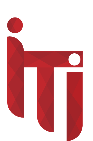
Information Technology Institute (ITI)

Smart Village

**Analyze And Model Data Warehouse**

**For Airline System Project**

**Prepared by:**

Amr El-saeed Mohamed Ellakany

Donia Ayman Ahmed

Hafsa Ahmed Zahran

Mohab Mahmoud Amro Abd Elghafar

Shahd Mohamed Ali

**Presented to:**

Ebtehal El-Hossiny

**Table of Contents**

**Contents**

[Introduction 3](#_Toc135770938)

[Business Matrix 3](#_Toc135770939)

[Dimensional Modeling Process 3](#_Toc135770940)

[1- Business process 3](#_Toc135770941)

[2- Granularity 3](#_Toc135770942)

[3- Dimensions Tables 3](#_Toc135770943)

[4- Facts Tables: 4](#_Toc135770944)

[Logical Model 5](#_Toc135770945)

[Tables Identifications 6](#_Toc135770946)

[1- Passengers Dimension 6](#_Toc135770947)

[2- Frequent Flyers Dimension 6](#_Toc135770948)

[3- Flights Dimension. 6](#_Toc135770949)

[4- Class Upgrade Dimension. 6](#_Toc135770950)

[5- Date Dimension. 6](#_Toc135770951)

[6- Reservation Channels Dimension. 6](#_Toc135770952)

[7- Fare Basis Dimension. 7](#_Toc135770953)

[8- Hotels Dimension. 7](#_Toc135770954)

[9- Interactions Dimension. 7](#_Toc135770955)

[10- Aircrafts Dimension. 7](#_Toc135770956)

[11- Airports Dimension. 7](#_Toc135770957)

[12- Payment Methods Dimension. 7](#_Toc135770958)

[13- Redeem Dimension. 8](#_Toc135770959)

[14- Staff Dimension 8](#_Toc135770960)

[15- Flights Activity Fact. 8](#_Toc135770961)

[16- Customer Care Fact. 8](#_Toc135770962)

[17- Frequent Flyers Fact. 9](#_Toc135770963)

[Physical Model 9](#_Toc135770964)

[Insert Dummy Data into The Physical Tables 23](#_Toc135770965)

[Insert The Measured Data into Fact Tables 30](#_Toc135770966)

[SQL Queries to Answer Business Questions: 33](#_Toc135770967)

[Indexes For the Data Warehousing 38](#_Toc135770968)

## Introduction

The project’s purpose is to analyze the flight activities of some airline companies and their frequent flyers and model its data warehouse schema.

## Business Matrix

## Dimensional Modeling Process

### Business process

Analyze the Airlines flights’ activities.

### Granularity

Flights Activity Fact is per Leg level.

Customer Care Fact is per Ticket Number level.

Frequent Flyers Fact is per Ticket Number level.

### Dimensions Tables

1. Passengers Dimension.
2. Frequent Flyers Dimension.
3. Flights Dimension.
4. Date Dimension.
5. Class Upgrade Dimension.
6. Channels Dimension.
7. Fare Basis Dimension.
8. Hotels Dimension.
9. Interactions Dimension.
10. Airports Dimension
11. Aircrafts Dimension
12. Payment methods Dimension
13. Redeem Dimension
14. Staff Dimension

### Facts Tables:

1. Flights Activity Fact.
2. Frequent Flyers Fact
3. Customer Care Fact.

## Logical Model

**A picture containing text, screenshot, parallel, diagram

Description automatically generated**Represents the facts and dimensions tables, and the relations between them.

## Tables Identifications

### Passengers Dimension

Represents data about passengers, like **Passenger** **ID**, **Name, Gender,** **Phone Number, email, address, city, country**, **Passenger Type**: **Regular or Frequent.**

### Frequent Flyers Dimension

Represents data about Frequent Flyers, like **Their ID**, **Tier, Club Membership Status: Non Member, Club Member, Mileage Tier and Membership Status**

**As follow:**

**Under 100,000 miles Silver**

**500,000-1,999,999 miles Platinum**

**100,000-499,999 miles Gold**

### Flights Dimension.

Represents data about the flight, like **flight ID**, **Flight Zone: Domestic or international and Flight type: Non-Stop, Direct, indirect.**

### Class Upgrade Dimension.

Represents data about the flight, like **ID**, **Purchased class: Economy, Prem Economy, First, Business and Flown Class, purchased Flown Group** **Economy-Economy, Economy-Prem Economy,…etc, class change indicator: No change , Upgrade, Downgrade**

### Date Dimension.

Represents the **Minute**, **Hour**, **Day**, **Month**, **Quarter**, and **Year** for each date stored for different business needs.

### Reservation Channels Dimension.

Represents the channel that the ticket was purchased from. Like **channel Id**, **channel type** It can be **Website** **or the agency itself.**

### Fare Basis Dimension.

Represents if the passenger fully paid the ticket price, or with a discount. It contains **id, code, status.**   
This represented using code numbers which:

**111 means Full Fare First Class.  
112 means Discount Fare First Class.  
221 means Full Fare Business Class.  
222 means Discount Fare Business Class.  
331 means Full Fare Economy Class.  
332 means Discount Fare Economy Class.**

### Hotels Dimension.

Represents data about the hotels that the passengers stayed on during the transit flights, like hotel ID, Stars, Night Price, city and country.

### Interactions Dimension.

Represents the passenger’s different interactions with customer services, it contains the **Interaction Time** if it was **Before**, **Within**, or **After** the flight and the **Interaction Type** if it was **Feedback**, **Inquiry**, or **Complaint.**

### Aircrafts Dimension.

Represents the data about the aircraft, it contains **ID, Model, Company and Seats Capacity**

### Airports Dimension.

Represents the information about airport, it contains **ID, Name , City and Country**

### Payment Methods Dimension.

Represents the information about the payment method, it contains **Payment method type: Credit or Cash**

### Redeem Dimension.

Represents the information about the type of Redeem that customer will change his mile point with, it contains **type and description.**

**As follow:**

**Flight awards Get a discount on your ticket total price.**

**Upgrade awards Get premium cabin upgrades.**

**Club and status awards Complimentary amenities and services are**

**available to make your travel more productive**

**and relaxing.**

### Staff Dimension

Represents the information about staff like: **ID, Name, Gender, Address, job title, department, phone and his supervisor.**

### Flights Activity Fact.

Represents the activity of the passenger during the flight and the data needed.

### Customer Care Fact.

Represents the passenger’s different **interactions** and the customer service’s employee who is responsible for it and in which flight this interaction happened and the passenger’s seat number and his ticket’s number.It measures the **Problem Severity** based on its **Type** and **Time** of the occurrence to work on different interactions based on their priorities:

**6 for Complaint Within the flight.  
4 for Complaint Before the flight.  
2 for Complaint After the flight.  
6 for Inquiry Within the flight.  
4 for Inquiry Before the flight.  
2 for Inquiry After the flight.**

It also shows the passengers’ **feedback rate [From 1 ~ 5]**

### Frequent Flyers Fact.

Represents the frequent flyers process and the **Miles Earned** that each passenger gained based on the **Miles Flown** also shows the **redeem type** to change these miles and the **promotion response**.

## Physical Model

The creation of tables using SQL queries. In this project, PL SQL is used for all queries.

1. **Create Passenger Dimension**

CREATE TABLE Passengers\_Dim

(

Passenger\_key NUMBER CONSTRAINT Passenger\_pk\_cons PRIMARY KEY,

Passenger\_ID VARCHAR2(250) CONSTRAINT Passenger\_ID\_unq UNIQUE,

Name VARCHAR2(250),

Gender VARCHAR2(50),

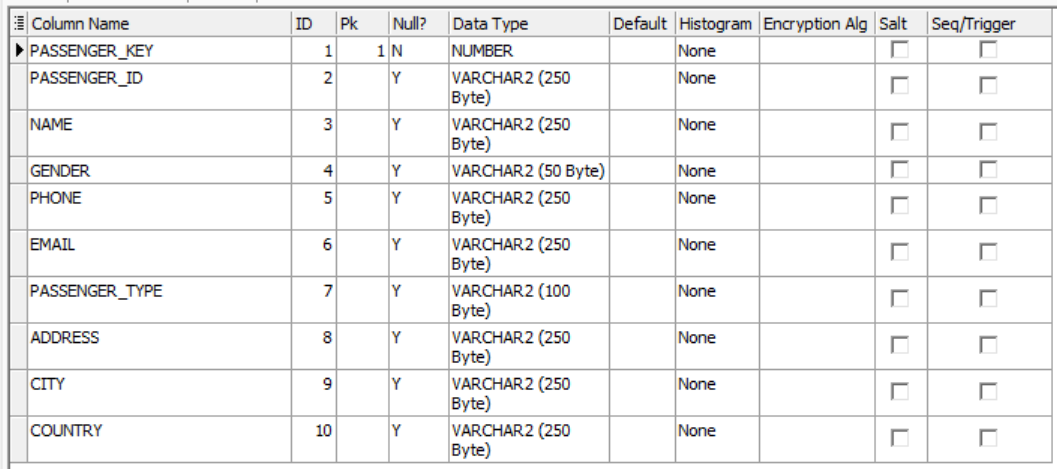
Phone VARCHAR2(250),

Email VARCHAR2(250),

Passenger\_Type VARCHAR2(100),

Address VARCHAR2(250),

City VARCHAR2(250),

 Country VARCHAR2(250) );

