**MINISTERUL EDUCAŢIEI ȘI CERCETĂRII AL REPUBLICII MOLDOVA**

**Universitatea Tehnică a Moldovei**

**Facultatea Calculatoare, Informatică şi Microelectronică**

**Departamentul Ingineria Software și Automatică**

**Programul de studii: Tehnologia informației**

**Proiectarea unei baze de date pentru un Sistem de Managment al unui companii Aeronautice**

**Lucru Individual**

|  |  |  |
| --- | --- | --- |
| **Student(ă):** | **\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Vlași**ț**chi Ștefan , TI-216** |
| **Coordonator universitate:** | **\_\_\_\_\_\_\_\_\_\_\_\_\_** | **Saranciuc Dorian, asist.univ.** |

**Chişinău, 2023**

**CUPRINS**

[1 Introducere și Scopul Proiectului 3](#_Toc155752976)

[2 Proiectarea logică a bazei de date 4](#_Toc155752977)

[2.1 Analiza domeniului bazei de date. Descrierea atributelor 4](#_Toc155752978)

[2.2 Descrierea structurii BD 4](#_Toc155752979)

[2.3 Descrierea asocierilor între relații 4](#_Toc155752980)

[2.4 Stabilirea ordinii de creare și populare cu tupluri a relațiilor bazei de date 4](#_Toc155752981)

[3 Implementare și Populare a Bazei de Date 4](#_Toc155752982)

[Concluzie 4](#_Toc155752983)

[Anexa A 4](#_Toc155752984)

# Introducere și Scopul Proiectului

În era digitală actuală, gestionarea eficientă a informațiilor reprezintă un aspect crucial pentru buna funcționare a oricărei organizații. Acest proiect se axează pe dezvoltarea unui sistem de management al companiei aeriene, o soluție software proiectată pentru a optimiza operațiunile și a facilita procesul de luare a deciziilor în cadrul unei companii aeriene.

Companiile aeriene se confruntă cu diverse provocări, cum ar fi gestionarea flotei de aeronave, programarea zborurilor, administrarea echipajelor și furnizarea de servicii clienților. Pentru a face față acestor provocări, sistemul propus vine în ajutor prin integrarea unei baze de date robuste și prin furnizarea unor funcționalități software avansate.

Scopul principal al acestui proiect constă în proiectarea, implementarea și testarea unui sistem de management al companiei aeriene, care să ofere o platformă centralizată pentru gestionarea eficientă a resurselor și informațiilor. Obiectivele specifice ale proiectului includ proiectarea unei baze de date solide pentru stocarea informațiilor despre aeronave, echipaje, clienți și zboruri, implementarea unui script SQL unic pentru crearea și popularea inițială a bazei de date, dezvoltarea unui model conceptual al datelor și realizarea unei diagrame a bazei de date pentru o înțelegere vizuală și eficientă a relațiilor dintre entități.

De asemenea, proiectul vizează definirea unei scheme logice a bazei de date, specificând cheile primare, secundare și relațiile externe, precum și descrierea detaliată a tipurilor de asocieri între entități și a constrângerilor de integritate referențială. Prin atingerea acestor obiective, ne propunem să furnizăm o soluție software comprehensivă și adaptabilă, menită să sprijine activitățile cotidiene ale unei companii aeriene.

Top of Form

# Proiectarea logică a bazei de date

# Analiza domeniului bazei de date. Descrierea atributelor

Domeniul lumii reale pentru care va fi creată baza de date (BD) este cel al unei companii aeriene. Aceasta include gestionarea și monitorizarea aspectelor legate de avioane, piloți, pasageri, zboruri și servicii asociate.

Entități și Atribute Cheie:

1. PlaneModel:

- Entitatea care reprezintă modelele de avioane.

- Atribute cheie: ModelNumber (identificator unic).

2. License:

- Informații despre licențele necesare pentru piloți.

- Atribute cheie: LicenseNo (identificator unic).

3. Pilot:

- Informații despre piloți și licențele lor.

- Atribute cheie: PilotID (identificator unic).

4. PlaneDetail:

- Detalii specifice despre fiecare avion în parte.

- Atribute cheie: PlaneID (identificator unic).

5. PassengerDetails:

- Detalii despre pasageri și documentele lor de identitate.

- Atribute cheie: PassengerID (identificator unic).

6. Flight:

- Informații despre zboruri.

- Atribute cheie: FlightNo și FlightDate (identificatori unici combinați).

7. PassengerInstance:

- Informații despre pasagerii prezenți pe fiecare zbor.

- Atribute cheie: PassengerID, FlightNo și FlightDate (identificatori unici combinați).

8. FlightAttendant:

- Detalii despre însoțitorii de zbor.

- Atribute cheie: AttendantID (identificator unic).

9. FlightInstance:

- Informații specifice despre fiecare instanță de zbor.

- Atribute cheie: InstanceID (identificator unic).

10. InstanceAttendant:

- Asociază însoțitorii de zbor cu instanțele de zbor.

- Atribute cheie: InstanceID și AttendantID (identificatori unici combinați).

11. Service:

- Informații despre serviciile oferite pe zboruri.

- Atribute cheie: ServiceID (identificator unic).

12. FlightService:

- Asociază serviciile cu fiecare zbor.

- Atribute cheie: FlightNo, FlightDate și ServiceID (identificatori unici combinați).

Baza de date acoperă toate aspectele relevante pentru o companie aeriană, inclusiv modelele de avioane, licențele piloților, informații despre pasageri, detalii despre zboruri și servicii asociate. Relațiile dintre entități permit gestionarea eficientă a operațiunilor zilnice ale companiei, inclusiv programarea și monitorizarea zborurilor, evidența pasagerilor și administrarea personalului navigant. Utilizarea cheilor primare și străine asigură integritatea datelor și relațiilor între diferitele tabele ale bazei de date.

