Table of Contents

[ANALYSIS 3](#_Toc121218561)

[(B) – ERD DESIGN 4](#_Toc121218562)

[(C) – RELATIONAL TABLES 5](#_Toc121218563)

[(D) – SQL SCRIPT 6](#_Toc121218564)

[CREATE TABLE STATEMENT FOR (PASSENGER) 6](#_Toc121218565)

[CREATE TABLE STATEMENT FOR (FLIGHT) 6](#_Toc121218566)

[CREATE TABLE STATEMENT FOR (Ticket) 7](#_Toc121218567)

[CREATE TABLE STATEMENT FOR (Booking) 7](#_Toc121218568)

[CREATE TABLE STATEMENT FOR (Employee) 7](#_Toc121218569)

[CREATE TABLE STATEMENT FOR (Enquiries) 8](#_Toc121218570)

[(E) – TABLE POPULATED WITH DATA 9](#_Toc121218571)

[Data in BOOKING Table 9](#_Toc121218572)

[Data in PASSENGER Table 10](#_Toc121218573)

[Data in FLIGHT table 11](#_Toc121218574)

[Data in TICKET table 12](#_Toc121218575)

[Data in EMPLOYEE table 13](#_Toc121218576)

[Data in ENQUIRIES table 14](#_Toc121218577)

[(F) - SQL QUERIES – ‘BETWEEN’ 15](#_Toc121218578)

[EXPLANATION: 15](#_Toc121218579)

[SQL QUERY: 15](#_Toc121218580)

[RESULT: 15](#_Toc121218581)

[(G) - SQL QUERIES – ‘DISTINCT’ 16](#_Toc121218582)

[EXPLANATION: 16](#_Toc121218583)

[SQL QUERY: 16](#_Toc121218584)

[RESULT: 16](#_Toc121218585)

[(H) - SQL QUERIES – ‘TO\_CHAR’ 17](#_Toc121218586)

[EXPLANATION: 17](#_Toc121218587)

[SQL QUERY: 17](#_Toc121218588)

[RESULT: 17](#_Toc121218589)

[(I) - SQL QUERIES – ‘AGGREGATE FUNCTION’ 18](#_Toc121218590)

[EXPLANTION: 18](#_Toc121218591)

[SQL QUERY: 18](#_Toc121218592)

[RESULT: 18](#_Toc121218593)

[(J) - SQL QUERIES – ‘GROUP BY’ 19](#_Toc121218594)

[EXPLANATION: 19](#_Toc121218595)

[SQL QUERY: 19](#_Toc121218596)

[RESULT: 19](#_Toc121218597)

[(K) - SQL QUERIES – ‘SUB QUERY’ 20](#_Toc121218598)

[EXPLANATION: 20](#_Toc121218599)

[SQL QUERY: 20](#_Toc121218600)

[RESULT: 20](#_Toc121218601)

[(L) – ‘DELETE command’ 21](#_Toc121218602)

[WORKED SUCCESSFULLY: 21](#_Toc121218603)

[ERRORED: 22](#_Toc121218604)

[(M) – ‘UPDATE command’ 23](#_Toc121218605)

[WORKED SUCCESSFULLY: 23](#_Toc121218606)

[ERRORED: 24](#_Toc121218607)

[(N) – ‘INSERT command’ 25](#_Toc121218608)

[WORKED SUCCESSFULLY: 25](#_Toc121218609)

[ERRORED: 26](#_Toc121218610)

## ANALYSIS

In my airline company, actively integrating personal data collection and processing will be used to improve passenger experience and performance. The passengers' data will be used effectively to increase airline revenue and promote the loyalty of our passengers. When managed appropriately, this data will be utilised to offer our airline a comprehensive insight into my clients.

for my airline to receive continuous feedback and evaluations from customers to enhance the services we established an independent data company with strong organizational abilities to assist our airline in collecting data from passengers.

The data will be used to categorise passengers and analyse their behaviour patterns to provide relevant services and promotions. This will improve passenger service, increases customer loyalty, and generates more income for the airline.

DATA WEREHOUSING

My company collects data and analytics to provide information to passengers about my airline's market, products, and services. The business involves gathering, organising, and analysing data from a variety of sources, including passenger data, employee data, and inquiries. By giving a full perspective of customer activity, this airline should be able to access whatever data I need. Although understanding a specific customer's transactional patterns sounds enticing, these airlines point out that there are frequent constraints, such as the difficulty of integration concerns and ensuring data is up to date.

BENEFITS AND COSTS

Creating a data analysis service is to assist my company and consumers in making educated decisions. A good data system will be required to maximize the potential profits from data. When the payoff is unknown, the airline will find it difficult to invest. However, there are clear long-term advantages like increase of passengers, investors, and an increasing number of flights.

ETHICAL

We ensure that every regulation protecting our clients' data is fulfilled, and that the employers' data is regularly analyses with feedback from our clients.