1. **Create Frequent Flyers Dimension**

CREATE TABLE Frequent\_flyers\_Dim

(

Freq\_key NUMBER CONSTRAINT Freq\_pk\_cons PRIMARY KEY,

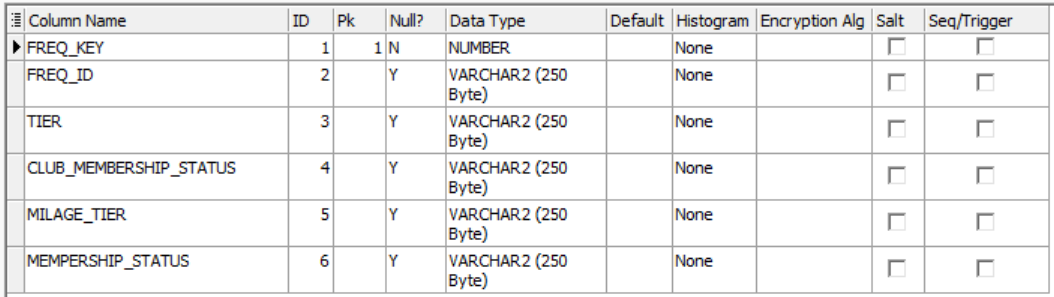
Freq\_ID VARCHAR2(250) CONSTRAINT Freq\_ID\_unq UNIQUE,

Tier VARCHAR2(250),

Club\_Membership\_Status VARCHAR2(250),

Milage\_Tier VARCHAR2(250),

Mempership\_Status VARCHAR2(250)

);

1. **Create Flights Dimension**

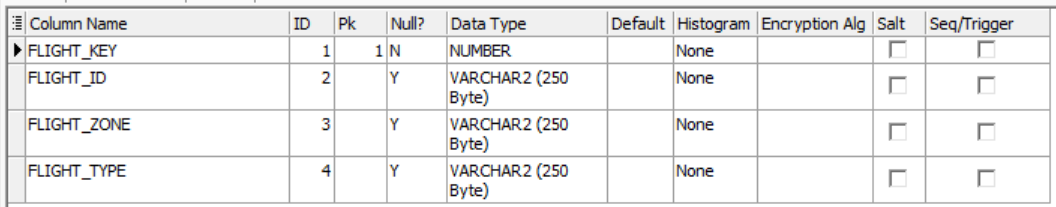
CREATE TABLE Flights\_Dim

(

Flight\_key NUMBER CONSTRAINT flight\_pk\_cons PRIMARY KEY,

Flight\_ID VARCHAR2(250) CONSTRAINT flight\_unq\_cons UNIQUE,

Flight\_zone VARCHAR2(250),

 Flight\_type VARCHAR2(250) );

1. **Create Class Upgrade Dimension**

CREATE TABLE Class\_Upgrade\_Dim

(

Class\_Upgrade\_key NUMBER CONSTRAINT class\_pk\_cons PRIMARY KEY,

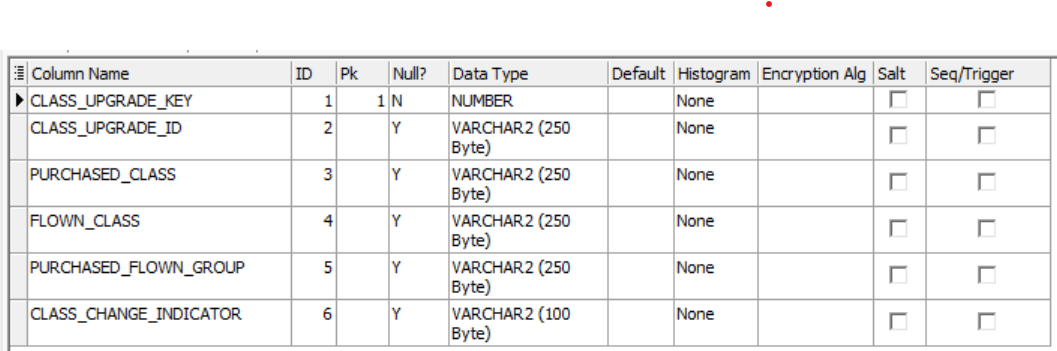
Class\_Upgrade\_ID VARCHAR2(250) CONSTRAINT classU\_ID\_unq UNIQUE,

Purchased\_Class VARCHAR2(250),

Flown\_Class VARCHAR2(250),

Purchased\_Flown\_Group VARCHAR2(250),

Class\_Change\_Indicator VARCHAR2(100)

);

1. **Create Date Dimension**

CREATE TABLE Date\_Dim

(

Full\_date DATE CONSTRAINT D\_pk\_cons PRIMARY KEY,

Year VARCHAR2(15),

Quarter VARCHAR2(15),

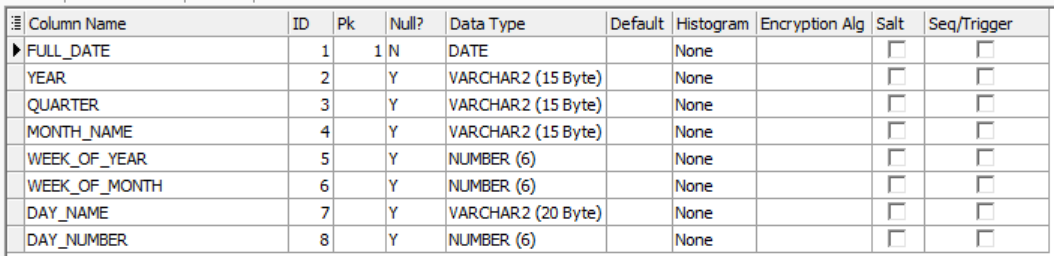
Month\_Name VARCHAR2(15),

Week\_of\_Year NUMBER(6),

Week\_of\_Month NUMBER(6),

Day\_Name VARCHAR2(20),

Day\_Number NUMBER(6)

 );

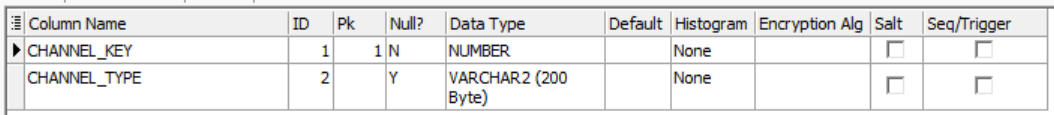
1. **Create Channel Dimension**

CREATE TABLE Channels\_Dim

(

Channel\_key NUMBER CONSTRAINT channel\_pk\_cons PRIMARY KEY,

Channel\_Type VARCHAR2(200)

 );

1. **Create Fare Basis Dimension**

CREATE TABLE Fare\_Basis\_Dim

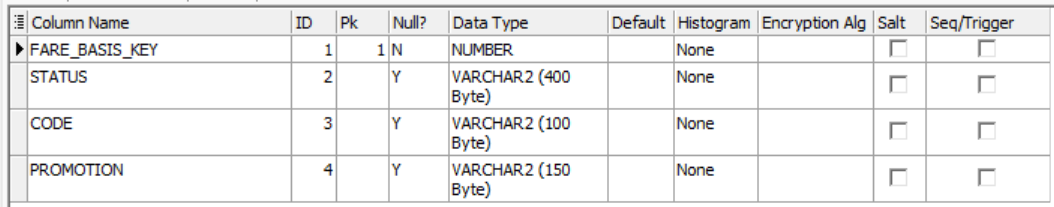
(

Fare\_Basis\_key NUMBER CONSTRAINT fare\_pk\_cons PRIMARY KEY,

Status VARCHAR2(400),

Code VARCHAR2(100),

Promotion VARCHAR2(150)

);

1. **Create Hotel Dimension**

CREATE TABLE Hotels\_DIM

(

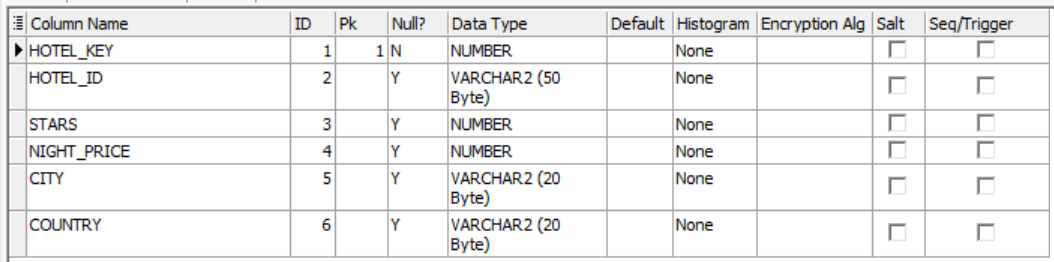
Hotel\_key NUMBER CONSTRAINT HT\_pk\_cons PRIMARY KEY,

Hotel\_ID VARCHAR2(50) CONSTRAINT HT\_unq\_cons UNIQUE,

Stars NUMBER,

Night\_Price NUMBER,

City VARCHAR2(20),

 Country VARCHAR2(20) );

1. **Create Interactions Dimension**

CREATE TABLE Interactions\_DIM

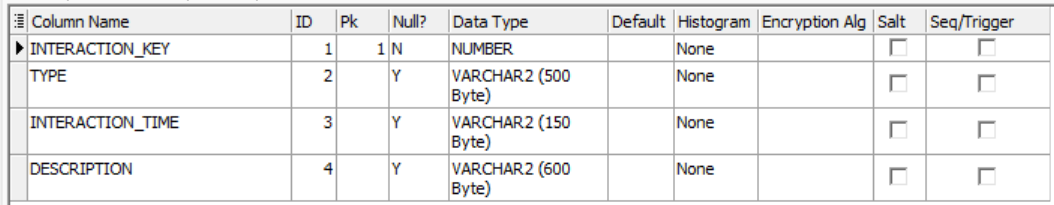
(

Interaction\_key NUMBER CONSTRAINT Int\_pk\_cons PRIMARY KEY,

TYPE VARCHAR2(500),

Interaction\_time VARCHAR2(150),

Description VARCHAR2(600)

