# Flight Log Technical Specification

Paul Bogard · January 31, 2016

## Contents

Classes ..... 4
Flight ..... 4
Associations ..... 4
Attributes ..... 4
Methods ..... 6
self.classes_list ..... 6
self.aircraft_first_flight(aircraft_family) ..... 6
self.airline_first_flight(airline) ..... 6
self.airport_first_visit(airport_id) ..... 6
self.tail_country(tail_number) ..... 6
Airport ..... 6
Associations ..... 6
Attributes ..... 7
Methods ..... 7
all_flights(logged_in) ..... 7
airline_frequency(logged_in) ..... 7
aircraft_frequency(logged_in) ..... 7
country_flag_path ..... 8
self.frequency_array(flight_array) ..... 8
Aircraft Family ..... 8

Associations ..... 8
Attributes ..... 8
Methods ..... 8
self.categories_list ..... 9
format_name ..... 9
full_name ..... 9
Airline ..... 9
Associations ..... 9
Attributes ..... 9
Methods ..... 10
format_name ..... 10
Route ..... 10
Associations ..... 10
Attributes ..... 10
Methods ..... 11
Trip ..... 11
Associations ..... 11
Attributes ..... 11
Methods ..... 11
User ..... 11
Associations ..... 11
Attributes ..... 12
Methods ..... 12
Database ..... 13
Database Design ..... 13

Size ..... 13

## Classes

### Flight

#### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `belongs_to :trip` | Trip | `id` |
| `belongs_to :origin_airport` | Airport | `id` |
| `belongs_to :destination_airport` | Airport | `id` |
| `belongs_to :airline` | Airline | `id` |
| `belongs_to :aircraft_family` | Aircraft Family | `id` |
| `belongs_to :operator` | Airline | `id` |
| `belongs_to :codeshare_airline` | Airline | `id` |

#### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique flight identifier |
| `origin_airport_id` | integer (required) | Maps to the id attribute of Airport |
| `destination_airport_id` | integer (required) | Maps to the id attribute of Airport |
| `trip_id` | integer (required) | Maps to the id attribute of Trip |
| `trip_section` | integer (required) | Used to break a trip into subsections |
| `departure_date` | date (required) | Departure date of the flight (in the local time of the departure airport) |
| `departure_utc` | datetime (required) | UTC departure date and time, used to sort flights |

| | | |
|---|---|---|
| `airline` | string | Airline branding the flight. For regional subsidiaries, use the parent airline; for codesharing, use the plane's livery. |
| `flight_number` | integer | The airline's assigned number for this flight |
| `codeshare_airline` | string | Airline the flight was purchased on and ticketed as |
| `codeshare_flight_number` | integer | The `codeshare_airline`'s assigned number for this flight |
| `operator` | string | Airline operating the flight. For mainline flights, this will likely be the same as the `airline` attribute. |
| `fleet_number` | string | The operator's internal fleet number for the aircraft used for this flight. |
| `aircraft_family` | string | Manufacturer and family type (e.g. "Boeing 737" and "Airbus A320") |
| `aircraft_variant` | string | Variant type and model (e.g. "737-800" and "A321") |
| `aircraft_name` | string | Operator's name for the aircraft used for the flight, if named. |
| `tail_number` | string | Tail number for the aircraft used for this flight. |
| `travel_class` | string | Class of travel (Economy, Business, or First) |
| `comment` | text | Comment |

### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

**self.classes_list**

Returns a hash of the possible travel classes, with the class IATA codes as the keys and the class names as the values.

**self.aircraft_first_flight**(aircraft_family)

Returns the departure_date of the first flight on this aircraft family as a date.

**self.airline_first_flight**(airline)

Returns the departure_date of the first flight on this airline as a date.

**self.airport_first_visit**(airport_id)

Returns the departure_date of the first visit to this airport as a date.

**self.tail_country**(tail_number)

Accepts a tail number string, and returns a string containing the country that this tail number is associated with.

Each country has its own format for tail numbers, as documented in ICAO Annex 7. This function shall examine the format of the tail number and use that to determine the resulting country.

