CASE STUDY

OF

AIRPORT

MANAGEMENT

SYSTEM

USING

RELATIONAL

DATABASE DESIGN

Case Study in Relational Database Design

- **Title:** Airport Management System

- **Student Names:** Pranav Arora(2010990537)
  Vishal Kumar(2010990786)
  Saksham Kaushal (2010990625)
  Shivam Soni (2010990665)
  Paranav Mahajan(2010990514)

- **Guide:** Jyoti Snehi

Contents

Abstract

The objective of this thesis is to get some hands-on experience for creating the relational schemas and implementing the data extraction queries related to them. Our case study "Airport Management System" is presented. Input for this case study is taken from its informal specification to a relational schema using entity-relationship modeling and its translation to the relational model, to database schema, to implementation of the database, to interactive SQL querying of the installed database (Oracle).

Acknowledgments

Chapter 1: Introduction

Database Management Systems

Database is a collection of data, in which we can retrieve, insert, delete data. It can also be organized into tables, schema, views, etc.

Database Management Systems is a software that is used to manage databases.

It provides us an interface that can be used to perform operations like database creation, creating tables, updating data, etc.

It helps us control data redundancy, and because it is centralized we can easily maintain the database.

We can easily backup the data and restore it in case of a software or hardware failure.

Relational Database Management System:

Relational Database Management System is a collection of programs that help us to create, delete, update, administer and interact with the database. The data in a RDBMS is stored in the form of tables. And it is accessed using *Structured Query Language.*

It is the most popular database system among companies. It also provides us with data dictionaries and metadata which is useful in data handling.They provide a systematic view of data which helps in better understanding and decision making process.

ER Model

ER Model stands for Entity Relationship Model, It displays the relationship of entity sets stored in a database.

It uses different symbols like rectangles to represent entities, ovals to define attributes and diamonds to represent relationships, etc.

It helps in creating the blueprint of the database. Also it gives a preview of how the tables should connect.

Main components and symbols of ER Model

Rectangles: This is used to represent the entity types.
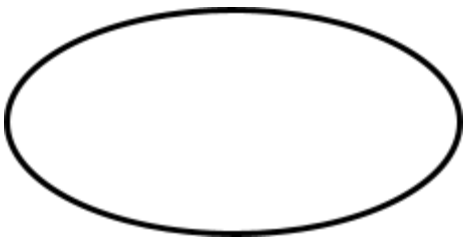
Ellipses: It is used to represent the attributes.

Diamond: It is used to represent relationship types.

Lines: It links attributes to entity types and entity types with other relationship types.
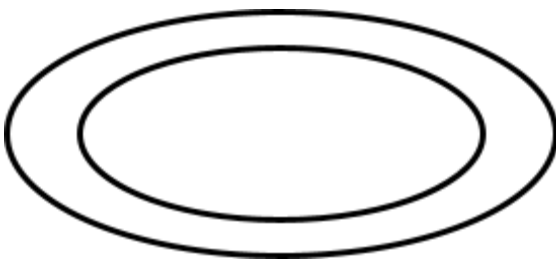
Double Ellipses: It is used to represent multi-valued attributes.

Brief introduction of case study

We have chosen to produce an Airport management System as our case study. In our project we have depicted how airport management works.This provides us with details like airports, the Airline companies, details about passengers and e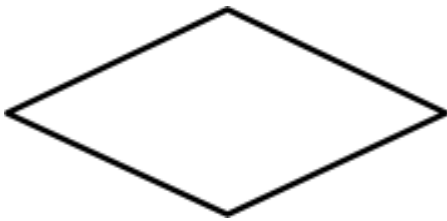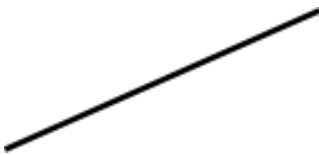mployees. Passengers are eligible for a discount of 10% if they are frequent flyers or are employees in the Airport. Detailed description is discussed further.

Objective of the case study

We made this project to make the work easier for the managing team of the airport. With this they can easily access the details of flights, details of passengers travelling, status of flights, Also it can help them Monitor the flight schedule.

Chapter 2: Airport Management System

An informal specification of the project

In this system a passenger can see how many seats are available in his particular flight, the flight status, and check if he is eligible for discounts.
If an employee travels, he/she is also eligible for employee discounts.
Passengers can also get discounts if they have flight points they can redeem those points.

An Entity-Relationship Model (Made using erdplus.com)

Case study Logical Model (made using Oracle SQL Developer)

