Nimbus DB: Airline Database Management System

INTRODUCTION

Welcome to NimbusDB, your ultimate Airline Management System. As the demand for air travel continues to surge, having a robust management system is essential for ensuring smooth operations and meeting passenger expectations. Our database management system is crafted to efficiently and securely handle vast amounts of data, encompassing airline operations, passenger information, and aircraft maintenance, all in a well-organized manner.

PURPOSE

At NimbusDB, we take care of the intricacies involved in data storage, management, and upkeep for airline companies. Our aim is to streamline your responsibilities in areas such as flight scheduling and planning, reservations and ticketing, crew coordination, aircraft maintenance, inventory management, passenger services, and data analysis and reporting. Our database's core purpose is to offer a centralized repository, where all relevant airline data is stored in one location, replacing the need for separate databases for meals, staff, flights, and other aspects. This approach ensures effortless data transfer, when necessary, with all records well-organized and accessible with just a click.

USERS

Our Database System is designed for employees in airline companies. It provides them with a comprehensive view that is unattainable through manual processes. This enables them to monitor operations thoroughly, enhancing efficiency and overall experience.

APPLICATIONS

Airline companies can utilize the database to record information about scheduled flights, their employees, airports, and the services offered. Additionally, the database can be used to generate valuable insights, such as identifying the most popular destinations to adjust flight frequencies, determining passengers' favourite meals, or recognizing loyal customers to reward them. It also aids in making strategic decisions regarding flight pricing, scheduling, and crew assignments.

DATABASE REQUIREMENTS

Assumptions made:

1) No connecting flights, only direct flights
2) Food variety – Veg or Non-Veg or Vegan, each instance can only choose one among them

Entity types and its attributes:

1) **Employee (Superclass)**

- Employee ID (Primary Key)
  - Data Type: Varchar (10)
- Name (Composite Attribute)
  - First Name + Last Name
  - Data Type: Varchar (100)
- Address (Composite Attribute)
  - Apartment Number + Street + City + State
  - Data Type: Varchar (200)
- DOB
  - Data Type: Date
- Age (Derived Attribute)
  - Data Type: int
  - Domain: 18 < Age < 60
- Date of Joining
  - Data Type: int
- Nationality
  - Data Type: Varchar (100)
- Gender
  - Data Type: Char (1)

Pilot (Subclass)

- License ID
  - Data Type: Varchar (20)
- Flight Hours
  - Data Type: Int
- Aircraft Type Rating
  - Data Type: Varchar (10)
- Rank

        - Data Type: Varchar (20)
    - Languages Known (Multi-valued Attribute)
        - Data Type: Varchar (100)

Flight Attendant (Subclass)

- Languages Known (Multi-valued Attribute)
    - Data Type: Varchar (100)
- Service Awards (Multi-valued Attribute)
    - Data Type: Varchar (200)
- Emergency Training (Multivalued Attribute)
    - Data Type: Varchar (100)

Ground Staff (Subclass)

- Role (Multi-valued Attribute)
    - Data Type: Varchar (100)
- Work Shifts (Multi-valued Attribute)
    - Data Type: Varchar (50)

Technicians (Subclass)

- Specialisations (Multi-valued Attribute)
    - Data Type: Varchar (100)

2) **Flights**

- Flight Number (Primary Key)
    - Data Type: Varchar (10)
- Scheduled Departure Date-Time (Composite Attribute)
    - Date + Time
    - Data Type: Date-Time
- Actual Departure Date-Time (Composite Attribute)
    - Date + Time
    - Data Type: Date-Time
- Flight Status
    - Data Type: Varchar (50)
- Scheduled Arrival Date-Time (Composite Attribute)
    - Date + Time
    - Data Type: Date-Time
- Actual Arrival Date-Time (Composite Attribute)
    - Date + Time
    - Data Type: Date-Time

3) **Aircrafts**

- Aircraft ID (Key Attribute)
  - Data Type - Varchar (10)
- Manufacturer's Name
  - Data Type - Char (200)
- Model Number
  - Data Type - Varchar (20)
- Purchase Date
  - Data Type - Date
- Purchase Price (in Dollars)
  - Data Type - int
- Distance Travelled (in km)
  - Data Type - int
- Passenger Capacity
  - Data Type - int
- Cargo Capacity (in kg)
  - Data Type - int