);

1. **Create Aircrafts Dimension**

CREATE TABLE Aircrafts\_Dim

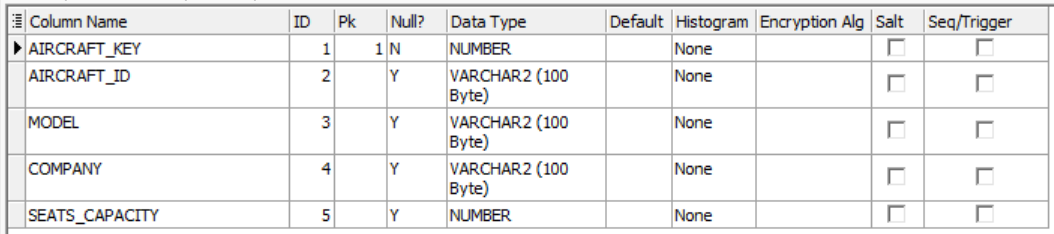
(

Aircraft\_key NUMBER CONSTRAINT aircraft\_pk\_cons PRIMARY KEY,

Aircraft\_ID VARCHAR2(100) CONSTRAINT Acraft\_unq\_cons UNIQUE,

Model VARCHAR2(100),

Company VARCHAR2(100),

 Seats\_Capacity NUMBER  
);

1. **Create Airports Dimension**

CREATE TABLE Airports\_Dim

(

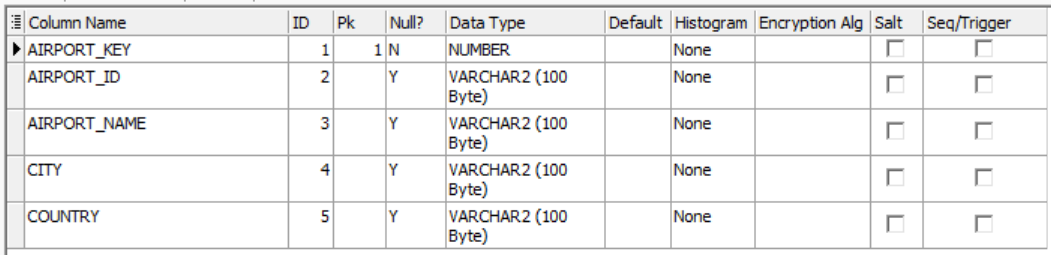
Airport\_key number CONSTRAINT airport\_pk\_cons PRIMARY KEY,

Airport\_ID VARCHAR2(100) CONSTRAINT Aport\_unq\_cons UNIQUE,

Airport\_name VARCHAR2(100),

City VARCHAR2(100),

Country VARCHAR2(100)

);

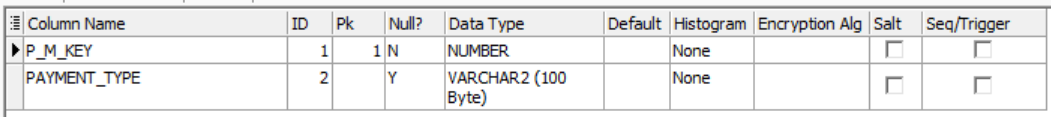
1. **Create Payment methods Dimension**

CREATE TABLE Payment\_Methods\_Dim

(

P\_M\_key NUMBER CONSTRAINT p\_m\_pk\_cons PRIMARY KEY,

Payment\_type VARCHAR2(100)

);

1. **Create Redeem Dimension**

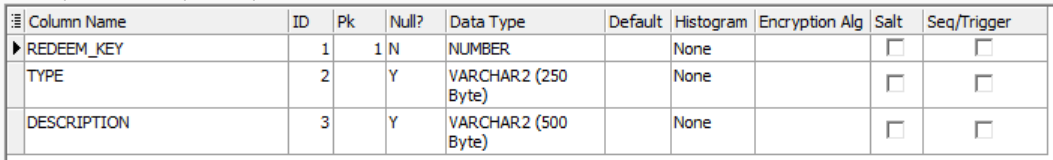
CREATE TABLE Redeem\_Dim

(

Redeem\_key NUMBER CONSTRAINT Rd\_pk\_cons PRIMARY KEY,

TYPE VARCHAR2(250),

Description VARCHAR2(500)

);  


1. **Create Staff Dimension:**

CREATE TABLE Staff\_Dim

(

Staff\_key NUMBER CONSTRAINT ST\_pk\_cons PRIMARY KEY,

Staff\_ID VARCHAR2(500) CONSTRAINT ST\_unq\_cons UNIQUE,

NAME VARCHAR2(500),

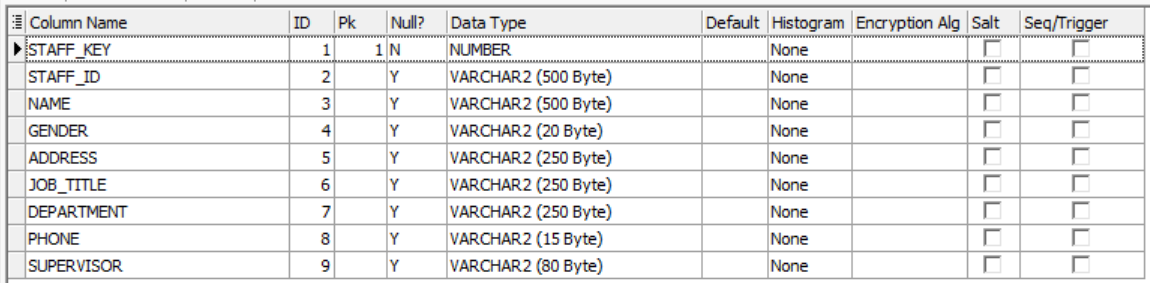
Gender VARCHAR2(20),

Address VARCHAR2(250),

Job\_Title VARCHAR2(250),

Department VARCHAR2(250),

Phone VARCHAR2(15),

Supervisor VARCHAR2(80));   


1. **Create Flights Activity Fact**

CREATE TABLE flights\_activity\_Fact

(

Leg\_key NUMBER CONSTRAINT Leg\_pk\_cons PRIMARY KEY,

Ticket\_number NUMBER CONSTRAINT TN\_unq\_cons UNIQUE,

Passenger\_key NUMBER,

Freq\_key NUMBER,

Aircraft\_key NUMBER,

Segment\_number NUMBER ,

Flight\_key NUMBER,

Departure\_airport NUMBER,

Arrival\_airport NUMBER,

Actual\_departure\_date DATE,

Schedualed\_departure\_time TIMESTAMP,

Actual\_departure\_time TIMESTAMP,

Actual\_arrival\_date DATE,

Schedualed\_arrival\_time TIMESTAMP,

Actual\_arrival\_time TIMESTAMP,

Flight\_duration NUMBER,

Reservation\_Ticket\_Date DATE,

Channel\_key NUMBER,

Payment\_method\_key NUMBER,

Fare\_basis\_key NUMBER,

Class\_upgrade\_key NUMBER,

Luggage\_Number NUMBER,

Base\_fare\_price NUMBER,

Taxes NUMBER,

Overweight\_Luggage\_price NUMBER,

Upgrage\_fees NUMBER,

Transaction\_fees NUMBER,

Promotion\_amount NUMBER,

Net\_price NUMBER,

Cost NUMBER,

Profit NUMBER,

Hotel\_key NUMBER,

overnight\_stays NUMBER

);

ALTER TABLE flights\_activity\_Fact

ADD(

CONSTRAINT pass\_fk\_cons FOREIGN KEY(Passenger\_key) REFERENCES Passengers\_Dim(Passenger\_key),

CONSTRAINT FF\_fk\_cons FOREIGN KEY(Freq\_key) REFERENCES Frequent\_flyers\_Dim(Freq\_key),

CONSTRAINT aircraft\_fk\_cons FOREIGN KEY(aircraft\_key) REFERENCES Aircrafts\_Dim(Aircraft\_key),

CONSTRAINT flight\_fk\_cons FOREIGN KEY(Flight\_key) REFERENCES Flights\_Dim(Flight\_key),

CONSTRAINT airport\_fk\_cons FOREIGN KEY(Departure\_airport) REFERENCES Airports\_Dim(Airport\_key),

CONSTRAINT airport\_fk\_conss FOREIGN KEY(Arrival\_airport) REFERENCES Airports\_Dim(Airport\_key),

CONSTRAINT actual\_date\_departure\_fk\_cons FOREIGN KEY(Actual\_departure\_date) REFERENCES Date\_Dim(Full\_date),

CONSTRAINT actual\_date\_arrival\_fk\_cons FOREIGN KEY(Actual\_arrival\_date) REFERENCES Date\_Dim(Full\_date),

