099
18	1018 FL1871	Airline B	Houston	Delayed	To Be Improved	1871
19	1019 FL1663	Airline B	Chicago	Cancelled	To Be Improved	1663
20	1020 FL1130	Airline A	New York	On Time	Best	1130
21	1021 FL1661	Airline B	New York	Cancelled	To Be Improved	1661
22	1022 FL1308	Airline A	Houston	Delayed	To Be Improved	1308
23	1023 FL1769	Airline A	Chicago	On Time	Best	1769
24	1024 FL1343	Airline B	Chicago	Delayed	To Be Improved	1343
25	1025 FL1491	Airline D	Phoenix	On Time	Best	1491
26	1026 FL1413	Airline D	Chicago	Cancelled	To Be Improved	1413
27	1027 FL1805	Airline D	Chicago	On Time	Best	1805
28	1028 FL1385	Airline D	Chicago	On Time	Best	1385
29	1029 FL1191	Airline D	Los Angeles	On Time	Best	1191

Here, I created a conditional column that classifies each flight as either ‘Best’ or ‘To Be Improved’ based on the flight’s status. This helps quickly filter high-performing flights.

This screenshot also shows the Column From Example feature in Power Query used to extract just the flight number portion from the full FlightNumber string.

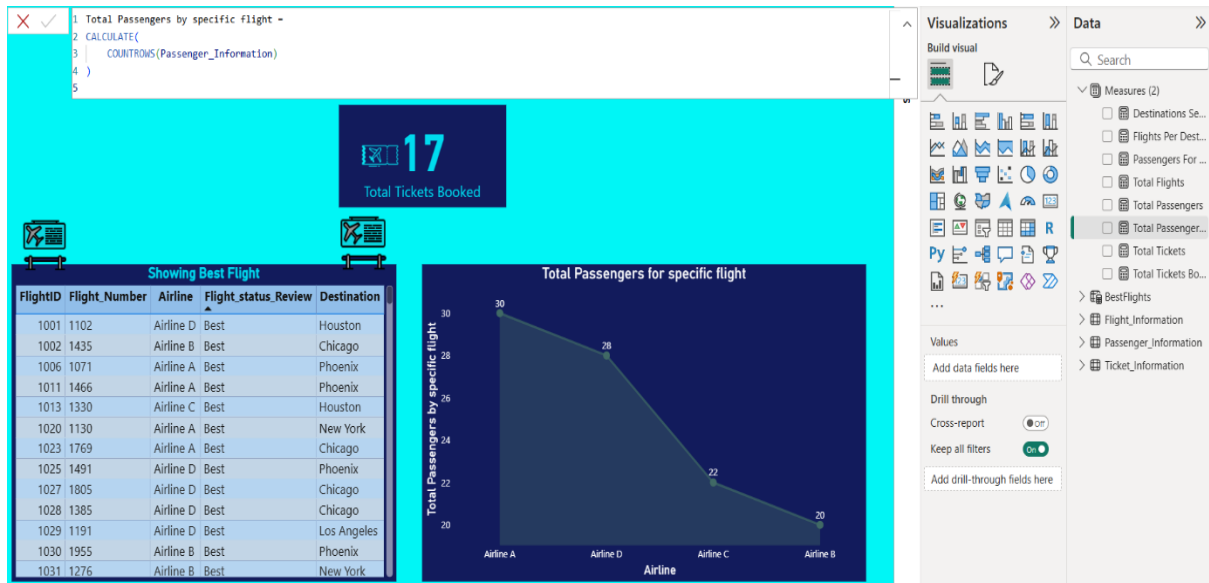
## Task 4: Calculations Using DAX

In this task, I used DAX to create calculated measures:

- Total passengers for a specific flight: Counted the number of unique passengers per flight
- Total tickets booked: Counted ticket records from the Ticket\_Information table
- Filtered table: Created a visual/table showing only the "Best" flights based on the conditional column