## Airport

### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `has_many :originating_flights` | Flight | `originating_airport_id` |
| `has_many :arriving_flights` | Flight | `destination_airport_id` |
| `has_many :first_routes` | Route | `airport1_id` |
| `has_many :second_routes` | Route | `airport2_id` |

### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique airport identifier |
| `iata_code` | string (required) | 3-letter IATA code. Must be unique. |
| `city` | string (required) | Usually the city, with additional information if ambiguous (e.g. “Dayton” and “Chicago-O’Hare” and “Portland (OR)”). |
| `country` | string (required) | The country that the airport is located. |
| `region_conus` | bool | True if the airport is in the CONUS region, False otherwise |

### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

**all_flights**(logged_in)

Returns a collection of Flights that have this airport as an origin or destination. If logged_in is false, hidden flights will not be included.

**airline_frequency**(logged_in)

Returns a hash of the airlines of the flights using this airport, and how many flights involving this airport each airline has. If logged_in is false, hidden flights will not be counted.

**aircraft_frequency**(logged_in)

Returns a hash of the aircraft families of the flights using this airport, and how many flights involving this airport each aircraft family has. If logged_in is false, hidden flights will not be counted.

**country_flag_path**

Returns the path to the country flag icon for the current airport.

**self.frequency_array**(flight_array)

Returns a hash of the airports visited in the flights listed in flight_array, and how many times each airport was visited within that list of flights. This method does not filter hidden flights, so care should be taken to pass in an appropriate flight_array.

## Aircraft Family

### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `has_many :flights` | Flight | `id` |

### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique aircraft family identifier |
| `family_name` | string (required) | Name of the aircraft family |
| `iata_aircraft_code` | string (required) | 3-letter IATA code of the most generic version of the aircraft family (e.g. 32S or 737, not 320 or 738) |
| `manufacturer` | string (required) | Manufacturer of the aircraft family |
| `category` | string | Category of aircraft family, selected from the options in self.categories_list |

### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

#### self.categories_list

Returns a hash of aircraft categories (Wide-body, Narrow-body, Regional Jet, Turboprop).

#### format_name

Returns the name of the aircraft family.

#### full_name

Returns the manufacturer and name of the aircraft family.

## Airline

### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `has_many :flights` | Flight | `id` |
| `has_many :operated_flights` | Flight | `operator_id` |
| `has_many :codeshared_flights` | Flight | `codeshare_airline_id` |

### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique airport identifier |
| `iata_airline_code` | string (required) | 2-letter IATA code, appended by a hyphen and airline name (lowercase, spaces to hyphens) if necessary for uniqueness. Must be unique. Used for places where uniqueness is required (for example, parameters and icon names). |
| `airline` | string (required) | The name of the airline |
| `is_only_operator` | bool | True if the airline does not sell its own flights, false otherwise. Used to |

| | |
|---|---|
| | determine whether this should be included in the Airlines list or just the operators list. |

### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

#### format_name

Returns the name of the airline.

## Route

### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `belongs_to :airport1` | Airport | `id` |
| `belongs_to :airport2` | Airport | `id` |

### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique route identifier |
| `airport1_id` | integer (required) | Airport 1 id |
| `airport2_id` | integer (required) | Airport 2 id |
| `distance_mi` | Integer | Great circle distance between the two airports (in miles) |

Routes consist of a pair of airports, which are indifferent to the direction flown. Thus, care must be taken to ensure duplicate pairs of airports don't end up in the route table. For example, `[airport1_id,airport2_id] = [5,9]` is the same route as

`[airport1_id,airport2_id] = [9,5]`, so only one of these should have a record in the Routes table. In order to ensure this, whenever a new record is submitted, it should check both combinations of airport1, airport2 to ensure a record doesn't already exist; if one does, the existing record shall have its `distance_mi` updated rather than creating a new record. If the record does not exist, then it shall be saved such that `airport1_id` is the lower of the two ids, and `airport2_id` is the higher.

#### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

### Trip

#### Associations

| Relationship | Class | Foreign Key |
|---|---|---|
| `has_many :flights` | Flight | `id` |

#### Attributes

| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique trip identifier |
| `name` | string (required) | Trip name |
| `hidden` | bool | True if the trip is only visible to verified users; False if visible to visitors |
| `comment` | text | Comment |

#### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

### User

#### Associations

None

### Attributes

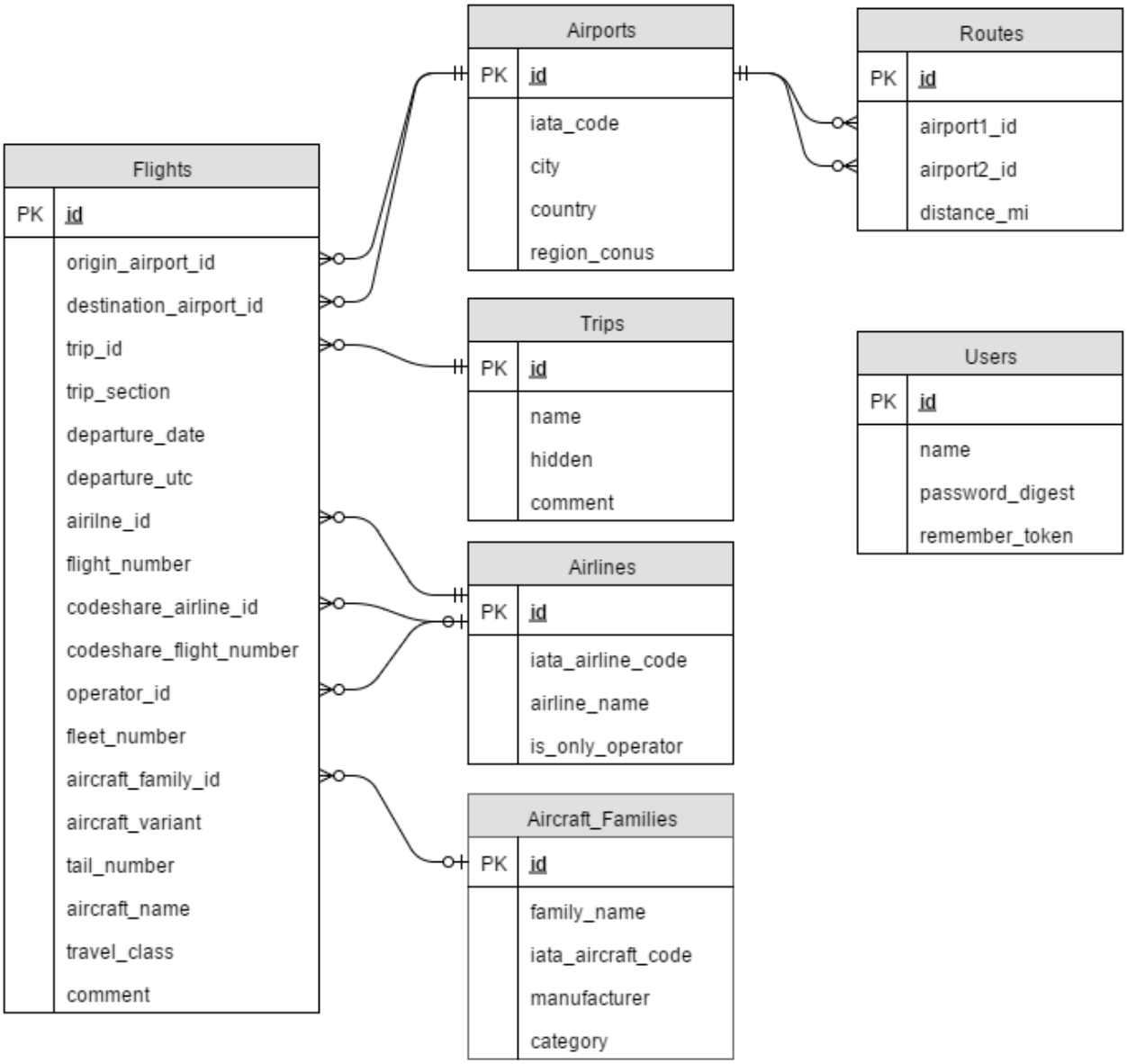
| Attribute | Type | Description |
|---|---|---|
| `id` | integer (required) | Unique trip identifier |
| `name` | string (required) | Trip name |
| `password_digest` | string (required) | Encrypted version of the user's password |
| `remember_token` | string | Login token storage |

### Methods

Standard Ruby on Rails ActiveRecord methods are available, but not listed in this document.