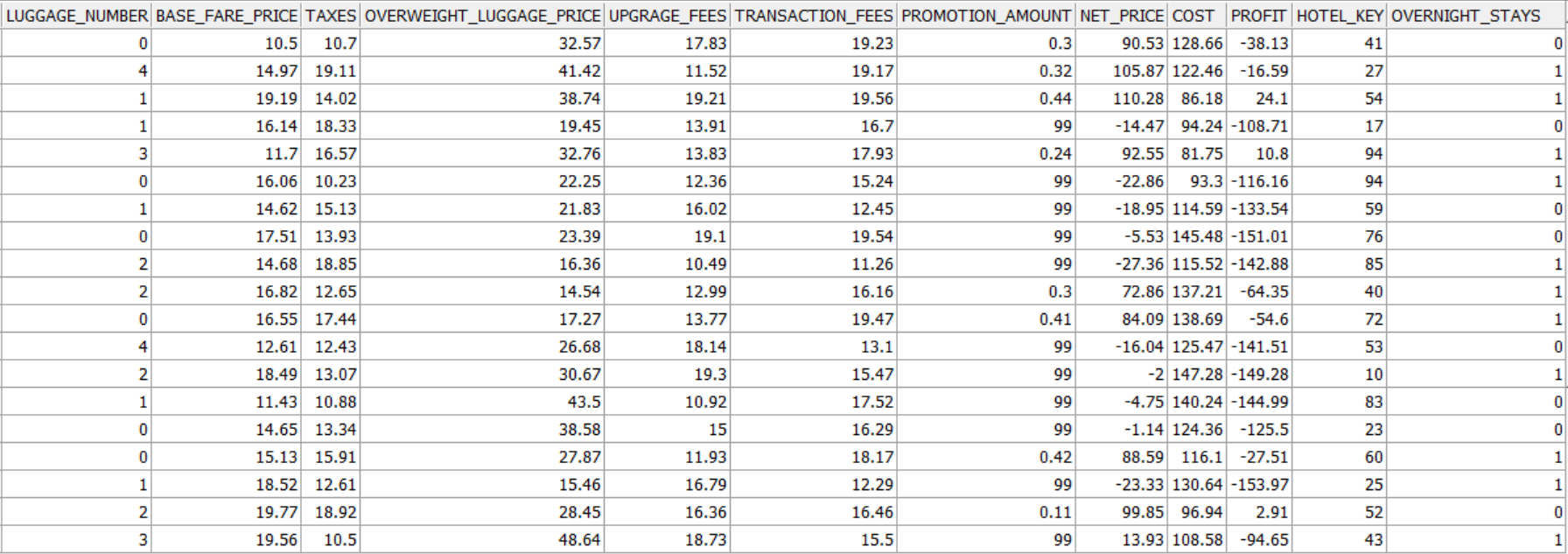
CONSTRAINT resv\_t\_date\_fk\_conssss FOREIGN KEY(Reservation\_Ticket\_Date) REFERENCES Date\_Dim(Full\_date),

CONSTRAINT ch\_fk\_conss FOREIGN KEY(Channel\_key) REFERENCES Channels\_Dim(Channel\_key),

CONSTRAINT pm\_fk\_conss FOREIGN KEY(Payment\_method\_key) REFERENCES Payment\_Methods\_Dim(P\_M\_key),

CONSTRAINT fb\_fk\_conss FOREIGN KEY(Fare\_basis\_key) REFERENCES Fare\_Basis\_Dim(Fare\_Basis\_key),

CONSTRAINT cu\_fk\_conss FOREIGN KEY(Class\_upgrade\_key) REFERENCES Class\_Upgrade\_Dim(Class\_Upgrade\_key),

 CONSTRAINT H\_fk\_consss FOREIGN KEY(Hotel\_key) REFERENCES Hotels\_DIM(Hotel\_key) );

1. **Create Customer Care Fact**

CREATE TABLE Customer\_Care\_fact

(

CC\_key NUMBER CONSTRAINT cc\_pk\_cons PRIMARY KEY ,

Ticket\_number NUMBER CONSTRAINT TN\_unq\_consss UNIQUE,

Passenger\_key NUMBER ,

Freq\_key NUMBER ,

Reservation\_Ticket\_Date DATE,

Segment\_number NUMBER,

Flight\_key NUMBER,

Interaction\_key NUMBER,

Staff\_key NUMBER,

Problem\_severity NUMBER,

Feedback\_Rate NUMBER,

Deprature\_airport NUMBER,

Arrival\_airport NUMBER,

Seat\_Number NUMBER

);

ALTER TABLE Customer\_Care\_fact

ADD(

CONSTRAINT pass\_fk\_consss FOREIGN KEY(Passenger\_key) REFERENCES Passengers\_Dim(Passenger\_key),

CONSTRAINT FF\_fk\_consss FOREIGN KEY(Freq\_key) REFERENCES Frequent\_flyers\_Dim(Freq\_key),

CONSTRAINT resv\_t\_date\_fk\_consss FOREIGN KEY(Reservation\_Ticket\_Date) REFERENCES Date\_Dim(Full\_date),

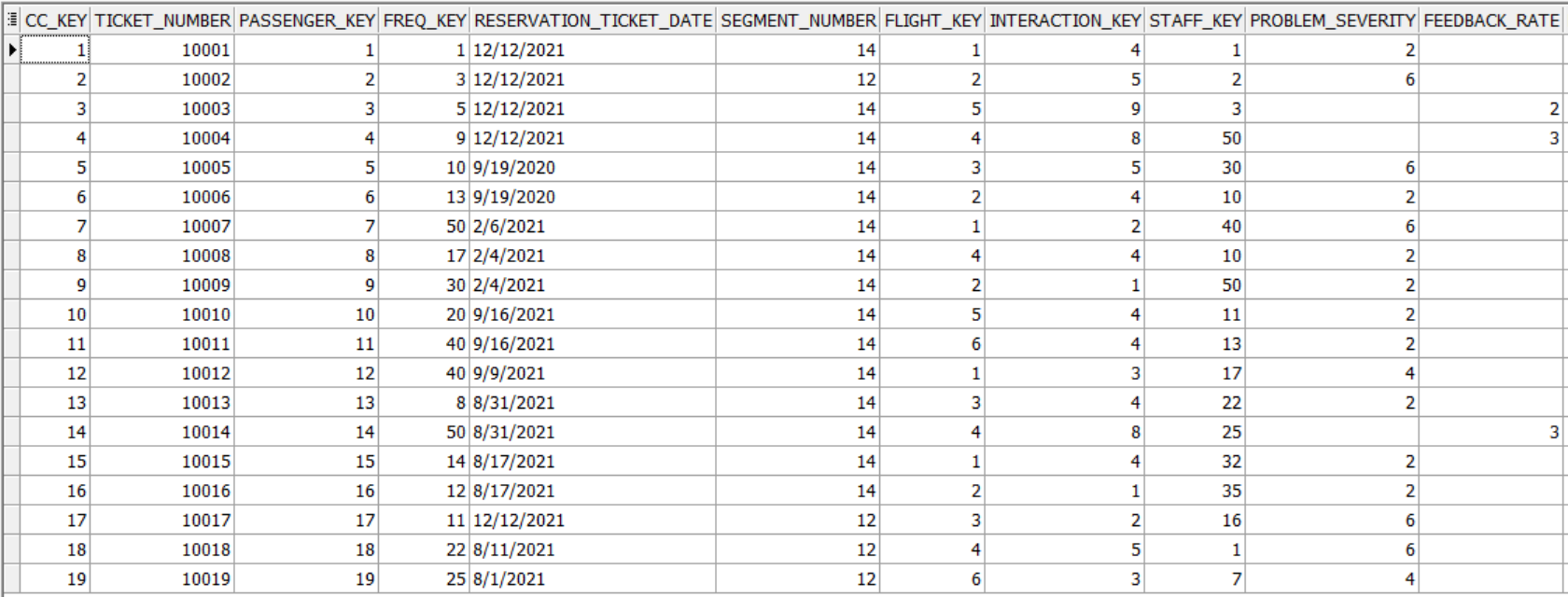
CONSTRAINT flight\_fk\_consss FOREIGN KEY(Flight\_key) REFERENCES Flights\_Dim(Flight\_key),

CONSTRAINT int\_fk\_consss FOREIGN KEY(Interaction\_key) REFERENCES Interactions\_DIM(Interaction\_key),

CONSTRAINT st\_fk\_consss FOREIGN KEY(Staff\_key) REFERENCES Staff\_Dim(Staff\_key),

CONSTRAINT airport\_fk\_consssss FOREIGN KEY(Deprature\_airport) REFERENCES Airports\_Dim(Airport\_key),

CONSTRAINT airport\_fk\_conssssssss FOREIGN KEY(Arrival\_airport) REFERENCES Airports\_Dim(Airport\_key)

 );

1. **Create Frequent Flyers Fact**

CREATE TABLE Frequent\_Flyers\_fact

(

FF\_Fact\_key Number CONSTRAINT freqf\_fact\_pk\_cons PRIMARY KEY ,

Ticket\_number NUMBER CONSTRAINT TTN UNIQUE,

Freq\_key NUMBER,

Flight\_date DATE,

Flight\_key NUMBER,

Deprature\_airport NUMBER,

Arrival\_airport NUMBER,

Miles\_flown NUMBER,

Miles\_earned NUMBER,

promotion\_response VARCHAR2(20),

Redeem\_key NUMBER

);

ALTER TABLE Frequent\_Flyers\_fact

ADD(

CONSTRAINT FF\_fk\_conssss FOREIGN KEY(Freq\_key) REFERENCES Frequent\_flyers\_Dim(Freq\_key),