4) **Passenger**

- Passenger ID (Primary Key)
  - Data Type: Varchar (10)
- Name (Composite Attribute)
  - Data Type: Varchar (100)
- ID Type
  - Data Type: Varchar (50)
- ID Number (Candidate Key)
  - Data Type: Varchar (50)
- Nationality
  - Data Type: Varchar (50)
- Email ID (Candidate Key)
  - Data Type: Varchar (100)
- Date of Birth
  - Data Type: Date
- Gender
  - Data Type: Char (1)
- Address (Composite Attribute)
  - Apartment Number + Street + City + State
  - Data Type: Varchar (200)
- Medical History (Multi-valued Attribute)
  - Data Type: Char (500)

5) **Airports**

- Airport Code (Key Attribute)
  - Data Type: Varchar (20)
- Name
  - Data Type: Char (100)
- Airport Type
  - International OR Domestic
  - Data Type: Char (1)
- Location (Composite Attribute)
  - Street + City + State + Country
  - Data Type: Varchar (200)
- Number of Gates
  - Data Type: int
- Number of Belts
  - Data Type: int
- POC (Multi-valued Attribute)
  - Data Type: Varchar

6) **Meals**

- Meal ID (Key Attribute)
  - Data Type: Varchar (20)
- Vendor's Name
  - Data Type: Char (100)
- Dishes Available (Multi-Valued Attribute)
  - Data Type: Char (50)
- Food Variety
  - Data Type: Char (30)
  - Domain: Veg/ Non-Veg/ Vegan
- Price (in Dollars)
  - Data Type: int

7) **Ticket (weak entity)**

- Category
  - Data Type: Varchar (50)
- Ticket Price (in Dollars)
  - Data Type: Int
- *Seat Allocated (Partial Key)

        - Data Type: Varchar (5)
    - *Flight Number (Partial Key)
        - Data Type: Varchar (10)
    - Baggage Allowance (in kgs)
        - Data Type: Int
    - Meal Included
        - Domain: YES/NO
        - Data Type: Char (1)

    * Seat Allocated and Flight Number are the Key Attributes (together)

8) **Baggage (weak entity)**

    - Baggage ID (Partial Key)
        - Data Type: Varchar (50)
    - Weight (in kgs)
        - Data Type: Int
    - Colour (Multi-valued Attribute)
        - Data Type: Varchar (100)
    - Size (Composite Attribute)
        - Length + Breadth + Height
        - Data Type: Varchar (20)

9) **Rating (weak entity)**

    - Rating Number (Partial Key)
        - Data Type: int
    - Feedback
        - Data Type: char (200)
    - Stars
        - Data Type: int
        - Domain (0 < Stars < 10)

Relationship types:

1) **Flight DEPARTS FROM Airport 1 ARRIVES AT Airport 2**
   - ATTRIBUTES
     - i) Departure Gate
       1. Data Type: Int
       2. Domain: 0 < Gate <= Number of gates of airport 1
     - ii) Arrival Baggage Belt
       1. Data Type: Int
       2. Domain: 0 < Gate <= Number of belts of airport 2
     - iii) Type
       1. Data Type: Char (1)
       2. Domain: International OR Domestic
   - DEGREE - 2
   - **Participation Constraints**:
     1) Flight (1, N): A flight must depart from and arrive at an airport, and multiple flights can use the same airport.
     2) Airport1 (1, N): An airport can handle multiple departing flights.
     3) Airport2 (1, N): An airport can handle multiple arriving flights.

2) **Flight USES Aircraft**
   - DEGREE - 2 (Binary)
   - CARDINALITY- 1:1
   - **Participation Constraints**:
     1) Flight (1,1): A flight must use one aircraft.
     2) Aircraft (0,1): An aircraft may or may not be used by a flight but can be used by only one flight at a time.

