Dalla Riva Alessandro
Scozzai Samuele
Propedo Demien
Tavano Matteo

Progetto di Basi di Dati e Laboratorio aa. 22-23

Implementazione di una base di dati relativa ad un sistema aeroportuale.

Nell'era dell'informazione, l'elaborazione, la gestione e l'analisi dei dati rivestono un ruolo sempre più cruciale in una vasta gamma di settori. La creazione di un database efficace e la sua gestione adeguata diventano quindi elementi fondamentali per la presa di decisioni informate e per l'ottimizzazione delle risorse aziendali. Questa relazione si concentra sul processo di sviluppo di un database, dall'idea iniziale all'analisi dei dati, attraverso le fasi di progettazione concettuale, progettazione logica, implementazione in SQL e sfruttamento delle potenzialità di analisi dei dati con R.

Nella società attuale, in cui dati di ogni tipo sono generati a ritmi vertiginosi, la necessità di organizzare e interpretare tali informazioni è diventata una priorità. L'efficacia con cui i dati vengono archiviati e interrogati può influenzare la qualità delle decisioni prese, dai livelli operativi a quelli strategici. Pertanto, la progettazione accurata e il corretto utilizzo di un database svolgono un ruolo determinante nel garantire l'affidabilità e l'integrità delle informazioni.

Nel corso di questa relazione, esploreremo le fasi coinvolte nel passaggio dalla concezione di un database all'analisi dei dati. Inizieremo descrivendo la fase iniziale di pianificazione, in cui si delineano gli obiettivi principali e i requisiti del database. Successivamente, esamineremo la progettazione concettuale, in cui verranno identificate le entità e le relazioni chiave del dominio di dati. Procederemo poi alla progettazione logica, traducendo il modello concettuale in uno schema relazionale. Continueremo con la fase di sviluppo, implementando il database utilizzando SQL e creando query per estrarre informazioni significative. Infine, esploreremo come utilizzare R per analizzare i dati del database, identificando pattern, trend e relazioni.

Attraverso questa relazione, dimostreremo come il processo di sviluppo di un database possa trasformare dati disorganizzati in una risorsa preziosa, pronta ad essere sfruttata per trarre conclusioni ed effettuare scelte ponderate

1. Raccolta e analisi dei requisiti (dati e operazioni).

La raccolta dei requisiti consiste nella completa identificazione dei problemi che l'applicazione deve risolvere e le caratteristiche che dovrebbe caratterizzare tale applicazione.

L'analisi dei requisiti consiste nel chiarimento e organizzazione della specifica dei requisiti che derivano da:

- utenti dell'applicazione;
- documentazione esistente connessa al problema, regole interne procedure business, leggi e regole.
- Possibili versioni precedenti dell'applicazione.

Utenti dell'applicazione:

Questo software è stato progettato appositamente per agevolare e ottimizzare le operazioni quotidiane dell'amministrazione aeroportuale, consentendo agli utenti di fornire un servizio efficiente e sicuro per passeggeri e compagnie aeree.

Documento di specifiche

Dominio: sistema informativo relativo ad una gestione di una base dati relativa ad un aeroporto.

- Ogni aeroporto sia identificato univocamente da un codice e sia caratterizzato da un nome, da una città e da una nazione. Si assuma che in una data città non vi possano essere due aeroporti con lo stesso nome (e che ogni città sia identificata univocamente dal nome).

- Ogni tipo di aeroplano sia identificato dal nome e sia caratterizzato dal nome dell'azienda costruttrice, dal numero massimo di posti a sedere e dall'autonomia di volo.

- Ogni aeroplano sia identificato da un codice e sia caratterizzato dal tipo (di aeroplano) e dal numero di posti a sedere messi effettivamente a disposizione per i passeggeri. Si assuma che uno stesso aeroplano possa essere utilizzato da compagnie diverse.

- Per ogni aeroporto, sia definito l'insieme dei tipi di aeroplano che possono decollare/atterrare.

- Ogni volo sia caratterizzato da un codice, che lo identifica univocamente, dalla compagnia aerea che offre il volo, dai giorni della settimana in cui viene offerto (si assuma che ogni volo sia effettuato al più una volta in un dato giorno, che gli orari di ogni volo siano sempre gli stessi e che ogni volo, anche quando costituito da più tratte, inizi e termini nella stessa giornata) e, per ogni classe (business, economica, ..), dal prezzo del biglietto (si assuma che biglietti di un dato volo relativi alla stessa classe abbiano tutti lo stesso prezzo).