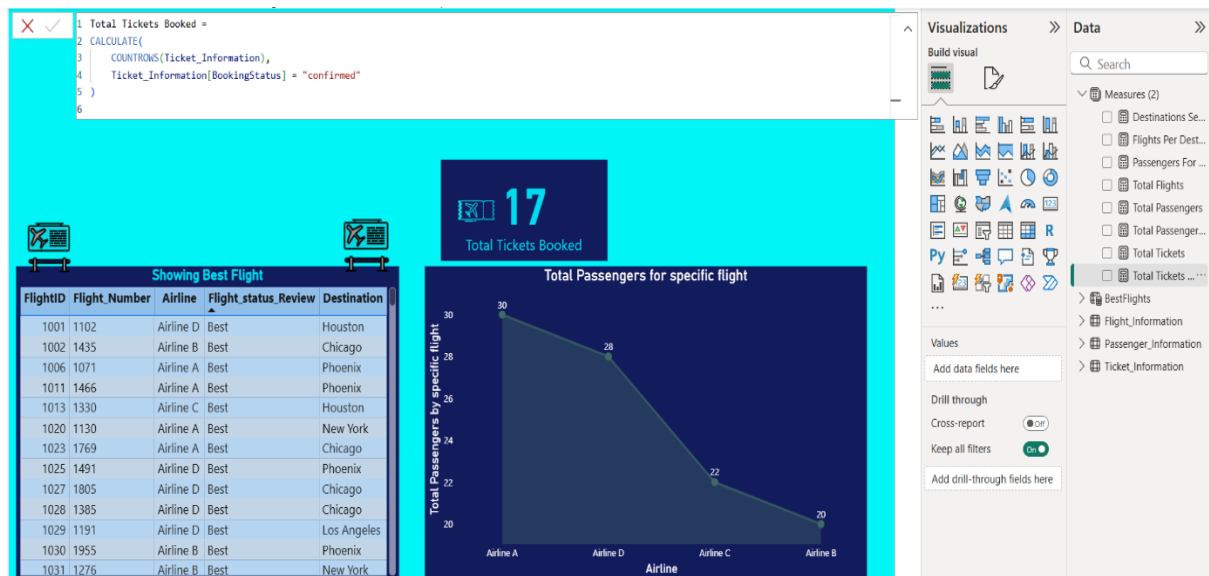
These measures helped in generating KPIs and analytical visuals to understand flight and passenger data better.

- **Total passengers for a specific flight Measure**



The DAX formula shown here calculates total passengers per flight using the COUNT or COUNTROWS function, enabling airline-specific passenger analysis.

- **Total tickets booked Measure**



This screenshot displays a DAX measure counting the number of tickets booked, derived from the Ticket\_Information table.