EMPLOYEE_DETAIL
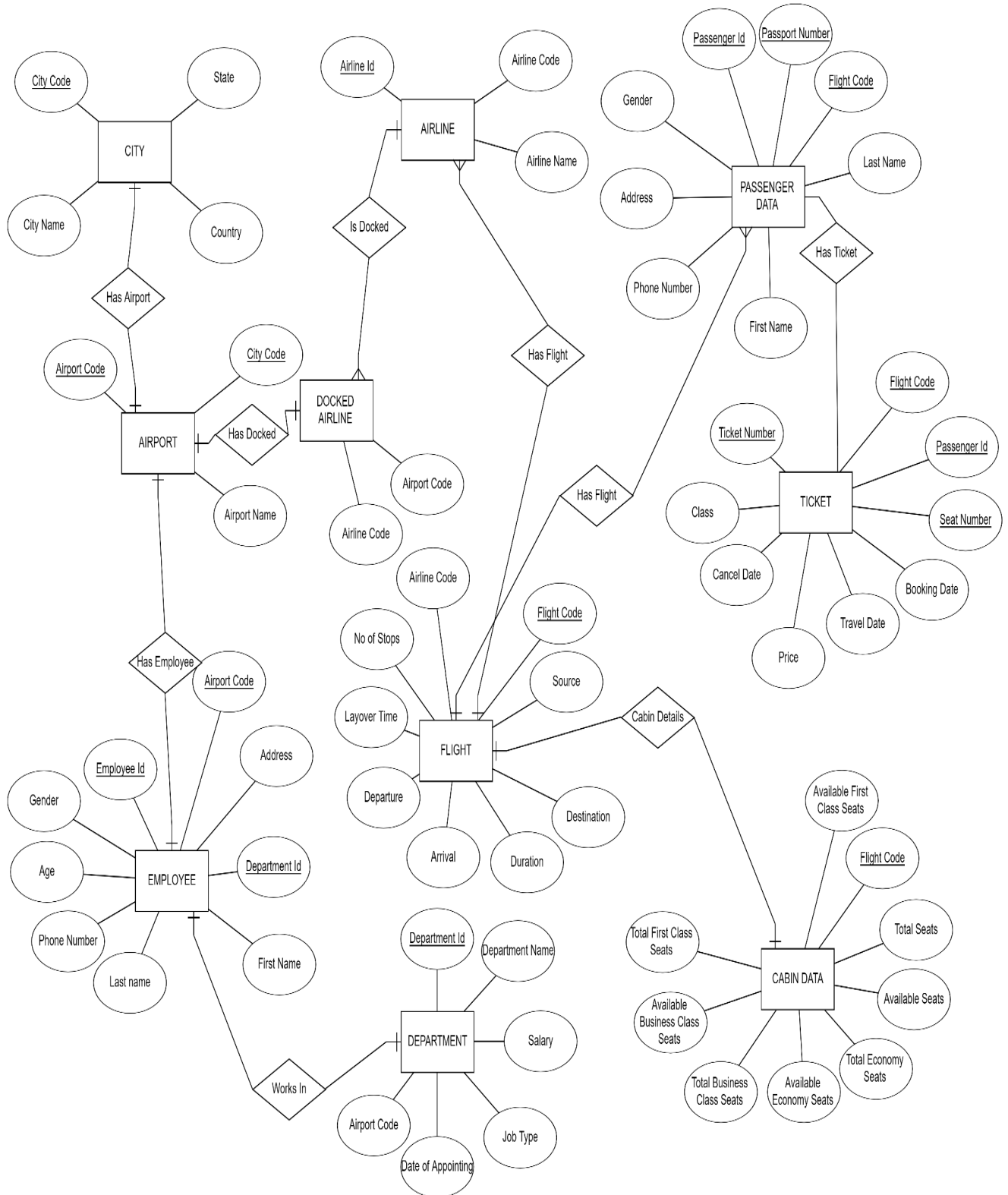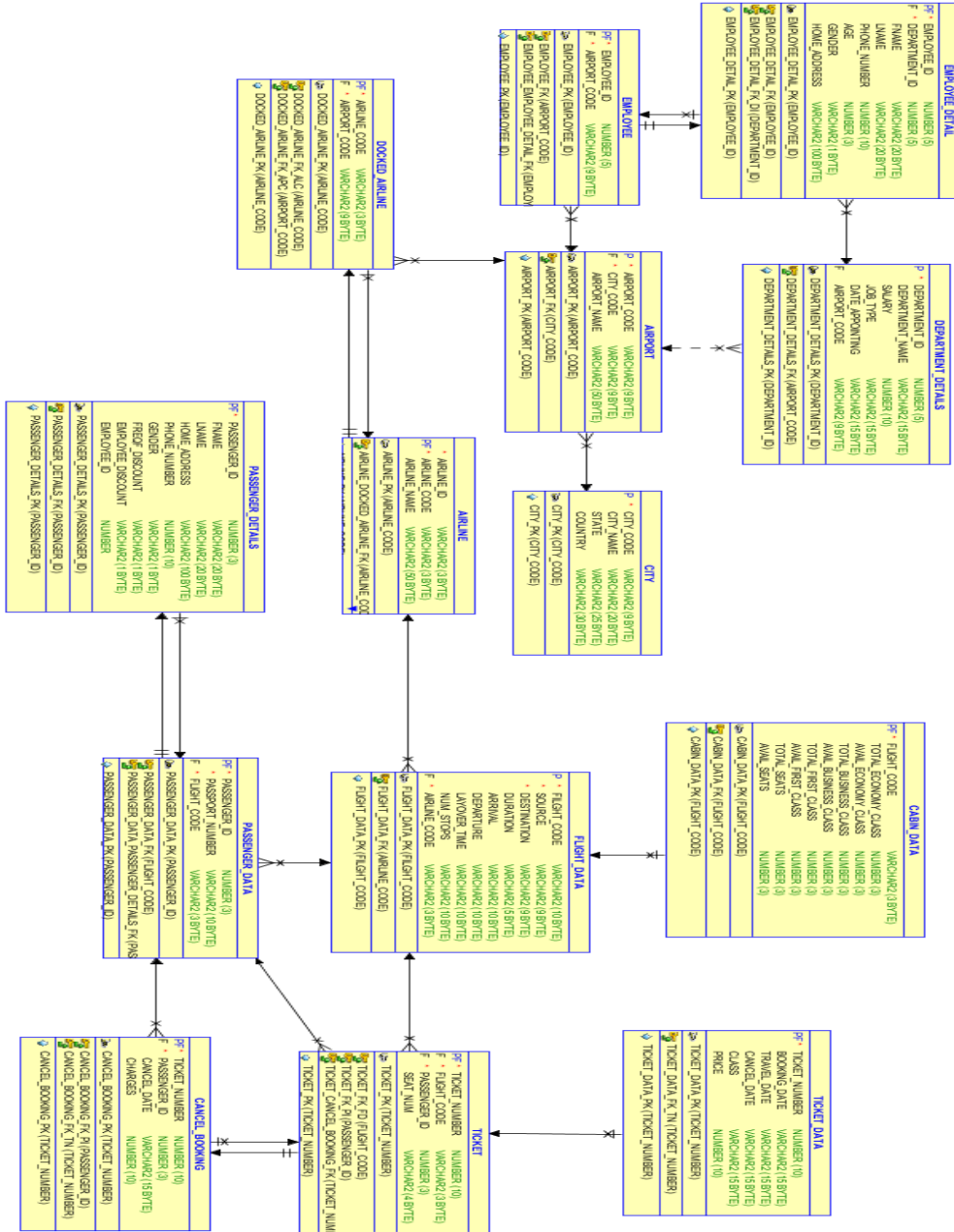- PF * EMPLOYEE_ID — NUMBER (5)
- F * DEPARTMENT_ID — NUMBER (5)
- * FNAME — VARCHAR2 (20 BYTE)
- * LNAME — VARCHAR2 (20 BYTE)
- PHONE_NUMBER — NUMBER (10)
- AGE — NUMBER (3)
- GENDER — VARCHAR2 (1 BYTE)
- HOME_ADDRESS — VARCHAR2 (100 BYTE)
- EMPLOYEE_DETAIL_FK (EMPLOYEE_ID)
- EMPLOYEE_DETAIL_FK_D (DEPARTMENT_ID)
- EMPLOYEE_DETAIL_PK (EMPLOYEE_ID)