- Ogni volo sia costituito da un insieme di tratte intermedie (una tratta sia una porzione di volo priva di scali intermedi), ciascuna identificata da un numero progressivo (prima tratta del volo, seconda tratta del volo, ..). Ogni tratta sia caratterizzata da un aeroporto di partenza, un aeroporto di arrivo, un orario previsto di partenza e un orario previsto di arrivo.

- Per ogni specifica istanza di tratta, la data in cui avrà luogo (ad esempio, la tratta Trieste-Monaco del 25 febbraio 2012), l'aeroplano utilizzato (si assuma che su una stessa tratta possano essere utilizzati in date diverse aeroplani diversi) e il numero di posti ancora disponibili.

- Ogni prenotazione relativa ad una certa istanza di tratta sia identificata dal posto prenotato (numero più lettera; ad esempio, posto 16D) e sia caratterizzata dal nome, dal cognome e dal recapito telefonico della persona che ha prenotato il posto.

Legenda:

Rosso: Entità

Blu: Relazioni

Verde: Attributi di entità

Viola: Attributi di relazioni

Marrone: ???

Possibili versioni precedenti dell'applicazione:

In fase di implementazione, si può assistere alla migrazione dei dati preesistenti verso il nuovo sistema. Ciò garantisce una transizione fluida, consentendo agli utenti di continuare a lavorare con dati storici e mantenere la continuità delle operazioni.

Specifiche in Natural language:

Il linguaggio è soggetto ad ambiguità e malinterpretazione, si necessita perciò di un'analisi specifica del documento per rimuovere ambiguità e curare la terminologia. Scegliamo perciò un livello di astrazione al fine di standardizzare la struttura della frase, evitando frasi complesse. Per identificare sinonimi e omonimi ricorriamo ad un glossario dei termini, che ci aiuterà anche a rendere espliciti i riferimenti incrociati.

Glossario dei termini:

Termine	Descrizione	Sinonimi	Collegamenti
Aeroporto	Struttura, infrastruttura che permette il decollo e l'atterraggio degli aerei	-	Aerei, Compagnie aeree, Tratte
Codice aeroporto	Codice alfanumerico che identifica univocamente un aeroporto	-	-
Nome aeroporto	Nome ufficiale dell'aeroporto, assegnazione per identificazione non univoca.	-	-
Città	Centro abitato e area limitrofa, a cui appartiene uno specifico aeroporto.	Comune	Aeroporto, Nazione
Nazione	Ente politico-territoriale.	Stato, paese	Aeroporto, Città
Aeroplano	Veicolo abilitato al trasporto aereo di passeggeri	Aereo	Tipo di aeroplano, Compagnie aeree, Tratte
Codice aeroplano	Codice alfanumerico che identifica univocamente un aeroplano	-	-
Nome aeroplano	Nome ufficiale dell'aeroplano	-	-
Azienda costruttrice	Azienda che ha prodotto l'aeroplano	Produttore	Aeroplano
Posti a sedere	Numero di posti a sedere disponibili su un aeroplano	Capacità a sedere	Aeroplano, Tratte
Autonomia di volo	Distanza massima che può essere percorsa dall'aeroplano senza dover effettuare rifornimento	-	Aeroplano
Tipo di aeroplano	Categoria o modello dell'aeroplano	Modello, Categoria	Aeroplano
Compagnia aerea	Azienda che offre servizi di trasporto aereo di passeggeri e merci	-	Volo, Tratte, Prenotazione
Codice volo	Codice alfanumerico che identifica univocamente un volo	-	Volo, Tratte, Prenotazione
Giorni della settimana	Giorni in cui viene offerto un volo	-	Volo
Classe	Categoria di prenotazione sul volo (es. business, economica)	-	Volo, Prenotazione
Prezzo del biglietto	Prezzo di un biglietto di un certo volo per una certa classe	Tariffa	Volo, Prenotazione