Diagram

Description automatically generated

# (B) – ERD DESIGN

## (C) – RELATIONAL TABLES

Passenger (Passenger\_id, Full\_Name, Mobile\_No, Address, Email, DOB)

Booking ( Booking\_id , Date\_, Destination, Departure, Ticket\_id\*)

Flight ( Flight\_id, Employee\_id, Passenger\_id\*, Booking\_id, Duration, Capacity, Departure, Destination)

Employee ( Employee\_id, Name , Qualification, Salary, Role)

Ticket (Ticket\_id, Passenger\_id , Date\_, Flight\_id\* )

Enquiry (Enqiry\_id, enqiry\_reason, Date, Booking\_id, Employee\_id\*)

## (D) – SQL SCRIPT

### CREATE TABLE STATEMENT FOR (PASSENGER)

create table Passenger  
(  
Passenger\_id int not null,  
Full\_Name varchar(100),  
CONSTRAINT Full\_Name check (Full\_Name = Upper(Full\_Name)) ,   
Mobile\_No varchar(20),  
Address varchar(150),  
Email varchar (50),  
DOB date,  
 primary key(Passenger\_id)  
)

### CREATE TABLE STATEMENT FOR (FLIGHT)

create table Flight  
(  
Flight\_id int not null,  
Passenger\_id int,  
Employee\_id int not null,  
Booking\_id int,  
Duration varchar(20),  
Capacity int,   
CONSTRAINT Capacity check (Capacity >= 100),  
Departure varchar(100),  
Destination varchar(100),  
primary key (Flight\_id),  
foreign key(Passenger\_id) references Passenger(Passenger\_id)  
)

### CREATE TABLE STATEMENT FOR (Ticket)

create table Ticket  
(  
Ticket\_id int not null,  
Passenger\_id int not null,  
Flight\_id int not null,  
Date\_ date,  
unique (Passenger\_id),  
primary key(Ticket\_id),  
foreign key (Flight\_id) references FLIGHT (Flight\_id)  
)

### CREATE TABLE STATEMENT FOR (Booking)

create table Booking  
(  
Booking\_id int not null,  
Date\_ date,  
Destination varchar(100),  
Booking\_type varchar(100),  
Departure varchar(100) ,  
CONSTRAINT departure check (departure = upper(departure)) ,  
Ticket\_id int not null,  
primary key (Booking\_id),  
foreign key (Ticket\_id) references Ticket(Ticket\_id)  
)

### CREATE TABLE STATEMENT FOR (Employee)

create table Employee  
(  
Employee\_id int not null,  
Name varchar (100),  
Qualification varchar (100),  
Salary number (6,2),  
CONSTRAINT Salary check (salary >=7000),  
Role varchar (100),  
primary key(Employee\_id)  
)

### CREATE TABLE STATEMENT FOR (Enquiries)

create table Enquiries  
(  
Enqiry\_id int not null,  
Date\_ date,  
enqiry\_reason varchar (1000),  
CONSTRAINT enqiry\_reason check (enqiry\_reason = lower),  
Booking\_id int,  
Employee\_id int not null,  
primary key(Enqiry\_id),  
foreign key(Employee\_id) references Employee(Employee\_id)  
)

## (E) – TABLE POPULATED WITH DATA

### Graphical user interface Description automatically generated with medium confidenceData in BOOKING Table

### Data in PASSENGER Table

**Graphical user interface, text, application

Description automatically generated**

### Graphical user interface, application Description automatically generatedData in FLIGHT table

### Graphical user interface Description automatically generatedData in TICKET table

### Data in EMPLOYEE table

### Graphical user interface, text, application Description automatically generatedData in ENQUIRIES table

## (F) - SQL QUERIES – ‘BETWEEN’

EXPLANATION:   
 The below query is using a BETWEEN function to select data from a table called ENQUIRUES whereby the data selected contains date between 01/01/2009 and 31/12/2018.

### SQL QUERY:

SELECT \* FROM enquiries  
WHERE date\_ BETWEEN '01-JAN-09' AND '31-DEC-18'

### RESULT:

Graphical user interface, text, application, email

Description automatically generated

### 

## (G) - SQL QUERIES – ‘DISTINCT’

### EXPLANATION:

DISTINCT function has been used in conjunction with a select statement to select columns ‘*flight\_id*’ and ‘*destination*’ from a table called FLGHT

### SQL QUERY:

SELECT DISTINCT flight\_id, destination FROM FLIGHT

### RESULT:

Graphical user interface, text, application, email

Description automatically generated

## (H) - SQL QUERIES – ‘TO\_CHAR’

### EXPLANATION:

TO\_CHAR function has been used in the below query to convert a column ‘*passenger\_id*’ which has a data type of *int* into a string value.