EMPLOYEE
- PF * EMPLOYEE_ID — NUMBER (5)
- F * AIRPORT_CODE — VARCHAR2 (9 BYTE)
- EMPLOYEE_FK (AIRPORT_CODE)
- EMPLOYEE_PK (EMPLOYEE_ID)
- EMPLOYEE_EMPLOYEE_DETAIL_FK (EMPLOY...)

DOCKED_AIRLINE
- PF * AIRLINE_CODE — VARCHAR2 (13 BYTE)
- F * AIRPORT_CODE — VARCHAR2 (9 BYTE)
- DOCKED_AIRLINE_FK_ALC (AIRLINE_CODE)
- DOCKED_AIRLINE_FK_APC (AIRPORT_CODE)
- DOCKED_AIRLINE_PK (AIRLINE_CODE)

DEPARTMENT_DETAILS
- P * DEPARTMENT_ID — NUMBER (5)
- * DEPARTMENT_NAME — VARCHAR2 (18 BYTE)
- SALARY — NUMBER (10)
- JOB_TYPE — VARCHAR2 (15 BYTE)
- DATE_APPOINTING — VARCHAR2 (15 BYTE)
- F AIRPORT_CODE — VARCHAR2 (9 BYTE)
- DEPARTMENT_DETAILS_FK (AIRPORT_CODE)
- DEPARTMENT_DETAILS_PK (DEPARTMENT_ID)

AIRPORT
- P * AIRPORT_CODE — VARCHAR2 (9 BYTE)
- F * CITY_CODE — VARCHAR2 (9 BYTE)
- * AIRPORT_NAME — VARCHAR2 (30 BYTE)
- AIRPORT_FK (CITY_CODE)
- AIRPORT_PK (AIRPORT_CODE)

CITY
- P * CITY_CODE — VARCHAR2 (9 BYTE)
- CITY_NAME — VARCHAR2 (20 BYTE)
- STATE — VARCHAR2 (25 BYTE)
- COUNTRY — VARCHAR2 (30 BYTE)
- CITY_PK (CITY_CODE)

AIRLINE
- P * AIRLINE_ID
- PF * AIRLINE_CODE — VARCHAR2 (13 BYTE)
- * AIRLINE_NAME — VARCHAR2 (30 BYTE)
- AIRLINE_DOCKED_AIRLINE_FK (AIRLINE_COD...)
- AIRLINE_PK (AIRLINE_CODE)