Tratta	Porzione di volo priva di scali intermedi	-	Volo, Tratte, Prenotazione
Aeroporto di partenza	Aeroporto da cui parte una tratta	-	Tratta, Prenotazione
Aeroporto di arrivo	Aeroporto a cui arriva una tratta	-	Tratta, Prenotazione
Orario previsto di partenza	Orario in cui è previsto che una tratta parta	-	Tratta, Prenotazione
Orario previsto di arrivo	Orario in cui è previsto che una tratta arrivi	-	Tratta, Prenotazione
Data della tratta	Data in cui si effettua una tratta	-	Tratta, Prenotazione
Tratta corrente	tratta progressiva, identificata da un numero, in caso di scali, si riferisce a un numero progressivo	Tratta reale, tratta intermedia	Tratta, Prenotazione, Aeroplano.

Riscrittura e ristrutturazione dei requisiti:

Dividiamo i paragrafi della consegna che trattano uno specifico argomento:

Frasi di natura generale: Si voglia modellare il seguente insieme di informazioni riguardanti un sistema di prenotazione dei voli aerei.

Frasi che si riferiscono all'aeroporto: Ogni aeroporto sia identificato univocamente da un codice e sia caratterizzato da un nome, da una città e da una nazione.

Frasi che si riferiscono ad una città in cui appartiene un aeroporto: Si assuma che in una data città non vi possano essere due aeroporti con lo stesso nome (e che ogni città sia identificata univocamente dal nome).

Frasi che si riferiscono ad un tipo di aeroplano:

Ogni tipo di aeroplano sia identificato dal nome e sia caratterizzato dal nome dell'azienda costruttrice, dal numero massimo di posti a sedere e dall'autonomia di volo.

Frasi che si riferiscono ad un aeroplano e dal tipo di aeroplano che lo caratterizza:

Ogni aeroplano sia identificato da un codice e sia caratterizzato dal tipo (di aeroplano) e dal numero di posti a sedere messi effettivamente a disposizione per i passeggeri. Si assuma che uno stesso aeroplano possa essere utilizzato da compagnie diverse.

Frasi che si riferiscono sia all'aeroporto che ai tipi di aeroplano:

Per ogni aeroporto, è definito l'insieme dei tipi di aeroplano che possono decollare/atterrare.

Frasi che si riferiscono ad un volo e alla compagnia aerea che lo offre:

Ogni volo sia caratterizzato da un codice, che lo identifica univocamente, dalla compagnia aerea che offre il volo, dai giorni della settimana in cui viene offerto (si assuma che ogni volo sia effettuato al più una volta in un dato giorno, che gli orari di ogni volo siano sempre gli stessi e che ogni volo, anche quando costituito da più tratte, inizi e termini nella stessa giornata) e, per ogni classe (business, economica, ..), dal prezzo del biglietto (si assuma che biglietti di un dato volo relativi alla stessa classe abbiano tutti lo stesso prezzo).

Frasi che si riferiscono ad un tipo di volo inerenti le tratte intermedie che comprende:

Ogni volo sia costituito da un insieme di tratte intermedie (una tratta sia una porzione di volo priva di scali intermedi), ciascuna identificata da un numero progressivo (prima tratta del volo, seconda tratta del volo, ..).

Frasi che si riferiscono alla tratta: Ogni tratta sia caratterizzata da un aeroporto di partenza, un aeroporto di arrivo, un orario previsto di partenza e un orario previsto di arrivo.

Frasi che si riferiscono alla specifica istanza di tratta:

Per ogni specifica istanza di tratta, la data in cui avrà luogo (ad esempio, la tratta Trieste-Monaco del 25 febbraio 2012), l'aeroplano utilizzato (si assuma che su una stessa tratta possano essere utilizzati in date diverse aeroplani diversi) e il numero di posti ancora disponibili.

Frasi che si riferiscono ad una prenotazione di un'istanza di tratta:

Ogni prenotazione relativa ad una certa istanza di tratta sia identificata dal posto prenotato (numero più lettera; ad esempio, posto 16D) e sia caratterizzata dal nome, dal cognome e dal recapito telefonico della persona che ha prenotato il posto.

Stesura requisiti operativi (dati e operazioni)

Delineiamo un insieme di requisiti operativi che il nostro sistema dovrà soddisfare, e nello specifico andiamo a mettere in evidenza alcune delle operazioni più frequenti con cui la nostra base dati dovrà interfacciarsi nel quotidiano uso dell'applicazione.