- Table Visual of Best Flights

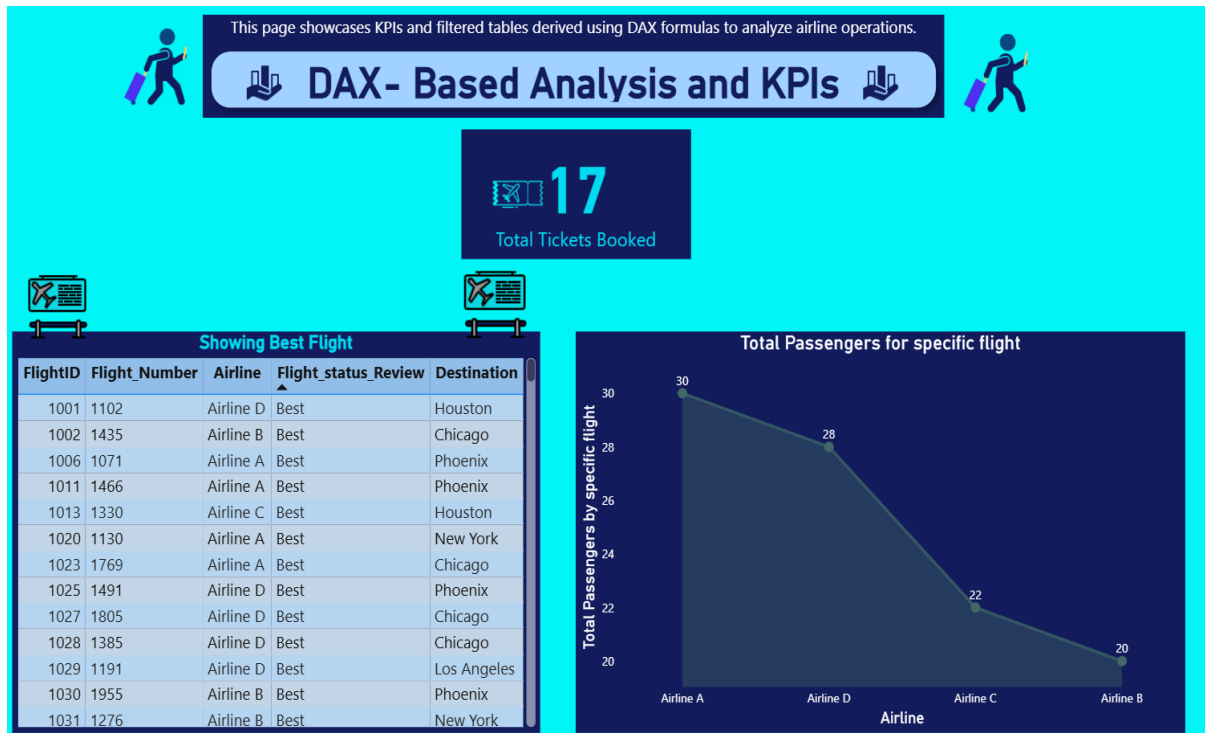


The image shows a Power BI table visual titled "Showing Best Flight". The table has five columns: FlightID, Flight\_Number, Airline, Flight\_status\_Review, and Destination. The data is filtered to show only flights with a "Best" status. The table is styled with a blue header and alternating light blue and white rows. There are airplane icons in the top corners of the visual area.

FlightID	Flight_Number	Airline	Flight_status_Review	Destination
1001	1102	Airline D	Best	Houston
1002	1435	Airline B	Best	Chicago
1006	1071	Airline A	Best	Phoenix
1011	1466	Airline A	Best	Phoenix
1013	1330	Airline C	Best	Houston
1020	1130	Airline A	Best	New York
1023	1769	Airline A	Best	Chicago
1025	1491	Airline D	Best	Phoenix
1027	1805	Airline D	Best	Chicago
1028	1385	Airline D	Best	Chicago
1029	1191	Airline D	Best	Los Angeles
1030	1955	Airline B	Best	Phoenix
1031	1276	Airline B	Best	New York

*This visual shows a filtered table displaying only 'Best' flights using a DAX measure or filter based on the conditional column.*

I created multiple DAX measures to perform key calculations using the data model. These calculations were then visualized on this page to provide focused insights using custom logic.



## Task 5: Visualization and Interactive Features

I created several visuals to explore the data interactively:

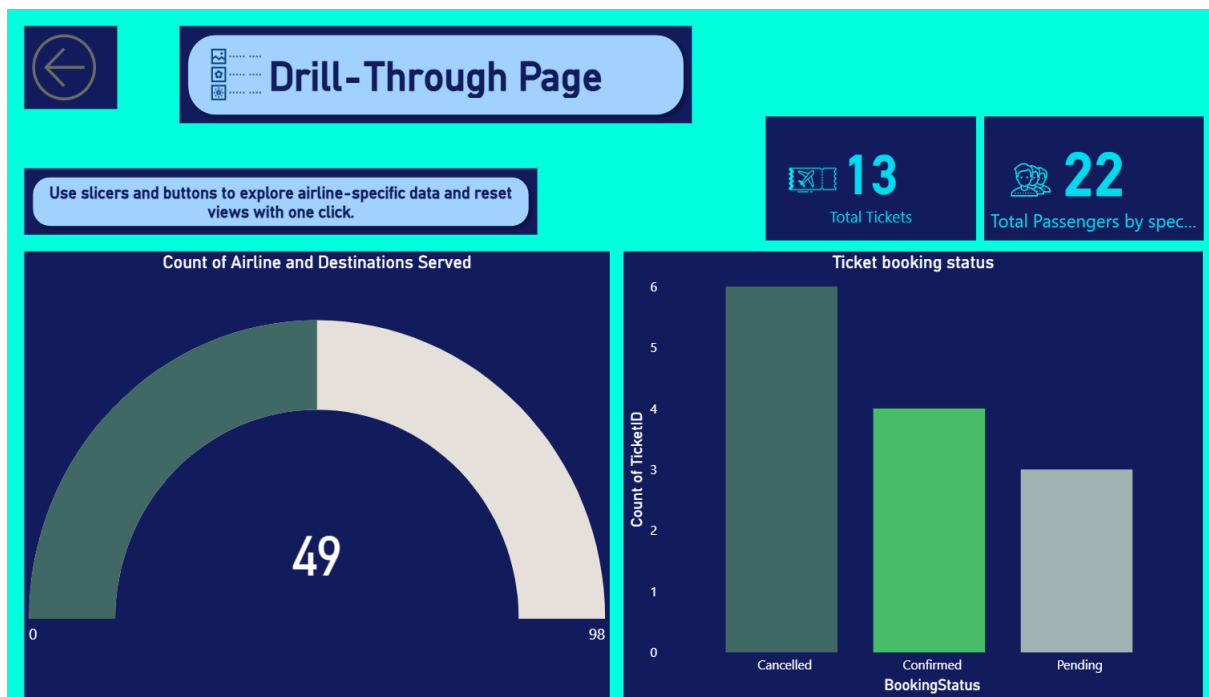
- **Passenger count by airline** using bar charts
- **Ticket booking statuses** using donut or pie charts
- **Flights by airline and destination** using stacked bar charts