PASSENGER_DETAILS
- PF * PASSENGER_ID — NUMBER (3)
- * FNAME — VARCHAR2 (20 BYTE)
- * LNAME — VARCHAR2 (20 BYTE)
- HOME_ADDRESS — VARCHAR2 (100 BYTE)
- PHONE_NUMBER — NUMBER (10)
- GENDER — VARCHAR2 (1 BYTE)
- FREOF_DISCOUNT — VARCHAR2 (1 BYTE)
- EMPLOYEE_DISCOUNT — VARCHAR2 (1 BYTE)
- EMPLOYEE_ID — NUMBER
- PASSENGER_DETAILS_FK (PASSENGER_ID)
- PASSENGER_DETAILS_PK (PASSENGER_ID)

FLIGHT_DATA
- P * FLIGHT_CODE — VARCHAR2 (10 BYTE)
- * SOURCE — VARCHAR2 (9 BYTE)
- * DESTINATION — VARCHAR2 (9 BYTE)
- DURATION — VARCHAR2 (5 BYTE)
- ARRIVAL — VARCHAR2 (10 BYTE)
- DEPARTURE — VARCHAR2 (10 BYTE)
- LAYOVER_TIME — VARCHAR2 (10 BYTE)
- NUM_STOPS — NUMBER (10)
- F AIRLINE_CODE — VARCHAR2 (13 BYTE)
- FLIGHT_DATA_FK (AIRLINE_CODE)
- FLIGHT_DATA_PK (FLIGHT_CODE)

CABIN_DATA
- PF * FLIGHT_CODE — VARCHAR2 (13 BYTE)
- TOTAL_ECONOMY_CLASS — NUMBER (3)
- AVAIL_ECONOMY_CLASS — NUMBER (3)
- TOTAL_BUSINESS_CLASS — NUMBER (3)
- AVAIL_BUSINESS_CLASS — NUMBER (3)
- TOTAL_FIRST_CLASS — NUMBER (3)
- AVAIL_FIRST_CLASS — NUMBER (3)
- TOTAL_SEATS — NUMBER (3)
- AVAIL_SEATS — NUMBER (3)
- CABIN_DATA_FK (FLIGHT_CODE)
- CABIN_DATA_PK (FLIGHT_CODE)

PASSENGER_DATA
- PF * PASSENGER_ID — NUMBER (3)
- * PASSPORT_NUMBER — VARCHAR2 (10 BYTE)
- F * FLIGHT_CODE — VARCHAR2 (10 BYTE)
- PASSENGER_DATA_FK (FLIGHT_CODE)
- PASSENGER_DATA_FK (PASSENGER_ID)
- PASSENGER_DATA_PK (PASSENGER_ID)

CANCEL_BOOKING
- PF * TICKET_NUMBER — NUMBER (10)
- F * PASSENGER_ID — NUMBER (3)
- CANCEL_DATE — VARCHAR2 (15 BYTE)
- CHARGES — NUMBER (10)
- CANCEL_BOOKING_FK (TICKET_NUMBER)
- CANCEL_BOOKING_FK_TN (TICKET_NUMBER)
- CANCEL_BOOKING_PK (TICKET_NUMBER)

TICKET
- PF * TICKET_NUMBER — NUMBER (10)
- F * FLIGHT_CODE — VARCHAR2 (13 BYTE)
- F * PASSENGER_ID — NUMBER (3)
- SEAT_NUM — VARCHAR2 (4 BYTE)
- TICKET_FK_FD (FLIGHT_CODE)
- TICKET_FK_PI (PASSENGER_ID)
- TICKET_CANCEL_BOOKING_FK (TICKET_NUM...)
- TICKET_PK (TICKET_NUMBER)

TICKET_DATA
- PF * TICKET_NUMBER — NUMBER (10)
- BOOKING_DATE — VARCHAR2 (15 BYTE)
- TRAVEL_DATE — VARCHAR2 (15 BYTE)
- CANCEL_DATE — VARCHAR2 (15 BYTE)
- CLASS — VARCHAR2 (15 BYTE)
- PRICE — NUMBER (10)
- TICKET_DATA_FK (TICKET_NUMBER)
- TICKET_DATA_PK (TICKET_NUMBER)