Operazione 1: Inserire un nuovo volo, definendo tratta, prezzo del biglietto per ogni classe. (10 o 15 volte al mese);

Operazione 2: Visualizza, per ogni aeroporto i tipi di aeroplano che vi possono atterrare o decollare. (5000 volte al giorno);

Operazione 3: Dato uno specifico volo, il sistema deve essere in grado di riconoscere le tratte intermedie necessarie e per ciascuna assegnare un aeroporto di partenza, uno di arrivo, gli orari per la partenza e per l'arrivo fornendone un numero progressivo. (10 o 15 volte al mese);

Operazione 4: Per ogni specifica istanza di tratta, è necessario registrare l'aeroplano utilizzato, la data e il numero di posti ancora disponibili. (8000 volte al giorno);

Operazione 5: Il sistema deve essere ora in grado di registrare la prenotazione effettuata dall'utente, tenendo traccia del numero identificativo e acquisendone le caratterizzazioni specifiche quali numero di telefono, nome e cognome. (100000 volte al giorno);

Operazione 6: Visualizzare tutti i passeggeri in una tratta di uno specifico volo (10000 volte al giorno);

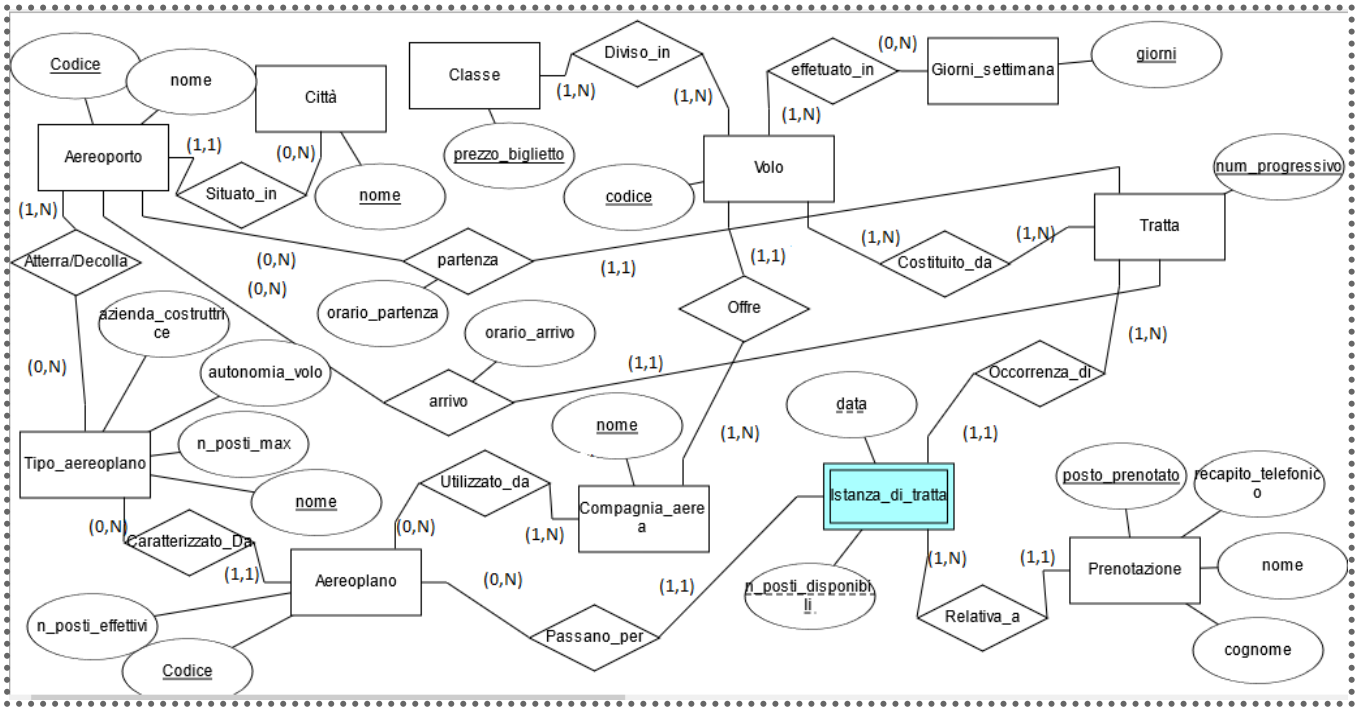
Operazione 7: Cancellare specifiche tratte di voli a causa di imprevisti (eventualmente notificare i clienti) (2000 volte al giorno).

