A Relational Database

Model for a Commercial

Airline

By James Fefes, Eugene Nicks,

Andrew Hotchkiss and Andrew Morrison

Table of Contents

1. Cover Page
2. Table of Contents
3. Introduction

3. Description of Business Process

1. Schema
2. E/R Diagram
3. DBMS and programming language selection

Introduction

We decided to model our database for an airline service that will manage and track employees, plane information, flights, customer data and reservations. Our task was to develop a relational database with a client-side interface to store and utilize the data. The design process included researching the processes and challenges in an airline database, identifying relations and putting them in an ER diagram, creating a schema, and selecting and implementing a GUI in PHP. Some of the challenges we faced and solved along the way were enforcing referential integrity, designing a method to organize seating, and creating views for different parties.

Description of Airline Business Processes

Commercial airlines provide the service of transporting large numbers of people in airplanes. Managing an airline as a whole requires that information is regularly updated on employees, planes, customers and flights. Typically, airlines sell tickets to reserve a seat on a flight for a designated time and date, and receive payment for the ticket. Customers are assigned seats on a flight depending on their departure and destination locations and sometimes other factors like seating class. They go to their designated gate at the time on their ticket and board, whereupon the plane takes off. The flights carry not only customers but employees, who need to be tracked, and flights need to be coordinated to avoid crashes and mixups on the runway. In addition, flights need to be coordinated with other airports for landings and layovers. All of these things are recorded and communicated to the interested parties in a convenient and frugal manner.

Schema

Key: Flight\_Num – Underlined is Primary Key

En\_Route\* - Asterisk is binary attribute

Location(City, etc.) – Multivalued attribute

Flights-

    Flight\_Num, PID, Dest\_Port\_ID, Current\_port\_ID, Departure\_T, Arrival\_T, Em\_ID, Cust\_ID, En\_Route\*, Gate\_D, Gate\_C, Seats\_Max

Employees-

    Em\_ID, FName, LName, Title, Salary

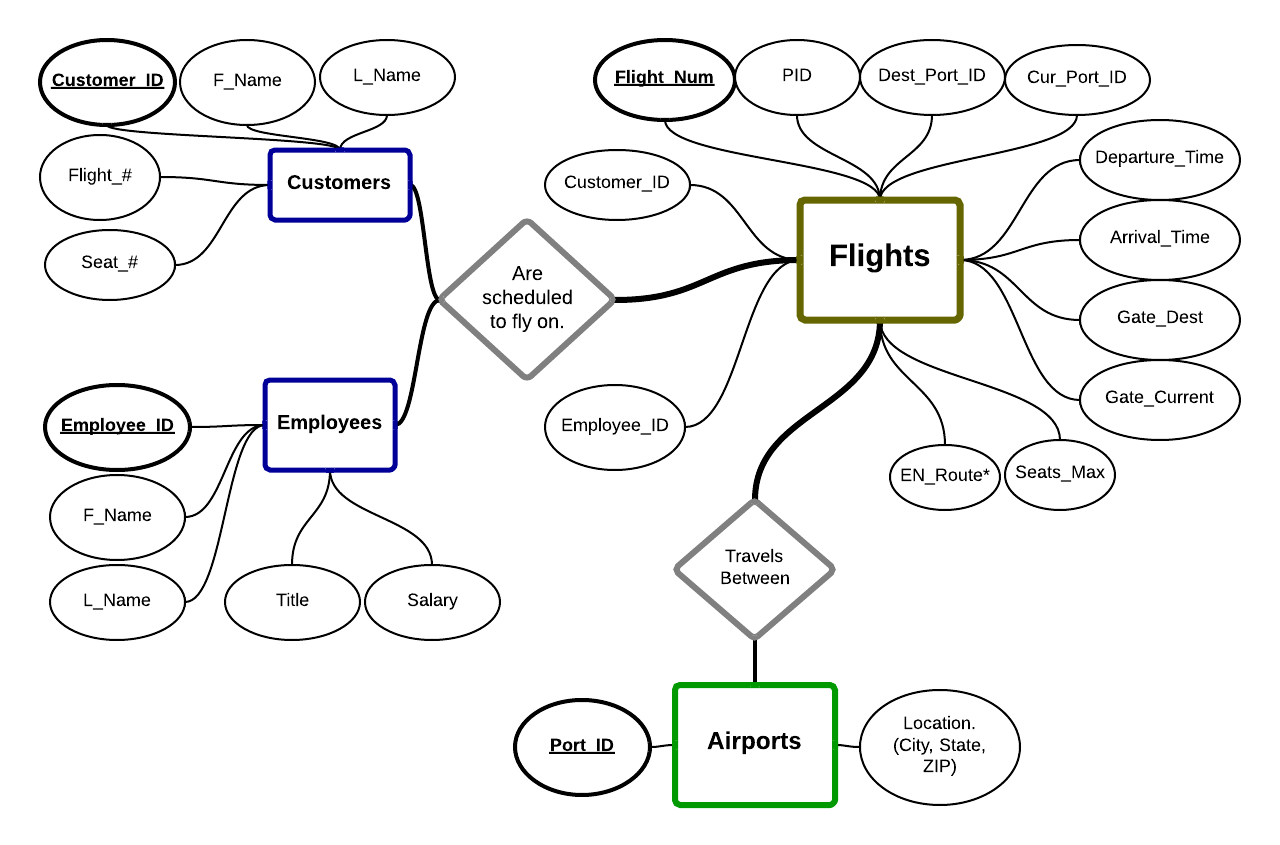
Customers-

    Cust\_ID, FName, LName, Flight\_No, Seat\_No

Airports-

    Port\_ID, Location(City, State, Zip, PO\_Box,)

Entity/Relationship Diagram



PHP - Graphical User Interface

We decided to use PHP to connect to our database and create the interfaces, or individual views. The GUI utilizes a menu with submenus for navigation. This simple style of interface provides convenience for the user, and manages division of data for the database. The program uses class inheritance to relate objects and organize data. The strategy of class inheritance also ensures consistency of data, as content will update simultaneously for each page.

MySQL

We selected MySQL as our database because it is commonly used, easy to connect to, and supports many different applications. After identifying all necessary entities and relationships, we added the attributes and set primary and foreign keys. A few constraints were added, like checks in attributes to ensure required fields were not left null, and a check for the binary ‘En\_Route’ attribute. The database was populated with sample information, then connected and tested with the PHP GUI.