Database Schema

```sql
CREATE table "CITY" (
    "CITY_CODE"  VARCHAR2(9) NOT NULL,
    "CITY_NAME"  VARCHAR2(20),
    "STATE"      VARCHAR2(25),
    "COUNTRY"    VARCHAR2(30),
    constraint  "CITY_PK" primary key ("CITY_CODE")
);
```

```sql
CREATE table "AIRPORT" (
    "AIRPORT_CODE" VARCHAR2(9) NOT NULL,
    "CITY_CODE"    VARCHAR2(9) NOT NULL,
    "AIRPORT_NAME" VARCHAR2(50),
    constraint  "AIRPORT_PK" primary key ("AIRPORT_CODE")
);
ALTER TABLE "AIRPORT" ADD CONSTRAINT "AIRPORT_FK"
FOREIGN KEY ("CITY_CODE")
REFERENCES "CITY" ("CITY_CODE")
ON DELETE CASCADE;
```

```sql
CREATE table "AIRLINE" (
    "AIRLINE_ID"   VARCHAR2(3) NOT NULL,
    "AIRLINE_CODE" VARCHAR2(3) NOT NULL,
    "AIRLINE_NAME" VARCHAR2(50),
    constraint  "AIRLINE_PK" primary key ("AIRLINE_CODE")
);
```

```sql
CREATE table "DOCKED_AIRLINE" (
    "AIRLINE_CODE" VARCHAR2(3) NOT NULL,
    "AIRPORT_CODE" VARCHAR2(9) NOT NULL,
    constraint  "DOCKED_AIRLINE_PK" primary key ("AIRLINE_CODE")
);
ALTER TABLE "DOCKED_AIRLINE" ADD CONSTRAINT "DOCKED_AIRLINE_FK_APC"
FOREIGN KEY ("AIRPORT_CODE")
REFERENCES "AIRPORT" ("AIRPORT_CODE")
ON DELETE CASCADE;
```

```sql
ALTER TABLE "DOCKED_AIRLINE" ADD CONSTRAINT "DOCKED_AIRLINE_FK_ALC"
FOREIGN KEY ("AIRLINE_CODE")
REFERENCES "AIRLINE" ("AIRLINE_CODE")
ON DELETE CASCADE;
```

```sql
CREATE table "FLIGHT_DATA" (
    "FILGHT_CODE"   VARCHAR2(10) NOT NULL,
    "SOURCE"        VARCHAR2(9) NOT NULL,
    "DESTINATION"   VARCHAR2(9) NOT NULL,
    "DURATION"      VARCHAR2(5),
    "ARRIVAL"       VARCHAR2(10),
    "DEPARTURE"     VARCHAR2(10),
    "LAYOVER_TIME"  VARCHAR2(10),
    "NUM_STOPS"     VARCHAR2(10),
    "AIRLINE_CODE"  VARCHAR2(3) NOT NULL,
    constraint  "FLIGHT_DATA_PK" primary key ("FILGHT_CODE")
);
ALTER TABLE "FLIGHT_DATA" ADD CONSTRAINT "FLIGHT_DATA_FK"
FOREIGN KEY ("AIRLINE_CODE")
REFERENCES "AIRLINE" ("AIRLINE_CODE")
ON DELETE CASCADE;


CREATE table "CABIN_DATA" (
    "FLIGHT_CODE"           VARCHAR2(10) NOT NULL,
    "TOTAL_ECONOMY_CLASS"   NUMBER(3),
    "AVAIL_ECONOMY_CLASS"   NUMBER(3),
    "TOTAL_BUSINESS_CLASS"  NUMBER(3),
    "AVAIL_BUSINESS_CLASS"  NUMBER(3),
    "TOTAL_FIRST_CLASS"     NUMBER(3),
    "AVAIL_FIRST_CLASS"     NUMBER(3),
    "TOTAL_SEATS"           NUMBER(3),
    "AVAIL_SEATS"           NUMBER(3),
    constraint  "CABIN_DATA_PK" primary key ("FLIGHT_CODE")
);
ALTER TABLE "CABIN_DATA" ADD CONSTRAINT "CABIN_DATA_FK"
FOREIGN KEY ("FLIGHT_CODE")
REFERENCES "FLIGHT_DATA" ("FILGHT_CODE")
ON DELETE CASCADE;


CREATE table "PASSENGER_DATA" (
    "PASSENGER_ID"      NUMBER(3) NOT NULL,
    "PASSPORT_NUMBER"   VARCHAR2(10) NOT NULL,
    "FLIGHT_CODE"       VARCHAR2(10) NOT NULL,
    constraint  "PASSENGER_DATA_PK" primary key ("PASSENGER_ID")
);
ALTER TABLE "PASSENGER_DATA" ADD CONSTRAINT "PASSENGER_DATA_FK"
FOREIGN KEY ("FLIGHT_CODE")
REFERENCES "FLIGHT_DATA" ("FILGHT_CODE")
ON DELETE CASCADE;


CREATE table "PASSENGER_DETAILS" (
    "PASSENGER_ID"      NUMBER(3) NOT NULL,
```

```sql
   "FNAME"          VARCHAR2(20),
   "LNAME"          VARCHAR2(20),
   "HOME_ADDRESS"     VARCHAR2(100),
   "PHONE_NUMBER"     NUMBER(10),
   "GENDER"          VARCHAR2(1),
   "FREQF_DISCOUNT"   VARCHAR2(1),
   "EMPLOYEE_DISCOUNT" VARCHAR2(1),
   "EMPLOYEE_ID"     NUMBER(10),