3) **Passenger BOOKS Ticket INCLUDES Meal (identifying relationship)**
   - ATTRIBUTES
     - PNR Number
       - Data Type: Varchar (50)
     - Booking ID (Key Attribute)
       - Data Type: Varchar (50)
   - DEGREE - 3 (Ternary)
   - **Participation Constraints**:
     1) Passenger (0, N): A passenger can book multiple tickets.
     2) Ticket (1,1): A ticket is associated with exactly one passenger.

3) Meal (0, N): A meal can be included in multiple tickets.
*If a single passenger books multiple tickets together, all will have the same PNR number but different booking IDs

4) **Baggage ASSOCIARED WITH Ticket (identifying relationship)**
   - DEGREE - 2 (Binary)
   - CARDINALITY- N:1
   - **Participation Constraints**:
     1) Baggage (1,1): A baggage is associated with exactly one ticket.
     2) Ticket (0, N): A ticket can have multiple baggage items.

5) **Ground staff POSTED at Airport**
   - DEGREE - 2 (Binary)
   - CARDINALITY- N:1
   - **Participation Constraints**:
     1) Ground Staff (1,1): A ground staff member is posted at exactly one airport.
     2) Airport (5,20): An airport has at least 5 and at most 20 ground staff members.

6) **Technician POSTED at Airport**
   - DEGREE - 2 (Binary)
   - CARDINALITY- N:1
   - **Participation Constraints**:
     1) Technician (1,1): A technician is posted at exactly one airport.
     2) Airport (2,5): An airport has at least 2 and at most 5 technicians.

7) **Flight SERVES Meal**
   - DEGREE - 2 (Binary)
   - CARDINALITY - M: N
   - **Participation Constraints**:
     1) Flight (1, M): A flight serves at least one meal.
     2) Meal (0, N): A meal can be served on multiple flights, or none.

8) **Flight attendant SUPERVISES Flight attendant (self-relationship)**
   - ATTRIBUTES
     - Flight Number (Foreign Key)
       - Data Type - Varchar (10)

- DEGREE - 1 (Unary)
- **Participation Constraints**:
    1) Head Flight Attendant (0, N): A head flight attendant can supervise multiple flight attendants.
    2) Flight Attendant (1,1): Each flight attendant is supervised by exactly one head flight attendant.

- CARDINALITY - 1: N
- A Flight can have only one Head Flight Attendant who supervises one or more Flight Attendants.

9) Ground staff MANAGES Ground staff

- DEGREE - 1 (Unary)
- CARDINALITY - 1: N
- **Participation Constraints**:
    1) Manager (0, N): A manager can manage multiple ground staff members.
    2) Ground Staff (1,1): Each ground staff member has one manager.

10) Rating GIVEN BY ticket (identifying relationship)

- DEGREE - 2 (Binary)
- **Participation Constraints**:
    1) Rating (1, N): A rating is given by multiple tickets.
    2) Ticket (1,1): Each ticket can give one rating.

- CARDINALITY - 1:1
- A Rating can be given by multiple passengers traveling. A single Ticket can be used to give only one Rating.

11) Pilot CAPTAIN of Flight WITH CABIN HEAD Flight attendant HANDLED by Ground staff

- DEGREE - 4
- For a particular flight, we have 1 pilot,1 flight attendant head and 1ground staff manager.
- A pilot can captain multiple flights scheduled at different times. Similarly, the flight attendant head and ground staff manager can also handle multiple flights scheduled at different times.
- **Participation Constraints**:
    1) Pilot (0, N): A pilot can captain multiple flights.
    2) Flight (1,1): Each flight has one pilot, one cabin head, and one ground staff manager.
    3) Flight Attendant (0, N): A cabin head can handle multiple flights.
    4) Ground Staff (0, N): A ground staff manager can handle multiple flights.

Functional Requirements:

MODIFICATIONS:

1) INSERT:

1. **Employee** entry when new employee joins
2. **Flight schedule** entry when a new flight is scheduled.
3. **Aircraft** entry when a new aircraft is purchased.
4. **Passenger** entry when a new passenger signs up.
5. **Meal** entry when a new meal option is added.
6. **Ticket** when a passenger books a ticket.
7. **Baggage** entry when passenger checks in baggage
8. **Rating** entry when passenger provides feedback.
9. **Airport** entry when a new airport is constructed.

