Fall 2007 CS186 Discussion Section: Week 2, 09/03 - 09/07

Your friendly TAs

September 4, 2007

1 Discussion Exercise

1. Develop a simple model of a flight reservation system. The model should include:

   - A plane (with a unique ID) is assigned to each flight. The assignment can be different each day. A flight that flies on Monday and Wednesday can be using different planes on each day.
   - Planes have a bunch of seats, usually identified by a seat number.
   - A particular flight number can only be used once a day.
   - A particular fight number can have different Source/Destination on a different date. For example, flight number "UA111" can be assigned to "Los Angeles to Oakland" on Monday and then changed to "San Francisco to Boston" on Tuesday.

   Include a few key attributes for each entity and relationship as you think is necessary (be creative). Some ideas to consider could include:

   - Plane type, manufacturer.
   - Source/Destination of a flight.

2. How to construct the ER model? Translate the model into relational tables using SQL commands.
3. How will the ER model and relational tables change if:

   - Every flight has a fixed "Source/Destination".
   - A flight uses the same plane each day.

1.1 Sample Solution

This is only ONE solution, there are many. The solution provided was designed to show as many features of ER diagrams as possible and there may be other (including better) models.

1.1.1 E-R Model

Notes about sample solution:

- This model stores redundant information in the flights entity. A new instance is created for each flight on each day, however generally flight information generally does not change. Some solutions include:
  - using a separate entity for flights and flight plans.
  - expand the uses relationship to include date (this introduces some other issues).
  - create a date entity and a relationship between flights and dates. These two entities and their relationship can then be aggregated.
- Passengers could reserve more than 1 seat per flight, and there can be passengers (including frequent flyers) who have never flown before! It is also assumed no passenger has the same first and last name.
- Terminals have to have at least one gate, and each gate must belong to exactly one terminal.
- Unlike real airlines, there is no way to overbook planes since each combination of flight/plane/seat can only have up to one passenger (due to the aggregation and arrow to travels_on).