To enhance user experience, I added **interactive slicers** for Airline and Destination. These slicers help users quickly filter and explore relevant data.

I also created **drillthrough pages**, allowing the user to right-click on an airline and navigate to a dedicated page showing its detailed data. Additionally, I used **buttons with bookmarks** for quick views of filtered dashboards.



This screenshot shows all key visuals and interactive features created. It includes a bar chart displaying passenger count by airline, a donut chart for ticket booking statuses, and a matrix showing flights by airline and destination.



This screenshot displays a dedicated drillthrough page for **Airline C**. I used Power BI's **Drillthrough** feature to allow users to right-click on any airline (from a chart or slicer) and navigate to a separate page that shows detailed visuals and metrics specific to that airline.

On this page, I've included key metrics such as passenger count, total tickets, flight distribution, and destination insights — all filtered automatically for Airline C. This helps decision-makers analyze performance and trends for a single airline in a focused way.

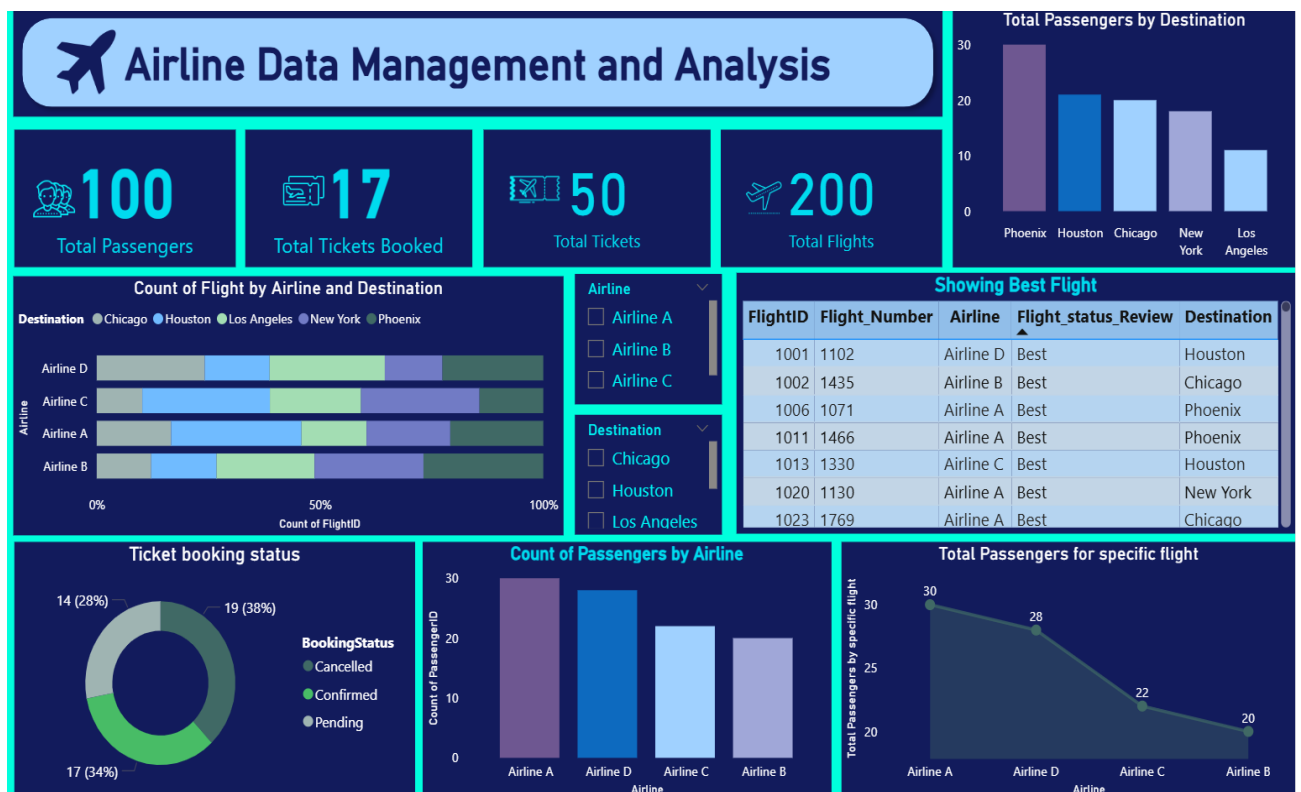
## Task 6: Final Dashboard and Power BI Service