2. Progettazione concettuale

Di seguito, riportiamo la progettazione concettuale della base di dati che andremo a rappresentare, mediante lo schema Entità/Relazioni.

Per la stesura di quest'ultimo ci avvaliamo della strategia "inside-out", inquadrando le componenti principali quali aereoporto, aeroplano, volo e tratta. Di queste vogliamo espandere la rappresentazione inquadrando le componenti principali coinvolte come entità e assegnando delle relazioni che soddisfano dei vincoli di cardinalità.

Utilizziamo *ERDplus* per la realizzazione grafica dello schema.



Assunzioni e vincoli sullo schema Entità/Relazioni:

- Assumiamo che in una città possano esserci più aeroporti, distinti da nomi diversi. In una città possono, inoltre, non esserci nessun aeroporto (es. se non esiste più o è stato rimosso). Un aeroporto è univocamente assegnato ad una e una sola città, non vi possono essere aeroporti che appartengono a più città (per esempio, Venezia-Mestre assumiamo che si trovi presso Venezia anche se effettivamente si trova sul suolo di Mestre).
- Nel caso di un aeroplano, assumiamo che possa momentaneamente non essere utilizzato da nessuna compagnia aerea, magari perchè non adatto.., oppure possa coesistere ed essere adoperato da più compagnie aeree. Allo stesso tempo, una compagnia aerea deve utilizzare almeno un aeroplano per offrire il servizio, altrimenti non avrebbe senso esista e può utilizzarne molteplici per coprire più tratte e offrire più voli. Assumiamo che una compagnia aerea sia identificata univocamente dal suo nome (Alitalia, Ryanair..).
- In ogni singolo aeroporto, possono atterrare/decollare più tipi di aeroplano, ma che almeno un tipo di aeroplano vi possa atterrare o decollare affinchè sia valido e in funzione. Un tipo di aereo, può non essere predisposto per l'utilizzo di uno specifico aeroporto (per indisposizione momentanea o definitiva) e al massimo utilizzare lo specifico aeroporto se abilitato.
- Assumiamo, che una compagnia aerea deve mettere a disposizione almeno 1 volo e ne possa al contempo offrire molteplici. Un volo è univocamente associato alla compagnia aerea che lo mette a disposizione (per esempio, il volo RA9666 offerto da Ryanair, dove le prime 4 lettere identificano Ryanair). Quindi, dovrà esistere una compagnia aerea che lo metta a disposizione, e non esistono voli che non coinvolgono alcuna compagnia aerea o che ne comprendono più di una (non assumiamo che un volo possa essere messo a disposizione sia da Fly emirates che da US Airlines).
- Un volo deve comprendere almeno una classe di prezzo (identificata dal prezzo del biglietto), e al più può comprenderne molteplici (come da consegna, business, economica…). La classe deve essere compresa almeno in un volo affinché abbia senso esista (pensiamo ad una classe ad hoc istituita unicamente per un volo). Una classe può anche essere compresa in più voli che la comprendono.
- Assumiamo che un volo sia offerto almeno in un giorno della settimana (e come da consegna, duri tutto il giorno, ovvero inizi e termini nello stesso giorno) e al più sia offerto in più giorni della settimana (come per esempio, un volo feriale effettuato giornalmente per motivi lavorativi). Un giorno della settimana, può non offrire uno specifico volo, pensiamo ad una domenica per una tratta internazionale, oppure ne può offrire molteplici e diversi (per esempio, il lunedì può offrire sia il volo RA9666 che il volo RA7766). Per semplicità, assumiamo che un giorno della settimana sia identificato dal suo nome.
- Un volo è costituito da una tratta al minimo (se non ha scali ed è diretto) o può essere suddiviso in più tratte intermedie. La stessa tratta è assegnata al minimo ad un volo che la comprende o possa essere inclusa in più tratte che la comprendono. Pensiamo alla tratta Milano-Parigi, che può essere compresa nel volo che parte da Sydney e arriva a New York, e assieme alla tratta che va da Trieste a Parigi. Il numero progressivo delle tratte, indica un ordinamento attraverso cui si susseguono le tratte.

