Progetto Basi di Dati

Massimiliano Sartoretto mat. 1008168,
Pavanello Mirko mat. 1009424

A.A. 2012/2013

Indice

Elenco delle figure

1 Abstract

Il progetto consiste nella realizzazione di una base di dati per la gestione di un'agenzia di viaggi aerei. Lo scopo è di fornire uno strumento che mantenga informazioni circa la gestione di viaggi aerei e offra, attraverso una semplice interfaccia web, un portale di ricerca e acquisto degli stessi. Gli utenti registrati potranno interagire con la base di dati cercando viaggi aerei, secondo le propire preferenze, da un'ampia gamma di opzioni garantita dalla gestione di molteplici compagnie. Il progetto si propone quindi di centralizzare in un unico strumento l'offerta del mercato di settore, aumentando la disponibilità di ricerca e comparazione dei servizi proposti dalle compagnie aderenti

2 Analisi dei requisiti

Il progetto consiste nel realizzare una base di dati per la gestione dei viaggi, delle compagnie e degli acquisti effettuati dagli utenti dell'applicazione web.

Un volo è la primitiva necessaria a definire successivamente un viaggio, è caratterizzato da:

- un numero univoco che lo identifica
- degli orari di partenza e arrivo
- una tratta che specifica gli aeroporti di partenza e arrivo
- una compagnia che ne è proprietaria

Ogni volo è realizzato da uno o più viaggi. Di un viaggio si vuole conoscere:

- il tipo di viaggio, diretto o con scali
- il volo che esegue, tutti i voli se sono più d'uno
- la tratta totale di percorrenza
- la data in cui sarà effettuato
- i dipendenti e la compagnia che lo eseguiranno
- l'aereo con cui sarà eseguito
- lo stato che indica se il viaggio è previsto, eseguito o soppresso
- delle tariffe che ne specificano il prezzo

Ogni viaggio potrà essere diretto o con scali. Un viaggio diretto sarà relativo ad un solo volo mentre un viaggio con scali sarà l'unione di più viaggi diretti. Un viaggio diretto può essere appaltato ad una compagnia diversa rispetto a quella che crea il volo corrispondente, pertanto di entrambe le eventuali compagnie si vuole mantenere memoria.
Agli utenti del sistema sarà richiesto di essere riconosciuti solo al momento dell'acquisto, quindi la possibilità di vedere i viaggi disponibili e le offerte non è soggetta a restrizioni. D'ora in avanti con utente verrá intesa una persona che si è registrata nel sistema e che possiede quindi dei dati di accesso. Per ogni utente si vogliono memorizzare le anagrafiche, i privilegi di cui gode e i dati di accesso, composti da un'email univoca e una password. Un utente è pertanto caratterizzato da:

- delle anagrafiche che lo identificano
- una password che viene memorizzata criptata
- dei privilegi che gli consentono di svolgere determinate operazioni

Ogni utente puó effettuare delle prenotazioni, per sé o per una persona non registrata al sistema, quindi in ognuna vi sarà specificato:

· il viaggio per cui è valida

· l'utente che l'ha effettuata

· il passeggero che viaggerá (non necessariamente l'utente che ha prenotato il viaggio)

· il tipo di seduta prenotata, se in prima o seconda classe, ed eventualmente in numero di posto a sedere

· le indicazioni sulla classe di validitá e, se fissato, il posto a sedere

· il numero e il tipo di bagagli

· lo stato che indica se la prenotazione é valida, annullata o rimborsata

Al momento di un acquisto, un utente può specificare come passeggero un'altra persona della quale interessano solo le anagrafiche necessarie per l'intestazione della prenotazione e un email, questi non ha modo di prenotare un viaggio poiché non possiede dati di accesso. Se la prenotazione di un utente é intestata ad un passeggero di etá inferiore ai quattordici anni saranno applicate delle tariffe ridotte al momento dell'acquisto, calcolate con dei parametri specificati nel viaggio scelto. L'utente deve avere la possibilitá di scegliere anche la classe in cui viaggiare, se l'aereo del viaggio che vuole prenotare ne ha piú d'una, e quanti e di che tipo sono i bagagli che vuole portare con sè. Nel sistema dovranno essere quindi memorizzate informazioni su tre diverse tipologie di persone, gli utenti che hanno accesso alla prenotazione dei viaggi, i passeggeri, che sono gli intestatari di una o piú prenotazioni effettuate da un utente, e i dipendenti delle compagnie dei quali si gestiscono i viaggi. Dovrá essere possibile gestire due tipologie di viaggi, diretti e con scali, e per questi l'inserimento in un pacchetto di offerte con ulteriori riduzioni sul prezzo.

3 Progettazione Concettuale

3.1 Descrizione testuale delle classi

3.1.1 Persona

La classe Persona raccoglie le informazioni di ogni dipendente, utente e passeggero
Attributi

◇ *nome: string* - nome

◇ *cognome: string* - cognome

◇ *nascita: date* - data di nascita

◇ *sesso: enum'M','F'* - sesso

◇ *email: string* - email dell'utente

3.1.2 Passeggero

La classe Passeggero estende la superclasse Persona e memorizza i dati di accesso di di ogni persona che compare in una o piú prenotazioni come passeggero del viaggio

3.1.3 Utente

La classe Utente estende la superclasse Persona e memorizza i dati di accesso di ogni persona registrata nella base di dati
Attributi

◇ *password: string* - password criptata con sha1

3.1.4 Ospite

La classe Ospite deriva da Utente ma non la estende con ulteriori attributi

3.1.5 Amministratore

La classe Amministratore deriva da Utente ma non la estende con ulteriori attributi

3.1.6 Dipendente

La classe Dipendente tiene memoria delle informazioni sui dipendenti delle compagnie i cui viaggi sono gestiti dal sistema
Attributi

- ◇ *matricola: integer* - matricola del dipendente

3.1.7 Assistente

La classe Assistente è una sottoclasse di dipendenti, non la estende con ulteriori attributi

3.1.8 Comandante

La classe Comandante è una sottoclasse di dipendenti, non la estende con ulteriori attributi

3.1.9 Vicecomandante

La classe Vicecomandante è una sottoclasse di dipendenti, non la estende con ulteriori attributi

3.1.10 Aereo

La classe Aereo mantiene le informazioni di base sui velivoli impiegati nei voli
Attributi

- ◇ *matricola: string «PK»* - numero di telaio
- ◇ *marca: string* - marca del velivolo
- ◇ *modello: string* - modello
- ◇ *anno: year* - anno di costruzione del velivolo
- ◇ *postiPrima: integer* - numero di posti di classe business
- ◇ *postiSeconda: integer* - numero di posti in turistica

3.1.11 Luogo

La classe Luogo mantiene le informazioni sulla posizione di una precisa cittá
Attributi

- ◇ *nomecittá: string* - cittá
- ◇ *nazione: string* - nazione

3.1.12 Aeroporto

La classe Aeroporto mantiene le informazioni sugli aeroporti dai quali puó partire o arrivare un volo
Attributi

- ◇ *nome: string* - nome dell'aeroporto

3.1.13 Compagnia

La classe Compagnia mantiene le informazioni sulle compagnie che possiedono o eseguono almeno un volo
Attributi

- ◇ *nome: string* - nome della compagnia
- ◇ *numTel: string* - numero di telefono
- ◇ *email: string* - email della compagnia
- ◇ *nazione: string* - nazione nativa della compagnia

3.1.14 Bagaglio

La classe Bagaglio mantiene le informazioni sulle tariffe imposte dalle compagnie ai bagagli che viaggiano nei voli da loro eseguiti
Attributi

- ◇ *peso: integer* - peso massimo consentito

3.1.15 Tratta

La classe Tratta rappresenta il collegamento tra due luoghi
Attributi

- ◇ *numeroTratta: integer* - numero identificativo

3.1.16 Viaggio

La classe Viaggio mantiene le informazioni sui viaggi
Attributi

- ◇ *giorno: date* - giorno in cui sará effettuato il viaggio
- ◇ *stato: enum'effettuato','previsto','soppresso'* - stato del viaggio
- ◇ *prezzoPrima: integer* - prezzo di prima classe
- ◇ *prezzoSeconda: integer* - prezzo di seconda classe

3.1.17 ViaggioDiretto

La classe ViaggioDiretto deriva da Viaggio e non estende la superclasse

3.1.18 ViaggioConScali

La classe ViaggioConScali deriva da Viaggio e non estende la superclasse

3.1.19 Volo

La classe Volo descrive gli attributi che determinano un particolare volo
Attributi

- ◇ *numero: string «PK»* - numero di volo
- ◇ *oraP: time* - ora di partenza
- ◇ *oraA: time* - ora di arrivo

3.1.20 Prenotazione

La classe Prenotazione mantiene le informazioni sull'acquisto di un viaggio
Attributi

- ◇ *numeroBagagli: integer* - numero di bagagli
- ◇ *stato: enum'valida','annullata','rimborsata'* - stato della prenotazione

3.1.21 PrenotazionePrimaClasse

La classe PrenotazionePrimaClasse estende Prenotazioni e mantiene le informazioni sull'acquisto di un viaggio in prima classe

3.1.22 PrenotazioneSecondaClasse

La classe PrenotazioneSecondaClasse deriva da Prenotazioni ma non la estende con ulteriori attributi

3.1.23 PostoPrimaClasse

La classe PostoPrimaClasse mantiene le informazioni sui posti di prima classe di ogni aereo
Attributi

- ◇ *numero: string* - numero del posto

3.2 Descrizione testuale delle associazioni

3.2.1 Offerta-Viaggio: "è composto da"

- ◇ Molteplicità N:1
  Un'offerta comprende un viaggio, un viaggo può essere compreso in piú offerte
- ◇ Totalità: Parziale verso Offerta / Totale verso Viaggio
  Un viaggio può non comparire in alcuna offerta