Tipuri de date diverse:

INT, VARCHAR(255), DATE, DECIMAL(10,2), DATETIME

Constrângeri de integritate:

**PRIMARY KEY:** PlaneModel.ModelNumber, License.LicenseNo, Pilot.PilotID, PlaneDetail.PlaneID, PassengerDetails.PassengerID, Flight.FlightNo, Flight.FlightDate, FlightAttendant.AttendantID, FlightInstance.InstanceID, InstanceAttendant (Composite PK)

**UNIQUE:** License.LicenseNo, PlaneDetail.RegistrationNo, PassengerDetails.PassportNo, FlightService (Composite Unique Constraint)

**NOT NULL:** Toate atributele care nu pot avea valori nule

**CHECK:** FlightArriveFrom != FlightDepartTo (în tabela Flight)

**DEFAULT:** Implicita pentru DateTimeLeave si DateTimeArrive in FlightInstance

Aceasta proiectare reflectă necesitățile bazei de date pentru o companie aeriană și respectă cerințele impuse pentru diverse tipuri de asocieri și constrângeri de integritate.

Tabelul 1.1 - Exemplu de specificare pentru relația **PlaneModel**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume atribut** | **Descriere atribut** | **Constrângere de domeniu (Tip de date)** | **Valori nedefinite** | **Constrângeri de comportament de atribut** |
| ModelNumber | Numărul de model al avionului. | INT (IDENTITY) | NOT NULL | PRIMARY KEY |
| ManufacturerName | Numele producătorului avionului. | VARCHAR(255) | NOT NULL | - |
| PlaneRange | Distanța maximă pe care avionul o poate parcurge. | INT | NOT NULL | - |
| CruiseSpeed | Viteza de croazieră a avionului. | INT | NOT NULL | - |

Tabelul 1.2 - Exemplu de specificare pentru relația PlaneModel.

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | ModelNumber | Nu pot exista două modele de avioane cu același număr de identificare |
| Constrângere de comportament de atribut (între atribute) | ManufacturerName | Numele producătorului trebuie să aibă cel puțin 3 caractere |
|  | PlaneRange, CruiseSpeed | Distanța și viteza de croazieră trebuie să fie mai mari decât zero |
| Constrângere de relație | - | - |

Tabelul 2.1 - Exemplu de specificare pentru relația **Licence**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| LicenseNo | Numărul licenței. | INT (IDENTITY) | Nu | PRIMARY KEY |
| LicenseType | Tipul licenței. | VARCHAR(255) | Nu | - |

Tabelul 2.2 - Exemplu de Specificare pentru Relația **License**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | LicenseNo | Nu pot exista două licențe cu același număr de identificare |
| Constrângere de comportament de atribut (între atribute) | LicenseType | Tipul licenței trebuie să fie unul din valorile prestabilite |
| Constrângere de relație | - | - |

Tabelul 3.1 - Exemplu de specificare pentru relația **Pilot**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| PilotID | Numărul de identificare al pilotului. | INT (IDENTITY) | Nu | PRIMARY KEY |
| LicenseNo | Numărul licenței de pilot. | INT | Nu | FOREIGN KEY REFERENCES License(LicenseNo) |
| FirstName | Prenumele pilotului. | VARCHAR(255) | Nu | - |
| LastName | Numele pilotului. | VARCHAR(255) | Nu | - |
| DOB | Data de naștere a pilotului. | DATE | Nu | CHECK(DOB < CURRENT\_DATE) |
| HoursFlown | Numărul de ore de zbor ale pilotului. | INT | Da | DEFAULT(0) |

Tabelul 3.2 - Exemplu de Specificare pentru Relația **Relația Pilot**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | PilotID | Nu pot exista doi piloți cu același număr de identificare |
| Cheie Secundară 1 | LicenseNo | Nu pot exista doi piloți cu același număr de licență |
| Cheie Secundară 2 | FirstName, LastName | Nu pot exista doi piloți cu același nume și prenume |
| Constrângere de comportament de atribut (între atribute) | DOB, HoursFlown | Data de naștere trebuie să fie înainte de data curentă |
| Constrângere de relație | - | - |

Tabelul 4.1 - Exemplu de specificare pentru relația **PlaneDetail**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| PlaneID | Numărul de identificare al avionului. | INT (IDENTITY) | Nu | PRIMARY KEY |
| ModelNumber | Numărul de model al avionului. | INT | Nu | FOREIGN KEY REFERENCES PlaneModel(ModelNumber) |
| RegistrationNo | Numărul de înregistrare al avionului. | VARCHAR(255) | Nu | UNIQUE |
| BuiltYear | Anul de fabricație al avionului. | INT | Nu | CHECK(BuiltYear < YEAR(GETDATE())) |
| FirstClassCapacity | Capacitatea de locuri la clasa întâi. | INT | Nu | DEFAULT(0) |
| EcoCapacity | Capacitatea de locuri la clasa economică. | INT | Nu | DEFAULT(0) |

Tabelul 4.2 - Exemplu de Specificare pentru Relația **PlaneDetail**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | PlaneID | Nu pot exista două avioane cu același număr de identificare |
| Cheie Secundară 1 | ModelNumber | Nu pot exista două avioane de același model |
| Cheie Secundară 2 | RegistrationNo | Nu pot exista două avioane cu același număr de înregistrare |
| Constrângere de comportament de atribut (între atribute) | BuiltYear, FirstClassCapacity, EcoCapacity | Anul de fabricație trebuie să fie înainte de anul curent |
| Constrângere de relație | - | - |

Tabelul 5.1 - Exemplu de specificare pentru relația **PassengerDetails**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| PassengerID | Numărul de identificare al pasagerului. | INT (IDENTITY) | Nu | PRIMARY KEY |
| PassportNo | Numărul de pașaport al pasagerului. | VARCHAR(255) | Nu | UNIQUE |
| PassengerFirstName | Prenumele pasagerului. | VARCHAR(255) | Nu | - |
| PassengerLastName | Numele pasagerului. | VARCHAR(255) | Nu | - |
| PassengerDOB | Data de naștere a pasagerului. | DATE | Nu | CHECK(PassengerDOB < CURRENT\_DATE) |

Tabelul 5.2 - Exemplu de Specificare pentru Relația **PassengerDetails**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | PassengerID | Nu pot exista doi pasageri cu același număr de identificare |
| Cheie Secundară | PassportNo | Nu pot exista doi pasageri cu același număr de pașaport |
| Constrângere de comportament de atribut (între atribute) | PassengerFirstName, PassengerLastName, PassengerDOB | Data de naștere trebuie să fie înainte de data curentă |
| Constrângere de relație | - | - |