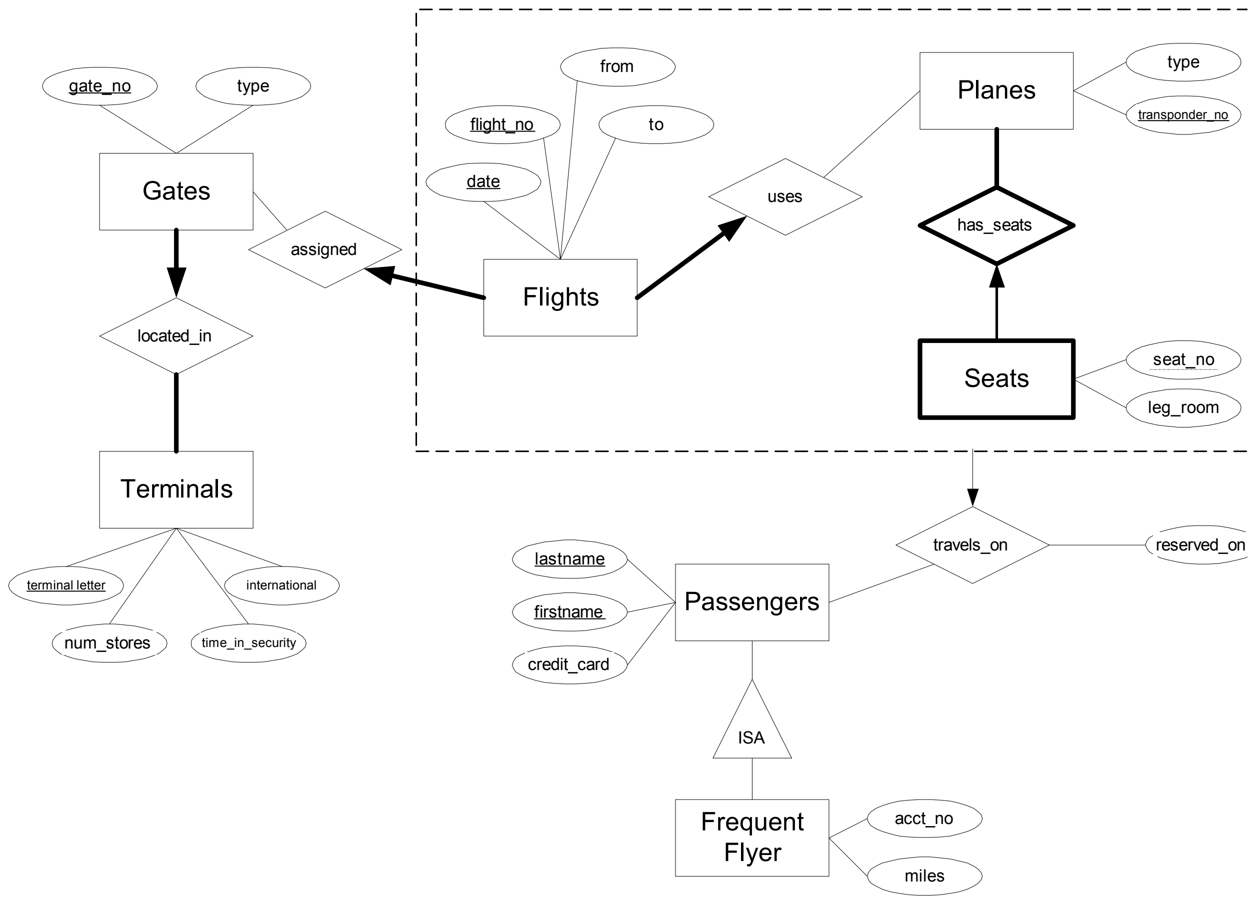

Figure 1: E-R Diagram

- Notice that flights is involved in both an aggregation (with flights, uses, planes, has_seates, and seats) and has a direct relationship with gates (which does not involve the aggregation).

There are other restrictions and anomalies because of the modeling decisions in the solution feel free to find them and point them out.

1.1.2 SQL Equivalent Schema

- `Flights(flight_no, date, from, to, plane_id)`,
  where `plane_id` is a foreign key to planes and is not null
- `Planes(plane_id, type)`
- `Seat(seat_no, plane_id, legroom)`,
  where `plane_id` is FK to planes, not null and when delete cascade.

```
CREATE TABLE Flights(
   date DATE,
   flight_no CHAR(5),
   from CHAR(20),
   to CHAR(20),
   plane_id INTEGER NOT NULL,
   PRIMARY KEY (date, flight_no),
   FOREIGN KEY (plane_id) REFERENCES Planes (transponder_no)
      ON DELETE NO ACTION);

CREATE TABLE Planes(
   transponder_no INTEGER,
   type CHAR(20),
   PRIMARY KEY (transponder_no));
```

```
CREATE TABLE Seat_assignment(
   seat_no CHAR(3),
   leg_room CHAR(20),
   plane_id INTEGER,
   PRIMARY KEY (plane_id, seat_no),
   FOREIGN KEY (plane_id) REFERENCES Planes (transponder_no)
      ON DELETE CASCADE );
```

You may want to notice the following:

- `Flights.plane_id` may not be NULL because each flight must be assigned one plane (participation constraint). A Flight instance may be assigned to only one Plane instance (one to many relationship), we can incorporate the Uses relation in the Flights table itself. The "ON DELETE NO ACTION" clause prevents a Plane instance from being deleted before the Flights that use the Plane get assigned a different one.
- `Planes.transponder_no` uniquely identifies a plane.
- Since Seats is a weak entity associated with a plane, we can merge the entity Seats and the relationship `has_seats` into a single table `Seat_assignment`. We allow cascaded deletes because Seats is a weak entity of the plane, and if the plane row is removed from the database the seats of that plane should be removed as well.

Problems

1. What if each flight has a fixed source/destination? (F $\rightarrow$ FrT)
   - `Flight_Route(route_id, From, To)`
   - `Flights(flight_no, date, plane_id)`
   - `Uses_Route(flight_no, route_id)`,
     where `route_id` is a FK to `Flight_Route` table and not null. On delete no action since the route may be used by other flights.
2. What if each flight has a fixed plane even on different days? (F $\rightarrow$ P)
   - `Flight(flight_no, plane_id)`
   - `Flight_Schedule(flight_no, date)`,
     where `date` cannot be null by virtue of being part of a key (right?) In the ER, we have a participation constraint from `Flights` to its diamond leading to `Date`.
   - `Flight_Route(route_id, From, To)`
   - `Uses_Route(flight_no, route_id)`,
     where `route_id` is a FK to `Flight_Route` table and not null. On delete no action since the route may be used by other flights.

2 Take Home Exercise

Try to add the following to the model:

- A passenger reserves a seat for a particular flight.
- Some passengers are frequent flyers, and therefore have account numbers and accumulated miles.
- Flights are assigned to a gate which is located in one terminal.
- A gate can handle multiple flights each day, and different flights on various days.
- There are terminals (usually identified by letters) which contain the gates (usually identified by numbers).