Database Management Systems Coursework 1

F28DM

Members: Joanne, Ismail Marashi, Ishaq Marashy, Savannah

Contributions:

Joanne- loaded database with data, report

Ismail- made the database, ER diagram

Ishaq- relational schema, Java

Savannah- loaded database with data

Extended description of the scenario

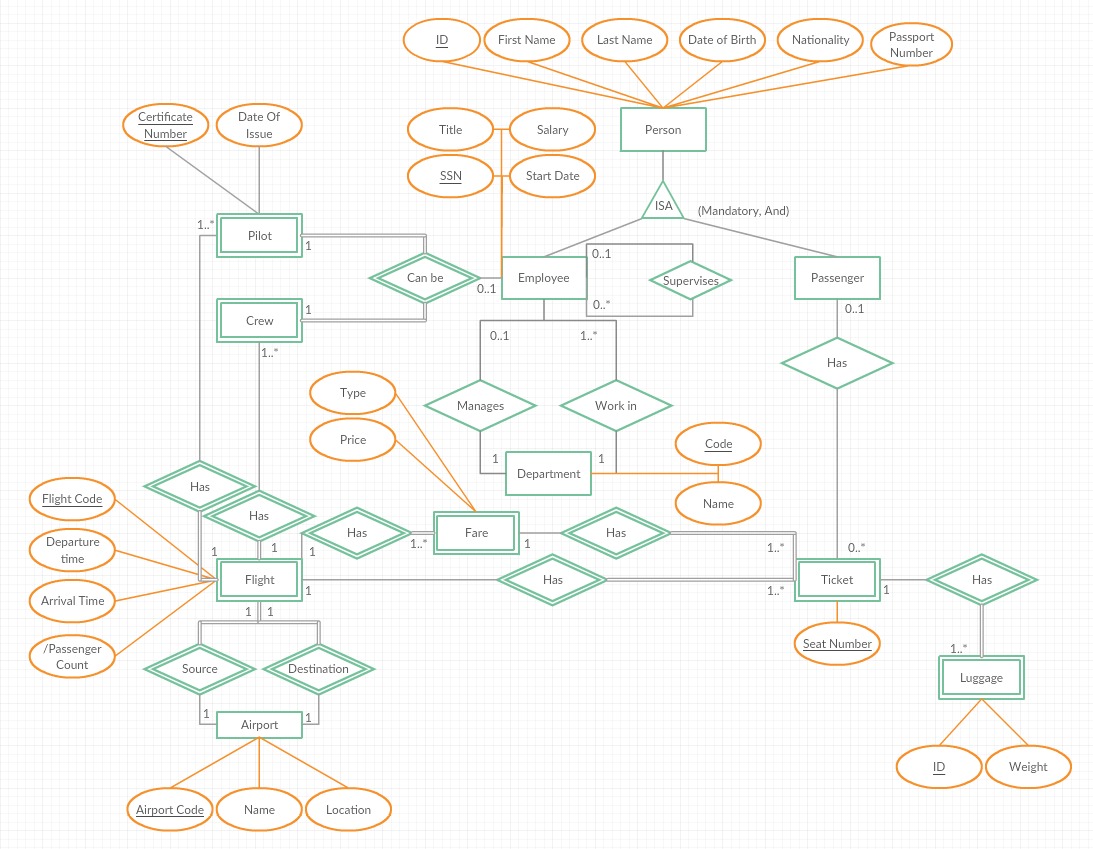
For the coursework, the main entities we have decided to include are: *person, employee, passenger, department, & airport* since these are the primary components that make up an air terminal. Additionally, the weak entities include *pilot, crew, flight, fare, ticket, & luggage* since their existence depend on the existence of the strong entities (ie: flight cannot exist without airport).