3.2.2 ViaggioConScali-ViaggioDiretto: "è composto da"

- ◇ Molteplicità N:M
  Un viaggio con scali è composto da più viaggi diretti, un viaggio diretto compone uno o più viaggi con scali
- ◇ Totalità: Parziale verso viaggioDiretto / Totale verso ViaggioConScali
  Un viaggio diretto può non essere parte di alcun viaggio con scali

3.2.3 ViaggioDiretto-Volo: "è espressione di"

- ◇ Molteplicità 1:N
  Un viaggio diretto esprime un volo, un volo è espresso da uno o più viaggi diretti
- ◇ Totalità: Totale verso ViaggioDiretto / Totale verso Volo

3.2.4 ViaggioDiretto-Compagnia: “è effettuato da”

- ◇ Molteplicità 1:N
  Un viaggio diretto è effettuato da una compagnia, una compagnia effettua uno o più viaggi diretti
- ◇ Totalità: Parziale verso Viaggio / Totale verso Compagnia
  Una compagnia potrebbe non eseguire nessun Viaggio

3.2.5 Viaggio-Tratta: “percorre”

- ◇ Molteplicità N:1
  Un viaggio percorre una tratta, una tratta è percorsa da uno o più viaggi
- ◇ Totalità: Totale verso Viaggio / Totale verso Tratta

3.2.6 ViaggioDiretto-Aereo: “è effettuato con”

- ◇ Molteplicità 1:N
  Un viaggio diretto è effettuato con un aereo, un aereo effettua uno o più viaggi diretti
- ◇ Totalità: Totale verso Viaggio / Totale verso Aereo

3.2.7 ViaggioDiretto-Comandante: “pilota”

- ◇ Molteplicità N:1
  Un viaggio diretto è pilotato da un comandante, un comandante pilota uno o più viaggi diretti
- ◇ Totalità: Totale verso Viaggio / Totale verso Comandante

3.2.8 ViaggioDiretto-Vicecomandante: “copilota”

- ◇ Molteplicità N:1
  Un viaggio diretto è copilotato da un vicecomandante, un vicecomandante copilota uno o più viaggi diretti
- ◇ Totalità: Totale verso Viaggio / Totale verso Vicecomandante

3.2.9 ViaggioDiretto-Assistente: “assiste”

- ◇ Molteplicità N:M
  Un viaggio diretto è assistito da uno o più assistenti, un assistente assiste uno o più viaggi diretti
- ◇ Totalità: Totale verso Viaggio / Totale verso Assistente

3.2.10 Viaggio-Amministratore: “inserisce”

- ◇ Molteplicità N:1
  Un viaggio è inserito da un amministratore, un amministratore inserisce uno o più viaggi
- ◇ Totalità: Parziale verso Viaggio / Totale verso Amministratore
  Un amministratore può non inserire alcun viaggio

3.2.11 Tratta-Aeroporto: “parte da”

- ◇ Molteplicità N:1
  Una tratta parte da un aeroporto, da un aeroporto partono una o più tratte
- ◇ Totalità: Totale verso Tratta / Totale verso Aeroporto

3.2.12 Tratta-Aeroporto: "arriva a"

- ◇ Molteplicità N:1
  Una tratta arriva in un aeroporto, in un aeroporto arrivano una o più tratte
- ◇ Totalità: Totale verso Tratta / Totale verso Aeroporto

3.2.13 Tratta-Volo: "percorre"

- ◇ Molteplicità N:1
  Una tratta è percorsa da uno o più voli, un volo percorre una tratta
- ◇ Totalità: Totale verso Tratta / Totale verso Volo

3.2.14 Compagnia-Aereo: "appartiene a"

- ◇ Molteplicità 1:N
  Una compagnia possiede uno o più aerei, un aereo appartiene ad una compagnia
- ◇ Totalità: Totale verso Compagnia / Totale verso Aereo

3.2.15 Compagnia-Dipendente: "lavora per"

- ◇ Molteplicità 1:N
  Per una compagnia lavorano uno o più dipendenti, un dipendente lavora per una compagnia
- ◇ Totalità: Totale verso Compagnia / Totale verso Dipendente

3.2.16 Aeroporto-Luogo: "ha sede"

- ◇ Molteplicità N:1
  Un aeroporto ha sede in un luogo, in un luogo hanno sede uno o più aeroporti
- ◇ Totalità: Totale verso Aeroporto / Totalo verso Luogo

3.2.17 Aereo-PostoPrimaClasse: "possiede"

- ◇ Molteplicità 1:N
  Un aereo possiede uno o più PostiPrimaClasse, un PostoPrimaClasse è relativo ad un solo aereo
- ◇ Totalità: Totale verso Aereo / Parziale verso PostiPrimaClasse
  Un aereo può non possedere posti di prima classe

3.2.18 PostoPrimaClasse-PrenotazionePrimaClasse: "è occupato da"

- ◇ Molteplicità 1:N
  Un posto di prima classe è occupato da una o o più prenotazioni di prima classe, una prenotazione di prima classe occupa un posto i prima classe
- ◇ Totalità: Totale verso Posto / Totale verso Prenotazione

3.2.19 Prenotazione-Utente: "effettua"

- ◇ Molteplicità N:1
  Una prenotazione è effettuata da un utente, un utente effettua una o più prenotazioni
- ◇ Totalità: Parziale verso Prenotazione / Totale verso Utente
  Un utente può non effettuare alcuna prenotazione

3.3 Descrizione delle gerarchie

Lo schema concettuale prevede tre gerarchie distinte:

La prima gerarchia, rappresentata in Figura 1, modella i tipi di persone che vengono gestite dalla base di dati. La classe **Persona** si estende in

- **Dipendente** : persona che lavora per una compagnia che propone i viaggi in vendita, classe a sua volta estesa da altre tre classi che specificano il tipo di dipendente modellato:
  - **Comandante**
  - **Vice-Comandante**
  - **Assistente**
- **Utente** : persona con dei dati di accesso che può prenotare un viaggio per sè o per altri, classe a sua volta estesa da due classi che specificano il tipo di utente, e in particolare i suoi privilegi:
  - **Ospite**
  - **Amministratore**
- **Passeggero** : persona priva di dati di accesso all'applicazione web che tuttavia risulta intestataria di almeno una prenotazione per un viaggio, viene sottoclassata dal tipo del passeggero
  - **Adulto**
  - **Bambino**

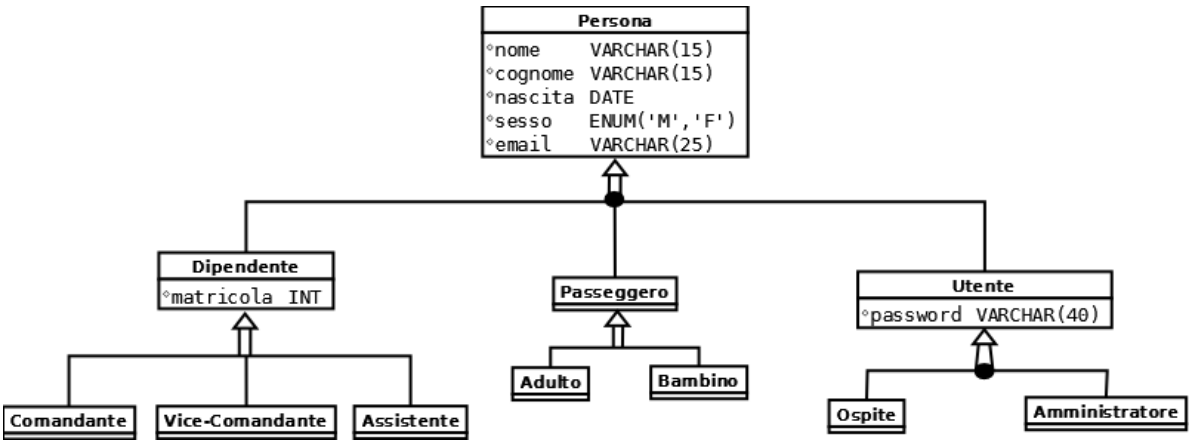

Figura 1: Gerarchia della classe Persona

La seconda gerarchia definisce il tipo viaggio, se diretto o non diretto, derivando dalla classe base **Viaggio** due sottoclassi:

- **Viaggio Diretto**
- **Viaggio Non Diretto**

La gerarchia rappresenta una partizione poichè non esistono altri tipi di viaggi ed è evidente che gli insiemi siano disgiunti, ne è mostrata una rappresentazione in Figura 2.

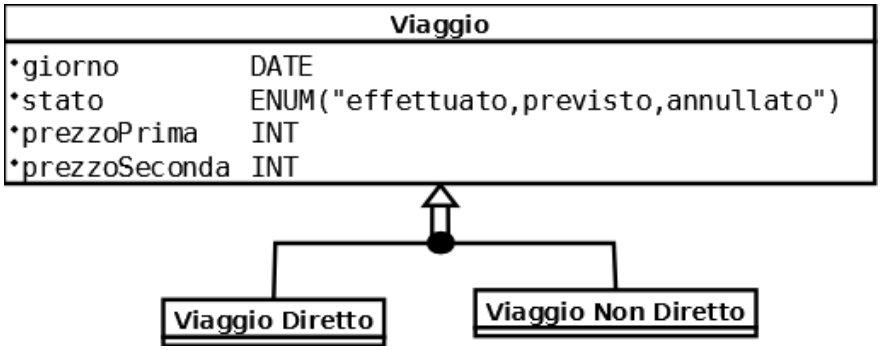

Figura 2: Gerarchia della classe Viaggio

L'ultima gerarchia definisce il tipo di **Prenotazione** derivandone due sottoclassi:

- **Prima Classe**
- **Seconda Classe**

Una prenotazione di prima classe tiene traccia del posto prenotato dall'utente, mentre in seconda classe i posti non sono numerati.