2) DELETE

1. **Employee** entry when an employee leaves the job.
2. **Aircraft** entry when it is retired, sold or replaced.
3. **Flight** scheduled entry when no longer relevant/ data cleanup.
4. **Meal** entry when meal option is removed.
5. **Passenger** entry when the passenger deletes their account.
6. **Ticket** once flights are concluded or passenger requirements are fulfilled.
7. **Baggage** once flights are concluded or passenger requirements are fulfilled.

3) UPDATE

1. **Pilot information** such as license information, flight hours, aircraft type ratings, rank, or languages spoken.

2. **Flight attendant** details, including languages spoken, service awards, or emergency training.
3. **Flight scheduled** for changes in departure/arrival times, gate/terminal assignments, and flight status
4. **Aircraft data** for changes in manufacturer details, model, purchase information, capacity, or flight distance.
5. **Passenger records** for changes in personal information such as name, address, or contact details.
6. **Airport information** for changes in code, name, location, or point of contact details.
7. **Ticket** details for changes in class, price, seat allocation, baggage allowance, or meal selection

RETRIEVALS:

1) SELECTION

1. List all attributes of **Ground Staff** in a particular Airport.
2. List details of all **Flights Scheduled** that fly from Airport A to Airport B
3. List details of all **Passengers** in a particular Flight Scheduled

2) PROJECTION

1. Retrieve names of **Technicians and their respective Specialization.**
2. List **Vendor and Price for menu** planning of Meal.
3. List the **Aircraft ID** of Aircrafts with X Capacity.

3) AGGREGATE

1.Find the **Maximum and Minimum Ticket Prices** by Class.
2. Calculate the **Average Rating** for a Flight scheduled.
3. Calculate **total weight** of all checked in Baggage.
4. **Total cost** of all Tickets with a given PNR.

4) SEARCH

1. Search for **Aircrafts purchased after 2001.**
2. Search for **Meals that contain chicken.**
3. Search for **Flight Attendants based on Language spoken.**

5) ANALYSIS

1. Analyse the **busiest Airports** and the **number of Flights** departing from them.
2. Find the **Airport with the longest average Flight delay time**.
3. Evaluate the **service quality of Flight Attendants using passenger Ratings.**

Additional features:

- In contrast to traditional airline company databases, our database includes a unique feature, storing the medical histories of passengers, which can be vital in emergency situations.
- The database allows the airline company to identify its top loyal passengers on a specific flight and this data can be used to upgrade their traveling class in case of vacant seats.
- The database retains passenger ratings for each flight, facilitating performance evaluations

of employees and ensuring quality of the meals served. This, in turn, helps in recognizing and rewarding employees based on their service quality.

Summary:

The Airline Management System is a robust database system designed to oversee and streamline airline operations, including employee management, flight scheduling, aircraft maintenance and passenger services. This system offers extensive features for efficiently handling data, generating reports, and conducting in-depth analyses, ultimately aimed at enhancing overall airline performance and ensuring a superior passenger experience.

