SkyHop

Airline Ticketing System

by

Varshith Peddineni
Venu Dattathreya Vemuru
Firoz Khan Patan
Yaswanth Ravi Teja Dammalapati
Kavya Mothukuri

OBJECTIVE

Our goal:
Evaluate and contrast three schema-design approaches for SkyHop’s ticketing database.

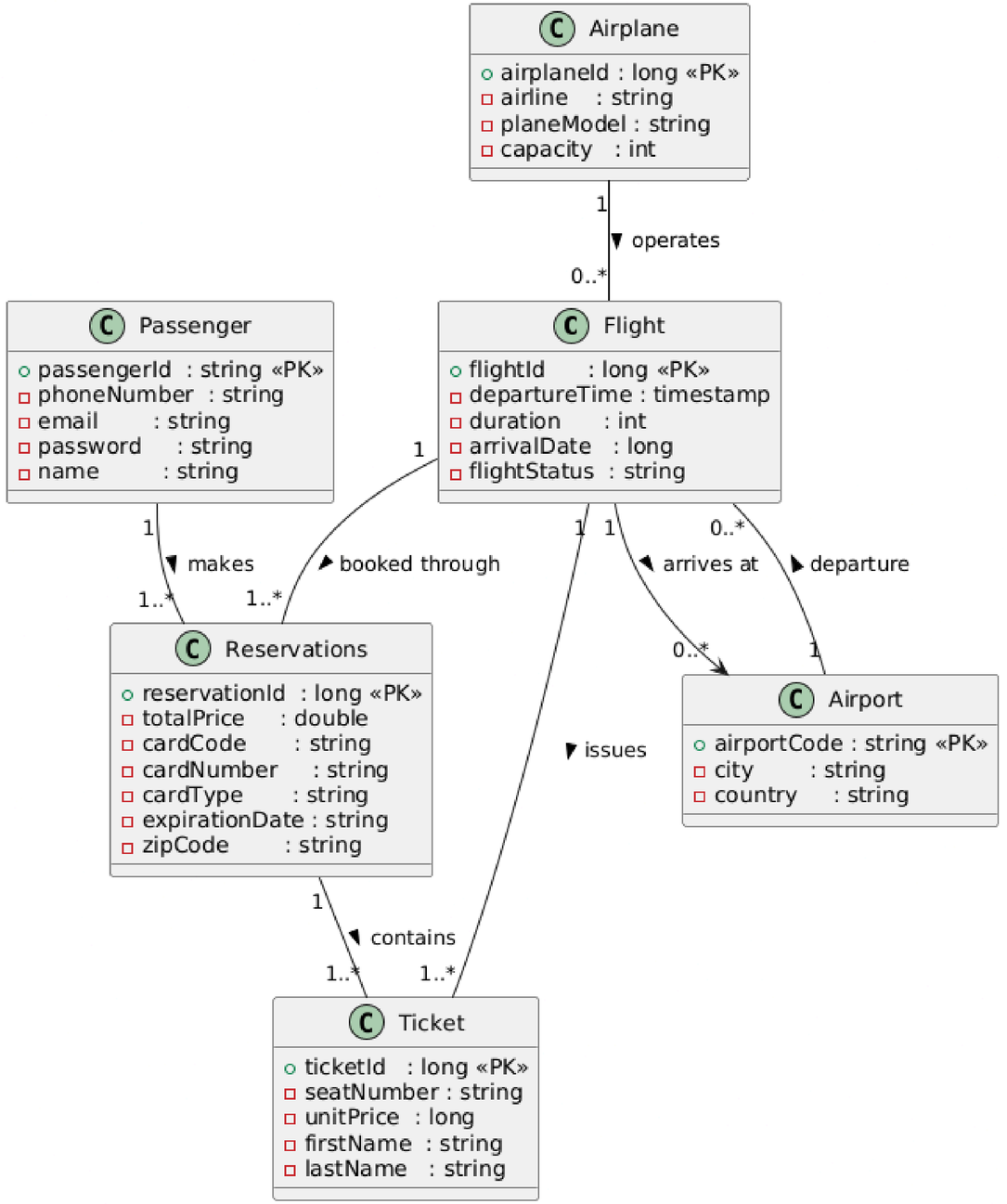
Compare the visual, intuitive UML model against normalized schemas (3NF and BCNF)

Assess each approach for redundancy elimination, dependency handling, and practical complexity

Identify the most balanced design for performance, maintainability, and data integrity

UML DIAGRAM

- 6 core tables: Airplane, Airport, Flight, Ticket, Reservations, Passenger.
- 6 primary keys: one unique ID column in each table.
- 7 foreign keys: every many-to-one link adds a column Flight has 3 (airplane + departure + arrival airports), Ticket has 2 (flight + reservation), Reservations has 2 (passenger + flight) ⇒ 3 + 2 + 2 = 7.