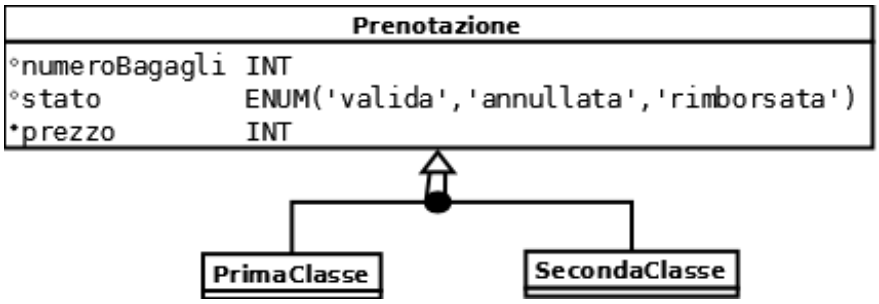

Figura 3: Gerarchia della classe Prenotazione

3.4 Schema concettuale in forma grafica

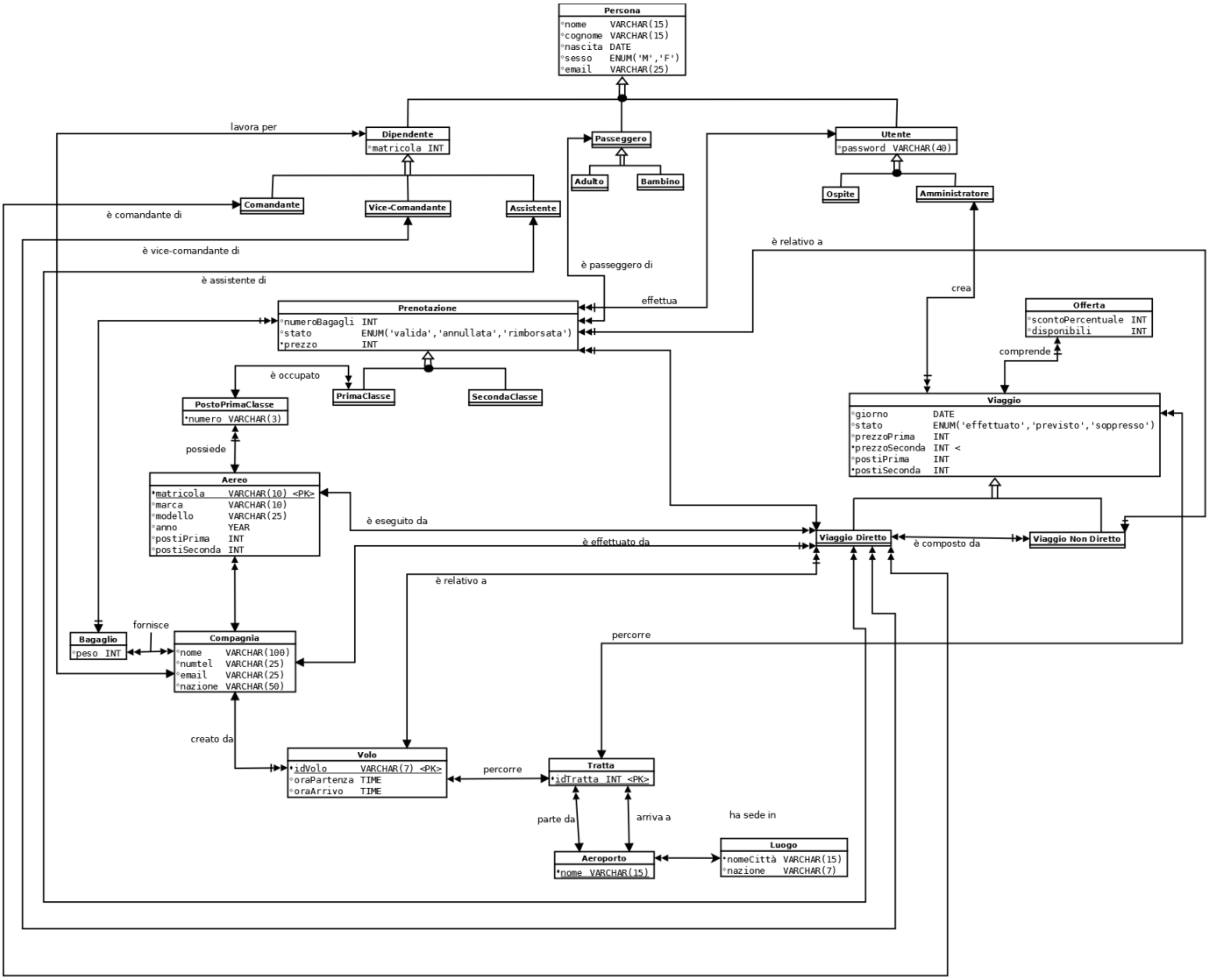

Figura 4: Schema concettuale

4 Progettazione Logica

4.1 Descrizione testuale dello schema relazionale

- **Aerei** (matricola: *varchar*, marca: *varchar*, modello: *varchar*, anno: *year*, postiPrima: *integer*, postiSeconda: *integer*, idCompagnia*: *integer*)
  - PK (matricola)
  - idCompagnia FK (Compagnie) not null
- **Aeroporti** (idAeroporto: *integer*, nome: *varchar*, idLuogo*: *integer*)
  - PK (idAeroporto)
  - idLuogo FK (Luoghi) not null
- **Anagrafiche** (idAnag: *integer*, nome: *varchar*, cognome: *varchar*, nascita: *date*, sesso: *enum{'M','F'}*, email: *varchar*, tipo: *enum{'adulto','bambino'}*)
  - PK (idAnag)
  - UNIQUE (email)
- **Assistenze** (idViaggio*: *integer*, matricola* *integer*)
  - PK (idViaggio, matricola)
  - idViaggio FK (ViaggiDiretti)
  - matricola FK (Dipendenti)
- **Bagagli** (idBagaglio: *integer*, peso: *integer*)
  - PK (idBagaglio)
- **Compagnie** (idCompagnia: *integer*, nome: *varchar*, numTel: *varchar*, email: *varchar*, nazione: *varchar*)
  - PK (idCompagnia)
- **Dipendenti** (idAnag*: *integer*, matricola: *integer*, grado: *enum{'assistente','comandante','vice'}*, idCompagnia: *integer*)
  - PK (idAnag)
  - idAnag FK (Anagrafiche)
- **Luoghi** (idLuogo: *integer*, nomecitta: *varchar*, nazione: *varchar*)
  - PK (idLuogo)
- **Offerte** (idViaggio*: *integer*, scontoperc: *integer*, disponibili: *integer*)
  - PK (idViaggio)
  - idViaggio FK (Viaggi)
- **PostiPrimaClasse** (numero: *varchar*, aereo*: *varchar*)
  - PK (numero, aereo)
  - aereo FK (Aerei) not null

- **Prenotazioni** (idPrenotazione: *integer*, idViaggio*: *integer*, diretto: *bool*, idViaggioConScali*: *integer*, acquirente*: *integer*, passeggero*: *integer*, numeroBagagli: *integer*, idBagaglio*: *integer*, type: *enum{'prima','seconda'}*, stato: *enum{'valido','annullato','rimborsato'}*, prezzoPrenotazione: *integer*, posto*: *varchar*)
  - PK (idPrenotazione)
  - idViaggio FK (ViaggiDiretti) not null
  - idViaggioConScali FK (ViaggiConScali)
  - acquirente FK (Utenti) not null
  - passeggero FK (Anagrafiche)
  - idBagaglio FK (Bagagli)
  - posto FK (PostiPrimaClasse)
- **Scali** (idViaggioConScali*: *integer*, idViaggioDiretto*: *integer*, ordine: *integer*)
  - PK (idViaggioConScali, idViaggioDiretto)
  - idViaggioConScali FK (ViaggiConScali)
  - idViaggioDiretto FK (ViaggiDiretti)
- **TariffeBagagli** (idBagaglio*: *integer*, idCompagnia*: *integer*, prezzo: *integer*)
  - PK (idBagaglio, idCompagnia)
  - idBagaglio FK (Bagagli)
  - idCompagnia FK (Compagnie)
- **Tratte** (idTratta: *integer*, da*: *integer*, a*: *integer*)
  - PK (idTratta)
  - da FK (Aeroporti)
  - a FK (Aeroporti)
- **Utenti** (idAnag*: *integer*, password: *varchar*, type: *enum{'Guest','Admin'}*)
  - PK (idAnag)
  - idAnag FK (Anagrafiche)
- **Viaggi** (idViaggio: *integer*, giorno: *date*, stato: *enum{'effettuato','previsto','soppresso'}*, prezzoPrima: *integer*, prezzoSeconda: *integer*, postiPrima: *integer*, postiSeconda: *integer*, idTratta*: *integer*, , inseritoDa*: *integer*)
  - PK (idViaggio)
  - idTratta FK (Tratte)
  - inseritoDa FK (Anagrafiche) not null
- **ViaggiConScali** (idViaggioConScali*: *integer*)
  - PK (idViaggioConScali)
  - idViaggioConScali FK (ViaggiConScali)
- **ViaggiDiretti** (idViaggioDiretto*: *integer*, idVolo*: *varchar*, aereo*: *varchar*, comandante*: *integer*, vice*: *integer*, ridottoPerc: *integer*, idCompagniaEsec*: *integer*)
  - PK (idViaggioDiretto)

- idViaggioDiretto FK (ViaggiDiretti)
- idVolo FK (Voli) not null
- aereo FK (Aerei)
- comandante FK (Dipendenti)
- vice FK (Dipendenti)
- idCompagniaEsec FK (Compagnie) not null

- **Voli** (idVolo: *integer*, oraP: *time*, oraA: *time*, idTratta*: *integer*, idCompagnia*: *integer*)
  - PK (idVolo)
  - idTratta FK (Tratte)
  - idCompagnia FK (Compagnie) not null