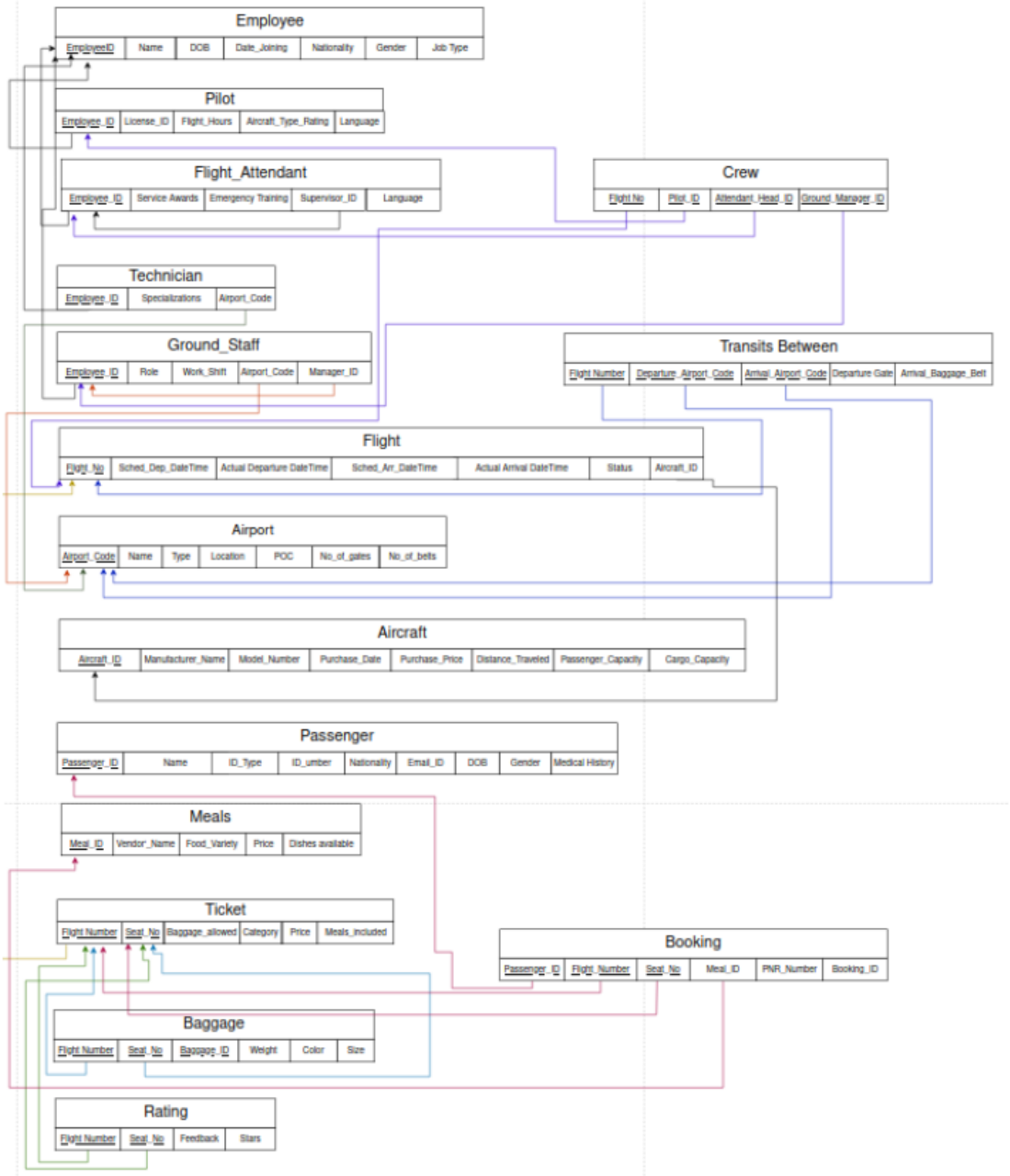
ER TO RELATIONAL MODEL:

1) Created a relation for each strong entity type. Decomposed composite attributes to simple attributes (e.g. Name: Fname, Lname).
2) For Employee (which is a superclass) - one relation comprising common attributes to all subclasses.
3) For all the disjoint subclasses - Individual relations containing Primary Key as EmployeeID and only those attributes which are unique to each subclass.
4) Created a relation for each weak entity type and included primary key of owner as foreign key in the attributes.
5) Incorporated 1:1 binary relationship using foreign key approach, where the primary key of the entity is added as a foreign key attribute to the entity with total participation.
   Example: Flight USES Aircraft
6) Incorporated 1: N and N:1 binary relationship using foreign key approach. Included as foreign key in N-side entity, the primary key of the other entity type.
   Example: Ground Staff POSTED AT Airport
7) Mapped binary M: N relationships using Cross Reference approach, where a separate relation is made which will have the combination of primary keys of the participating entity types as its primary key. Each of these primary keys is a foreign key for the new relation. Other attributes represent the attributes of the relationship in the ER Model.
   Example: Flight SERVES Meals
8) Mapped multivalued attributes- For each entity type, create a separate relation for multi-valued attributes whose foreign key is the primary key of the corresponding. entity.
   Example: Service Awards for Flight Attendants
9) Mapped N-ary relation types- Created a relation whose primary key is combination of primary keys of the participating entities with other attributes being the attributes of the relationship type. Each of these primary keys is a foreign key for the new relation.
   Example: TRANSITS BETWEEN relationship

Minor Changes

- The attribute 'Address' in entity types- Employee and Passenger was removed.
- Some attribute names have been abbreviated (For example Flight Number written as
  Flight No.)
- The attribute 'Meal Included' from Ticket was removed.
- The attribute type for a few attributes were changed like Work-Shift, Role, Colour and
  Size changed from multi-valued to simple.