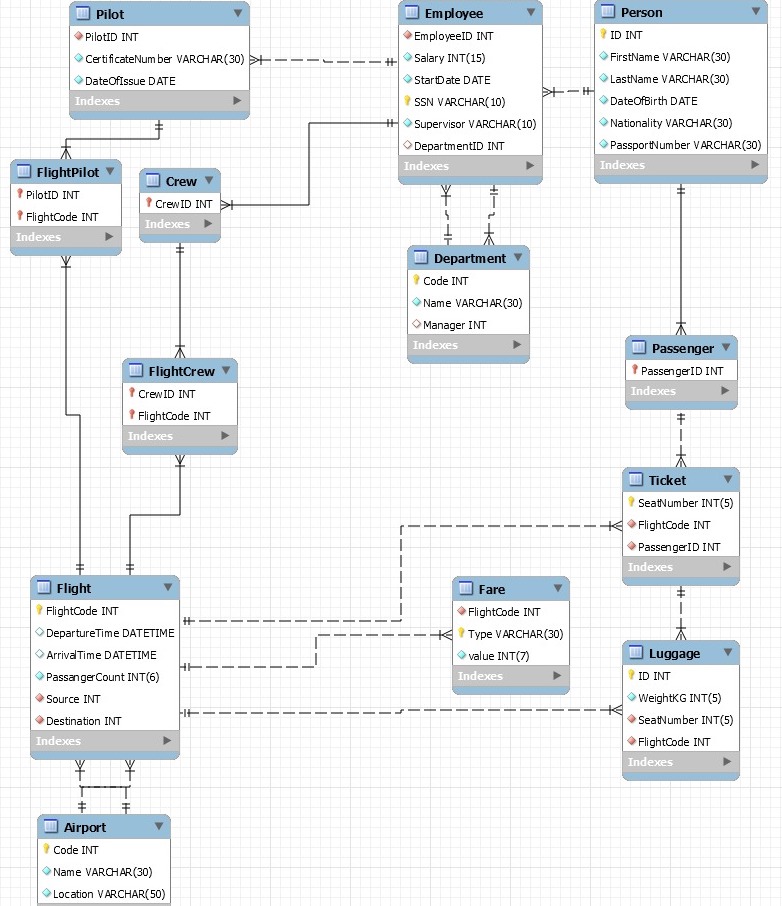
We have created *person* for the specialization aspect of this coursework wherein a person in the database can either be an employee in the airport or a passenger. Furthermore, we have included a recursive relationship within *employee* to indicate that some employees can be supervisors who supervise other employees who belong in different departments.