### SQL QUERY:

SELECT PASSENGER\_ID,DOB, TO\_CHAR (PASSENGER\_ID, '000000000')  
FROM PASSENGER

### RESULT:

Graphical user interface, text, application, email

Description automatically generated

## - SQL QUERIES – ‘AGGREGATE FUNCTION’

### EXPLANTION:

In the below query, two tables (FLIGHT and TICKET) have been joined together using left outer join.  
The query is to count the number of employees assigned to a specific flight, using their flight\_id and employee\_id.

### SQL QUERY:

SELECT flight.flight\_id, COUNT(flight.employee\_id) FROM FLIGHT   
LEFT OUTER JOIN TICKET   
ON ticket.flight\_id= flight.flight\_id  
GROUP BY flight.flight\_id

### RESULT:

Graphical user interface, text

Description automatically generated

## (J) - SQL QUERIES – ‘GROUP BY’

### EXPLANATION:

Within the query below, two tables(Enquiries and Employee) have been join using right outer join.  
A COUNT aggregate has been used on column called qualifications which is within the employee table.  
The results are to count the number of employees with a specific qualifications and enquiry reason recorded.

### SQL QUERY:

SELECT enquiries.enqiry\_reason, COUNT(employee.qualification),max(employee.qualification)   
FROM enquiries  
RIGHT OUTER JOIN employee   
ON employee.employee\_id = enquiries.employee\_id  
group by enquiries.enqiry\_reason

### RESULT:

Graphical user interface, text, email

Description automatically generated

## (K) - SQL QUERIES – ‘SUB QUERY’

### EXPLANATION:

The subquery which was generated from table(Passenger) was then join to the main query which was generated from table (Flight), using left outer join. Basically, the results select columns from each query, that is flight\_id, passenger\_id, full\_name, duration.

### SQL QUERY:

SELECT f.flight\_id, p.passenger\_id, p.full\_name, f.duration   
FROM flight F  
LEFT OUTER JOIN (  
SELECT \* FROM passenger   
WHERE dob BETWEEN '01-JAN-1980' AND '31-DEC-1989') P   
on P.passenger\_id = f.passenger\_id

### RESULT:

Graphical user interface, text, application, email

Description automatically generated

## (L) – ‘DELETE command’

### WORKED SUCCESSFULLY:

#### QUERY:

DELETE FROM employee  
WHERE qualification = 'Degree'

#### EXPLANATION:

The above query deletes rows where it has “Degree” under the *qualification* column.

#### EVIDENCE:

Graphical user interface, text, application, email

Description automatically generated

### ERRORED:

#### QUERY:

DELETE FROM booking   
(  
CASE   
 WHEN ticket\_id IS NULL   
 THEN 2000 ELSE ticket\_id   
 END  
 )

#### EXPLANATION:

The query above is meant to delete a specific row with the CASE statement.

#### EVIDENCE:

Graphical user interface, text, application

Description automatically generated

## (M) – ‘UPDATE command’

### WORKED SUCCESSFULLY:

#### QUERY:

UPDATE ticket   
SET  
date\_ = NULL  
WHERE passenger\_id = 2020

#### EXPLANATION:

The above query executes a command, where *passenger\_id* is 2020, change its corresponding data in the *date\_* column and NULL (Blank).

#### EVIDENCE:

Graphical user interface, text, application, email

Description automatically generated

### ERRORED:

#### QUERY:

UPDATE ticket   
SET  
passenger\_id = NULL  
WHERE ticket\_id = 2020

#### EXPLANATION:

The above query errored because the *passenger\_id* is a foreign key in the *ticket* table which was set to NOT NULL in the *ticket* table and *passenger* table which it happens to be a primary key.

#### EVIDENCE:

Graphical user interface, text, application, email

Description automatically generated

## (N) – ‘INSERT command’

### WORKED SUCCESSFULLY:

#### QUERY:

INSERT INTO ENQUIRIES  
(  
enqiry\_id,  
date\_,  
enqiry\_reason,  
BOOKING\_ID,  
EMPLOYEE\_ID  
)  
 VALUES (1000, '01-dec-22', 'Unknown', 2000, 3000)

#### EXPLANATION:

The above query inserts data into the ­*ENQUIRY* table with the exact data provided in order.

#### EVIDENCE:

Graphical user interface, text, application, email

Description automatically generated

### ERRORED:

#### QUERY:

INSERT INTO passenger  
(  
PASSENGER\_ID,  
full\_name,  
mobile\_no,  
address,  
email,  
dob  
)  
 VALUES (Null, 'No\_Name', '+477586576', 'Uknown', 'No\_Name@icloud.co.uk', '01-Feb-99')

#### EXPLANATION:

The above query is meant to insert data into the *PASSENGER* table with the data provided. However, the column *passenger\_id* is a primary key which doesn’t allow nulls.

#### EVIDENCE:

Graphical user interface, text, application, email

Description automatically generated