   constraint  "PASSENGER_DETAILS_PK" primary key ("PASSENGER_ID")
);
ALTER TABLE "PASSENGER_DETAILS" ADD CONSTRAINT "PASSENGER_DETAILS_FK"
FOREIGN KEY ("PASSENGER_ID")
REFERENCES "PASSENGER_DATA" ("PASSENGER_ID")
ON DELETE CASCADE;


CREATE table "TICKET" (
   "TICKET_NUMBER" NUMBER(10) NOT NULL,
   "FLIGHT_CODE"   VARCHAR2(10) NOT NULL,
   "PASSENGER_ID"  NUMBER(3) NOT NULL,
   "SEAT_NUM"      VARCHAR2(4),
   constraint  "TICKET_PK" primary key ("TICKET_NUMBER")
);
ALTER TABLE "TICKET" ADD CONSTRAINT "TICKET_FK_FD"
FOREIGN KEY ("FLIGHT_CODE")
REFERENCES "FLIGHT_DATA" ("FILGHT_CODE")
ON DELETE CASCADE;
ALTER TABLE "TICKET" ADD CONSTRAINT "TICKET_FK_PI"
FOREIGN KEY ("PASSENGER_ID")
REFERENCES "PASSENGER_DATA" ("PASSENGER_ID")
ON DELETE CASCADE;
CREATE table "TICKET_DATA" (
   "TICKET_NUMBER" NUMBER(10) NOT NULL,
   "BOOKING_DATE"  VARCHAR2(15),
   "TRAVEL_DATE"   VARCHAR2(15),
   "CANCEL_DATE"   VARCHAR2(15),
   "CLASS"         VARCHAR2(15),
   "PRICE"         NUMBER(10),
   constraint  "TICKET_DATA_PK" primary key ("TICKET_NUMBER")
);

ALTER TABLE "TICKET_DATA" ADD CONSTRAINT "TICKET_DATA_FK_TN"
FOREIGN KEY ("TICKET_NUMBER")
REFERENCES "TICKET" ("TICKET_NUMBER")
ON DELETE CASCADE;


CREATE table "CANCEL_BOOKING" (
   "TICKET_NUMBER" NUMBER(10) NOT NULL,
   "PASSENGER_ID"  NUMBER(3) NOT NULL,
   "CANCEL_DATE"   VARCHAR2(15),
```

```sql
    "CHARGES"      NUMBER(10),
    constraint  "CANCEL_BOOKING_PK" primary key ("TICKET_NUMBER")
);
ALTER TABLE "CANCEL_BOOKING" ADD CONSTRAINT "CANCEL_BOOKING_FK_TN"
FOREIGN KEY ("TICKET_NUMBER")
REFERENCES "TICKET" ("TICKET_NUMBER")
ON DELETE CASCADE;
ALTER TABLE "CANCEL_BOOKING" ADD CONSTRAINT "CANCEL_BOOKING_FK_PI"
FOREIGN KEY ("PASSENGER_ID")
REFERENCES "PASSENGER_DATA" ("PASSENGER_ID")
ON DELETE CASCADE;


CREATE table "EMPLOYEE" (
    "EMPLOYEE_ID"  NUMBER(5) NOT NULL,
    "AIRPORT_CODE" VARCHAR2(9) NOT NULL,
    constraint  "EMPLOYEE_PK" primary key ("EMPLOYEE_ID")
);
ALTER TABLE "EMPLOYEE" ADD CONSTRAINT "EMPLOYEE_FK"
FOREIGN KEY ("AIRPORT_CODE")
REFERENCES "AIRPORT" ("AIRPORT_CODE")
ON DELETE CASCADE;

CREATE table "EMPLOYEE_DETAIL" (
   "EMPLOYEE_ID"   NUMBER(5) NOT NULL,
   "DEPARTMENT_ID" NUMBER(5) NOT NULL,
   "FNAME"       VARCHAR2(20),
   "LNAME"       VARCHAR2(20),
   "PHONE_NUMBER"  NUMBER(10),
   "AGE"         NUMBER(3),
   "GENDER"      VARCHAR2(1),
   "HOME_ADDRESS"  VARCHAR2(100),
   constraint  "EMPLOYEE_DETAIL_PK" primary key ("EMPLOYEE_ID")
);
ALTER TABLE "EMPLOYEE_DETAIL" ADD CONSTRAINT "EMPLOYEE_DETAIL_FK"
FOREIGN KEY ("EMPLOYEE_ID")
REFERENCES "EMPLOYEE" ("EMPLOYEE_ID")
ON DELETE CASCADE;

ALTER TABLE "EMPLOYEE_DETAIL" ADD CONSTRAINT "EMPLOYEE_DETAIL_FK_DI"
FOREIGN KEY ("DEPARTMENT_ID")
REFERENCES "DEPARTMENT_DETAILS" ("DEPARTMENT_ID")
ON DELETE CASCADE;


CREATE table "DEPARTMENT_DETAILS" (
   "DEPARTMENT_ID"   NUMBER(5) NOT NULL,
   "DEPARTMENT_NAME" VARCHAR2(15),
   "SALARY"        NUMBER(10),
   "JOB_TYPE"       VARCHAR2(15),
```

```
  "DATE_APPOINTING" VARCHAR2(15),
  "AIRPORT_CODE"   VARCHAR2(9),
  constraint  "DEPARTMENT_DETAILS_PK" primary key ("DEPARTMENT_ID")
);
ALTER TABLE "DEPARTMENT_DETAILS" ADD CONSTRAINT "DEPARTMENT_DETAILS_FK"
FOREIGN KEY ("AIRPORT_CODE")
REFERENCES "AIRPORT" ("AIRPORT_CODE")
ON DELETE CASCADE;
```