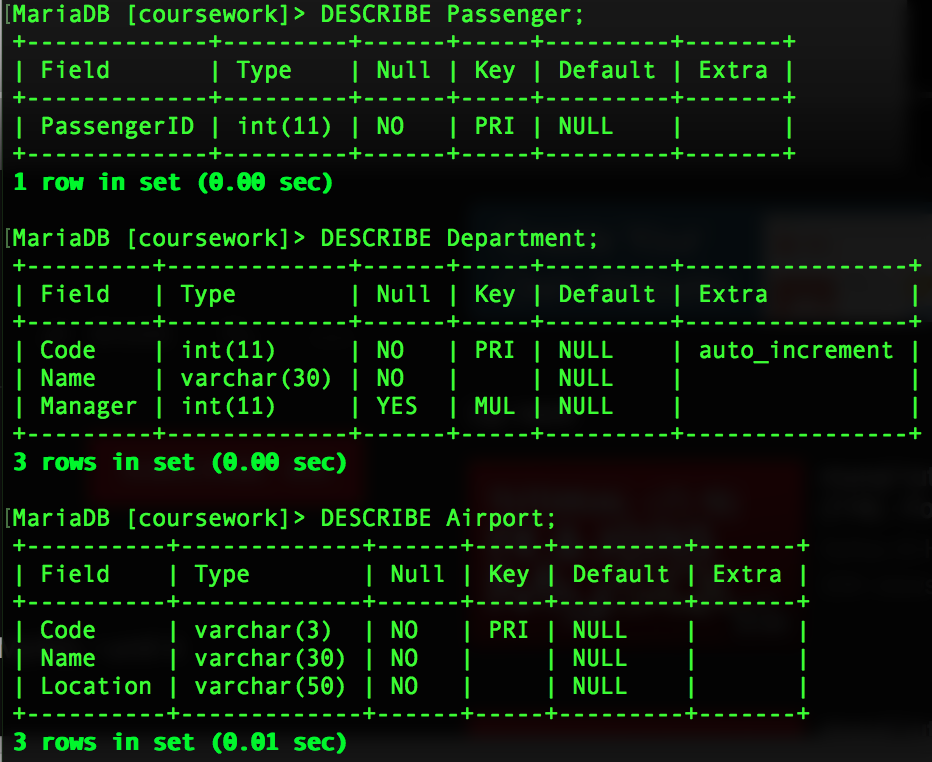
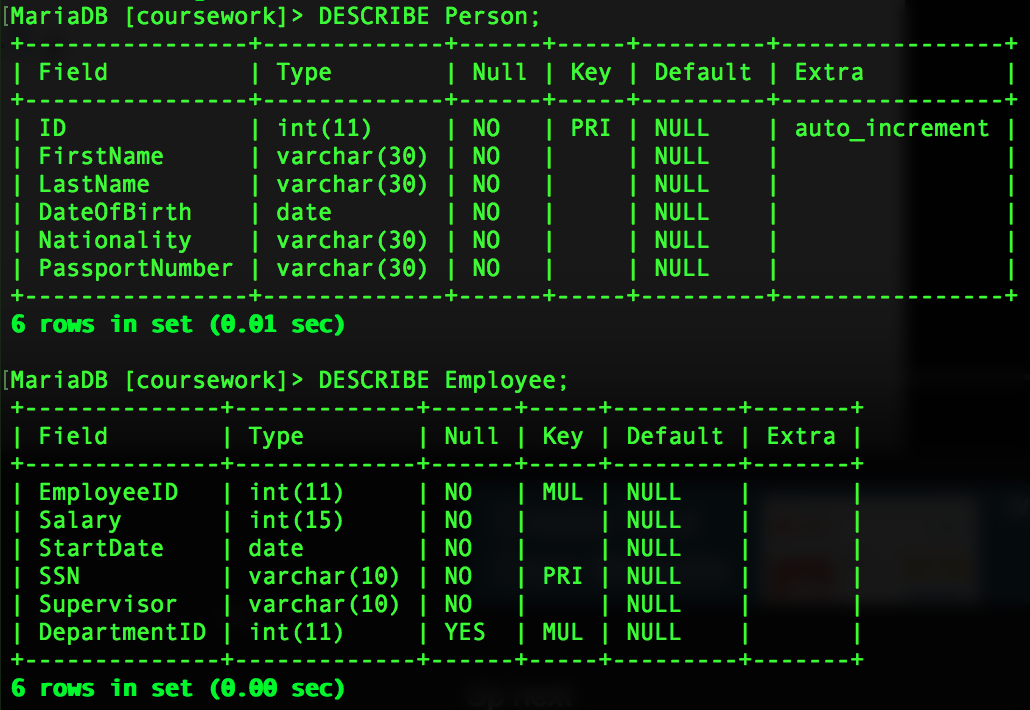
Assuming that employees can be more than a supervisor, a “can be” relationship was added which insinuates that they can be a *pilot* or a *crew* who belong in a *flight* which has a *ticket* and *fare* connectedly. Finally, the flight arrives and departs from *airport*.

