CS 411 Group 112

NetIds: zxchoo2, hansenp2, pnp4

Relational Schema

1. **Users**(user_id:INT [**PK**], username:VARCHAR(255), password:VARCHAR(255), email:VARCHAR(255))

2. **Searches**(search_id:INT [**PK**], date_searched:DATETIME, search_start_date:DATE, search_end_date:DATE, search_airline:VARCHAR(255), user_id:INT [**FK to Users.user_id**])

3. **Flights**(flight_number:INT [**PK**], date_of_flight:DATE [**FK to DayofDate.date_of_flight**] , airline:VARCHAR(255), origin_airport_iata:VARCHAR(255) [FK to Airports.airport_iata_code], dest_airport_iata:VARCHAR(255) [**FK to Airports.airport_iata_code**], scheduled_departure:INT, departure_time:INT, scheduled_time:INT, elapsed_time:REAL, distance:REAL, scheduled_arrival:INT, arrival_time:INT)

4. **DayofDate** (date_of_flight:DATE[**PK**], day_of_week:VARCHAR(255))

5. **Cancelled**(flight_number [**PK**, **FK to Flights.flight_number**], cancelled:VARCHAR(255), cancellation_reason:VARCHAR(255))

6. **Delays**(flight_number [**PK**, **FK to Flights.flight_number**], departure_delay:INT, arrival_delay:INT)

7. **Airports**(airport_iata_code:VARCHAR(255) [**PK**], airport:VARCHAR(255), city:VARCHAR(255), state:VARCHAR(255), country:VARCHAR(255), latitude:REAL, longitude:REAL)

8. **Address** airport:VARCHAR(255)[**PK**, **FK to Airports.airport**], city:VARCHAR(255), state:VARCHAR(255), country:VARCHAR(255), latitude:REAL, longitude:REAL)

9. **Airlines**(airline_iata_code:VARCHAR(255) [**PK**], airline_name:VARCHAR(255))

Relationships

1. Airlines-Flights: 1 - many
2. User - Searches: 1 - many
3. Delays - Flights: 1 - 1
4. Cancellation - Flights: 1 - 1
5. Searches - Airlines : many - 1
6. Airports - Airlines : many - many
7. Flights - DayofDates: many - 1
8. Airports - Address: 1 - 1
9. Flights - Airports: many (2) - one

To clarify: For entry 9, each flight is linked to exactly 2 airports.

Functional Dependencies

User_id -> username, password, email

Search_id -> data_searched, search_start_date, search_end_date, search_airline, user_id

Flight_number -> date_of_flight, airline, origin_airport_iata, dest_airport_iata, scheduled_departure, departure_time, scheduled_time, elapsed_time, distance, scheduled_arrival, arrival_time

Date_of_flight -> day_of_week

Flight_number -> cancelled, cancellation_reason

Flight_number -> departure_delay, arrival_delay

Airport_iata_code -> airport

Airport -> city, state, country, latitude, longitude

Airline_iata_code -> airline_name

Assumptions

We assume that knowledge of delays that happened in 2015 will help forecast delays in the future

1. Airlines - Flights: 1 - many
   - We assume that flights can only have one airline
2. User - Searches: 1 - many
   - We assume that each search can only belong to one user
   - We assume that one user can make more than one search
3. Delays - Flights: 1 - 1
   - We assume that each flight can only be delayed one time, in the sense that multiple delays will be cumulative and treated as one delay.
   - We assume that delays only correspond to a unique flight.
4. Cancellation - Flights: 1 - 1
   - We assume that flights can only be cancelled one time
5. Searches - Airlines : many - 1
   - We assume that users can search on the same airport multiple times
6. Airports - Airlines : many - many
   - We assume that there airlines can share airports
   - We assume that airlines own more than one airport
7. Flights - DayofDates: many - 1
   - We assume that each flight has one “day” that the flight is considered to be under, even if the flight is overnight. This “day” will correspond to the day that the flight departs.
8. Airports - Address: 1 - 1
   - We assume that airports can only have one address
   - We assume that there is only one airport at each address
9. Flights - Airports: many(2) - one