Tabelul 6.1 - Exemplu de specificare pentru relația **Flight**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| FlightNo | Numărul zborului. | VARCHAR(255) | Nu | PRIMARY KEY |
| FlightDate | Data zborului. | DATE | Nu | PRIMARY KEY |
| FlightDepartTo | Orașul de plecare. | VARCHAR(255) | Nu | - |
| FlightArriveFrom | Orașul de sosire. | VARCHAR(255) | Nu | CHECK(FlightDepartTo <> FlightArriveFrom) |

Tabelul 6.2 - Exemplu de Specificare pentru Relația **Flight**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | FlightNo, FlightDate | Nu pot exista două zboruri cu același număr și aceeași dată |
| Constrângere de comportament de atribut (între atribute) | FlightDepartTo, FlightArriveFrom | Orasele de plecare si sosire trebuie sa fie diferite |
| Constrângere de relație | - | - |

Tabelul 7.1 - Exemplu de specificare pentru relația **PassengerInstance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| PassengerID | Numărul de identificare al pasagerului. | INT | Nu | FOREIGN KEY REFERENCES PassengerDetails(PassengerID) |
| FlightNo | Numărul zborului. | VARCHAR(255) | Nu | FOREIGN KEY REFERENCES Flight(FlightNo) |
| FlightDate | Data zborului. | DATE | Nu | FOREIGN KEY REFERENCES Flight(FlightDate) |
| SeatNo | Numărul de loc. | INT | Nu | - |
| Class | Clasa de călătorie. | VARCHAR(255) | Nu | - |
| Fare | Tariful pasagerului. | DECIMAL(10, 2) | Nu | - |

Tabelul 7.2 - Exemplu de Specificare pentru Relația **PassengerInstance**

|  |  |  |
| --- | --- | --- |
| **Tip C7onstrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | PassengerID, FlightNo, FlightDate | Nu pot exista două înregistrări pentru același pasager, zbor și dată |
| Constrângere de comportament de atribut (între atribute) | SeatNo, Class, Fare | Numărul de loc trebuie să fie unic pentru fiecare pasager și zbor |
| Constrângere de relație | - | - |

Tabelul 8.1 - Exemplu de specificare pentru relația **FlightAttendant**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| AttendantID | Numărul de identificare al însoțitorului de zbor. | INT (IDENTITY) | Nu | PRIMARY KEY |
| FirstName | Prenumele însoțitorului de zbor. | VARCHAR(255) | Nu | - |
| LastName | Numele însoțitorului de zbor. | VARCHAR(255) | Nu | - |
| DOB | Data de naștere a însoțitorului de zbor. | DATE | Nu | CHECK(DOB < CURRENT\_DATE) |
| HireDate | Data angajării însoțitorului de zbor. | DATE | Nu | CHECK(HireDate < CURRENT\_DATE) |

Tabelul 8.2 - Exemplu de Specificare pentru Relația **FlightAttendant**

|  |  |  |
| --- | --- | --- |
| **Tip Constrângere** | **Atributele la Care se Aplică Constrângerea** | **Descrierea Constrângerii** |
| Cheie Primară | AttendantID | Nu pot exista doi însoțitori de zbor cu același număr de identificare |
| Cheie Secundară | FirstName, LastName | Nu pot exista doi însoțitori de zbor cu același nume și prenume |
| Constrângere de comportament de atribut (între atribute) | DOB, HireDate | Data angajării trebuie să fie înainte de data curentă |
| Constrângere de relație |  |  |

Tabelul 9.1 - Exemplu de specificare pentru relația **FlightInstance**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| FlightNo | Numărul zborului. | VARCHAR(255) | Nu | FOREIGN KEY REFERENCES Flight(FlightNo) |
| FlightDate | Data zborului. | DATE | Nu | FOREIGN KEY REFERENCES Flight(FlightDate) |
| PilotID | Numărul de identificare al pilotului. | INT | Nu | FOREIGN KEY REFERENCES Pilot(PilotID) |
| PlaneID | Numărul de identificare al avionului. | INT | Nu | FOREIGN KEY REFERENCES PlaneDetail(PlaneID) |

Tabelul 10.1 - Exemplu de specificare pentru relația **InstanceAttendant**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| FlightNo | Numărul zborului. | VARCHAR(255) | Nu | FOREIGN KEY REFERENCES Flight(FlightNo) |
| FlightDate | Data zborului. | DATE | Nu | FOREIGN KEY REFERENCES Flight(FlightDate) |
| AttendantID | Numărul de identificare al însoțitorului de zbor. | INT | Nu | FOREIGN KEY REFERENCES FlightAttendant(AttendantID) |

Tabelul 11.1 - Exemplu de specificare pentru relația **Service**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| ServiceID | Numărul de identificare al serviciului. | INT (IDENTITY) | Nu | PRIMARY KEY |
| ServiceName | Numele serviciului. | VARCHAR(255) | Nu | - |

Tabelul 12.1 - Exemplu de specificare pentru relația **FlightService**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nume Atribut** | **Descriere Atribut** | **Tipul de Date** | **Valori Nedefinite NULL** | **Constrângeri de Comportament** |
| FlightNo | Numărul zborului. | VARCHAR(255) | Nu | FOREIGN KEY REFERENCES Flight(FlightNo) |
| FlightDate | Data zborului. | DATE | Nu | FOREIGN KEY REFERENCES Flight(FlightDate) |
| ServiceID | Numărul de identificare al serviciului. | INT | Nu | FOREIGN KEY REFERENCES Service(ServiceID) |

# Descrierea **structurii** BD

Baza de date pentru Linia Aeriană:

**bd-AirLine =**

{

PlaneModel,

License,

Pilot,

PlaneDetail,

PassengerDetails,

Flight,

PassengerInstance,

FlightAttendant,

FlightInstance,

InstanceAttendant,

Service,

FlightService

}

Structura fiecărei relații din BD:

1. **PlaneModel**:

- PlaneModel(**ModelNumber** [PK], ManufacturerName, PlaneRange, CruiseSpeed)

2. **License**:

- License(**LicenseNo** [PK], LicenseType)

3. **Pilot**:

- Pilot(**PilotID** [PK], LicenseNo [FK], FirstName, LastName, DOB, HoursFlown)

4. **PlaneDetail**:

- PlaneDetail(**PlaneID** [PK], ModelNumber [FK], RegistrationNo, BuiltYear, FirstClassCapacity, EcoCapacity)

5. **PassengerDetails**:

- PassengerDetails(**PassengerID** [PK], PassportNo, PassengerFirstName, PassengerLastName, PassengerDOB)

6. **Flight**:

- Flight(**FlightNo** [PK], **FlightDate** [PK], FlightDepartTo, FlightArriveFrom)

7. **PassengerInstance**:

- PassengerInstance(**PassengerID** [FK], FlightNo [FK], FlightDate [FK], SeatNo, Class, Fare)

8. **FlightAttendant**:

- FlightAttendant(**AttendantID** [PK], FirstName, LastName, DOB, HireDate)

9. **FlightInstance**:

- FlightInstance(FlightNo [FK], FlightDate [FK], PilotID [FK], PlaneID [FK])

10. **InstanceAttendant**:

- InstanceAttendant(FlightNo [FK], FlightDate [FK], AttendantID [FK])

11. **Service**:

- Service(ServiceID [PK], ServiceName)

12. **FlightService**:

- FlightService(FlightNo [FK], FlightDate [FK], ServiceID [FK])

Mulțimea de chei pentru fiecare relație:

1. PlaneModel:

- K={**ModelNumber**, ManufacturerName, PlaneRange, CruiseSpeed}

- FK={ModelNumber}

2. License:

- K={**LicenseNo**, LicenseType}

- FK={LicenseNo}

3. Pilot:

- K={**PilotID**, LicenseNo, FirstName, LastName, DOB, HoursFlown}

- FK={LicenseNo}

4. PlaneDetail:

- K={**PlaneID**, ModelNumber, RegistrationNo, BuiltYear, FirstClassCapacity, EcoCapacity}

- FK={ModelNumber}

5. PassengerDetails:

- K={**PassengerID**, PassportNo, PassengerFirstName, PassengerLastName, PassengerDOB}

6. Flight:

- K={**FlightNo**, **FlightDate**, FlightDepartTo, FlightArriveFrom}

7. PassengerInstance:

- K={**PassengerID**, **FlightNo**, FlightDate, SeatNo, Class, Fare}

- FK={PassengerID, FlightNo, FlightDate}

8. FlightAttendant:

- K={**AttendantID**, FirstName, LastName, DOB, HireDate}

9. FlightInstance:

- K={**FlightNo**, **FlightDate**, PilotID, PlaneID}

- FK={FlightNo, FlightDate, PilotID, PlaneID}

10. InstanceAttendant:

- K={**FlightNo**, **FlightDate**, AttendantID}

- FK={FlightNo, FlightDate, AttendantID}

11. Service:

- K={**ServiceID**, ServiceName}

12. FlightService:

- K={**FlightNo**, **FlightDate**, ServiceID}

- FK={FlightNo, FlightDate, ServiceID}

Aceasta este structura bazei de date, evidențiind mulțimile de chei potențiale și externe ale fiecărei relații. Cheile primare sunt evidențiate cu bold.

# Descrierea asocierilor între relații

Asocierile dintre relațiile bazei de date:

**1. 1:1 între PlaneModel și PlaneDetail:**

- Relațiile PlaneModel și PlaneDetail sunt legate printr-o asociere de tip unu la unu - 1:1.

- Descrierea asocierii:

- Un model de avion poate avea doar un set de detalii (de exemplu, capacitatea de transport în clasă întâi și economică, anul construcției).

- Detaliile unui avion sunt specifice unui singur model de avion.

- Implementare: La crearea și actualizarea datelor, mai întâi se operează asupra relației PlaneModel, care este independentă și nu conține cheie externă, apoi asupra relației PlaneDetail, care conține cheia externă (ModelNumber) și face referire la prima relație.

**2. 1:N între PlaneModel și PlaneInstance:**

- Relațiile PlaneModel și PlaneInstance sunt legate printr-o asociere de tip unu la mulți - 1:N.

- Descrierea asocierii:

- Un model de avion poate avea mai multe instanțe (exemplare) în cadrul zborurilor (de exemplu, mai multe avioane de același model participând la diferite zboruri).

- O instanță de avion este specifică unui singur model de avion.

- Implementare: La crearea și actualizarea datelor, mai întâi se operează asupra relației PlaneModel, care este independentă și nu conține cheie externă, apoi asupra relației PlaneInstance, care conține cheia externă (ModelNumber) și face referire la prima relație.

**3. N:M între Flight și Passenger:**

- Relațiile Flight și PassengerInstance sunt legate printr-o asociere de tip mulți la mulți - N:M.

- Descrierea asocierii:

- Un zbor poate avea mai mulți pasageri, iar un pasager poate participa la mai multe zboruri.

- Tabela intermediară PassengerInstance gestionează această asociere, conținând cheile externe ale ambelor relații (FlightNo, FlightDate, PassengerID).

- Implementare: La crearea și actualizarea datelor, se operează cu relațiile Flight și PassengerInstance, iar modificările sunt gestionate prin tabela intermediară.

Aceste descrieri ale asocierilor reflectă legăturile dintre entitățile bazei de date și orientează modul în care datele sunt inserate sau actualizate în funcție de constrângerile referențiale.

# Stabilirea ordinii de creare și populare cu tupluri a relațiilor bazei de date

**1. PlaneModel și PlaneDetail:**

- Aceste două relații pot fi create în primul rând, deoarece sunt independente și nu depind de alte relații.

2. **FlightAttendant**:

- Depinde de relația FlightInstance, deoarece FlightAttendant este legată prin cheia externă la FlightInstance.

3. **License**:

- Poate fi creată independent, deoarece nu depinde de alte relații.