ER diagram

The ER diagram contains 11 entities. It encapsulates the features that were a requisite to the coursework such as a recursive relationship displayed by the employee entity wherein an employee who can be a supervisor supervises the other employees; specialization (mandatory or) presented by the person entity in which a person can either be employee *or* a passenger as shown in the diagram below:

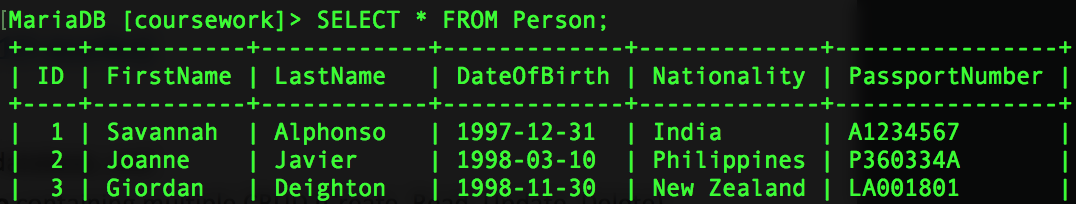


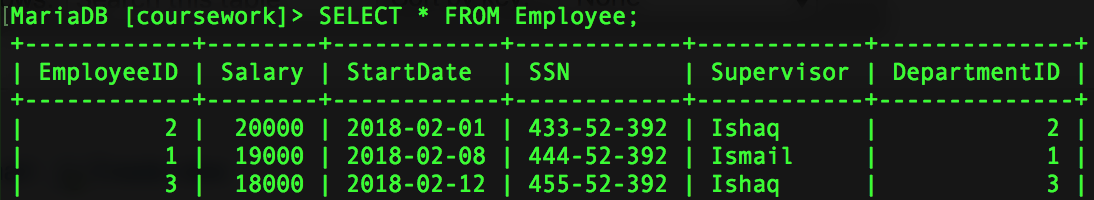
Translation into relational schema

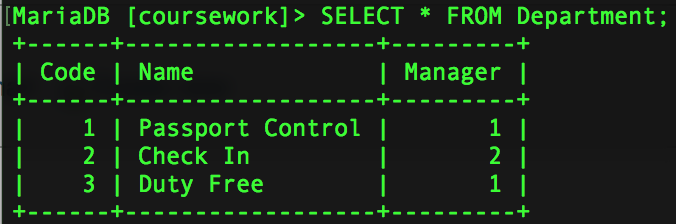
data dictionary:

Loading data

All the data in the database were loaded using <http://localhost/phpmyadmin/>.

Sample data from table *person:*

Sample data from related table *employee* wherein employeeID(int) references ID from person and DepartmentID references Code(int) from Department table

Sample data from department table wherein code(int) is referenced by Employee

Queries over the database

* **search for all passengers on a given flight**

SELECT FirstName, LastName

FROM Person as p

JOIN Passenger pass ON pass.PassengerID = p.ID

JOIN Ticket t ON t.PassengerID = p.ID

WHERE FlightCode= 1

ORDER BY p.LastName;

*output: displays names of passengers on given flight sorted by their last name*

* **search for all the names who are passengers as well as employees**

CREATE VIEW EMPLOYEEPASSANGER

AS SELECT SELECT P.id as EmployeeID, CONCAT(P.FirstName,' ',P.LastName) AS FULLNAME,p.nationality,p.passportnumber from

PERSON AS P WHERE P.id in (

SELECT EmployeeID

FROM Employee

UNION

SELECT PassengerID

FROM Passenger) order by p.id;

*output: creates another view (employeepassengers) with the names of people who are passengers & employees*

* **search for all flights going to a given city code**

CREATE VIEW dxbflights AS

SELECT FlightCode

FROM Flight

WHERE Destination = “DXB”;

*output: creates another view (dxbflights) with all the flight codes going to that city*

* **view flight plan**

CREATE VIEW (Flight Plan) AS

SELECT Code, Destination , DepartureTime, ArrivalTime

FROM Aiport, Flight

WHERE Aiport.Code = Flight.Source

AND Flight.Destination = “LAX";

*output: creates another view (Flight Plane) with all the information regarding flight to a given destination*

* **display who supervises which employees**

SELECT EmployeeID, Supervisor

FROM Employee

*output: displays employee IDs & their respective supervisor*

* **view passenger details**

CREATE VIEW (Personal Details) AS

SELECT FirstName, LastName , Nationality , SSN

FROM Person , Employee

WHERE Person.ID = Emlpoyee.EmlpoyeeID

AND Person.Nationality = “USA";