I designed the final dashboard with a clean layout using Power BI Desktop and then published it to Power BI Service. The dashboard includes multiple pages showing visual insights, KPIs, interactive filters, and detailed airline-specific data.

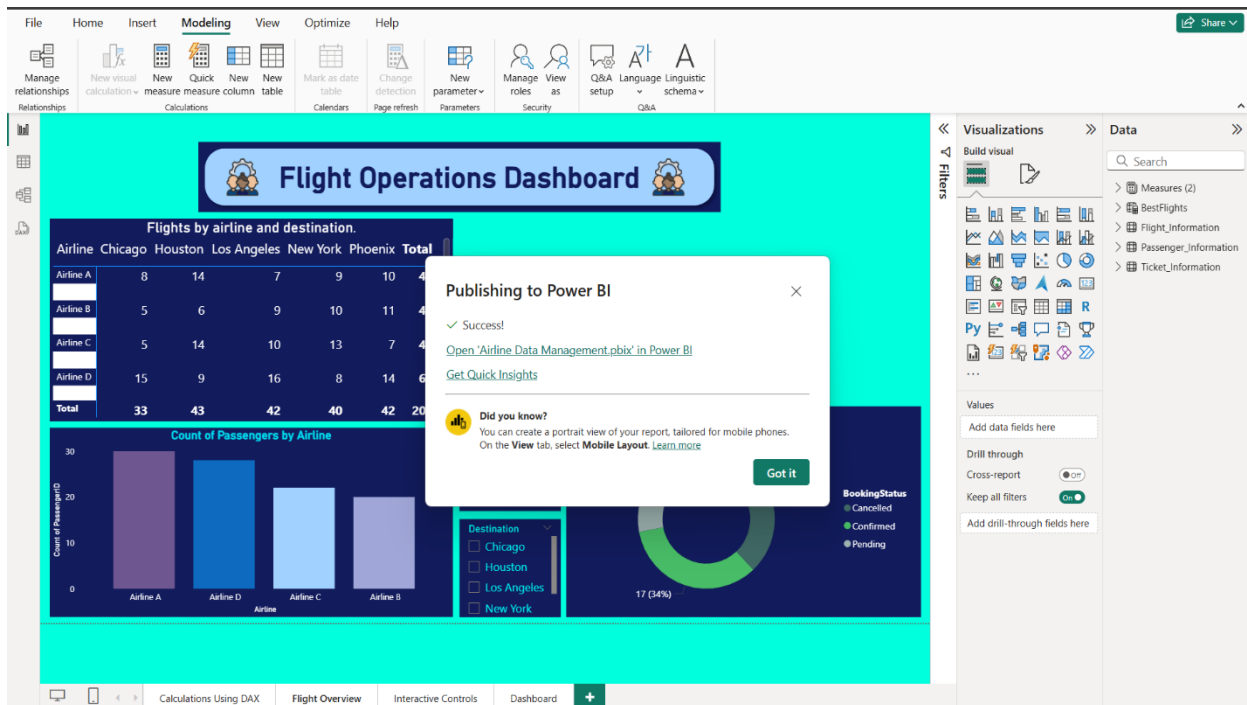
- I configured **Row-Level Security (RLS)** so that users assigned to the “Airline A” role only see data related to Airline A.
- I also set up a **scheduled refresh at 5 PM daily** by connecting the dataset to Excel files stored on OneDrive and enabling refresh settings in the Power BI Service.