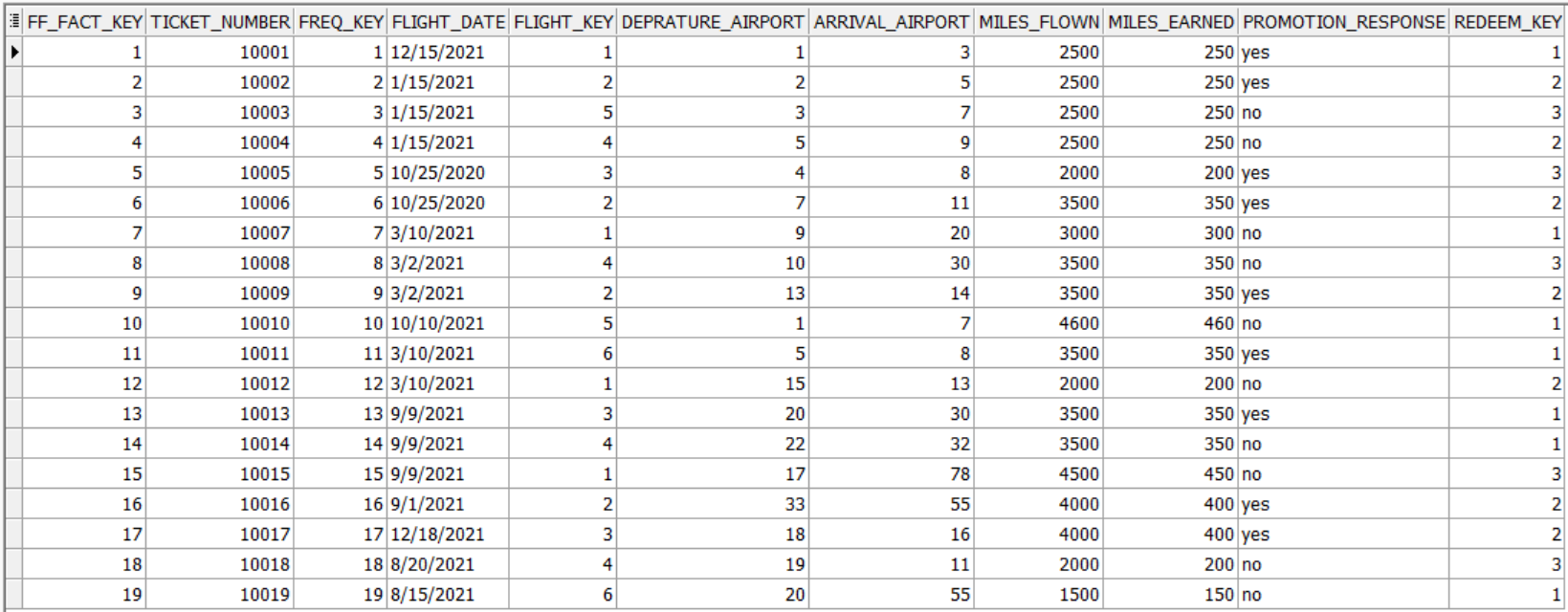
CONSTRAINT f\_date\_fk\_consss FOREIGN KEY(Flight\_date) REFERENCES Date\_Dim(full\_date),

CONSTRAINT flight\_fk\_conssss FOREIGN KEY(Flight\_key) REFERENCES Flights\_Dim(Flight\_key),

CONSTRAINT airport\_fk\_consssssssss FOREIGN KEY(Deprature\_airport) REFERENCES Airports\_Dim(Airport\_key),

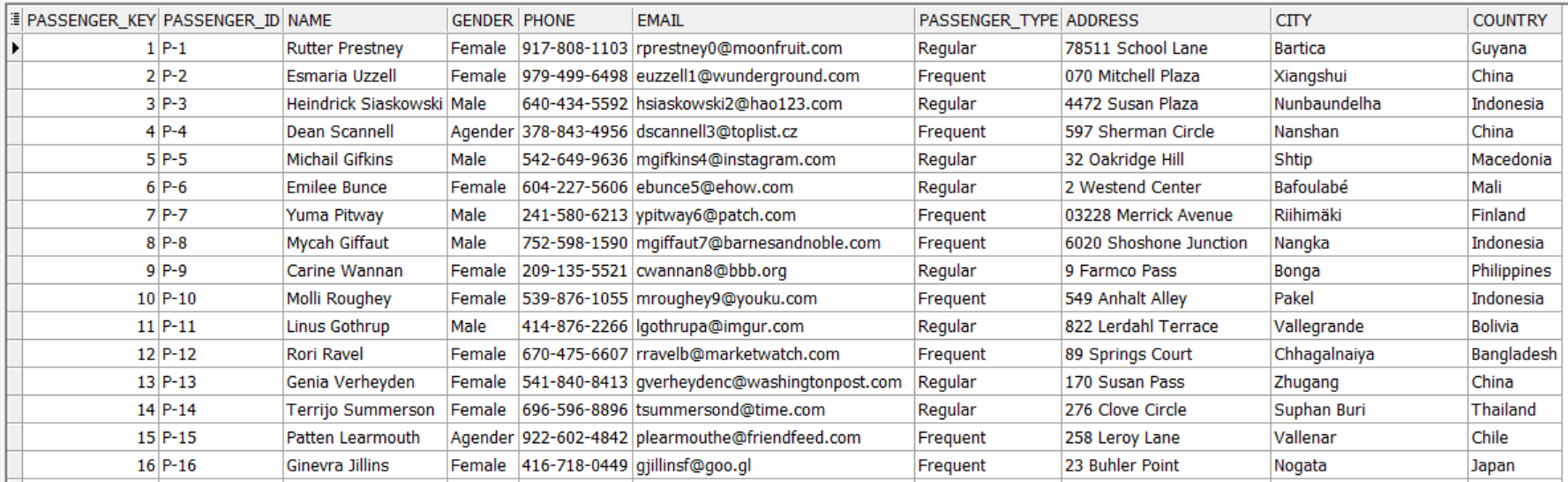
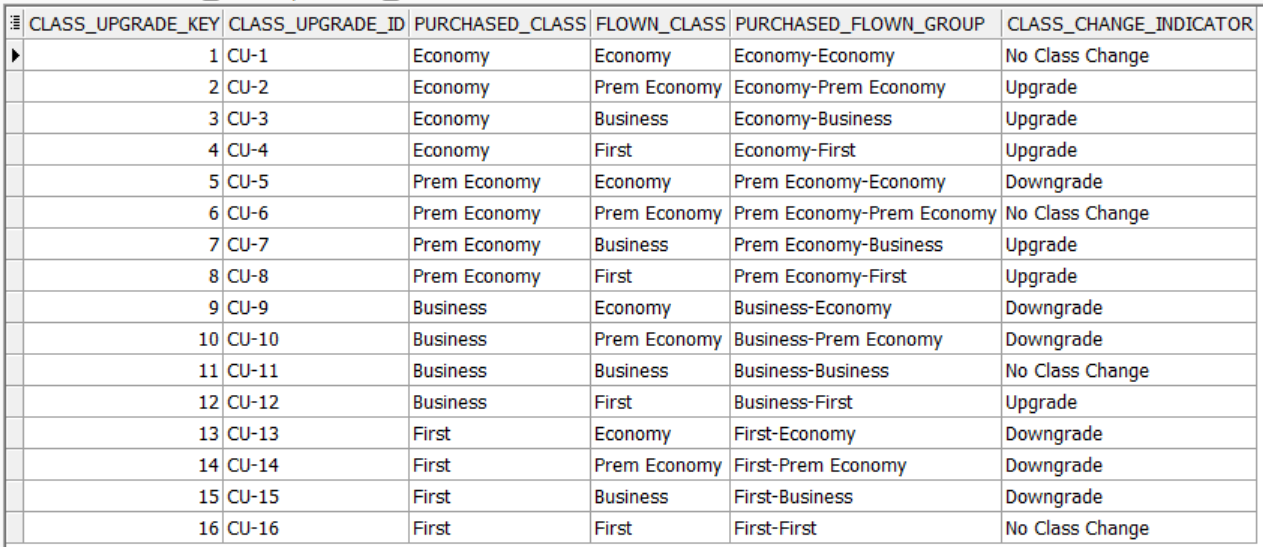
CONSTRAINT airport\_fk\_consssssssssss FOREIGN KEY(Arrival\_airport) REFERENCES Airports\_Dim(Airport\_key),

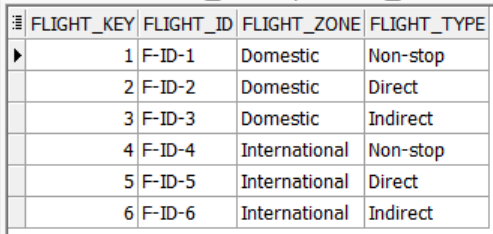
CONSTRAINT rd\_fk\_consss FOREIGN KEY(Redeem\_key) REFERENCES Redeem\_Dim(Redeem\_key)