*output: displays name, nationality & SSN of people who are of a given nationality*

* **view flight details**

CREATE VIEW FLIGHTDETAIL AS SELECT DISTINCT F.FlightCode,A.Location AS Source,B.Location AS Destination,F.DepartureTime,F.ArrivalTime,COUNT(F.FlightCode) AS Passengers

FROM FLIGHT AS F, Airport AS A, Airport AS B, Ticket AS T

WHERE F.Source = A.Code AND F.Destination = B.Code AND T.FlightCode=F.FlightCode group by F.FlightCode;

*output: displays the code, source, destination, depature time ,arrival time, and passenger count of a flight.*

CREATE VIEW EMPLOYEEPASSANGER AS SELECT SELECT P.id as EmployeeID, CONCAT(P.FirstName,' ',P.LastName) AS FULLNAME,p.nationality,p.passportnumber from

PERSON AS P WHERE P.id in ( SELECT EmployeeID

FROM Employee

UNION

SELECT PassengerID

FROM Passenger) order by p.id;

CREATE VIEW PersonalDetail AS

SELECT CONCAT(P.LastName,', ',P.FirstName) AS FULLNAME , Nationality , SSN

FROM Employee AS E, PERSON AS P

WHERE P.ID = E.EmPLoyeeID group by p.id;

CREATE VIEW CREWDETail AS

SELECT C.CREWID, CONCAT(P.LastName,', ',P.FirstName) AS FULLNAME , Nationality , SSN

FROM Employee AS E, PERSON AS P,CREW AS C

WHERE P.ID = E.EmPLoyeeID AND C.CrewID=P.ID group by p.id;

CREATE VIEW PILOTDETAIL AS

SELECT C.PilotID, CONCAT(P.LastName,', ',P.FirstName) AS FULLNAME,C.CertificateNumber,C.DateOfIssue , Nationality , SSN

FROM Employee AS E, PERSON AS P,PILOT AS C

WHERE P.ID = E.EmPLoyeeID AND C.PilotID=P.ID group by p.ID;

CREATE VIEW TICKETDETAIL AS

SELECT P.ID, CONCAT(P.LastName,', ',P.FirstName) AS FULLNAME,P.Nationality,P.PassportNumber ,

T.FlightCode,T.seatnumber,T.FARETYPE, CONCAT(A.NAME,', ',A.Location) AS Source,CONCAT(AA.NAME,', ',AA.Location) AS Destination

FROM PERSON AS P , Passenger AS PP, Ticket AS T,FLIGHT AS F, AIRPORT AS A, AIRPORT AS AA WHERE

P.ID=PP.PassengerID AND PP.PassengerID=T.PassengerID AND F.FlightCode=T.FlightCode AND F.Source=A.Code AND F.Destination=AA.CODE;

Transactions:

*temporary change of salary from the employee with ID 1*

UPDATE Emoployee SET Salary = Salary + 1000 WHERE EmployeeID = 1;

*altering USA to UK, needs to be a permanent update hence the COMMIT TRANSACTION*

BEGIN TRANSACTION

UPDATE Person SET Nationality = "UK" WHERE ID = 8;

COMMIT TRANSACTION

*to be used when executing the table again , to keep the new update*

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED

Indexes

* **index for passport numbers called Passport**

CREATE INDEX Passport ON Person (PassportNumber);

*This is needed because it’s more convenient when we are searching for people that we use their passport number as a unique identifier as only one person in the work can have that number*

* **index for finding flight code to a given destination**

CREATE INDEX dxb ON Flight(FlightCode, Destination);

*This is needed to quickly see all flight codes to a certain destination*

* **index for finding flight code to a given destination**

CREATE INDEX dxb ON Flight(FlightCode, Destination);

*This is needed to quickly see all flight codes to a certain destination*

* **index for an employees ssn.**

CREATE INDEX SSN ON employee(employeeid,ssn);

*This is needed to optimize searching for employee using their ssn*