UML Diagram to Relational Model

Table number: 6
Primary key: 6
Foreign key: 7
Tables:

R1: (Airplane): airplaneId, airline, planeModel, capacity
R2: (Airport): airportCode, city, country
R3: (Flight): flightId, departureTime, duration, flightStatus, *airplaneid*, *startAirportCode*, *endAirportCode*
R4: (Passenger): passengerId, phoneNumber, email, password, name
R5: (Ticket): ticketId, seatNumber, unitPrice, *flightId*, *reservationId*, firstName, lastName
R6: (Reservation): reservationId, totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, *passengerId, flightId*

Third Normal Form - 3NF

airplaneId --> airline, planeModel, capacity
airportCode --> city, country
flightId --> departureTime, duration, flightStatus, airplaneid, startAirportCode, endAirportCode
passengerId --> phoneNumber, email, password, name
ticketId --> seatNumber, unitPrice, flightId, reservationId, firstName, lastName
reservationId --> totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, passengerId, flightId

Third Normal Form – 3NF

Step 1 : Minimal Coverage (RHS are singleton sets)

- airplaneId --> airline
- airplaneId --> planeModel
- airplaneId --> capacity
- airportCode -> city
- airportCode -> country
- flightId -> departureTime
- flightId -> duration
- flightId -> flightStatus
- flightId -> airplaneid
- flightId -> startAirportCode
- flightId -> endAirportCode
- passengerId -> phoneNumber
- passengerId -> email
- passengerId -> password
- passengerId -> name
- ticketId -> seatNumber
- ticketId -> unitPrice
- ticketId -> flightId
- ticketId -> reservationId
- ticketId -> firstName
- ticketId -> lastName
- reservationId -> totalPrice
- reservationId -> cardCode
- reservationId -> cardNumber
- reservationId -> cardType
- reservationId -> expirationDate
- reservationId -> zipCode
- reservationId -> passengerId
- reservationId -> flightId

Step 2 : Minimal Coverage (LHS has no extraneous attributes)

- airplaneId --> airline
- airplaneId --> planeModel
- airplaneId --> capacity
- airportCode -> city
- airportCode -> country
- flightId -> departureTime
- flightId -> duration
- flightId -> flightStatus
- flightId -> airplaneid
- flightId -> startAirportCode
- flightId -> endAirportCode
- passengerId -> phoneNumber
- passengerId -> email
- passengerId -> password
- passengerId -> name
- ticketId -> seatNumber
- ticketId -> unitPrice
- ticketId -> flightId
- ticketId -> reservationId
- ticketId -> firstName
- ticketId -> lastName
- reservationId -> totalPrice
- reservationId -> cardCode
- reservationId -> cardNumber
- reservationId -> cardType
- reservationId -> expirationDate
- reservationId -> zipCode
- reservationId -> passengerId
- reservationId -> flightId

Step 3 : Minimal Coverage (No redundant FD’s)

- airplaneId --> airline
- airplaneId --> planeModel
- airplaneId --> capacity
- airportCode -> city
- airportCode -> country
- flightId -> departureTime
- flightId -> duration
- flightId -> flightStatus
- flightId -> airplaneid
- flightId -> startAirportCode
- flightId -> endAirportCode
- passengerId -> phoneNumber
- passengerId -> email
- passengerId -> password
- passengerId -> name
- ticketId -> seatNumber
- ticketId -> unitPrice
- ticketId -> flightId
- ticketId -> reservationId
- ticketId -> firstName
- ticketId -> lastName
- reservationId -> totalPrice
- reservationId -> cardCode
- reservationId -> cardNumber
- reservationId -> cardType
- reservationId -> expirationDate
- reservationId -> zipCode
- reservationId -> passengerId
- reservationId -> flightId

Third Normal Form – 3NF

Step 2 : Merge the FD with the same LHS

airplaneId --> airline, planeModel, capacity
airportCode --> city, country
flightId --> departureTime, duration, flightStatus, airplaneid, startAirportCode, endAirportCode
passengerId --> phoneNumber, email, password, name
ticketId --> seatNumber, unitPrice, flightId, reservationId, firstName, lastName
reservationId --> totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, passengerId, flightId