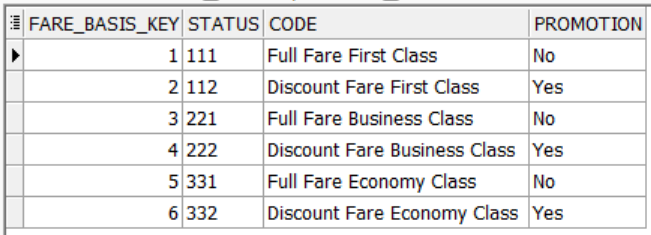
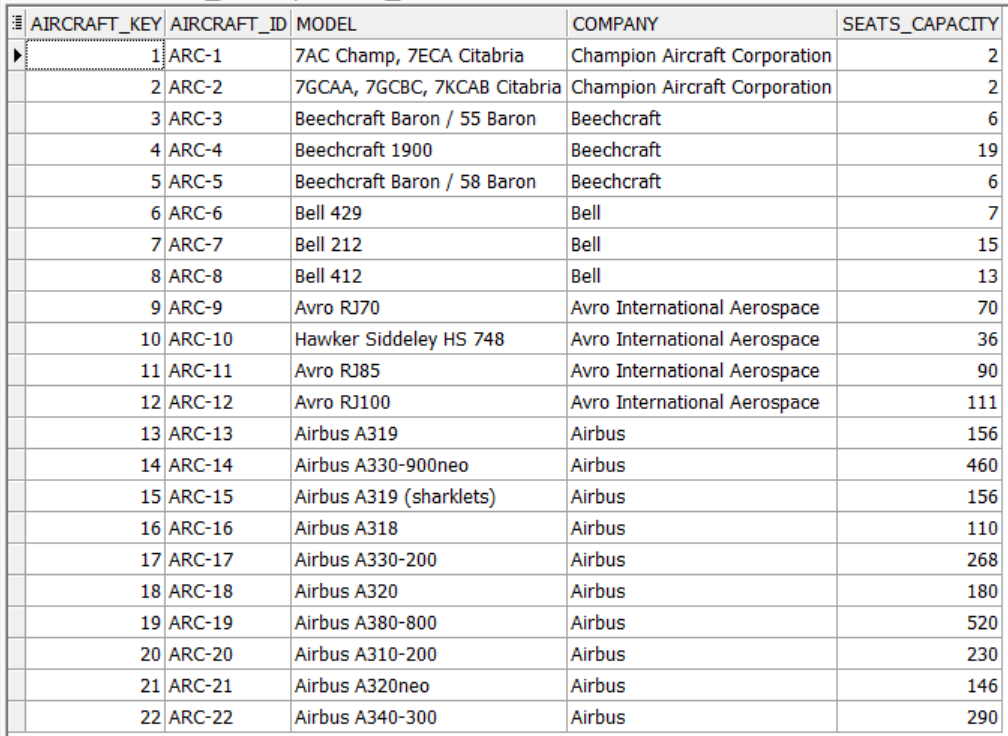
 );

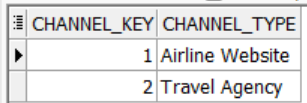
## Insert Dummy Data into The Physical Tables

The data inserted was using excel files that have dummy data and PLSQL blocks generating random data

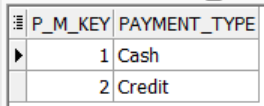
1. Passengers Dimension Data
2. Frequent Flyers Dimension Data
3. Class Upgrade Dimension Data
4. Flights Dimension Data

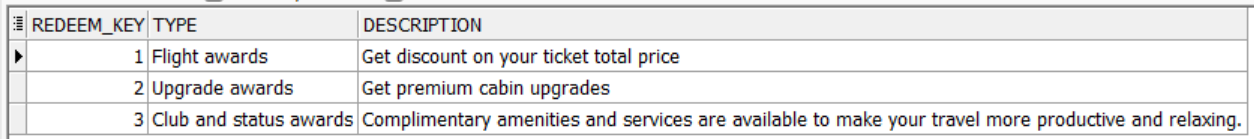
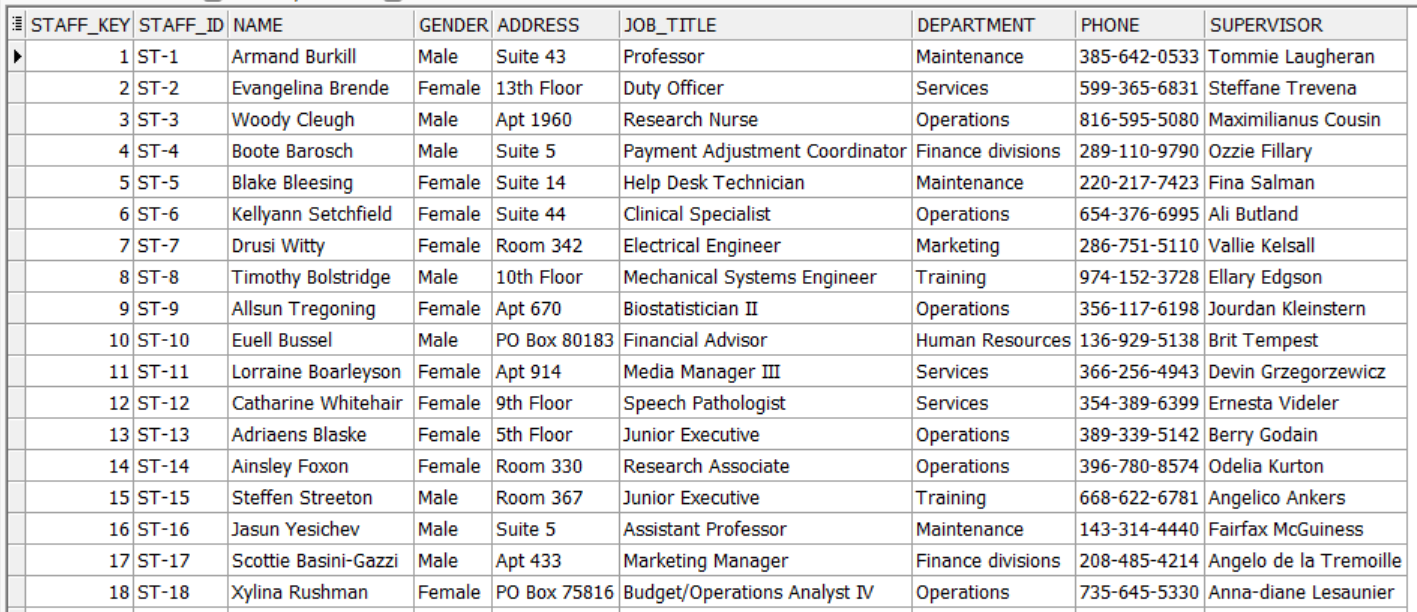
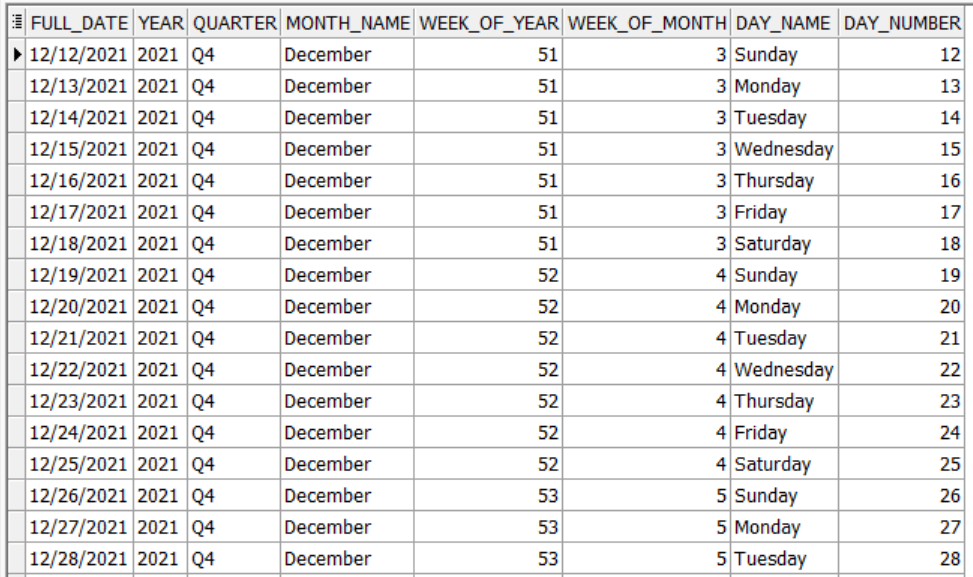
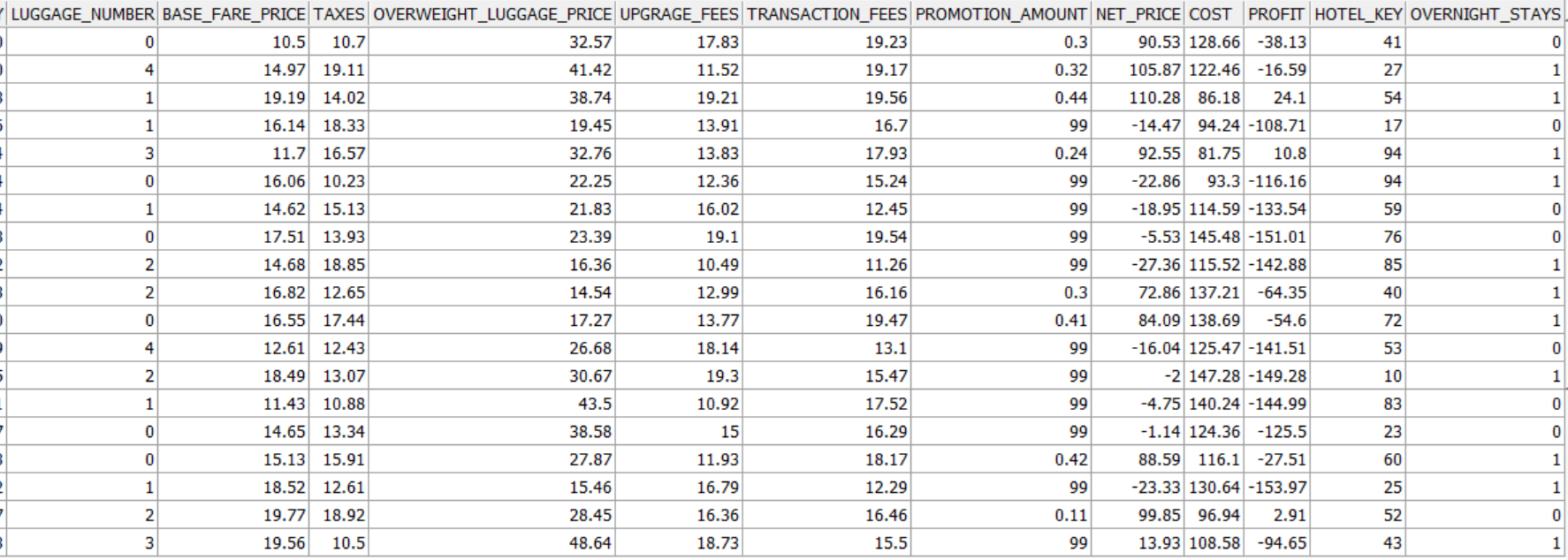