4. **Pilot**:

- Poate fi creată independent, deoarece nu depinde de alte relații.

5. **PlaneInstance**:

- Depinde de relația PlaneModel, deoarece PlaneInstance este legată prin cheia externă la PlaneModel.

6. **PassengerDetails**:

- Poate fi creată independent, deoarece nu depinde de alte relații.

7. **Flight**:

- Depinde de relația PlaneInstance, deoarece Flight este legată prin cheia externă la PlaneInstance.

8. **PassengerInstance**:

- Depinde de relațiile Flight și PassengerDetails, deoarece PassengerInstance este legată prin cheile externe la Flight și PassengerDetails.

9. **InstanceAttendant**:

- Depinde de relațiile FlightInstance și FlightAttendant, deoarece InstanceAttendant este legată prin cheile externe la FlightInstance și FlightAttendant.

10. **Service**:

- Poate fi creată independent, deoarece nu depinde de alte relații.

11. **FlightService**:

- Depinde de relațiile FlightInstance și Service, deoarece FlightService este legată prin cheile externe la FlightInstance și Service.

# Implementare și Populare a Bazei de Date

Pentru a materializa proiectarea conceptuală într-o implementare practică, am elaborat un script SQL unic (vezi Anexa A) destinat creării și populating bazei de date. Acest script este meticulos conceput pentru a fi executat fără erori, asigurându-se astfel că structura și datele corespund în totalitate proiectului nostru inițial.

Scriptul oferă un mediu curat prin verificarea existenței bazei de date și, dacă este cazul, prin ștergerea acesteia pentru a permite recrearea. Urmează crearea unei baze de date noi și comutarea execuției scriptului la această bază pentru continuarea operațiunilor.

Procesul de implementare este însoțit de o etapă crucială de testare și validare, în care am dezvoltat scenarii de test ce acoperă diverse aspecte ale funcționalității bazei de date. Această etapă asigură corectitudinea și fiabilitatea datelor manipulate, verificând conformitatea acestora cu cerințele proiectului.

Pentru a garanta coerența și integritatea datelor, scriptul include comenzi specifice de verificare, precum asigurarea cheilor primare, secundare și a constrângerilor de integritate referențială. În plus, se realizează crearea unor indici pentru optimizarea performanțelor.

Exemple de Funcționare

Pentru a ilustra funcționarea corectă a bazei de date conform așteptărilor, prezentăm câteva exemple de interogări și analize realizate asupra datelor:

-- 3.1.1. Uniune între două tabele (Exemplu: Pilot și PassengerDetails)

SELECT

PilotID AS ID,

FirstName,

LastName,

DOB

FROM Pilot

UNION

SELECT

PassengerID AS ID,

PassengerFirstName AS FirstName,

PassengerLastName AS LastName,

PassengerDOB AS DOB

FROM PassengerDetails;

-- 3.1.2 Intersecție între Flight și PassengerInstance

SELECT

F.FlightNo,

F.FlightDate,

NULL AS FlightDepartTo,

NULL AS FlightArriveFrom,

NULL AS Distance

FROM Flight F

INTERSECT

SELECT

PI.FlightNo,

PI.FlightDate,

NULL AS FlightDepartTo,

NULL AS FlightArriveFrom,

NULL AS Distance

FROM PassengerInstance PI;

-- 3.1.3. Diferență între două tabele (Exemplu: Pilot și PassengerDetails)

SELECT

PilotID AS PersonID,

FirstName,

LastName,

DOB,

NULL AS PassportNo

FROM Pilot

EXCEPT

SELECT

PassengerID AS PersonID,

PassengerFirstName AS FirstName,

PassengerLastName AS LastName,

PassengerDOB AS DOB,

PassportNo

FROM PassengerDetails;

-- 3.1.4 Exemplu de complement activ pentru Piloti care nu au licenta

SELECT \*

FROM Pilot

WHERE LicenseNo IS NULL;

--3.1.5 Exemplu de produs cartezian între Pilot și PlaneDetail

SELECT \*

FROM Pilot, PlaneDetail;

-- 3.1.6 Exemplu de selecție și proiecție pentru a obține numele și data nașterii pentru piloții cu mai mult de 500 de ore zburate

SELECT FirstName, LastName, DOB

FROM Pilot

WHERE HoursFlown > 500;

-- 3.1.7Exemplu de joncțiune theta între Pilot și License bazată pe condiția Pilot.LicenseNo = License.LicenseNo

SELECT \*

FROM Pilot

JOIN License ON Pilot.LicenseNo = License.LicenseNo;

--3.1.8 Exemplu de joncțiune naturală între Pilot și License, bazată pe coloanele cu același nume

SELECT \*

FROM Pilot p

JOIN License l ON p.LicenseNo = l.LicenseNo;

-- 3.1.9 Exemplu de semi joncțiune între Pilot și License bazată pe condiția Pilot.LicenseNo = License.LicenseNo

SELECT \*

FROM Pilot

WHERE EXISTS (

SELECT 1

FROM License

WHERE Pilot.LicenseNo = License.LicenseNo

);

--3.1.10 Exemplu de joncțiune externă de stânga între Flight și FlightInstance

SELECT \*

FROM Flight f

LEFT JOIN FlightInstance fi ON f.FlightNo = fi.FlightNo AND f.FlightDate = fi.FlightDate;

-- 3.1.11Exemplu de joncțiune externă de dreapta între Flight și FlightInstance

SELECT \*

FROM Flight f

RIGHT JOIN FlightInstance fi ON f.FlightNo = fi.FlightNo AND f.FlightDate = fi.FlightDate;

--3.1.12 Exemplu de joncțiune externă completă între Flight și FlightInstance

SELECT \*

FROM Flight f

FULL JOIN FlightInstance fi ON f.FlightNo = fi.FlightNo AND f.FlightDate = fi.FlightDate;

-- 3.1.13Exemplu de divizare între Flight și FlightInstance

SELECT \*

FROM Flight f

WHERE NOT EXISTS (

SELECT 1

FROM FlightInstance fi

WHERE f.FlightNo = fi.FlightNo AND f.FlightDate = fi.FlightDate

);

