CREATE BITMAP INDEX Country_IDX ON Passenger_Dim(Country);

This index is suitable because the Country column has a relatively low number of
distinct values compared to the total number of rows in the table.

Bitmap Index on Year column: Assuming the Year column has relatively low cardinality,
a bitmap index can be efficient for queries filtering data based on year.
CREATE BITMAP INDEX Year_IDX ON Date_Dim(Year);

B-tree Index on Date_ column:This can be useful for range queries or equality on date.
CREATE INDEX Date_Date_IDX ON Date_Dim(Date_);

B-tree Index on Type column: becasue it is used frequently for filtering on interaction type,
a B-tree index on the Type column can be beneficial.

CREATE INDEX Type_IDX ON Interaction_Dim(Type);

CREATE CLUSTERED INDEX idx_passenger_sid ON Passenger_Dim (passenger_sid);

Created clustered index on the passenger_sid column ,Since passenger_sid contains unique and ascending values, the clustered index arranges rows in the table in ascending order of passenger_sid . This sorting improves data retrieval efficiency, as the database engine can perform efficient range scans operations
—---
CREATE INDEX Type_IDX ON sales_channal_Dim(sales_channal_name);

We created a b-tree index on the sales channel name as the marketing team always search for it , and the column has few distinct value we used b-tree index since B-tree indexes are well-suited for columns with unique values and support efficient lookup, range scans, and sorting operations.
—---
CREATE BITMAP INDEX idx_discount_ratio ON promotion_dim (discount_ratio);

Since that the discount ratios might have duplicate values, and does not contain a lot of distinct value we used a Bitmap index
,—---

CREATE INDEX idx_airport_id ON airport_dim (airport_sid);-

As the airport_id has unique values and a few distinct value and we choose to use B-tree index in place, queries that involve filtering or joining based on the airport identifier will benefit from improved performance

—--

CREATE INDEX idx_type ON Property_Dim(Type);

As the Type column in Property_Dim table has unique values and a few distinct value and we choose to use B-tree index in place

So If there are queries frequently filtering or grouping properties based on their type, creating an index on the Type column can improve query performance by facilitating efficient data retrieval for type-based filtering or grouping operations.

—--

CREATE INDEX idx_language_preference ON Passenger_Dim(Language_Preference);

As the Language_Preference column in Passenger_Dim table has unique values and a few distinct value and we choose to use B-tree index in place

So If there are queries frequently filtering passengers based on their language preference, creating an index on the Language_Preference column can improve query performance

—---

In this case we used partitioning for many columns that we in search and have a few distinct values , like the “type” column in the “property dim” table , the” frequent flyer tier “column in the “passenger profile dim” table , the airport “name” column in the “airport_dim” table , the “sales_channal_type” column in the “sales_channal_dim” and

the “year” column in the date dimension a lot in filtration and group by , we decide to partition by the year column

CREATE TABLE Date_Dim_Partitioned (Date_key NUMBER(10) NOT NULL, Year VARCHAR2(50) NOT NULL,

    Quarter VARCHAR2(50) NOT NULL, Month VARCHAR2(50) NOT NULL,Day VARCHAR2(50) NOT NULL, Day_of_Week VARCHAR2(50) NOT NULL,Weekend_Indicator VARCHAR2(50) NOT NULL, Holiday_Indicator VARCHAR2(50) NOT NULL,Date_ DATE NOT NULL)

PARTITION BY RANGE (Year) INTERVAL (1) (PARTITION p_initial VALUES LESS THAN (MAXVALUE));