1. Fare Basis Dimension Data
2. Aircrafts Dimension Data
3. Airports Dimension Data
4. Channel Dimension Data



1. Payment methods Dimension Data



1. Interactions Dimension Data
2. Redeem Dimension Data
3. Staff Dimension Data
4. Date Dimension Data  
   
5. Hotels Dimension Data
6. Flights Activity Fact
7. Frequent Flyers Fact

A picture containing text, number, screenshot, font

Description automatically generated

1. Table

   Description automatically generatedCustomer Care Fact Data

## Insert The Measured Data into Fact Tables

1. **Insert into Flights Activity Fact [Net price & Profit]**

DECLARE

CURSOR Cur IS

SELECT Leg\_key, Net\_price, Base\_fare\_price, Taxes, Overweight\_Luggage\_price, Upgrage\_fees, Transaction\_fees, Promotion\_amount, Cost, Profit

from Flights\_activity\_fact

FOR UPDATE OF Net\_price ;

BEGIN

FOR Rec IN Cur

Loop

UPDATE Flights\_activity\_fact

SET Net\_price = (Base\_fare\_price+ Taxes+nvl(Overweight\_Luggage\_price,0)+nvl(Upgrage\_fees,0)+Transaction\_fees)-nvl(Promotion\_amount ,0)

WHERE CURRENT OF Cur;

UPDATE Flights\_activity\_fact

SET Profit =Net\_price-Cost

WHERE CURRENT OF Cur;

END LOOP;

END;

1. **Insert into Customer Care Fact [Seat number]**

DECLARE

CURSOR Cur IS

SELECT Flight\_key ,Seat\_Number

from Customer\_Care\_fact

FOR UPDATE OF Seat\_Number;

BEGIN

FOR Rec IN Cur

LOOP

IF (Rec.Flight\_key in (1,2,3,4,5,6))

THEN

UPDATE Customer\_Care\_fact

SET Seat\_Number = TRUNC(***dbms\_random.value***(1,300),0)

WHERE CURRENT OF Cur;

END IF;

END LOOP;

END;

1. **Insert into Customer Care Fact [Problem severity & Feedback Rate]**

DECLARE

v\_type VARCHAR(250);

v\_time VARCHAR(250);

CURSOR Cur IS

SELECT Interaction\_key ,Problem\_severity

from Customer\_Care\_fact

FOR UPDATE OF Problem\_severity;

BEGIN

FOR Rec IN Cur

LOOP

SELECT Interaction\_time, TYPE

INTO v\_time, v\_type

FROM Interactions\_DIM

WHERE Interaction\_key = Rec.INTERACTION\_KEY;

IF (v\_time = 'Within Flight' AND v\_type = 'Complaints')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity = 6

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'Before Flight' AND v\_type = 'Complaints')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity = 2

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'After Flight' AND v\_type = 'Complaints')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity = 4

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'Within Flight' AND v\_type = 'Inquire')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity= 6

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'Within Flight' AND v\_type = 'Feedback')

THEN

UPDATE Customer\_Care\_fact

SET Feedback\_Rate = TRUNC(***dbms\_random.value***(1,5),0)

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'Before Flight' AND v\_type = 'Inquire')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity = 2

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'Before Flight' AND v\_type = 'Feedback')

THEN

UPDATE Customer\_Care\_fact

SET Feedback\_Rate= TRUNC(***dbms\_random.value***(1,5),0)

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'After Flight' AND v\_type = 'Feedback')

THEN

UPDATE Customer\_Care\_fact

SET Feedback\_Rate = TRUNC(***dbms\_random.value***(1,5),0)

WHERE CURRENT OF Cur;

ELSIF (v\_time = 'After Flight' AND v\_type = 'Inquire')

THEN

UPDATE Customer\_Care\_fact

SET Problem\_severity = 4

WHERE CURRENT OF Cur;

END IF;

END LOOP;

END;

## SQL Queries to Answer Business Questions:

**1- Which Flight types do frequent flyers take more depending on their tier?**

SELECT fd.Flight\_type, ff.Tier, COUNT(fa.Freq\_key) AS Frequent\_Flyers\_number

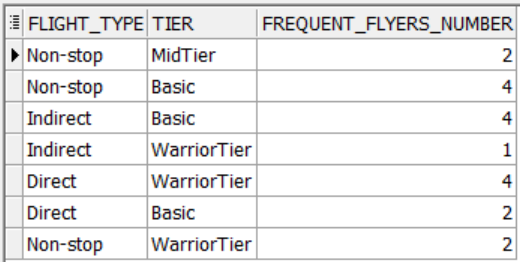
FROM flights\_activity\_Fact fa

JOIN Flights\_DIM fd

ON fa.Flight\_key = fd.Flight\_key

JOIN Frequent\_flyers\_Dim ff

ON ff.Freq\_key = fa.Freq\_key

GROUP BY fd.Flight\_type, ff.Tier ;

**2- Where are the most used hotels for overnights?**

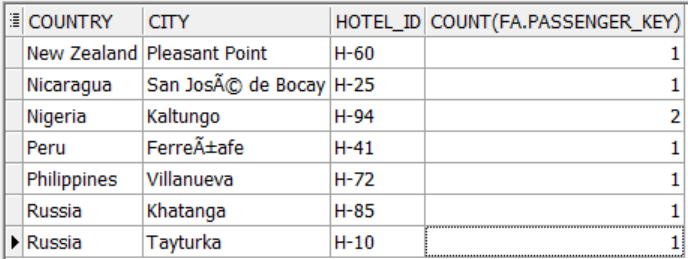
SELECT hd.Country, hd.City, hd.Hotel\_ID , count(fa.passenger\_key)

FROM Hotels\_DIM hd

JOIN flights\_activity\_Fact fa

ON hd.Hotel\_key = fa.Hotel\_key

GROUP BY hd.Country, hd.City, hd.Hotel\_ID

ORDER BY hd.Country ;

**3-** **What is the average number of nights passengers stay in each hotel?**

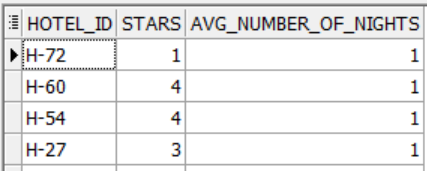
SELECT h.Hotel\_ID, h.Stars, TRUNC(AVG(fa.overnight\_stays),0) AVG\_number\_of\_nights

FROM Hotels\_DIM h

JOIN flights\_activity\_Fact fa

ON fa.Hotel\_key = h.Hotel\_key

GROUP BY h.Hotel\_ID, h.Stars

ORDER BY AVG\_number\_of\_nights DESC;

**4- What is the percentage of passengers to aircraft seats Capacity for every flight in each aircraft If there were stop stations?**

SELECT fa.Flight\_key, fa.Segment\_number, ac.Aircraft\_key,

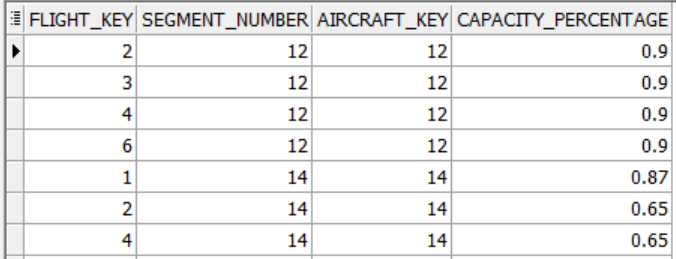
Round( (COUNT(fa.Passenger\_key) / ac.Seats\_Capacity)\* 100,2) AS Capacity\_Percentage

FROM flights\_activity\_Fact fa

JOIN Aircrafts\_Dim ac

ON fa.Aircraft\_key = ac.Aircraft\_key

GROUP BY fa.Flight\_key, fa.Segment\_number, ac.Aircraft\_key, ac.Seats\_Capacity

ORDER BY Capacity\_Percentage DESC ;

**5-** **What are the numbers of complaints based on their Problem severity per year?**

SELECT d.Year, i.TYPE, i.Interaction\_time, f.Problem\_severity,

COUNT (f.Problem\_Severity) AS "No. of Complaints", i.Description

FROM Interactions\_DIM i

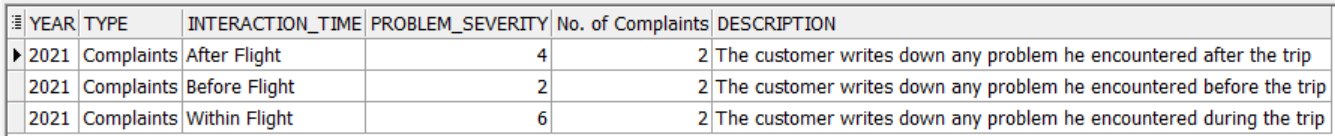
JOIN Customer\_Care\_fact f

ON i.Interaction\_key = f.Interaction\_key

JOIN Date\_Dim d

ON d.Full\_date = f.Reservation\_Ticket\_Date

WHERE i.TYPE = 'Complaints'