# 3.2. Crearea interogărilor cu funcţii de agregare

--3.2.1 Exemplu de interogare cu funcții de agregare MIN, MAX și AVG

SELECT

MIN(HoursFlown) AS MinHoursFlown,

MAX(HoursFlown) AS MaxHoursFlown,

AVG(HoursFlown) AS AvgHoursFlown

FROM Pilot;

-- 3.2.2Exemplu de interogare cu funcția de agregare SUM

SELECT

PlaneModel.ManufacturerName,

SUM(PlaneDetail.FirstClassCapacity) AS TotalFirstClassCapacity,

SUM(PlaneDetail.EcoCapacity) AS TotalEcoCapacity

FROM PlaneModel

JOIN PlaneDetail ON PlaneModel.ModelNumber = PlaneDetail.ModelNumber

GROUP BY PlaneModel.ManufacturerName;

-- 3.2.3Exemplu de interogare cu funcția de agregare COUNT

SELECT

ManufacturerName,

COUNT(\*) AS NumberOfPlaneModels

FROM PlaneModel

GROUP BY ManufacturerName;

# 3.3. Crearea interogărilor cu grupare

-- 3.3.1 Exemplu de interogare cu grupare pe o singură relație

SELECT

FlightNo,

COUNT(\*) AS NumberOfPassengers,

AVG(Fare) AS AverageFare

FROM PassengerInstance

GROUP BY FlightNo

HAVING COUNT(\*) > 0;

--3.3.2 Exemplu de interogare cu grupare pe mai multe relații și utilizarea de aliasuri pentru a evita coloanele ambigue

SELECT

FI.FlightNo,

FI.FlightDate,

PM.ManufacturerName,

COUNT(\*) AS NumberOfPassengers,

AVG(PI.Fare) AS AverageFare

FROM PassengerInstance PI

JOIN FlightInstance FI ON PI.FlightNo = FI.FlightNo AND PI.FlightDate = FI.FlightDate

JOIN PlaneDetail PD ON FI.PlaneID = PD.PlaneID

JOIN PlaneModel PM ON PD.ModelNumber = PM.ModelNumber

GROUP BY FI.FlightNo, FI.FlightDate, PM.ManufacturerName

HAVING AVG(PI.Fare) = NULL;

# 3.4. Subinterogări

--3.4.1 Exemplu de subinterogare cu operator de comparare

SELECT PilotID, FirstName, LastName

FROM Pilot

WHERE HoursFlown > (SELECT AVG(HoursFlown) FROM Pilot);

-- 3.4.2 Exemplu de subinterogare cu operator IN

SELECT PassengerID, PassengerFirstName, PassengerLastName

FROM PassengerDetails

WHERE PassengerID IN (SELECT PassengerID FROM PassengerInstance WHERE Class = 'First');

--3.4.3 Exemplu de subinterogare cu operator ALL

SELECT PilotID, FirstName, LastName

FROM Pilot

WHERE HoursFlown > ALL (SELECT HoursFlown FROM Pilot WHERE LicenseNo = 1);

-- 3.4.4 Exemplu de subinterogare cu operator ANY

SELECT PilotID, FirstName, LastName

FROM Pilot

WHERE HoursFlown > ANY (SELECT HoursFlown FROM Pilot WHERE LicenseNo = 1);

-- 3.4.5Exemplu de subinterogare cu operator EXISTS

SELECT FlightNo, FlightDate

FROM Flight

WHERE EXISTS (SELECT \* FROM PassengerInstance WHERE FlightNo = Flight.FlightNo AND FlightDate = Flight.FlightDate);

-- 3.4.6 Exemplu de subinterogare corelată

SELECT PilotID, FirstName, LastName

FROM Pilot P

WHERE HoursFlown > (SELECT AVG(HoursFlown) FROM Pilot WHERE LicenseNo = P.LicenseNo);

--3.4.7 Exemplu de comandă cu subinterogare după FROM

SELECT \*

FROM (

SELECT PilotID, FirstName, LastName

FROM Pilot

WHERE HoursFlown > 500

) AS Subquery;

-- 3.4.8 Exemplu de comandă cu subinterogare după SELECT

SELECT PilotID, FirstName, LastName, (SELECT AVG(HoursFlown) FROM Pilot) AS AvgHoursFlown

FROM Pilot;

# Concluzie

În urma proiectului nostru de proiectare și implementare a bazei de date pentru o companie aeriană, am obținut rezultate semnificative în gestionarea eficientă a informațiilor legate de aeronave, echipaje, clienți, zboruri și servicii oferite. Recapitulând rezultatele cheie ale acestui proiect, putem evidenția:

Dezvoltarea și implementarea cu succes a unei baze de date robuste, care acoperă aspecte esențiale precum modelele de avioane, detalii despre piloți și echipaj, date ale pasagerilor și informații despre zboruri.

Crearea unui model conceptual al datelor, ilustrând structura conceptuală a bazei de date și evidențiind entitățile și relațiile dintre acestea.

Realizarea unei diagrame a bazei de date, furnizând o reprezentare vizuală a structurii bazei de date și evidențiind modul în care entitățile sunt organizate și interconectate.

Definirea și prezentarea schemei logice a bazei de date, inclusiv specificații privind cheile primare, secundare și constrângerile de integritate referențială.

Descrierea tipurilor de asocieri între relații, precum "Un la Mulți", "Mulți la Unul" și "Mulți la Mulți", evidențiind modul în care datele sunt legate în cadrul bazei de date.

În timpul proiectului, am identificat unele aspecte care pot beneficia de îmbunătățiri și dezvoltări viitoare:

Optimizarea performanței bazei de date prin revizuirea și ajustarea indexării și a structurii tabelelor pentru a asigura răspunsuri rapide și eficiente la interogări.

Extinderea funcționalităților bazei de date pentru a include rapoarte și analize suplimentare, oferind o perspectivă mai detaliată asupra operațiunilor companiei aeriene.

Implementarea unui sistem de securitate robust pentru protejarea datelor sensibile și asigurarea conformității cu reglementările privind confidențialitatea.

Continuarea testării și validării bazei de date în condiții variate pentru a asigura stabilitatea și fiabilitatea acesteia în medii de producție.