Sequences

EMPLOYEE ID SEQUENCE

```
create sequence "EMPLOYEE_ID_SEQ"
start with 0
increment by 1
maxvalue 99999
minvalue 0
nocache
nocycle
noorder;
```

TICKET NUMBER SEQUENCE

```
create sequence "TICKET_NUMBER_SEQ"
start with 123
increment by 1
maxvalue 99999
minvalue 0
nocache
nocycle
noorder;
```

PASSENGER ID SEQUENCE

```
create sequence "PASSENGER_ID_SEQ"
start with 1
increment by 1
maxvalue 999
minvalue 0
nocache
nocycle
order;
```

Procedures

NEW_PASSENGER_PRO

```
create or replace procedure "NEW_PASSENGER_PRO"
(passport_number IN VARCHAR2, flight_code IN VARCHAR2,
firstname IN VARCHAR2, lastname IN VARCHAR2,
homeaddress IN VARCHAR2, phonenumber IN NUMBER,
p_gender IN VARCHAR2, p_freqfdiscount IN VARCHAR2,
p_employeediscount IN VARCHAR2, p_employeeid IN VARCHAR2,
p_class IN VARCHAR2, p_seat IN VARCHAR2,
p_price IN NUMBER)
```

```
is
BEGIN
INSERT INTO PASSENGER_DATA ("PASSENGER_ID","PASSPORT_NUMBER","FLIGHT_CODE")
VALUES
(PASSENGER_ID_SEQ.nextval, passport_number,flight_code);
COMMIT;
INSERT INTO PASSENGER_DETAILS
("PASSENGER_ID","FNAME","LNAME","HOME_ADDRESS",
"PHONE_NUMBER","GENDER","FREQF_DISCOUNT","EMPLOYEE_DISCOUNT","EMPLOYEE_ID
") VALUES
(PASSENGER_ID_SEQ.currval, firstname, lastname, homeaddress, phonenumber, P_gender,
p_freqfdiscount, p_employeediscount, p_employeeid);
COMMIT;
INSERT INTO TICKET ("TICKET_NUMBER", "FLIGHT_CODE", "PASSENGER_ID", "SEAT_NUM")
VALUES
(TICKET_NUMBER_SEQ.nextval, flight_code, PASSENGER_ID_SEQ.currval, p_seat);
COMMIT;
INSERT INTO TICKET_DATA
("TICKET_NUMBER","BOOKING_DATE","TRAVEL_DATE","CANCEL_DATE","CLASS","PRICE")
VALUES
(TICKET_NUMBER_SEQ.currval, sysdate, sysdate+7, '', p_class, p_price);
COMMIT;
END;
/
```

NEW_EMPLOYEE_PRO

```
create or replace procedure "NEW_EMPLOYEE_PRO"
(airport_code IN VARCHAR2,department_id IN NUMBER,
p_fname IN VARCHAR2,p_lname IN VARCHAR2,
p_phone_num IN NUMBER,p_age IN NUMBER,
p_gender IN VARCHAR2,p_home_address IN VARCHAR2)
is
BEGIN
    INSERT INTO EMPLOYEE ("EMPLOYEE_ID","AIRPORT_CODE") VALUES
    (EMPLOYEE_ID_SEQ.nextval, airport_code);
    COMMIT;

    INSERT INTO EMPLOYEE_DETAILS ("EMPLOYEE_ID","DEPARTMENT_ID","FNAME",
      "LNAME","PHONE_NUMBER","AGE","GENDER","HOME_ADDRESS")
    VALUES
    (EMPLOYEE_ID_SEQ.currval,
department_id,p_fname,p_lname,p_phone_num,p_age,p_gender,p_home_address);
    COMMIT;
END;
/;
```

```sql
create or replace procedure "UPDATE_SEATS_PRO"
(p_flight_code IN VARCHAR2,
p_class IN VARCHAR2,p_seat_num IN VARCHAR2)
is
begin
DECLARE
   T_AVAIL_SEATS number; AVAIL_ECO_SEATS number; AVAIL_BUS_SEATS number;
   AVAIL_FIR_SEATS number;
   FL_CLASS VARCHAR(30);
BEGIN
   SELECT AVAIL_SEATS INTO T_AVAIL_SEATS FROM CABIN_DATA WHERE
FLIGHT_CODE = p_flight_code;
   SELECT AVAIL_ECONOMY_CLASS INTO AVAIL_ECO_SEATS FROM CABIN_DATA
WHERE FLIGHT_CODE = p_flight_code;
   SELECT AVAIL_BUSINESS_CLASS INTO AVAIL_BUS_SEATS FROM CABIN_DATA
WHERE FLIGHT_CODE = p_flight_code;
   SELECT AVAIL_FIRST_CLASS INTO AVAIL_FIR_SEATS FROM CABIN_DATA WHERE
FLIGHT_CODE = p_flight_code;
   IF T_AVAIL_SEATS > 0 THEN
     dbms_output.put_line('ALL SEATS ARE FULL IN FLIGHT');
   ELSE
     IF p_class = 'economy' OR p_class = 'ECONOMY' OR p_class = 'ECO' THEN
        FL_CLASS:='AVAIL_ECONOMY_CLASS';
        IF AVAIL_ECO_SEATS <= 0 THEN
          dbms_output.put_line('ALL SEATS ARE FULL IN ECONOMY CLASS');
        ELSE
          UPDATE CABIN_DATA SET AVAIL_SEATS=AVAIL_SEATS-1,
          AVAIL_ECONOMY_CLASS=AVAIL_ECONOMY_CLASS-1
          WHERE FLIGHT_CODE=p_flight_code;
          COMMIT;
        END IF;
     ELSIF p_class = 'BUSINESS' OR p_class = 'business' OR p_class = 'busi' THEN
        FL_CLASS:='AVAIL_BUSINESS_CLASS';
        FL_CLASS:='AVAIL_ECONOMY_CLASS';
        IF AVAIL_ECO_SEATS <= 0 THEN
          dbms_output.put_line('ALL SEATS ARE FULL IN BUSINESS CLASS');
        ELSE
          UPDATE CABIN_DATA SET AVAIL_SEATS=AVAIL_SEATS-1,
          AVAIL_BUSINESS_CLASS=AVAIL_BUSINESS_CLASS-1
          WHERE FLIGHT_CODE=p_flight_code;
          COMMIT;
        END IF;
     ELSE
        FL_CLASS:='AVAIL_FIRST_CLASS';
        IF AVAIL_ECO_SEATS <= 0 THEN
          dbms_output.put_line('ALL SEATS ARE FULL IN BUSINESS CLASS');
        ELSE
```

```
        UPDATE CABIN_DATA SET AVAIL_SEATS=AVAIL_SEATS-1,
        AVAIL_FIRST_CLASS=AVAIL_FIRST_CLASS-1
        WHERE FLIGHT_CODE=p_flight_code;
        COMMIT;
      END IF;
   END IF;
  END IF;
END;
END;
```