These steps ensure the dashboard is both secure and automatically up-to-date.

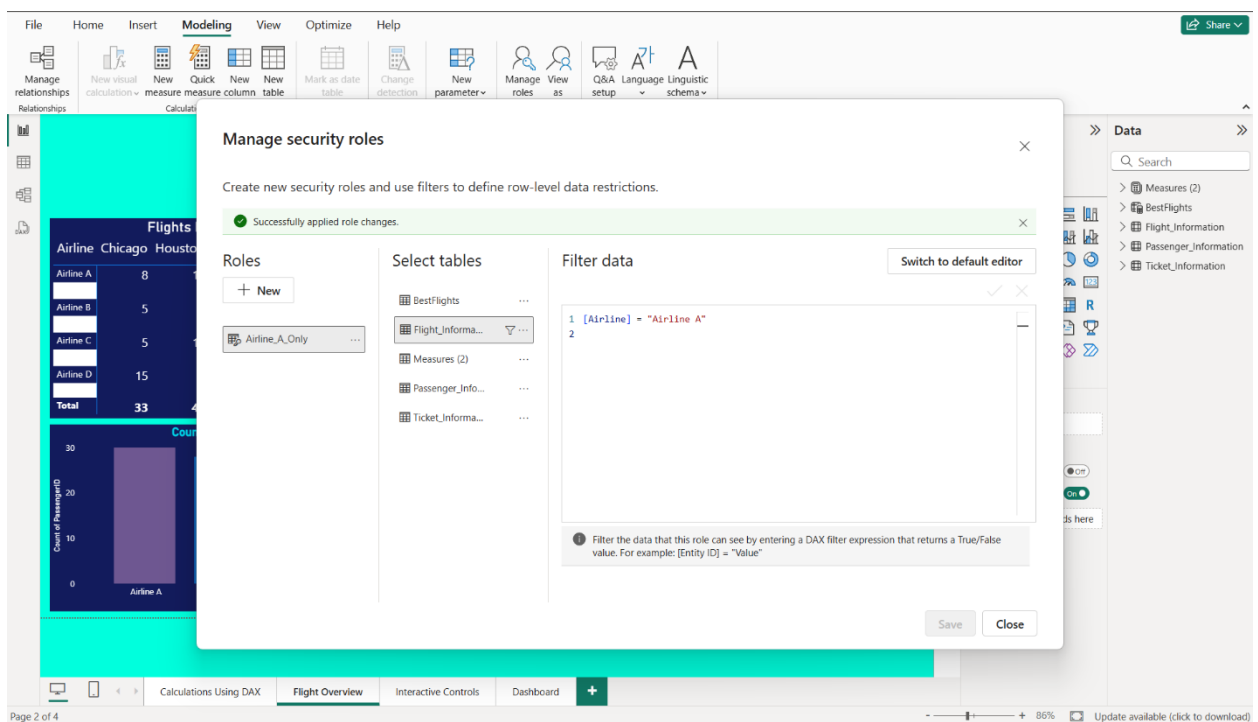
### Final Dashboard View (Published)