4.2 Considerazioni sulle scelte progettuali

4.2.1 Trasformazione delle gerarchie

Per la prima gerarchia si è scelto di risolvere attraverso tabella unica il livello inferiore poichè in tutti i casi le sottoclassi non presentano membri propri, un campo dati di tipo *enum* funge quindi da discriminante. Salendo nella gerarchia per lo stesso motivo abbiamo scelto di inglobare la classe Passeggero nella classe base Persona, rinominandola in Anagrafiche. Infine si è scelto di operare con partizionamento verticale la derivazione coinvolgente - Anagrafiche , Dipendente , Utente - per due motivi:

- gli attributi comuni alle due relazioni sono cinque, la ridondanza sarebbe stata quindi evidente
- la classe base, nella forma di Passeggero, ha una freccia entrante da Prenotazioni, vincolo che sarebbe stato difficile mantenere attraverso partizionamento orizzontale

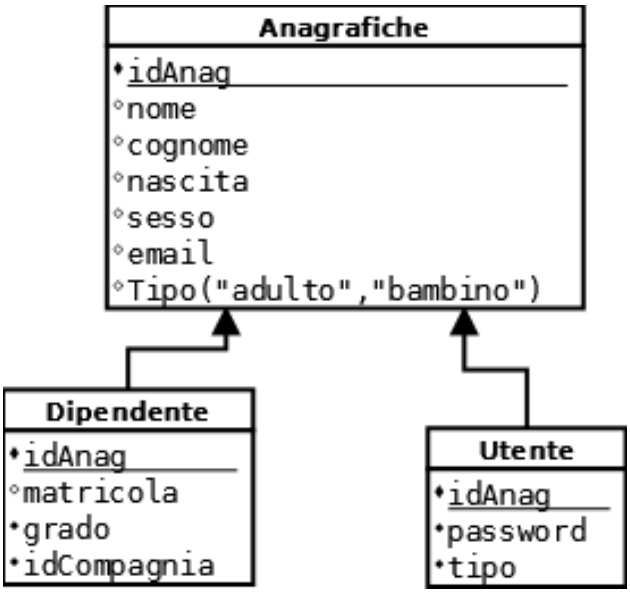

Figura 5: Trasformazione gerarchia della classe Persona

Anche per la seconda gerarchia si è scelto il partizionamento verticale: come prima la motivazione principale risiede nel fatto che la classe base Viaggio ha delle frecce entranti, da Prenotazioni e da Offerte. La trasformazione delle due gerarchie nel modello relazionale è mostrata di seguito in Figura 5.

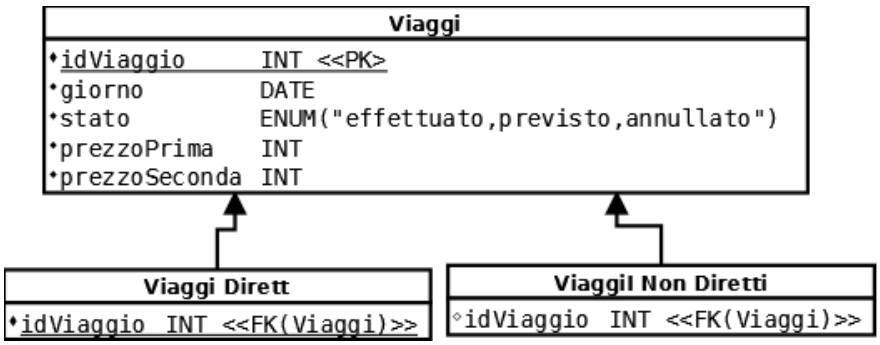

Figura 6: Trasformazione gerarchia della classe Viaggio

Infine per l'ultima gerarchia, su Prenotazione, si è scelto il metodo di tabella unica aggiungendo alla relazione un attributo discriminatore *type* e l'attributo *posto* che può essere *null*.
Questa è visibile di seguito in Figura 7

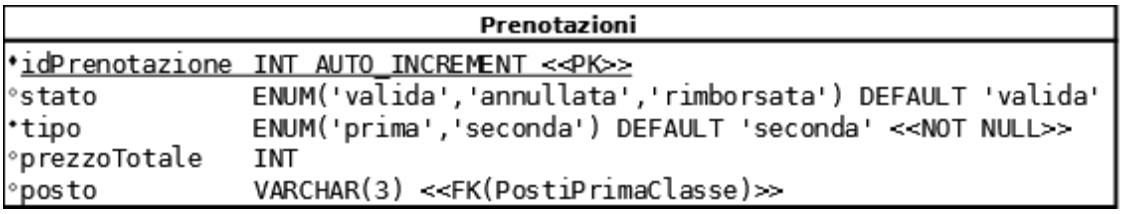

Figura 7: Trasformazione gerarchia della classe Prenotazione

4.2.2 Vincoli semantici

Nella progettazione della base di dati abbiamo introdotto dei vincoli che non possono essere catturati dallo schema:

- Tutti i viaggi, diretti e non, si svolgono nell'arco di uno stesso giorno. Non è per questo motivo stato previsto un attributo che specifichi il giorno di arrivo, essendo questo coindidente con il giorno di partenza
- Ogni compagnie deve offrire ai clienti il servizio di bagaglio in stiva di peso pari a 20Kg, in modo da permettere sempre una prenotazione con bagaglio anche in viaggi con scali
- Un viaggio con scali può avere al più 3 scali, e quindi essere composto da 4 viaggi diretti