Step 3 : Form a table for each FD

R1: (Airplane): airplaneId, airline, planeModel, capacity
R2: (Airport): airportCode, city, country
R3: (Flight): flightId, departureTime, duration, flightStatus, airplaneid, startAirportCode, endAirportCode
R4: (Passenger): passengerId, phoneNumber, email, password, name
R5: (Ticket): ticketId, seatNumber, unitPrice, flightId, reservationId, firstName, lastName
R6: (Reservation): reservationId, totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, passengerId, flightId

Third Normal Form - 3NF

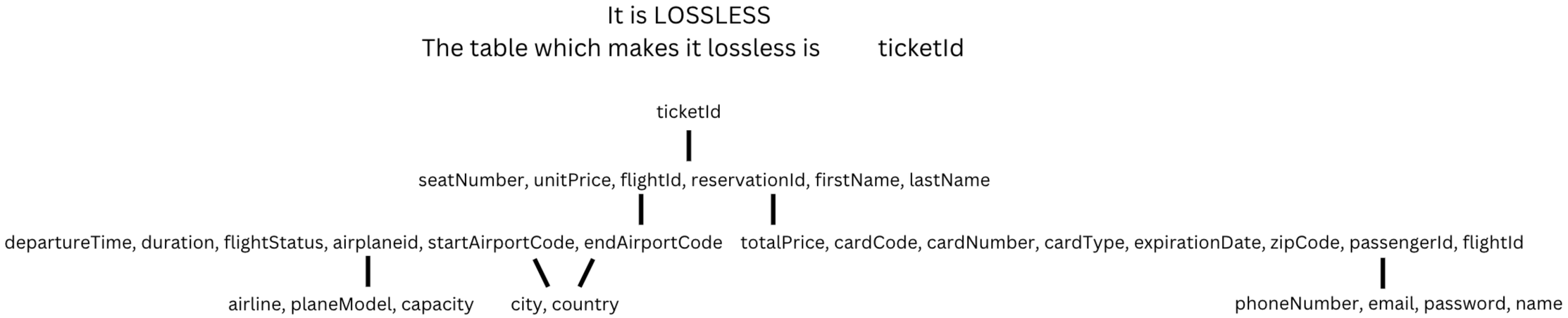
Step 4 : Remove a subset table

R1: (Airplane): airplaneId, airline, planeModel, capacity
R2: (Airport): airportCode, city, country
R3: (Flight): flightId, departureTime, duration, flightStatus, airplaneid, startAirportCode, endAirportCode
R4: (Passenger): passengerId, phoneNumber, email, password, name
R5: (Ticket): ticketId, seatNumber, unitPrice, flightId, reservationId, firstName, lastName
R6: (Reservation): reservationId, totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, passengerId, flightId

Step 5 : Check for losslesness

R1: (Airplane): airplaneId, airline, planeModel, capacity
R2: (Airport): airportCode, city, country
R3: (Flight): flightId, departureTime, duration, flightStatus, airplaneid, startAirportCode, endAirportCode
R4: (Passenger): passengerId, phoneNumber, email, password, name
R5: (Ticket): ticketId, seatNumber, unitPrice, flightId, reservationId, firstName, lastName
R6: (Reservation): reservationId, totalPrice, cardCode, cardNumber, cardType, expirationDate, zipCode, passengerId, flightId

Third Normal Form - 3NF



So, as you can see in the image above, we normalized the Relational Model using Third Normal Form (3NF Synthesis).
These were the steps we followed:

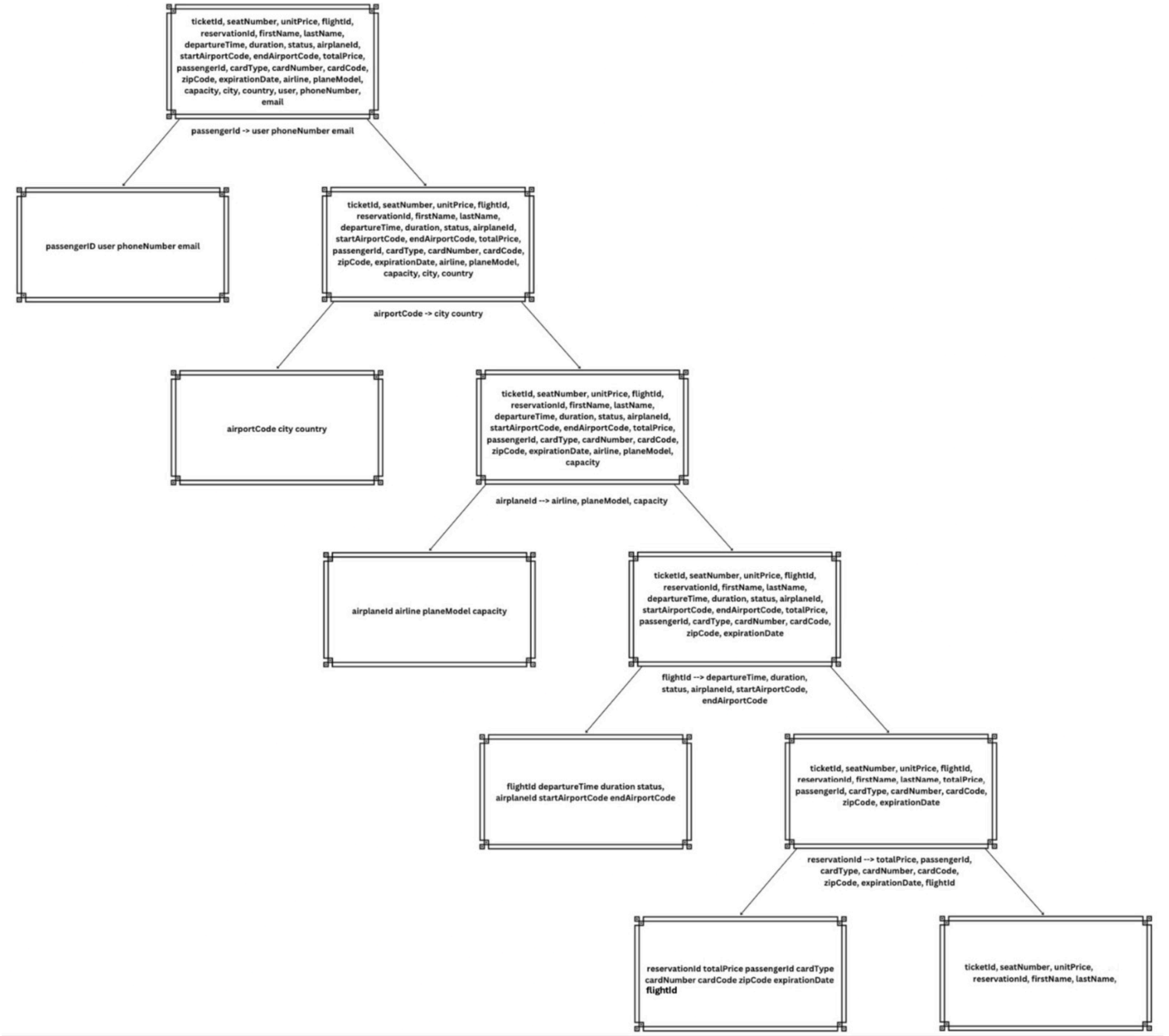
1. Minimal Coverage
   a. Handle Singleton Sets
   b. No Extraneous Attributes
   c. No Redundant FDs
2. Merge FDs with the same Left-Hand Side (LHS)
3. Form a Table for each FD
4. Remove Subset Tables
5. Check for Losslessness

Boyce Codd Normal Form – BCNF

Iterative FD splits: We partitioned the Relational Model step-by-step by each functional dependency (passengerId →..., airportCode →..., airplaneId →..., flightId →...).

Conflict at reservationId: When splitting on reservationId, the Ticket table's dependency on flightId created a violation.

BCNF abandoned: Adding flightId back to preserve data broke BCNF, so we rejected this schema.



Comparison

Uml

→Visual representation of database structure and relationships

→Does not address functional dependencies

→Not a database normalization method

3NF

→Reduces redundancy

→Removes transitive dependencies

→Is a practical implementation of normalization

BCNF

→Is a stricter normalization technique

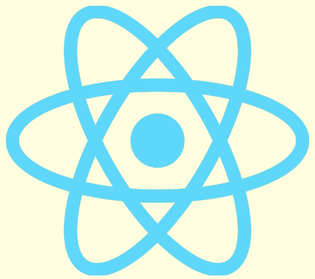
→Removes all redundancies

→Normalization can result in a more complex schema

TechStack

Software Architecture and Components

Frontend: React – HTML, CSS, JavaScript



Backend: Java, SpringBoot, Node.js

Database: MySQL

DEMO