În concluzie, acest proiect a reprezentat o etapă semnificativă în dezvoltarea unei baze de date eficiente pentru gestionarea operațiunilor unei companii aeriene. Observațiile acumulate și sugestiile pentru îmbunătățiri viitoare vor contribui la evoluția și optimizarea continuă a acestei soluții de gestionare a datelor.

# Anexa A

Codul SQL

-- Write your creation script here

IF DB\_ID('AirLine2') IS NOT NULL

BEGIN

PRINT 'Database exists - dropping.';

USE master;

ALTER DATABASE AirLine2 SET SINGLE\_USER WITH ROLLBACK IMMEDIATE;

DROP DATABASE AirLine2;

END

GO

-- Now that we are sure the database does not exist, we create it.

PRINT 'Creating database.';

CREATE DATABASE AirLine2;

GO

-- Now that an empty database has been created, we will make it the active one.

-- The table creation statements that follow will therefore be executed on the newly created database.

USE AirLine2;

GO

-- Create PlaneModel table

CREATE TABLE PlaneModel (

ModelNumber INT IDENTITY(1,1) PRIMARY KEY,

ManufacturerName VARCHAR(255),

PlaneRange INT,

CruiseSpeed INT

);

-- Create License table

CREATE TABLE License (

LicenseNo INT PRIMARY KEY,

LicenseType VARCHAR(255),

LicenseExpireDate DATE

);

-- Create Pilot table

CREATE TABLE Pilot (

PilotID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(255),

LastName VARCHAR(255),

DOB DATE,

HoursFlown INT,

LicenseNo INT UNIQUE,

FOREIGN KEY (LicenseNo) REFERENCES License(LicenseNo)

);

-- Create PlaneDetail table

CREATE TABLE PlaneDetail (

PlaneID INT IDENTITY(1,1) PRIMARY KEY,

ModelNumber INT,

RegistrationNo VARCHAR(255) UNIQUE,

BuiltYear INT,

FirstClassCapacity INT,

EcoCapacity INT,

FOREIGN KEY (ModelNumber) REFERENCES PlaneModel(ModelNumber)

);

-- Create PassengerDetails table

CREATE TABLE PassengerDetails (

PassengerID INT IDENTITY(1,1) PRIMARY KEY,

PassengerFirstName VARCHAR(255),

PassengerLastName VARCHAR(255),

PassengerDOB DATE,

PassportNo VARCHAR(255) UNIQUE,

FullName AS (PassengerFirstName + ' ' + PassengerLastName),

Age AS (DATEDIFF(year, PassengerDOB, GETDATE()))

-- Add other necessary columns as needed

);

-- Create Flight table

CREATE TABLE Flight (

FlightNo INT IDENTITY(1,1),

FlightDate DATE,

FlightDepartTo VARCHAR(255),

FlightArriveFrom VARCHAR(255),

Distance INT,

CHECK (FlightArriveFrom != FlightDepartTo),

PRIMARY KEY (FlightNo, FlightDate)

);

-- Create PassengerInstance table

CREATE TABLE PassengerInstance (

PassengerID INT,

FlightNo INT,

FlightDate DATE,

SeatNo VARCHAR(255),

Class VARCHAR(255),

Fare DECIMAL(10,2),

PRIMARY KEY (PassengerID, FlightNo, FlightDate),

FOREIGN KEY (PassengerID) REFERENCES PassengerDetails(PassengerID),

FOREIGN KEY (FlightNo, FlightDate) REFERENCES Flight(FlightNo, FlightDate)

-- Add other necessary columns as needed

);

-- Create FlightAttendant table

CREATE TABLE FlightAttendant (

AttendantID INT IDENTITY(1,1) PRIMARY KEY,

FirstName VARCHAR(255),

LastName VARCHAR(255),

DOB DATE,

HireDate DATE,

MentorID INT,

FOREIGN KEY (MentorID) REFERENCES FlightAttendant(AttendantID)

);

-- Create FlightInstance table

CREATE TABLE FlightInstance (

InstanceID INT IDENTITY(1,1) PRIMARY KEY,

FlightNo INT,

FlightDate DATE,

PlaneID INT,

PilotAboardID INT,

CoPilotAboardID INT,

FSM\_AttendantID INT,

DateTimeLeave DATETIME,

DateTimeArrive DATETIME,

CHECK (DateTimeArrive > DateTimeLeave),

CHECK (CoPilotAboardID != PilotAboardID),

FOREIGN KEY (FlightNo, FlightDate) REFERENCES Flight(FlightNo, FlightDate),

FOREIGN KEY (PlaneID) REFERENCES PlaneDetail(PlaneID),

FOREIGN KEY (PilotAboardID) REFERENCES Pilot(PilotID),

FOREIGN KEY (CoPilotAboardID) REFERENCES Pilot(PilotID),

FOREIGN KEY (FSM\_AttendantID) REFERENCES FlightAttendant(AttendantID)

);

-- Create InstanceAttendant table

CREATE TABLE InstanceAttendant (

InstanceID INT,

AttendantID INT,

PRIMARY KEY (InstanceID, AttendantID),

FOREIGN KEY (InstanceID) REFERENCES FlightInstance(InstanceID),

FOREIGN KEY (AttendantID) REFERENCES FlightAttendant(AttendantID)

);

-- Create Service table

CREATE TABLE Service (

ServiceID INT PRIMARY KEY,

ServiceName VARCHAR(255),

ServicePrice DECIMAL(10,2)

);

-- Create FlightService table

CREATE TABLE FlightService (

FlightNo INT,

FlightDate DATE,

ServiceID INT,

PRIMARY KEY (FlightNo, FlightDate, ServiceID),

FOREIGN KEY (FlightNo, FlightDate) REFERENCES Flight(FlightNo, FlightDate),

FOREIGN KEY (ServiceID) REFERENCES Service(ServiceID)

);

-- Create an index on the RegistrationNo column in the PlaneDetail table

CREATE INDEX IX\_PlaneDetail\_RegistrationNo ON PlaneDetail (RegistrationNo);