Case Study Interactive Queries

```
INSERT INTO CITY ("CITY_CODE","CITY_NAME","STATE","COUNTRY")
VALUES('LKUS','Louisville','Kentucky','United States');
INSERT INTO CITY ("CITY_CODE","CITY_NAME","STATE","COUNTRY")
VALUES('CCI','Chandigarh','Chandigarh','India');
```

```
INSERT INTO AIRPORT ("AIRPORT_NAME","AIRPORT_CODE","CITY_CODE")
VALUES('Louisville International Airport','SDF','LKUS');
INSERT INTO AIRPORT ("AIRPORT_NAME","AIRPORT_CODE","CITY_CODE")
VALUES('Chandigarh International Airport','IXC','CCI');
```

```
INSERT INTO AIRLINE ("AIRLINE_ID","AIRLINE_NAME","AIRLINE_CODE") VALUES('AI','Air
India Limited','098');
INSERT INTO AIRLINE ("AIRLINE_ID","AIRLINE_NAME","AIRLINE_CODE")
VALUES('LH','Lufthansa', '220');
```

```
INSERT INTO DOCKED_AIRLINE ("AIRLINE_CODE","AIRPORT_CODE") VALUES('001','SDF');
INSERT INTO DOCKED_AIRLINE ("AIRLINE_CODE","AIRPORT_CODE") VALUES('001','JFK');
```

```
INSERT INTO
FLIGHT_DATA("FLIGHT_CODE","SOURCE","DESTINATION","DURATION","ARRIVAL","DEPART
URE","LAYOVER_TIME","NUM_STOPS","AIRLINE_CODE")
VALUES('AI2014','BOM','DFW','24hr','02:10','03:15' ,'3', '1','098');
INSERT INTO
FLIGHT_DATA("FLIGHT_CODE","SOURCE","DESTINATION","DURATION","ARRIVAL","DEPART
URE","LAYOVER_TIME","NUM_STOPS","AIRLINE_CODE")
VALUES('EY1234','JFK','TPA','3hrs','19:20','20:05' ,'5', '2','607');
```

Insertion of Passenger Related Data using procedure

```
BEGIN
  NEW_PASSENGER_PRO('A1234568','AI2014','Pranav','Arora',
'123-sector-28-panchkula',9877012312,'M','N','N','','FIRST-CLASS','B021',12345);
END;
```

Insertion of Employee Related Data using procedure

```
BEGIN
  NEW_EMPLOYEE_PRO('SDF','125','carl','lewis',0123456795,26,'M','7820 MCCALLUM
COURTS, APT 234, AKRON, OH');
END;
```

Conclusion

While working on this project, we learnt a lot about creating a database and implementing all the queries related to creation of the database, modifying it and fetching the data from it. We have a good hold on the concepts related to ER diagrams and normalization now.
We also learned about procedures, sequences by this project and triggers.

Bibliography

- https://www.javatpoint.com/dbms-functional-dependency

- https://docs.oracle.com/cd/B19306_01/server.102/b14200/statements_6009.htm

- https://docs.oracle.com/cd/B12037_01/server.101/b10759/statements_6014.htm

- https://www.techonthenet.com/oracle/sequences.php

- https://www.techonthenet.com/oracle/procedures.php

- https://www.javatpoint.com/dbms-er-model-concept

- https://www.guru99.com/er-diagram-tutorial-dbms.html

- https://www.techonthenet.com/oracle/tables/alter_table.php