- Removed address from employee and passenger



Normalization:

1) 1NF

If a relation contains a **composite or multi-valued attribute**, it violates the first normal form, or the relation is in the first normal form if it does not contain any composite or multi-valued attribute. A relation is in first normal form if every attribute in that relation is a single-valued attribute.

The table has following **Multivalued Attributes**:

- Languages known.
- Flight Attendant's Service awards
- Technician's Specialization
- Flight Attendant's Emergency Training
- Airport's POC
- Serves
- Passenger's's Medical History
- Dishes Available (Meals)

The table has the following **Composite Attributes**:

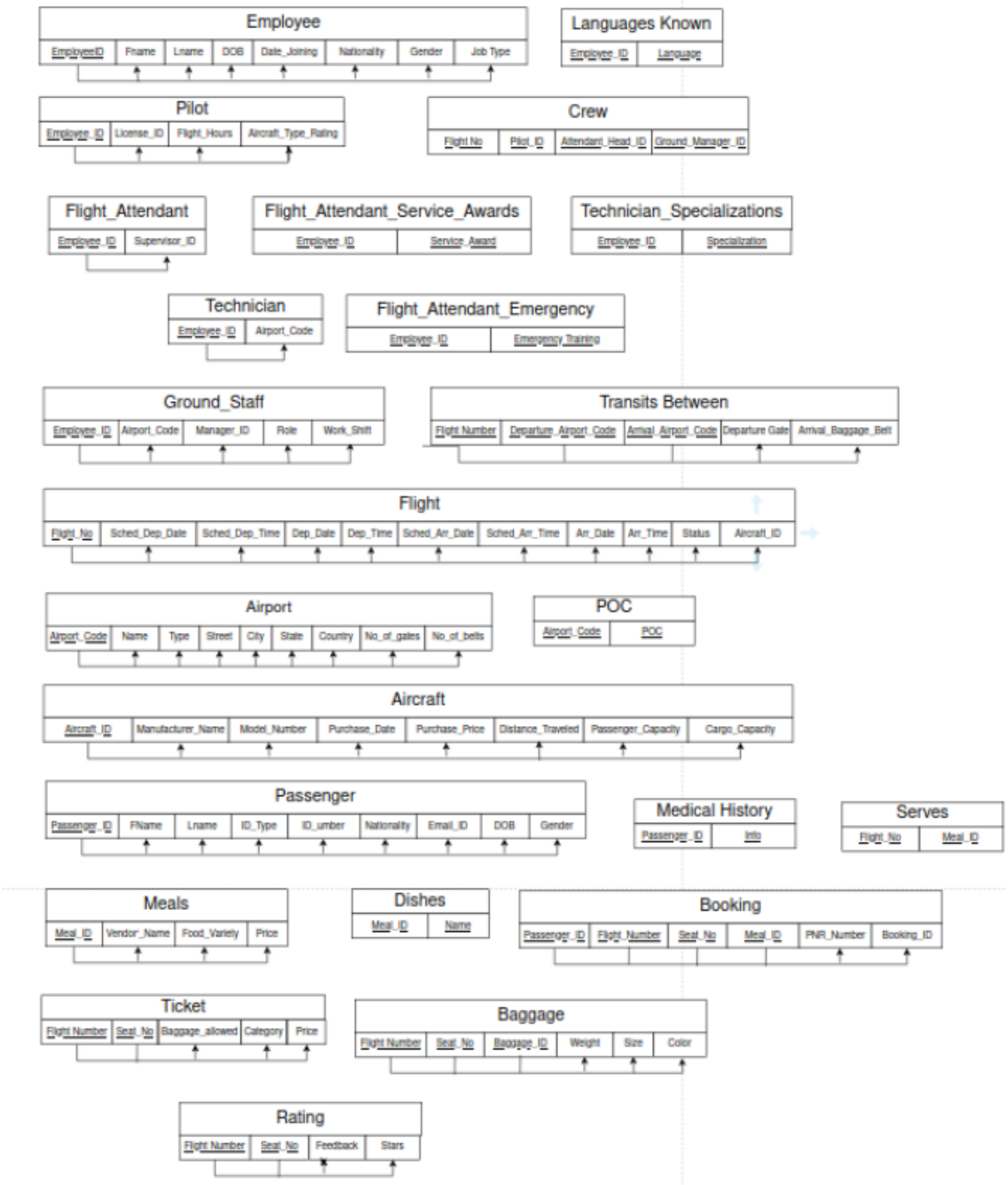
- Scheduled Departure Date Time
- Actual Departure Date Time
- Scheduled Arrival Date Time
- Actual Arrival Date Time
- Name
- Location