GROUP BY d.YEAR, i.TYPE, i.Interaction\_time, f.Problem\_severity, i.Description ;

**6- Which is the most severity interaction complaint for each year?**

SELECT d.Year, i.TYPE, MAX(f.Problem\_Severity)

FROM Interactions\_DIM i

JOIN Customer\_Care\_fact f

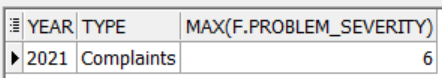
ON i.Interaction\_key = f.Interaction\_key

JOIN Date\_Dim d

ON d.Full\_date = f.Reservation\_Ticket\_Date

WHERE i.TYPE = 'Complaints'

GROUP BY d.Year, i.TYPE ;



**7- Find the interaction type for inquiries and complaints and the employee who worked on it**

SELECT i.TYPE, i.Interaction\_time, f.Problem\_severity, st.NAME AS Customer\_Services\_name

FROM Interactions\_DIM i

JOIN Customer\_Care\_fact f

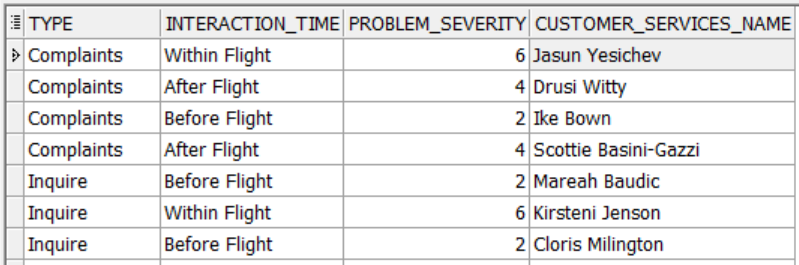
ON i.Interaction\_key = f.Interaction\_key

JOIN Staff\_Dim st

ON f.Staff\_key = st.Staff\_key

WHERE i.TYPE = 'Complaints' OR i.Type = 'Inquire'

ORDER BY i.TYPE;



**8- What are the numbers of Feedbacks Ratings based on their degree of rate per year?**

SELECT d.Year, i.TYPE, f.Feedback\_Rate,

COUNT(f.Feedback\_Rate) AS "No. of Feedback Ratings", i.Description

FROM Interactions\_DIM i

JOIN Customer\_Care\_fact f

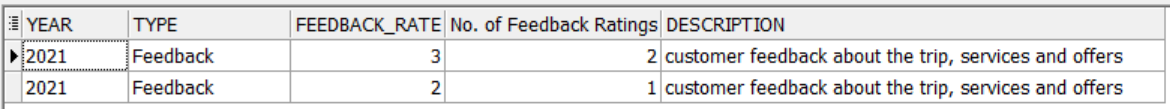
ON i.Interaction\_key = f.Interaction\_key

JOIN Date\_Dim d

ON d.Full\_date = f.Reservation\_Ticket\_Date

WHERE i.TYPE = 'Feedback'

GROUP BY d.Year, i.TYPE, f.Feedback\_Rate, i.Description

ORDER BY COUNT(f.Feedback\_Rate) DESC ;

**9- Which Reservation Channel contributes more to the sales of the company?**

SELECT ch.Channel\_Type AS "Reservation Channel", SUM(fa.Net\_price) AS Total\_sales

FROM flights\_activity\_Fact fa

JOIN Channels\_Dim ch

ON fa.Channel\_key = ch.channel\_key

GROUP BY ch.Channel\_Type

ORDER BY Total\_sales DESC;



**10- What are the profits of the company per Quarter?**

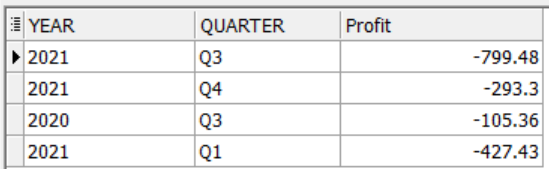
SELECT d.Year, d.Quarter, SUM(fa.Profit) AS "Profit"

FROM flights\_activity\_Fact fa

JOIN Date\_Dim d

ON fa.Reservation\_Ticket\_Date = d.full\_date

GROUP BY d.Year, d.Quarter ;



## Indexes For the Data Warehousing

1. Introduction

Data warehouse solutions have recently grown in significance for decision-makers. Most of the queries against a large data warehouse are complex and iterative. The ability to answer these queries efficiently is a critical issue in the data warehouse environment. Indexing is a way to optimize the performance by minimizing the number of disk accesses required when a query is processed. If the right index structures are built on columns, the performance of queries, especially ad hoc queries, will be greatly enhanced. In this report, we discuss indexing techniques being used. Indexing in a data warehouse may be tricky; too few indexes will result in quick data loads but delayed query responses, while too many indexes will affect data loads slow down and storage requirements go up, but query response is good.

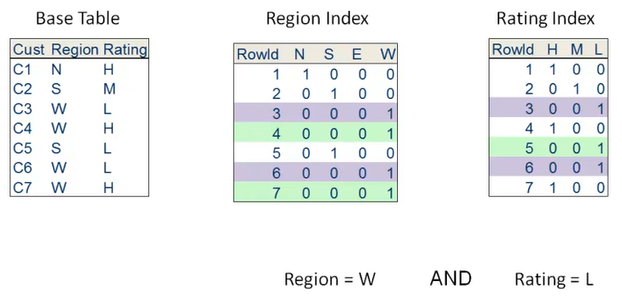
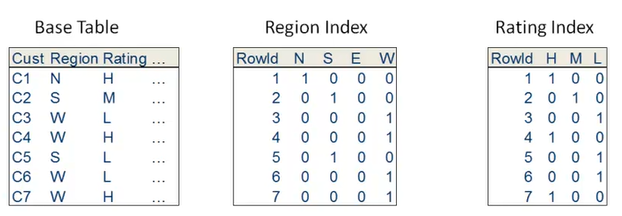
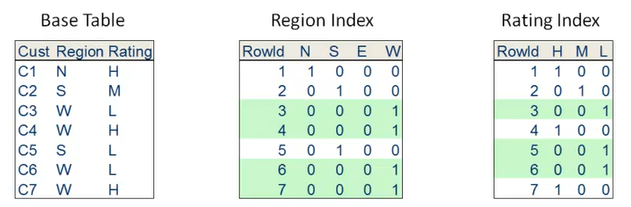
1. Factors used to determine which indexing technique should be built on a Column

Characteristics of indexed column

1. A column has its own characteristics which we can use to choose a proper index. These characteristics are given below:  
   • Cardinality data: The cardinality data of a column is the number of distinct values in the column. It is better to know that the cardinality of an indexed column is low or high since an indexing technique may work efficiently only with either low cardinality or high cardinality.  
   • Distribution: The distribution of a column is the occurrence frequency of each distinct value of the column. The column distribution guides us to determine which index type we should take.  
   • Value range: The range of values of and indexed column guides us to select an appropriate index type. For example, if the range of a high cardinality column is small, an indexing technique based on bitmap should be used. Without knowing this information, we might use a B-Tree resulting in a degradation of system performance.
2. Understanding the data and the usage in the SQL language. Knowing the columns that will be queried helps us choose appropriate index types for them. For example, which columns will likely be a part of the selection list, join constraints, application constraints, the ORDER BY clause, or the GROUP BY clause?
3. Indexing Techniques
   1. B+-Tree indexes

The most used data structures to improve query processing time in operational databases. A B+-Tree indexes on an attribute C of a relation R is a balanced tree providing associative access to the tuples of R on the basis of the values of the C key. The leaves of this tree are linked to each other and store the pointers to the disk blocks containing tuples. Internal nodes create a sort of map to find key values quickly. The top level of the index is called the root. The lowest level is called the leaf node. All other levels in between are called branches. Both the root and branch contain entries that point to the next level in the index. Leaf nodes consisting of the index key and pointers pointing to the physical location in which the corresponding records are stored.

* 1. Bitmap Index  
     The bitmap representation is an alternate method of the row ids representation. It is simple to represent and uses less space and CPU efficient than row ids. when the number of distinct values of the indexed column is low. The indexes improve complex query performance by applying low-cost Boolean operations such as OR, AND, and NOT in the selection predicate on multiple indexes at one time to reduce search space before going to the primary source data. Many variations of the Bitmap Index (Pure Bitmap Index, Encoded Bitmap, etc.) aiming to reduce space requirement as well as improve query performance.



SELECT Customers FROM Base Table WHERE Region = W AND Rating = L

* 1. Join and Star Indexes

Even though star schemata explicitly aim at reducing the number of joins necessary to retrieve data, joins still mainly determine the cost of OLAP queries. Researchers have been working for a long time to reduce the join execution time. To do this, they have developed more efficient join algorithms.

A join index between two relations and . stores couples of tuples RIDS that fulfill the <operator> : join predicate between two attributes and .

Join indexes strongly improve performance because they preprocess and explicitly store RIDs of the matching tuples in two relations. To check for the tuples to fulfill your join predicate, you no longer need to check all the combinations, but you need to scan your index.

1. References

**Book:**

[Matteo Golfarelli](https://www.amazon.com/Matteo-Golfarelli/e/B00279KCKU/ref=dp_byline_cont_book_1), *Data Warehouse Design, Modern Principles and Methodologies*

**Paper:**

*Indexing Techniques for Data Warehouses’ Queries*

*A Comparative Study of Indexing Techniques in Data Warehouse*

*Indexing Techniques to Enhance the Performance of Data Warehouses*

*The impact of indexes on data warehouse performance*

**Website:**

<https://docs.oracle.com/cd/B12037_01/server.101/b10736/indexes.htm#i1006549>