-- Insert data into PlaneModel table

INSERT INTO PlaneModel (ManufacturerName, PlaneRange, CruiseSpeed)

VALUES

('Boeing', 5000, 600),

('Airbus', 4500, 550),

('Embraer', 3000, 500),

('Bombardier', 2000, 450),

('Cessna', 1000, 300),

('Lockheed Martin F-35 Lightning II', 1500, 300),

('McDonnell Douglas MD-80', 2590, 480),

('Bombardier Challenger 300', 500, 400),

('Airbus A380', 8000, 100),

('Piper PA-28 Cherokee', 5500, 1450),

('Sukhoi Su-30', 2222, 870),

('Beechcraft King Air', 2500, 400),

('Boeing 777', 7952, 842),

('Douglas DC-3', 3579, 350);

-- Insert data into License table

INSERT INTO License (LicenseNo, LicenseType, LicenseExpireDate)

VALUES

(1, 'Commercial', '2025-01-01'),

(2, 'Private', '2024-01-01'),

(3, 'Instructor', '2023-01-01'),

(4, 'Student', '2022-01-01'),

(5, 'Sport', '2021-01-01'),

(6, 'Cargo', '2022-05-05');

-- Insert data into Pilot table

INSERT INTO Pilot (FirstName, LastName, DOB, HoursFlown, LicenseNo)

VALUES

('John', 'Doe', '1980-01-15', 1000, 1),

('Jane', 'Smith', '1985-05-20', 800, 2),

('Jack', 'Brown', '1990-03-10', 600, 3),

('Jill', 'Jones', '1995-07-05', 400, 4),

('Jim', 'Wilson', '2000-11-25', 200, 5);

-- Insert data into PlaneDetail table

INSERT INTO PlaneDetail (ModelNumber, RegistrationNo, BuiltYear, FirstClassCapacity, EcoCapacity)

VALUES

(1, 'ABC123', 2010, 20, 150),

(2, 'XYZ789', 2015, 25, 180),

(3, 'DEF456', 2012, 15, 120),

(4, 'UVW345', 2016, 30, 200),

(5, 'GHI789', 2014, 10, 100);

-- Insert data into PassengerDetails table

INSERT INTO PassengerDetails (PassengerFirstName, PassengerLastName, PassengerDOB, PassportNo)

VALUES

('Alice', 'Adams', '1990-01-01', 'AA123456'),

('Bob', 'Baker', '1991-02-02', 'BB234567'),

('Charlie', 'Clark', '1992-03-03', 'CC345678'),

('David', 'Davis', '1993-04-04', 'DD456789'),

('Eve', 'Evans', '1994-05-05', 'EE567890');

-- Insert data into Flight table

INSERT INTO Flight (FlightDate, FlightDepartTo, FlightArriveFrom, Distance)

VALUES

('2021-01-01', 'New York', 'London', 5000),

('2021-01-02', 'Paris', 'Berlin', 1000),

('2021-01-03', 'Tokyo', 'Beijing', 2000),

('2021-01-04', 'Sydney', 'Auckland', 3000),

('2021-01-05', 'Rio de Janeiro', 'Buenos Aires', 4000);

-- Insert data into PassengerInstance table

INSERT INTO PassengerInstance (PassengerID, FlightNo, FlightDate, SeatNo, Class)

VALUES

(1, 1, '2021-01-01', 'A1', 'First'),

(2, 2, '2021-01-02', 'B2', 'Eco'),

(3, 3, '2021-01-03', 'C3', 'First'),

(4, 4, '2021-01-04', 'D4', 'Eco'),

(5, 5, '2021-01-05', 'E5', 'First');

-- Insert data into FlightAttendant table

INSERT INTO FlightAttendant (FirstName, LastName, DOB, HireDate, MentorID)

VALUES

('Frank', 'Foster', '1980-06-06', '2010-01-01', NULL),

('Grace', 'Green', '1981-07-07', '2011-02-02', 1),

('Harry', 'Hall', '1982-08-08', '2012-03-03', 2),

('Ivy', 'Ingram', '1983-09-09', '2013-04-04', 3),

('Kevin', 'King', '1984-10-10', '2014-05-05', 4);

-- Insert data into FlightInstance table

INSERT INTO FlightInstance (FlightNo, FlightDate, PlaneID, PilotAboardID, CoPilotAboardID, FSM\_AttendantID, DateTimeLeave, DateTimeArrive)

VALUES

(1, '2021-01-01', 1, 1, 2, 1, '2021-01-01 10:00:00', '2021-01-01 18:00:00'),

(2, '2021-01-02', 2, 3, 4, 2, '2021-01-02 11:00:00', '2021-01-02 13:00:00'),

(3, '2021-01-03', 3, 5, 1, 3, '2021-01-03 12:00:00', '2021-01-03 16:00:00'),

(4, '2021-01-04', 4, 2, 3, 4, '2021-01-04 13:00:00', '2021-01-04 17:00:00'),

(5, '2021-01-05', 5, 4, 5, 5, '2021-01-05 14:00:00', '2021-01-05 20:00:00');

-- Insert data into InstanceAttendant table

INSERT INTO InstanceAttendant (InstanceID, AttendantID)

VALUES

(1, 1),

(1, 2),

(2, 2),

(2, 3),

(3, 3),

(3, 4),

(4, 4),

(4, 5),

(5, 5),

(5, 1);

-- Insert data into Service table

INSERT INTO Service (ServiceID, ServiceName, ServicePrice)

VALUES

(1, 'Meal', 10.00),

(2, 'Drink', 5.00),

(3, 'Snack', 3.00),

(4, 'Wifi', 15.00),

(5, 'Entertainment', 20.00);

-- Insert data into FlightService table

INSERT INTO FlightService (FlightNo, FlightDate, ServiceID)

VALUES

(1, '2021-01-01', 1),

(1, '2021-01-01', 2),

(2, '2021-01-02', 2),

(2, '2021-01-02', 3),

(3, '2021-01-03', 3),

(3, '2021-01-03', 4),

(4, '2021-01-04', 4),

(4, '2021-01-04', 5),

(5, '2021-01-05', 5),

(5, '2021-01-05', 1);