4.3 Schema logico in forma grafica

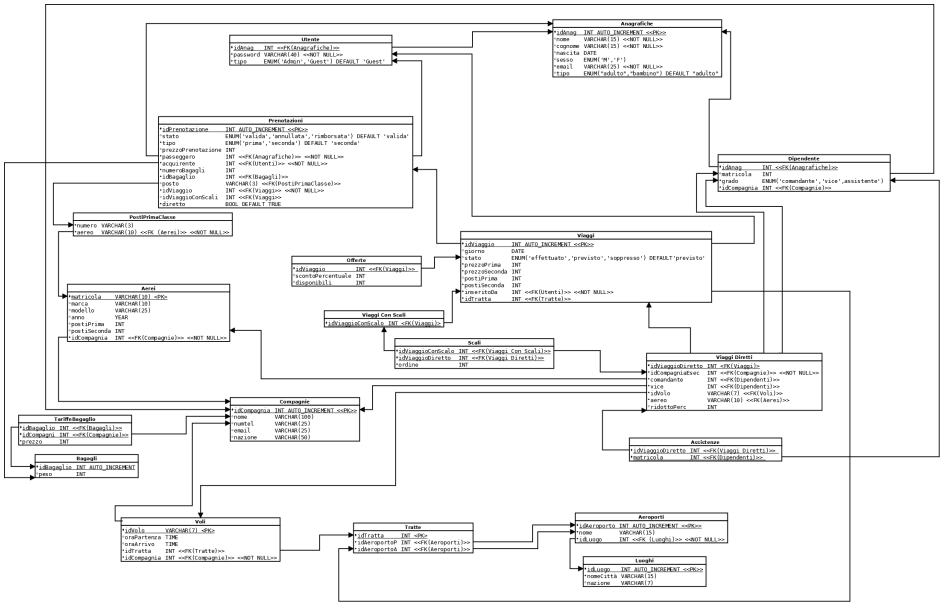

Figura 8: Schema logico relazionale

5 Definizione dello schema logico

```
/* Crea la tabella Anagrafiche */

CREATE TABLE Anagrafiche (
        idAnag          INT AUTO_INCREMENT PRIMARY KEY,
        nome            VARCHAR(15),
        cognome         VARCHAR(15),
        nascita         DATE,
        sesso           ENUM('M','F') DEFAULT "M",
        email           VARCHAR(25),
        tipo            ENUM('adulto','bambino') DEFAULT "adulto",
        UNIQUE (email)
) ENGINE=InnoDB;

/* Crea la tabella Utenti */

CREATE TABLE Utenti (
        idAnag          INT PRIMARY KEY,
        password        VARCHAR(40),
        type            ENUM('Guest','Admin') DEFAULT "Guest",
        FOREIGN KEY (idAnag)    REFERENCES Anagrafiche (idAnag)
                                ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB;
```

```
/* Crea la tabella Dipendenti */

CREATE TABLE Dipendenti (
            idAnag                  INT PRIMARY KEY,
            matricola     INT(10),
            grado                   ENUM('assistente','comandante','vice'),
            idCompagnia   INT,
            UNIQUE (matricola),
            FOREIGN KEY (idAnag) REFERENCES Anagrafiche (idAnag)
                                    ON UPDATE CASCADE,
            FOREIGN KEY (idCompagnia)     REFERENCES Compagnie (idCompagnia)
                                          ON UPDATE CASCADE
) ENGINE=InnoDB;

/* Crea la tabella Aerei */

CREATE TABLE Aerei (
        matricola       VARCHAR(10) PRIMARY KEY,
        marca           VARCHAR(10),
        modello         VARCHAR(25),
        anno                    YEAR,
        postiPrima      INT(3),
        postiSeconda    INT(3),
        idCompagnia     INT NOT NULL,
        FOREIGN KEY (idCompagnia) REFERENCES Compagnie (idCompagnia)
                                  ON UPDATE CASCADE
) ENGINE=InnoDB;

/* Crea la tabella Luoghi */

CREATE TABLE Luoghi (
        idLuogo                 INT AUTO_INCREMENT PRIMARY KEY,
        nomecitta       VARCHAR(40),
        nazione         VARCHAR(30)
) ENGINE=InnoDB;

/* Crea la tabella Aeroporti */

CREATE TABLE Aeroporti (
        idAeroporto     INT AUTO_INCREMENT PRIMARY KEY,
        nome          VARCHAR(40),
        idLuogo         INT NOT NULL,
        FOREIGN KEY (idLuogo)   REFERENCES Luoghi (idLuogo)
                                ON UPDATE CASCADE
) ENGINE=InnoDB;

/* Crea la tabella Compagnie */

CREATE TABLE Compagnie (
        idCompagnia     INT AUTO_INCREMENT PRIMARY KEY,
        nome            VARCHAR(30),
        numTel          VARCHAR(25),
```

```
        email           VARCHAR(50),
        nazione         VARCHAR(50)
)ENGINE=InnoDB;

/* Crea la tabella Bagagli */

CREATE TABLE Bagagli(
        idBagaglio      INT AUTO_INCREMENT PRIMARY KEY,
        peso            INT(2)
)ENGINE=InnoDB;

/* Crea la tabella TariffeBagagli */

CREATE TABLE TariffeBagagli(
        idBagaglio      INT,
        idCompagnia     INT,
        prezzo          INT,
        PRIMARY KEY(idBagaglio,idCompagnia),
        FOREIGN KEY(idBagaglio) REFERENCES Bagagli(idBagaglio)
                                ON UPDATE CASCADE,
        FOREIGN KEY(idCompagnia) REFERENCES Compagnie(idCompagnia)
                                ON UPDATE CASCADE
)ENGINE=InnoDB;

/* Crea la tabella Tratte */

CREATE TABLE Tratte (
        idTratta        INT AUTO_INCREMENT PRIMARY KEY,
        da                      INT,
        a                       INT,
        FOREIGN KEY (a) REFERENCES Aeroporti (idAeroporto)
                        ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY (da) REFERENCES Aeroporti (idAeroporto)
                        ON DELETE CASCADE  ON UPDATE CASCADE
)ENGINE=InnoDB;

/* Crea la tabella Voli */

CREATE TABLE Voli (
        idVolo          VARCHAR(7) PRIMARY KEY,
        oraP            TIME,
        oraA            TIME,
        idTratta        INT,
        idCompagnia     INT NOT NULL,
        FOREIGN KEY (idCompagnia)REFERENCES Compagnie (idCompagnia)
                                ON DELETE CASCADE ON UPDATE CASCADE,
        FOREIGN KEY(idTratta)   REFERENCES Tratte (idTratta)
                                ON UPDATE CASCADE
) ENGINE=InnoDB;

/* Crea la tabella Viaggi */
```

```
CREATE TABLE Viaggi (
        idViaggio       INT AUTO_INCREMENT PRIMARY KEY,
        giorno          DATE,
        stato           ENUM('effettuato','previsto','soppresso') DEFAULT 'previsto',
        prezzoPrima INT,
        prezzoSeconda INT,
        postiPrima INT,
        postiSeconda INT,
        idTratta        INT,
        inseritoDa   INT NOT NULL,
        FOREIGN KEY (inseritoDa) REFERENCES Utenti (idAnag)
                                ON UPDATE CASCADE,
        FOREIGN KEY (idTratta) REFERENCES Tratte (idTratta)
                                ON UPDATE CASCADE
) ENGINE=InnoDB;


/* Crea la tabella ViaggiDiretti */

CREATE TABLE ViaggiDiretti (
        idViaggioDiretto        INT PRIMARY KEY,
        idVolo  VARCHAR(7) NOT NULL,
        aereo   VARCHAR(10),
        comandante      INT(10),
        vice            INT(10),
        ridottoPerc INT,
        idCompagniaEsec INT NOT NULL,
        FOREIGN KEY (idViaggioDiretto) REFERENCES Viaggi (idViaggio)
                                ON UPDATE CASCADE,
        FOREIGN KEY (aereo) REFERENCES Aerei (matricola)
                                ON UPDATE CASCADE,
        FOREIGN KEY (comandante) REFERENCES Dipendenti (matricola)
                                ON UPDATE CASCADE,
        FOREIGN KEY (vice) REFERENCES Dipendenti (matricola)
                                ON UPDATE CASCADE,
        FOREIGN KEY (idCompagniaEsec) REFERENCES Compagnie (idCompagnia)
                                ON UPDATE CASCADE,
        FOREIGN KEY (idVolo) REFERENCES Voli (idVolo)
                                ON UPDATE CASCADE
) ENGINE=InnoDB;

/* Crea la tabella delle Offerte */

CREATE TABLE Offerte (
       idViaggio        INT PRIMARY KEY,
       scontoperc       INT,
       disponibili      INT,
       FOREIGN KEY (idViaggio) REFERENCES Viaggi (idViaggio)
                                ON UPDATE CASCADE
) ENGINE=InnoDB;
```

```
/* Crea la tabella delle Assistenze */

CREATE TABLE Assistenze (
        idViaggio           INT,
        matricola           INT(10),
       PRIMARY KEY (idViaggio,matricola),
       FOREIGN KEY (idViaggio) REFERENCES Viaggi (idViaggio)
                                ON UPDATE CASCADE,
       FOREIGN KEY (matricola) REFERENCES Dipendenti (matricola)
                                ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB;


/* Crea la tabella Posti */

CREATE TABLE PostiPrimaClasse(
         numero             VARCHAR(3),
         aereo              VARCHAR(10) NOT NULL,
         PRIMARY KEY(numero, aereo),
         FOREIGN KEY (aereo) REFERENCES Aerei (matricola)
                                ON DELETE CASCADE ON UPDATE CASCADE

)ENGINE=InnoDB;

/* Crea la tabella Prenotazioni */

CREATE TABLE Prenotazioni (
         idPrenotazione  INT(100) AUTO_INCREMENT PRIMARY KEY,
         idViaggio       INT NOT NULL,
         diretto BOOL DEFAULT TRUE,
         idViaggioConScali INT DEFAULT NULL,
         acquirente      INT NOT NULL,
         passeggero      INT,
         numeroBagagli   INT(3),
         idBagaglio      INT DEFAULT NULL,
         type            ENUM('prima','seconda') DEFAULT 'seconda',
         stato           ENUM('valido','annullato','rimborsato') DEFAULT 'valido',
         prezzoPrenotazione INT,
         posto           VARCHAR(3),
         FOREIGN KEY (posto) REFERENCES PostiPrimaClasse (numero)
                                ON UPDATE CASCADE,
         FOREIGN KEY (idBagaglio) REFERENCES Bagagli (idBagaglio)
                                ON UPDATE CASCADE,
         FOREIGN KEY (idViaggio) REFERENCES Viaggi (idViaggio)
                                ON UPDATE CASCADE,
         FOREIGN KEY (idViaggioConScali) REFERENCES ViaggiConScali (idViaggioConScali)
                                ON UPDATE CASCADE,
         FOREIGN KEY (acquirente)REFERENCES Utenti (idAnag)
                                ON DELETE CASCADE ON UPDATE CASCADE,
         FOREIGN KEY (passeggero)REFERENCES Anagrafiche (idAnag)
                                ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB;
```

```
/* Crea la tabella ViaggiConScali */

CREATE TABLE ViaggiConScali(
        idViaggioConScali       INT PRIMARY KEY,
        FOREIGN KEY (idViaggioConScali) REFERENCES Viaggi (idViaggio)
                                                        ON UPDATE CASCADE
)ENGINE=InnoDB;

/* Crea la tabella Scali*/

CREATE TABLE Scali(
        idViaggioConScali       INT,
        idViaggioDiretto        INT,
        ordine                  INT,
        PRIMARY KEY(idViaggioConScali, idViaggioDiretto),
        FOREIGN KEY (idViaggioConScali) REFERENCES ViaggiConScali (idViaggioConScali)
                                                        ON UPDATE CASCADE,
        FOREIGN KEY (idViaggioDiretto) REFERENCES ViaggiDiretti (idViaggioDiretto)
                                                        ON UPDATE CASCADE

)ENGINE=InnoDB;
```

6 Query e Viste

6.1 Query significative

- Per ogni compagnia il numero di viaggi effettuati e l'incasso totale dei viaggi effettuati tale da essere maggiore della media degli incassi

```
1 SELECT c.nome,COUNT(*) AS NumeroViaggiEffettuati,SUM(p.prezzoPrenotazione)
      AS Incasso
2 FROM Compagnie c JOIN ViaggiDiretti vd ON(c.idCompagnia=vd.idCompagniaEsec
      ) JOIN Viaggi v ON(vd.idViaggioDiretto=v.idViaggio) JOIN Prenotazioni
      p ON(v.idViaggio=p.idViaggio)
3 WHERE p.stato='valido' AND v.stato='effettuato'
4 GROUP BY c.idCompagnia
5 HAVING SUM(p.prezzoPrenotazione)>AVG(p.prezzoPrenotazione)
```

- Per ogni viaggio previsto, il numero di volo, da dove parte a dove arriva, il numero dei posti di prima classe ancora disponibili

```
1 SELECT vo.idVolo AS NumeroVolo,l1.nomeCitta AS Partenza,l2.nomeCitta AS
      Arrivo,pps.numero,a.marca AS MarcaAereo,a.modello AS ModelloAereo
2 FROM postiPrimaClasse pps JOIN ViaggiDiretti vd ON (pps.aereo=vd.aereo)
      JOIN Aerei a ON(a.matricola=pps.aereo) JOIN Viaggi v ON(v.idViaggio=vd
      .idViaggioDiretto) JOIN Voli vo ON(vo.idVolo=vd.idVolo) JOIN Tratte t
      ON (v.idTratta=t.idTratta) JOIN Luoghi l1 ON (t.da=l1.idLuogo) JOIN
      Luoghi l2 ON (t.a=l2.idLuogo)
3 WHERE pps.numero NOT IN (SELECT p.posto FROM Prenotazioni p WHERE p.
      idViaggio=vd.idViaggioDiretto AND p.type='prima' ) AND v.stato='
      previsto'
```

6.2 Viste

6.2.1 viewDipendenti

```
1 CREATE VIEW viewComandanti AS
2 SELECT d.matricola, a.nome, a.cognome, a.sesso, a.nascita, c.nome AS Compagnia
3 FROM Dipendenti d NATURAL JOIN Anagrafiche a JOIN Compagnie c ON (d.
      idCompagnia=c.idCompagnia)
4 WHERE d.grado='comandante';
```