1) To remove multivalued attributes for each entity type, create a separate relation for multi-valued attributes whose foreign key is the primary key of the corresponding entity. The **Primary Key for the new table formed is the combination of the Foreign Key and the Multivalued Attribute.**

   For Example: Service Awards for Flight Attendants, Medical history of Passengers.

2) Secondly to remove **composite attributes, instead of a single column, a composite key uses multiple columns to form the identifier.**

   For Example: Scheduled date and scheduled time are placed in separate columns instead of one in flights, name of employee broken down into first name and last name in employee.



2) 2NF

The second Normal Form (2NF) is based on the concept of fully functional dependency. The second Normal Form applies to relations with **composite keys,** that is, relations with a primary key composed of two to or more attributes. A relation with a single-attribute primary key is automatically in at least 2NF.

A relation is in 2NF if it has No Partial Dependency, i.e., no non-prime attribute (attributes that are not part of any candidate key) is dependent on any proper subset of any candidate key of the table.

Removal of Partial Dependencies -

1. In Transits Between Table -

❖ The Type (of flight, International/Domestic) can be determined only with the help of Flight Number. Hence, we made a different table for determining the Type.

❖ The Departure Gate can be retrieved by using only the Flight Number and the Departure_Airport_Code similarly the Arrival_Baggage_Belt can be retrieved using the Flight Number and the Arrival_Airport_Code.

Hence, we can split the table into these different tables to ensure that there are No Partial Dependencies.

2. In Rating Table -

❖ The Feedback and Stars given for a particular rating can be retrieved using Flight Number and Seat Number or Flight Number and

Rating Number (Since, a person in a given flight is allowed to give ratings only once). Therefore, we can split the table into two different tables to ensure that there are No Partial Dependencies.

3) 3NF

In essence, for a database to be in third normal form (3NF):

1. It must already be in 2NF.

2. It should eliminate transitive dependencies. This means that **every non-prime attribute (an attribute not part of any candidate key) should be functionally dependent on the primary key, rather than on another non-prime attribute.**

Note: If P -> Q and Q -> R are two functional dependencies, then P -> R is known as a transitive dependency. When normalizing a 2NF relation to 3NF, we remove these transitive dependencies.

By removing transitive dependencies, 3NF ensures that each non-prime attribute is related to the primary key, improving data integrity, and minimizing update anomalies.

Removing Transitive Dependencies -

❖ In the AIRCRAFT table, the attributes manufacturer, purchase price, passenger

capacity and cargo capacity have been eliminated as these are dependent on

model_no. and model_no. in turn depends on the aircraft_id.

❖ To eliminate these transitive dependencies, model_no., manufacturer, purchase

price, passenger capacity and cargo capacity have been added to a new table Aircraft_Model_Details, with model_no. as the primary key.

Assumption: Aircrafts of the same model_no. have the same manufacturer, purchase price, passenger capacity and cargo capacity.

❖ Similarly, in the AIRPORT table the attributes state and country have been removed as these can be determined from the city which depends on the Airport Code. These have been added to another table ADDRESS with city as the primary key.

Assumption: City names are unique, and no two states have the same city names.