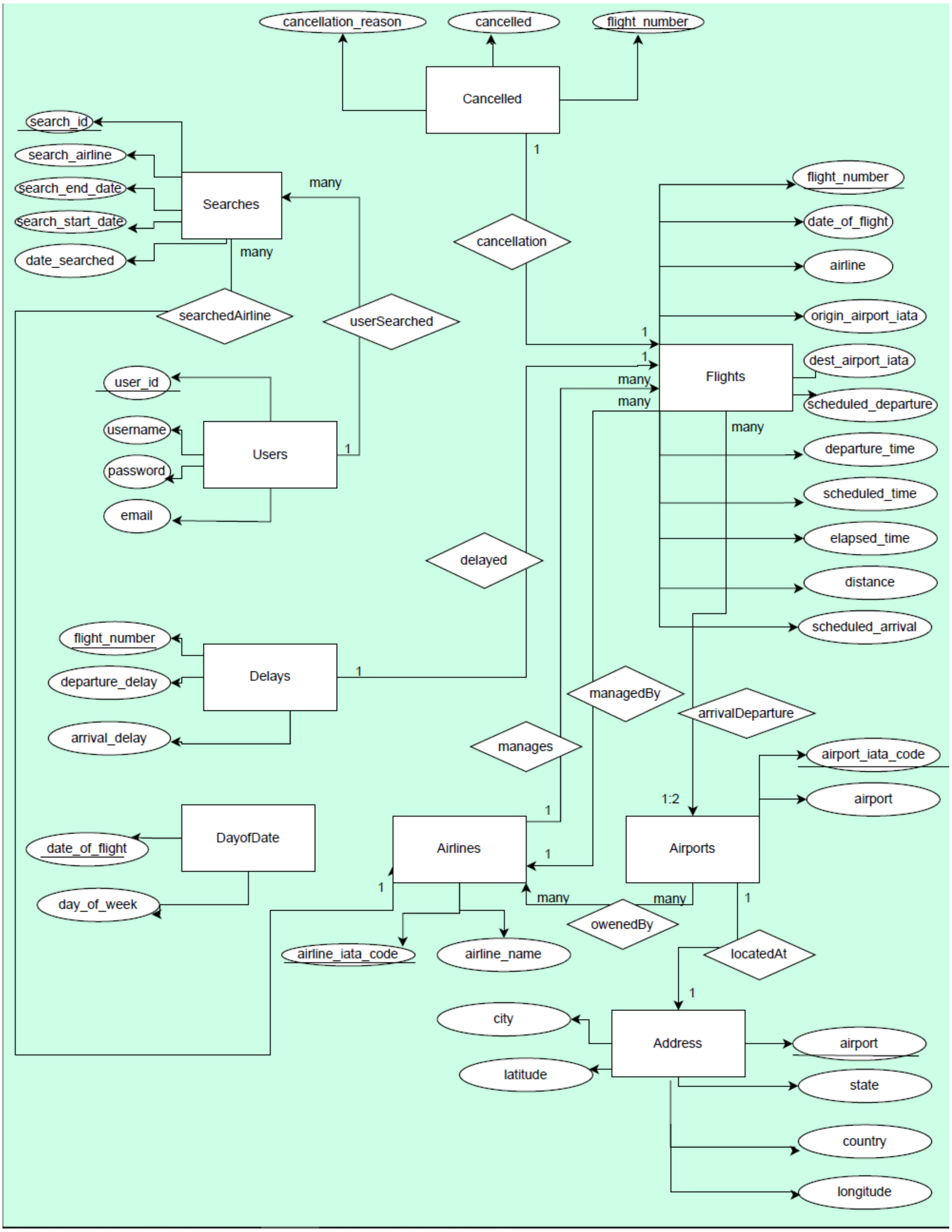
   - We assume that flights consist of only a start and end destination, no layovers

Normalization

We normalized into 3NF form. Before Normalization, the information in the DayofDate and Address tables did not exist, and the information contained within was held in the Flights and Airports tables respectively. When we normalized into 3NF form, we listed out all the functional dependencies as a minimal basis. Besides Date_of_flight->Day_of_week and Airport->city, state, country, latitude, longitude, all the other functional dependencies were keys determining the rest of the attributes. We dropped Day_of_week, city, state, country, latitude, longitude from their original tables to create a minimal basis. We then created tables for all of the functional dependencies in the minimal basis, and made sure that there was a table for each key of our original 7 entities. The end result of this was the creation of tables housing the dropped attributes (Day_of_week, city, state, country, latitude, longitude) which have a FK to the primary key of the table they were dropped from.

We chose 3NF because it is generally considered sufficient for most practical database designs and is easier to achieve than BCNF. We also think that 3NF is a good choice because it results in fewer tables and is easier to maintain. With the simplicity of this database, there are not a lot of complex operations or updates which could result in anomalies. With BCNF, there would have been more tables and more complexity.

ER Diagram


cancellation_reason
cancelled
flight_number
Cancelled
1
search_id
search_airline
search_end_date
search_start_date
date_searched
Searches
many
many
cancellation
flight_number
date_of_flight
airline
origin_airport_iata
dest_airport_iata
scheduled_departure
departure_time
scheduled_time
elapsed_time
distance
scheduled_arrival
searchedAirline
userSearched
1
1
many
many
Flights
many
user_id
username
password
email
Users
1
delayed
flight_number
departure_delay
arrival_delay
Delays
1
managedBy
arrivalDeparture
manages
airport_iata_code
airport
1:2
1
1
DayofDate
date_of_flight
day_of_week
Airlines
Airports
1
many
many
1
owenedBy
locatedAt
airline_iata_code
airline_name
1
city
latitude
Address
airport
state
country
longitude