6.2.2 viewTratte

```
1 CREATE VIEW viewTratte AS
2 SELECT t.idTratta AS Tratta, a1.nome AS Partenza, a2.nome AS Arrivo
3 FROM Tratte t JOIN Aeroporti a1 ON (t.da=a1.idAeroporto) JOIN Aeroporti a2 ON
      (t.a=a2.idAeroporto);
```

6.2.3 viewViaggiDiretti

```
1 CREATE VIEW viewViaggiDiretti AS
2 SELECT v.idViaggio, v.giorno, vt.Partenza AS da, vt.Arrivo AS a, l1.nomecitta
      AS luogoP, l2.nomecitta AS luogoA, vo.oraP, vo.oraA, TIMEDIFF(vo.oraA, vo.
      oraP) AS durata, v.stato, v.prezzoPrima, v.prezzoSeconda,
3                   v.postiPrima, v.postiSeconda, c.nome AS compagnia, v.
                        inseritoDa AS admin
4 FROM Viaggi v JOIN ViaggiDiretti vd ON (v.idViaggio=vd.idViaggioDiretto) JOIN
      viewTratte vt ON (v.idTratta=vt.Tratta)
5                   JOIN Voli vo ON vd.idVolo=vo.idVolo JOIN Compagnie c ON (vd.
                        idCompagniaEsec=c.idCompagnia) JOIN Tratte t ON
6                   vt.Tratta=t.idTratta JOIN Luoghi l1 ON (t.da=l1.idLuogo) JOIN
                        Luoghi l2 ON (t.a=l2.idLuogo);
```

6.2.4 viewViaggiConScali

```
1 CREATE VIEW viewViaggiConScali AS
2 SELECT v.idViaggio, v.giorno, vt.Partenza AS da, vt.Arrivo AS a, l1.nomecitta
      AS luogoP, l2.nomecitta AS luogoA, v.stato, v.prezzoPrima, v.prezzoSeconda
      , v.postiPrima,
3                   v.postiSeconda,v.inseritoDa AS admin
4 FROM Viaggi v JOIN ViaggiConScali vcs ON (v.idViaggio=vcs.idViaggioConScali)
      JOIN viewTratte vt ON (v.idTratta=vt.Tratta) JOIN Tratte t ON (vt.Tratta=t
      .idTratta)
5         JOIN Luoghi l1 ON (t.da=l1.idLuogo) JOIN Luoghi l2 ON (t.a=l2.idLuogo)
              ;
```

7 Stored Procedure e Triggers

Di seguito le principali stored procedures usate

7.1 Procedure

7.1.1 Gestione Utenti

```
1  CREATE PROCEDURE inserisciUtente (IN nome VARCHAR(15),IN cognome VARCHAR(15),
       IN nascita DATE,IN sesso VARCHAR(1),IN mail VARCHAR(25), IN psw VARCHAR
       (40),IN tipo VARCHAR(5),OUT inserito BOOL)
2  BEGIN
3          DECLARE Test INT;
4          DECLARE ultimoId INT;
5          SELECT COUNT(*) INTO Test FROM Anagrafiche a WHERE a.email=mail;
6          IF Test=0 THEN
7                  INSERT INTO Anagrafiche (nome, cognome, nascita, sesso, email)
                       VALUES (nome,cognome,nascita,sesso,mail);
8                  SELECT MAX(idAnag) INTO ultimoId FROM Anagrafiche;
9                  INSERT INTO Utenti VALUES (ultimoId,psw,tipo);
10                 SET inserito=1;
11         ELSE
12                 SET inserito=0;
13         END IF;
14 END;
```

```
1  CREATE PROCEDURE eliminaUtente (IN idAnagraf INT)
2  BEGIN
3          DELETE FROM Prenotazioni WHERE acquirente=idAnagraf OR passeggero=
               idAnagraf;
4          DELETE FROM      Utenti WHERE idAnag=idAnagraf;
5          DELETE FROM Anagrafiche WHERE idAnag=idAnagraf;
6  END;
```

7.1.2 inserisciDipendente

```
1  CREATE PROCEDURE inserisciDipendente  (IN nome VARCHAR(15),IN cognome VARCHAR
       (15),IN nascita DATE,IN sesso VARCHAR(1),IN mail VARCHAR(25), IN matricola
        INT,IN grado VARCHAR(10),IN compagnia INT,OUT inserito BOOL)
2  BEGIN
3          DECLARE Test INT;
4          DECLARE ultimoId INT;
5          SELECT COUNT(*) INTO Test FROM Dipendenti d WHERE d.email=mail;
6          IF Test=0 THEN
7                  INSERT INTO Anagrafiche VALUES (nome,cognome,nascita,sesso,
                       mail);
8                  SELECT COUNT(*) INTO ultimoId FROM Anagrafiche;
9                  INSERT INTO Dipendenti VALUES (ultimoId,matricola,grado,
                       compagnia);
10                 SET inserito=1;
11         ELSE
12                 SET inserito=0;
13         END IF;
14 END;
```

7.1.3 inserisciViaggio

```
1  CREATE PROCEDURE inserisciViaggio(IN Volo VARCHAR(7),IN giorno DATE,IN
       prezzoPrima INT,IN prezzoSeconda INT, IN idTratta INT,IN inseritoDa INT,
```

```
    IN compagnia INT,IN aereo VARCHAR(10),IN comandante INT(10), IN vice INT
    (10), IN ridottoPerc INT)
2 BEGIN
3         DECLARE ultimoId INT;
4         INSERT INTO Viaggi (giorno, prezzoPrima, prezzoSeconda, postiPrima,
              postiSeconda, idTratta, inseritoDa) VALUES (giorno, prezzoPrima,
              prezzoSeconda, getPosti(0,aereo), getPosti(1,aereo), idTratta,
              inseritoDa);
5         SELECT MAX(IdViaggio) INTO ultimoId FROM Viaggi;
6         INSERT INTO ViaggiDiretti VALUES (ultimoId, Volo, aereo, comandante,
              vice, ridottoPerc, compagnia);
7 END;
```

7.1.4 inserisciViaggioConScali

```
1 CREATE PROCEDURE inserisciViaggioConScali (IN giorno DATE,IN prezzoPrima INT,
    IN prezzoSeconda INT,IN postiPrima INT, IN postiSeconda INT,IN idTratta
    INT,IN inseritoDa INT,OUT VIAGGIO INT)
2 BEGIN
3         INSERT INTO Viaggi (giorno, prezzoPrima, prezzoSeconda, postiPrima,
              postiSeconda, idTratta, inseritoDa) VALUES (giorno, prezzoPrima,
              prezzoSeconda, postiPrima, postiSeconda, idTratta, inseritoDa);
4         SELECT MAX(IdViaggio) INTO VIAGGIO FROM Viaggi;
5         INSERT INTO ViaggiConScali VALUES (VIAGGIO);
6 END;
```

7.1.5 inserisciViaggioConScali

```
1 CREATE PROCEDURE inserisciViaggioConScali (IN giorno DATE,IN prezzoPrima INT,
    IN prezzoSeconda INT,IN postiPrima INT, IN postiSeconda INT,IN idTratta
    INT,IN inseritoDa INT,OUT VIAGGIO INT)
2 BEGIN
3         INSERT INTO Viaggi (giorno, prezzoPrima, prezzoSeconda, postiPrima,
              postiSeconda, idTratta, inseritoDa) VALUES (giorno, prezzoPrima,
              prezzoSeconda, postiPrima, postiSeconda, idTratta, inseritoDa);
4         SELECT MAX(IdViaggio) INTO VIAGGIO FROM Viaggi;
5         INSERT INTO ViaggiConScali VALUES (VIAGGIO);
6 END;
```

7.1.6 scalaPosti

```
1 CREATE PROCEDURE ScalaPosti ( IN nbiglietti INT, IN prima INT,IN idv INT,IN
      scali INT)
2 BEGIN
3 DECLARE idvs INT;
4 DECLARE idvd INT;
5 DECLARE Cur CURSOR FOR SELECT idViaggioDiretto,idViaggioConScali FROM Scali;
6 DECLARE EXIT HANDLER FOR NOT FOUND
7 BEGIN END;
8
9 IF scali=0 THEN
```

```
10          IF prima=1 THEN
11                      UPDATE Viaggi SET postiPrima=postiPrima-nbiglietti
                            WHERE idViaggio=idv;
12                  ELSE
13                      UPDATE Viaggi SET postiSeconda=postiSeconda-nbiglietti
                            WHERE idViaggio=idv;
14          END IF;
15  END IF;
16  IF scali=1 THEN
17          IF prima=1 THEN
18          UPDATE Viaggi SET postiPrima=postiPrima-nbiglietti WHERE idViaggio=idv
                ;
19                      OPEN Cur;
20                          LOOP
21                              FETCH Cur INTO idvd,idvs;
22                              IF idvs=idv THEN
23                                  UPDATE Viaggi SET postiPrima=
                                        postiPrima-nbiglietti
                                        WHERE idViaggio=idvd;
24                              END IF;
25                          END LOOP;
26                      CLOSE Cur;
27                  ELSE
28                      UPDATE Viaggi SET postiSeconda=postiSeconda-nbiglietti
                            WHERE idViaggio=idv;
29                      OPEN Cur;
30                          LOOP
31                              FETCH Cur INTO idvd,idvs;
32                              IF idvs=idv THEN
33                                  UPDATE Viaggi SET postiSeconda
                                        =postiSeconda-nbiglietti
                                        WHERE idViaggio=idvd;
34                              END IF;
35                          END LOOP;
36                      CLOSE Cur;
37          END IF;
38  END IF;
39  END;
```

7.2 Funzioni

7.2.1 getPosti

```
1  CREATE FUNCTION getPosti(classe BOOL, aereo VARCHAR(10)) RETURNS INT
2  BEGIN
3          DECLARE Posti INT;
4          IF(classe=0) THEN
5                  SELECT postiPrima INTO Posti FROM Aerei WHERE matricola=aereo;
6          ELSE
7                  SELECT postiSeconda INTO Posti FROM Aerei WHERE matricola=
                       aereo;
8          END IF;
9          RETURN Posti;
```

```
10 END;
```