# Database

## Database Design




## Size

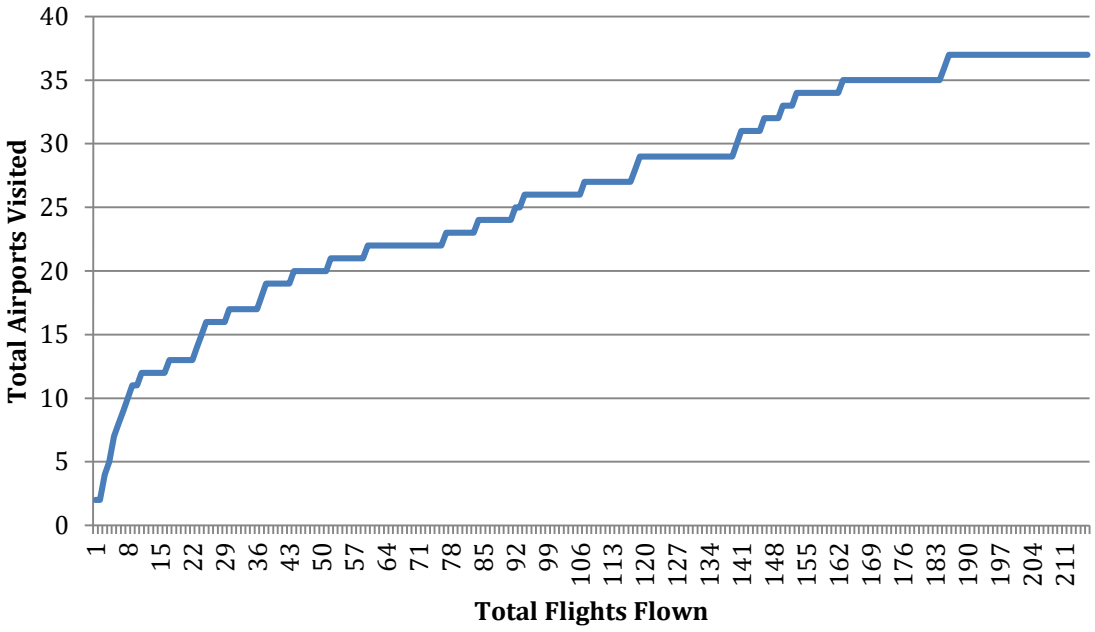
Per the functional specification, this site is intended for a single user (Paul Bogard), which will keep the size small.

For a conservative maximum number of records, assume an average of one trip per day and two flights per day for forty years.

$$\left(\frac{1 \text{ trip}}{1 \text{ day}}\right)\left(\frac{365.25 \text{ days}}{1 \text{ year}}\right)(40 \text{ years}) = 14610 \text{ trips}$$

$$\left(\frac{2 \text{ flights}}{1 \text{ day}}\right)\left(\frac{365.25 \text{ days}}{1 \text{ year}}\right)(40 \text{ years}) = 29220 \text{ flights}$$

At the time of the initial writing of this spec, Paul's flight log contained 219 flights and 37 airports. The number of airports as a function of flights appears to be less than linear. This is logical: the more flights are flown, the more likely it is that the flight will involve airports that have been visited in the past.




To get the worst-case prediction, though, we will assume a linear relationship with a ratio of 37 airports per 219 flights (and a y-intercept of zero).

$$29220 \text{ flights}\left(\frac{37 \text{ airports}}{219 \text{ flights}}\right) = 4937 \text{ airports}$$

Even at these extraordinarily worst-case numbers, these table sizes are easily within the capabilities of PostgreSQL.