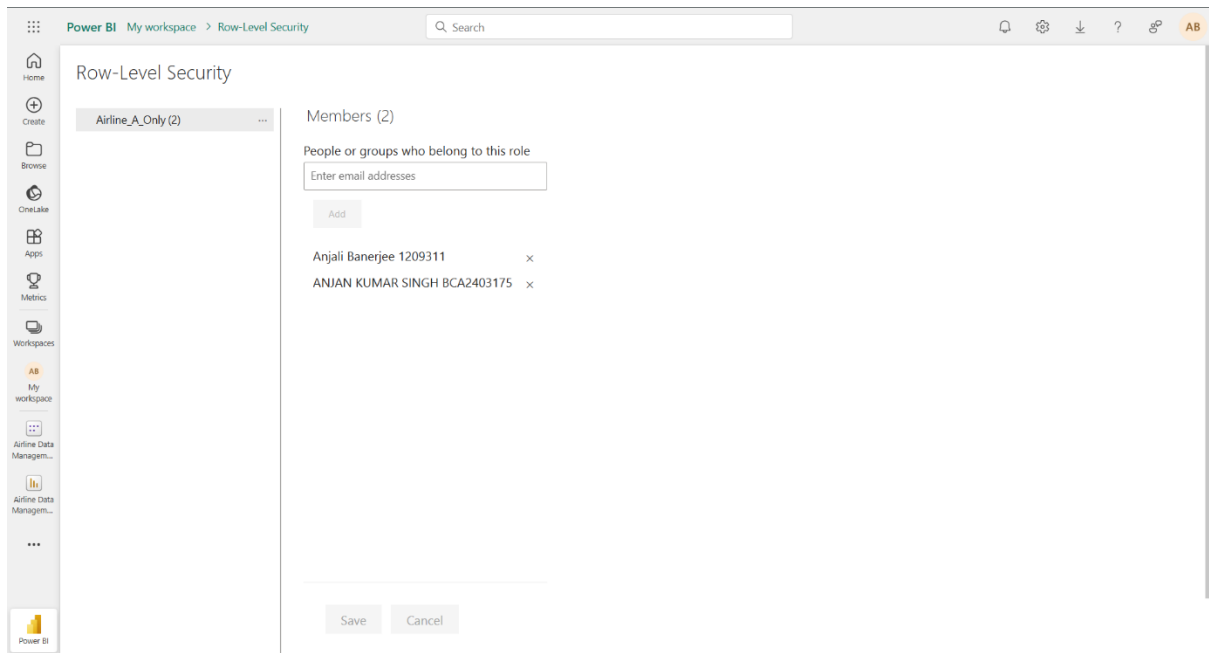
*This is the complete dashboard view as published to Power BI Service, showing how visuals are arranged and grouped meaningfully.*



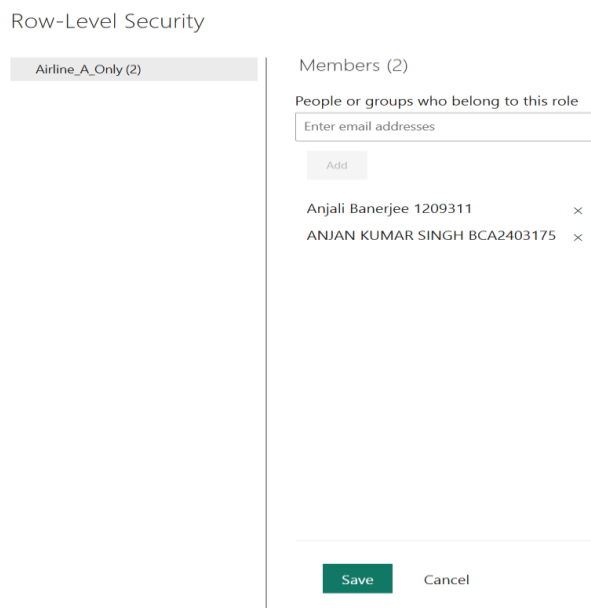
*This screenshot displays published dashboard on power BI service.*



*This screenshot shows the role setup that restricts Airline A users to only view data relevant to them using [Airline] = 'Airline A' logic.*



*This shows how the Airline A role was assigned to a specific user in Power BI Service under the Security tab.*



*This shows a close view of how the Airline A role was assigned to a specific user in Power BI Service under the Security tab.*

## Conclusion

This Power BI project has provided me with valuable hands-on experience in the complete data analysis life cycle — from data cleaning and modeling to creating calculated measures and building interactive dashboards.

I began by cleaning and preparing the datasets (Flight\_Information, Ticket\_Information, and Passenger\_Information) in Power Query Editor, ensuring the data was reliable and consistent. By

establishing proper relationships using FlightID and understanding cardinality, I laid a strong foundation for accurate data modeling.

Using DAX, I created custom calculations such as total passengers, ticket counts, and filtered flight views, which added deep analytical power to the report. The visuals — including bar charts, donut charts, and stacked visuals — offered key insights into airline performance, ticket statuses, and destinations served.

Interactive features like slicers, drillthrough pages, quick views, and bookmarks made the dashboard dynamic and user-friendly. I also implemented Row-Level Security (RLS) to restrict data access for specific airlines and scheduled a daily refresh at 5 PM to keep the report updated.

Overall, this project has helped me understand how to convert raw datasets into an insightful and interactive dashboard using Power BI. It has strengthened my analytical thinking, data modeling, and visualization skills — and prepared me for real-world reporting tasks in data-driven environments.