7.3 Trigger

7.3.1 setStatoViaggi

```
1  CREATE EVENT `StatoViaggi` ON SCHEDULE EVERY1 DAY STARTS '2013-07-15 00:00:00'
2  ON COMPLETION NOT PRESERVE ENABLE COMMENT 'Cambia lo stato dei viaggi eseguiti
       '
3   DO call setViaggiEffettuati();
4
5  CREATE PROCEDURE setViaggiEffettuati()
6  BEGIN
7          DECLARE Done INT DEFAULT 0;
8          DECLARE idV INT;
9          DECLARE Cur CURSOR FOR
10                 SELECT idViaggio FROM Viaggi v WHERE v.stato = '
                       previsto' AND v.giorno < CURDATE();
11         DECLARE CONTINUE HANDLER FOR NOT FOUND
12                 SET Done=1;
13
14         OPEN Cur;
15         REPEAT
16                 FECTH Cur INTO idV;
17                 IF NOT Done THEN
18                         UPDATE Viaggi v SET v.stato = 'effettuato' WHERE
                               idViaggio=idV;
19                         DELETE FROM Offerte WHERE idViaggio=idV;
20                 ENDIF;
21         UNTIL Done END REPEAT;
22         CLOSE Cur;
23 END;
```

8 Interfaccia WEB

8.1 Struttura

L'interfaccia web è raggiungibile, una volta collegati al server *@studenti.math.unipd.it*, attraverso le login *msartore* e *mpavanel*. Sono disponibili per l'accesso due account, il primo garantirà l'accesso come ospite mentre il secondo come amministratore

- username: *guest@airlines.it*
  password: *password*
- username: *admin@airlines.it*
  password: *password*

. Ricordiamo che anche senza una login è possibile avere una visione ristretta del sito.
L'applicazione web si divide in tre parti:

1. Un lato utenti, accessibile a tutti, dal quale è possibile vedere i viaggi disponibili, quelli in offerta, e ricercarne specifici secondo dei filtri personalizzati
2. Un lato clienti, accessibile solo agli utenti registrati, dal quale è possibile acquistare dei viaggi e visionare informazioni riepilogative sulla cronologia degli acquisti

3. Un lato amministrativo, raggiungibile solo attraverso un login con privilegi, dal quale è possibile inserire nuovi voli, nuovi viaggi, assegnare i dipendenti per i viaggi, inserire nuove offerte e gestire i privilegi degli utenti registrati al sito

Per le parti 2 e 3 il mantenimento dello stato è ottenuto attraverso l'uso delle sessioni. Di seguito alcune pagine che mostrano le parti salienti della struttura dell'applicazione.

8.2 Login

- login.php

```
<? ob_start(); ?>
<?php session_start(); ?>
<html>
<head>
        <title>
                Airlines
        </title>
        <head>
                <link rel="stylesheet" type="text/css" href="component/style.
                    css">
        </head>
</head>

<body link="red" alink="yellow" vlink="green">
        <?php
        if(isset($_GET['cmd']))
        {
                $path=$_SERVER['PHP_SELF'];
                $cmd=$_GET['cmd'];
                switch($cmd)
                {

                        case "out":     $_SESSION=array();
                                        session_destroy();
                                        header("Location: $path");
                                break;
                        case "nauth": header("Location: $path.?a=nauth");
                                break;
                }
        }
        else
        {
                if(isset($_SESSION['Admin']) | isset($_SESSION['Guest'])){
                        echo "<h3>Prima effettua il <a href=\"login.php?cmd=
                            out\">logout</a><br/></h3>";
                }

        ?>
                <br />
                <br />
                <br />

                <div align="center" style="margin-top:120px">
                <form method="POST" action="component/check.php" class="form">
                <table cellpadding="1" style="border-right:1px solid #000000;
                    border-bottom:2px solid #000000; padding:7px">
```

```
                        <tr height="30px">
                                <td align="center"><h2 class="tt">Autenticati
                                    </h2></td>
47                      </tr>
                        <tr>
                                <td>
                                <table width="260" border="1" bordercolor="
                                    #6397D0" cellspacing="0" align="center"
                                    class="table">
                                  <tr align="center">
52                                      <td width="96" align="right" class="sm
                                            ">
                                        <? if(isset($_GET['a'])){
                                                $alert=$_GET['a'];
                                                if($alert=='nauth')
                                                        echo"(!)";}
57                                      ?>
                                        E-mail:</td>
                                        <td align="left"><input type="text"
                                            name="mail" id="mail" size="25"
                                            /></td>
                                  </tr>
                                  <tr align="center">
62                                      <td align="right" class="sm">Password
                                            :</td>
                                        <td align="left"><input name="password
                                            " type="password" id="password"
                                            size="25" maxlength="8" /></td>
                                  </tr>
                                </table>
                                </td>
67                      </tr>
                        <tr>
                                <td align="right" height="40px">
                                <?
                                if(isset($_GET['e']))
72                              {
                                        if($_GET['e']=="ae")
                                                echo "<span class=\"error\">
                                                    Errore autenticazione &
                                                    nbsp  </span>";
                                }
                                ?>
77                              <input type="submit" name="login" id="login"
                                    value="Accedi" class="button" />
                                </td>
                        </tr>
                </table>
                </form>
82              <table cellpadding="1" style="border-right:1px solid #000000;
                    border-bottom:2px solid #000000; padding:7px">
                        <tr height="30px">
                                <td align="center"><h2 class="tt">
                                    Registrazione</h2></td>
                        </tr>
                        <tr>
87                              <td>
```

```
                                <p class="mm">Se non sei ancora registrato
                                    procedi ed in pochi passi avrai pieno
                                    accesso al sito</p>
                                </td>
                        </tr>
                        <tr>
92                              <td align="center" height="40px">
                                <a href="registration.php?cmd=log"><input type
                                    ="submit" href name="reg" id="login" value
                                    ="Registrati" class="button" /></a>
                                </td>
                        </tr>
                </table>
97              </div>
                <?
                }
                ?>
</body>
102 </html>
<? ob_flush();?>
```

- check.php

```
<?php
2       session_start();
        $insert=$_POST['password'];
        $login=$_POST['mail'];
        if($insert!="" && $login!="")
        {
7               require "db_connection.php";
                $query="SELECT * FROM Anagrafiche NATURAL JOIN Utenti WHERE
                    email=\"$login\"";
                $result = mysql_query($query,$conn) or die("Query fallita" .
                    mysql_error($conn));
                $arr = mysql_fetch_assoc($result);
                $pwd = $arr['password'];
12              if ($pwd == sha1($insert))
                {
                        if($arr['type'] == "Guest"){
                                //header("Location: http://localhost:8888/
                                    default.php");
                                header("Location: /basidati/~msartore/default.
                                    php");
17                              $_SESSION['Privileges'] = $arr['type'];
                                $_SESSION['email'] = $arr['email'];
                                $_SESSION['id'] = $arr['idAnag'];
                        }
                        else{
22                              if(isset($_SESSION['acquista']))
                                        $path=$_SESSION['acquista']."&prima=".
                                            $_SESSION['bigliettiPrima']."&
                                            seconda=".$_SESSION['
                                            bigliettiSeconda'];
                                else{
                                        //$path="http://localhost:8888/admin/
                                            administration.php";
```

```
                                        $path="/basidati/~msartore/admin/
                                            administration.php";
                                        }

                                header("Location: $path");
                                $_SESSION['Privileges'] = $arr['type'];
                                $_SESSION['email'] = $arr['email'];
                                $_SESSION['id'] = $arr['idAnag'];
                        }
                }
                else
                {
                        //header("Location: http://localhost:8888/login.php?e=
                            ae");
                        header("Location: /basidati/~msartore/login.php?a=
                            nauth");
                }
        }
        else
        {
                //header("Location: http://localhost:8888/login.php?e=ae");
                header("Location: /basidati/~msartore/login.php?e=ae");
        }
?>
```

8.3 Default

- default.php

```
<?php session_start();?>
<html>
<head>
        <title>
                Airlines
        </title>
        <head>
                <link rel="stylesheet" type="text/css" href="\component\style.
                    css">
                <meta http-equiv="Content-Type" content="text/html; charset=
                    UTF-8" />
        </head>
</head>

<body link="#002089" alink="#002089" vlink="#002089">

<div id="personale" align="center" >
<?php
        if(isset($_REQUEST["cmd"]))
                if($_REQUEST["cmd"]=="logout")
                        {
                                $_SESSION=array();
                                session_destroy();
                                header("Location:/basidati/~msartore/default.
                                    php");
                        }
        if(isset($_SESSION["Privileges"])){
```

```
                echo "Benvenuto ".$_SESSION["email"] .", <a href=\"default.php
                    ?cmd=logout\" >Logout</a>";
                echo "<p>Vai alla tua <a href=\"personale.php\" >pagina
                    personale</a></p>";
                echo "<p>Vedi le <a href=\"research.php?cmd=offerte\" >Offerte
                    </a></p>";
        }
        else{
                echo "<p>Vedi le <a href=\"research.php?cmd=offerte\" >Offerte
                    </a></p>";
                echo "Devi essere loggato o registrato per effettuare una
                    prenotazione <a href=\"login.php\" class=\"postlink\"
                    target=\"_new\">Login o Registrati</a>";
        }
?>
</div>

<div id="filtri" align="center" style="float:right; width:25%;" >
        <?php require_once "filter.php";

                if(isset($_REQUEST['err']))
                {
                        echo "<p style=\"color:red;\">Errore data. Ripetere la
                             ricerca</p>";
                }
                if(isset($_COOKIE["Destinazioni"]))
                {
                        $destinazioni=explode(',',$_COOKIE["Destinazioni"]);
                        echo "<h3 style=\"color:blue\"> Ultima ricerca
                            effettuata</h3>
                                        <p>DA: $destinazioni[0]</p>
                                        <p>A: $destinazioni[1]</p>";
                }

        ?>
</div>

<div id="voliDelGiorno" align="center" color="123456" style="width:75%; float:
    left;">
        <?php
                require_once "component/db_connection.php";
                $query="SELECT * FROM viewViaggiDiretti WHERE postiSeconda>1
                    AND stato='previsto' AND giorno>NOW() ORDER BY giorno ASC
                    LIMIT 0,20";
                $result=mysql_query($query,$conn);

                if(isset($_SESSION["Privileges"]))
                {

                echo "<h4>Voli previsti a breve <br></h4>
                        <table align=\"top-left\" border=\"2px\" bordercolor
                            =\"#99AF99\" style=\"margin:0px\">
                        <tr>
                                <th>Partenza</th>
                                <th>Arrivo</th>
                                <th>Durata</th>
                                <th>Giorno</th>
```

```
                        <th>Prezzo</th>
                        <th>Acquista</th>

                </tr>";
while($row=mysql_fetch_array($result))
{
echo "
                <form method=\"GET\" action=\"details.php\" >
                <tr>
                        <td>$row[4] $row[2] $row[6]</td>
                        <td>$row[5] $row[3] $row[7]</td>
                        <td>$row[8]</td>
                        <td>$row[1]</td>
                        <td>$row[11],00E</td>
                        <td height=\"25\">
                        <input type=\"hidden\" name=\"voloa\" value=\"
                            diretto\">
                        <input type=\"hidden\" name=\"idv\" value=\"
                            $row[0]\">
                        <input type=\"image\" src=\"images\go.png\"
                            value=\"Dettagli\ height=\"30\" width
                            =\"30\" alt=\"Acquista\"></td>
                </tr>
                </form>
                ";
}
echo "</table>";
}
else{

echo "<h4>Voli della giornata <br>"; echo date('Y-m-d'); echo"
    </h4>
                <table align=\"top-left\" border=\"2px\" bordercolor
                    =\"#99AF99\" style=\"margin:0px\">
                <tr>
                        <th>Partenza</th>
                        <th>Arrivo</th>
                        <th>Durata</th>
                        <th>Giorno</th>
                        <th>Prezzo</th>
                </tr>";
while($row=mysql_fetch_array($result))
{
echo "<form method=\"GET\" class=\"form\">
<tr>
                <td>$row[4] $row[2] $row[6]</td>
                <td>$row[5] $row[3] $row[7]</td>
                <td>$row[8]</td>
                <td>$row[1]</td>
                <td>$row[11],00E</td>
</tr>
</form>";

}
echo "</table>";
}
```

```
        ?>
</div>

</body>
</html>
```

8.4 Administration

- administration.php

```
<?php session_start(); ?>
<html>
        <head>
                <title>
                        Airlines
                </title>
                <head>
                        <link rel="stylesheet" type="text/css" href="../
                            component/style.css">
                </head>
        </head>

        <body link="#002089" alink="#002089" vlink="#002089">
                <?php
                if(isset($_SESSION['Privileges']) && $_SESSION['Privileges']==
                    "Admin"){
                        require "../component/db_connection.php";
                        require "banneradmin.php";
                        include "sidebar.php";
                        $id=$_SESSION['id'];
                        $query="SELECT nome, cognome FROM Anagrafiche WHERE
                            idAnag=$id";
                        $result = mysql_query($query,$conn) or die("Query
                            fallita" . mysql_error($conn));
                        if($row = mysql_fetch_row($result))
                                $string = $row[0]." ".$row[1];
                        <div class=\"content\">
                        <div style=\"padding-left:7%\">
                        <table border=\"1\" bordercolor=\"#99FFFF\"
                            cellspacing=\"0\" align=\"center\" class=\"table\"
                             cellpadding=\"3\" >
                        <tr>
                                <td colspan=\"7\" align=\"center\"><h2>
                                    Riepilogo viaggi inseriti da $string</h2
                                    ></td>
                        </tr>
                        <th>Giorno</th>
                        <th>Da</th>
                        <th>A</th>
                        <th>Partenza</th>
                        <th>Arrivo</th>
                        <th>Compagnia</th>";

                        $query="SELECT * FROM viewViaggiDiretti WHERE admin=
                            $id";
                        $result = mysql_query($query,$conn) or die("Query
                            fallita" . mysql_error($conn));
```

```
                        while ($row = mysql_fetch_row($result)
                        {
                          echo"
                          <tr>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[1] </label></td>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[4] </label></td>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[5] </label></td>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[6] </label></td>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[7] </label></td>
                                <td align=\"center\" style=\"padding-right:10
                                    px\"><label> $row[14] </label></td>
                          </tr>";
                        }
                        echo"
                        </table>
                        </div>
                        </div>";
                        }
                else
                        include "error.php";
                ?>
        </body>
</html>
```

- manageprivileges.php

```
<?php session_start(); ?>
<html>
        <head>
                <title>
                        Airlines
                </title>
                <head>
                        <link rel="stylesheet" type="text/css" href="../
                            component/style.css">
                </head>
        </head>

        <body link="#002089" alink="#002089" vlink="#002089">
                <?php
                if(isset($_SESSION['Privileges']) && $_SESSION['Privileges']==
                    "Admin"){
                        require "../component/db_connection.php";
                        include "banneradmin.php";
                        include "sidebar.php";
                        if(isset($_GET['userid']) && $_REQUEST['buttonForm']==
                            "Aggiorna"){
                                $id=$_GET['userid'];
                                $query="UPDATE Utenti SET type='$_POST[type]'
                                    WHERE idAnag=$id";
                                $result = mysql_query($query,$conn) or die("
                                    Query fallita" . mysql_error($conn));
```

```
}
elseif(isset($_GET['userid']) && $_REQUEST['buttonForm
    ']=="Elimina"){
        $id=$_GET['userid'];
        $query="call eliminaUtente('$id');";
        $result = mysql_query($query,$conn) or die("
            Query fallita" . mysql_error($conn));
}

echo "<div class=\"content\">
                <div style=\"padding-left:7%\">
                <table border=\"1\" bordercolor=\"#99
                    FFFF\" cellspacing=\"0\" align=\"
                    center\" class=\"table\"
                    cellpadding=\"3\" >
                <tr>
                        <td colspan=\"8\" align=\"
                            center\"><h2>Manage
                            privileges</h2></td>
                </tr>
                <th>id</th>
                <th>nome</th>
                <th>cognome</th>
                <th>email</th>
                <th colspan=\"2\">permessi</th>
                <th colspan=\"2\">azioni</th>";

$query="SELECT idAnag, nome, cognome, email, type FROM
     Anagrafiche a NATURAL JOIN Utenti";
$result = mysql_query($query,$conn) or die("Query
    fallita" . mysql_error($conn))
while ($row = mysql_fetch_row($result))
{
        echo "<form method=\"POST\" action=\"
            manageprivileges.php?userid=$row[0]\"
            class=\"form\">
        <tr>
        <td align=\"center\" style=\"padding-right:10
            px\"><label> $row[0] </label></td>
        <td align=\"center\" style=\"padding-right:10
            px\"><label> $row[1] </label></td>
        <td align=\"center\" style=\"padding-right:10
            px\"><label> $row[2] </label></td>
        <td align=\"center\" style=\"padding-right:10
            px\"><label> $row[3] </label></td>";
        if($row[4]=="Guest"){
                echo"<td><input type=\"radio\" name=\"
                    type\" value=\"Guest\" checked=\"
                    checked\"/>&nbspGuest  &
                    nbsp</td>
                <td><input type=\"radio\" name=\"type
                    \" value=\"Admin\"/>&nbspAdmin&
                    nbsp</td>";
        }
        else{
                echo"<td><input type=\"radio\" name=\"
                    type\" value=\"Guest\"/>&nbspGuest
```

```
                                         </td>
                                    <td><input type=\"radio\" name=\"type
                                        \" value=\"Admin\" checked=\"
                                        checked\"/>&nbspAdmin  &
                                        nbsp</td>";
                                }
                                echo"
                                <td align=\"center\">
                                        <button type=\"submit\" name=\"
                                            buttonForm\" value=\"Aggiorna\"><
                                            img src=\"..\images\update_user.
                                            png\" alt=\"Aggiorna utente\"></
                                            button>
                                </td>
                                <td align=\"center\">
                                <button type=\"submit\" name=\"buttonForm\"
                                    value=\"Elimina\"><img src=\"..\images\
                                    delete_user.png\" alt=\"Elimina utente
                                    \"></button>
                                </td>
                                </tr>
                                </form>";
                        }
                        echo"
                        </table>
                        </div>
        </div>";
                }
                else
                        include "error.php";
                ?>
        </body>
</html>
```

- sidebar.php

```
<div id='cssmenu'>
<ul>
   <li><a href='administration.php'><span>Home</span></a></li>
   <li><a href='managevoli.php?option=insert'><span>Voli</span></a></li>
   <li class='active has-sub'><a href='#'><span>Viaggi</span></a>
      <ul>
         <li><a href='manageviaggi.php?option=insert'><span>Diretti</span></a
            ></li>
         <li><a href='manageviaggiscali.php?option=insert'><span>Con scali</
            span></a></li>
      </ul>
   </li>
   <li><a href='manageassistenze.php'><span>Assistenze</span></a></li>
   <li class='active has-sub'><a href='#'><span>Offerte</span></a>
      <ul>
         <li><a href='manageofferte.php?option=insert'><span>Inserisci</span
            ></a></li>
         <li><a href='manageofferte.php?option=edit'><span>Modifica</span></a
            ></li>
      </ul>
   </li>
```

```
   <li><a href='manageprivileges.php'><span>Gestisci privilegi</span></a></li>
</ul>
</div>
```