- Assumiamo che da un aeroporto, possono non partire istanze specifiche di tratte, ovvero non lo comprendono, nell'andata o nel ritorno, e al più possa essere sede di partenza/arrivo di più tratte. Una tratta deve avere uno e solo uno aeroporto di arrivo e di partenza (potrebbe non averlo nel caso per esempio, che il volo non atterri, per sinistri avvenuti durante il volo, ma non lo consideriamo). L'orario di partenza e di arrivo lo attribuiamo alla relazione precedentemente definita.
- L'istanza di tratta, la consideriamo come entità debole definita solamente in presenza di tratta, ovvero abbia senso solo in caso di presenza di quest'ultima. Un'istanza di tratta, utilizza un solo aeroplano e al contempo, un aeroplano può essere utilizzato in più istanze di tratte (in date differenti ovviamente) e può avere, ovviamente, più prenotazioni relative a più passeggeri (Mario Rossi, Carlo Bruni), mentre la prenotazione è univoca e relativa a tale persona che la effettua.
- Infine, assumiamo che ogni aeroplano sia caratterizzato da un tipo di aeroplano (modello specifico Boeing747, Boeing323) e che non ci possano essere aeroplani non caratterizzati da nessun tipo di aeroplano. Un tipo di aeroplano può caratterizzare più aeroplani, se per esempio viene ampiamente utilizzato come il Boeing747, o possa non essere coinvolto nella relazione se non viene associato a nessun aeroplano (pensiamo ad un aeroplano datato o con delle specifiche non consone all'incarico).

3. Progettazione logica

Lo scopo della progettazione logica è costruire uno schema relazionale che correttamente ed efficientemente rappresenti tutte le informazioni descritte dallo schema E/R, prodotto nella fase precedente.

E' utile dividere lo sviluppo logico in due fasi: ristrutturazione dello schema E/R e traslazione nel modello logico.

Lo scopo della ristrutturazione dello schema E/R mira ad ottimizzare due parametri: il costo di un'operazione e lo spazio di archiviazione richiesto.

Per valutare questi due parametri progettiamo le tabelle dei volumi e delle operazioni.

Produzione della tabella dei volumi e della tabella delle operazioni

Procediamo con la creazione della tabella dei volumi, riassumiamo all'interno delle tabelle (volume e operazioni) il volume dei dati in questione e le caratteristiche generali delle operazioni. Indichiamo con E se il concetto rappresenta un Entità, oppure R se si tratta invece di una relazione. Il volume proposto, è costruito a partire dalla quantità di dati che andremo a stimare.

Tabella dei volumi:

Concetto	Tipo	Volume
Aeroporto	E	100/200
Città	E	1000
Tipo Aeroplano	E	50
Aeroplano	E	500
Compagnia Aerea	E	20/30
Volo	E	5000
Classe	E	3/4
Giorno Settimana	E	7
Tratta	E	200
Istanza di tratta	E	10000
Prenotazioni	E	70000
Situato In	R	100/200
Atterra/Decolla	R	50
Caratterizzato da	R	500
Utilizzato da	R	500
Offre	R	5000
Diviso in	R	10000/20000
Effettuato in	R	30000
Costituito da	R	100000
Occorrenza di	R	1000
Relativa a	R	200000
Partenza	R	1000
Arrivo	R	1000

Produzione della tabella delle operazioni:

Nella tabella delle operazioni, indichiamo invece con I se l'operazione è interattiva e con B quelle di batch. La frequenza proposta, rispecchia quella citata in precedenza.

Operazione	Tipo	Frequenza
Operazione 1	I	10 o 15 volte al mese
Operazione 2	I	5000 volte al giorno
Operazione 3	I	10 o 15 volte al mese
Operazione 4	I	8000 volte al giorno
Operazione 5	I	100000 volte al giorno
Operazione 6	I	10000 volte al giorno
Operazione 7	I	2000 volte al giorno

Navigation schema

La gestione di un database può essere complessa, ma si è creato questo schema per semplificare il processo e rendere l'accesso ai dati rapido, efficiente e intuitivo, permettendo di migliorare la leggibilità dello schema E/R.

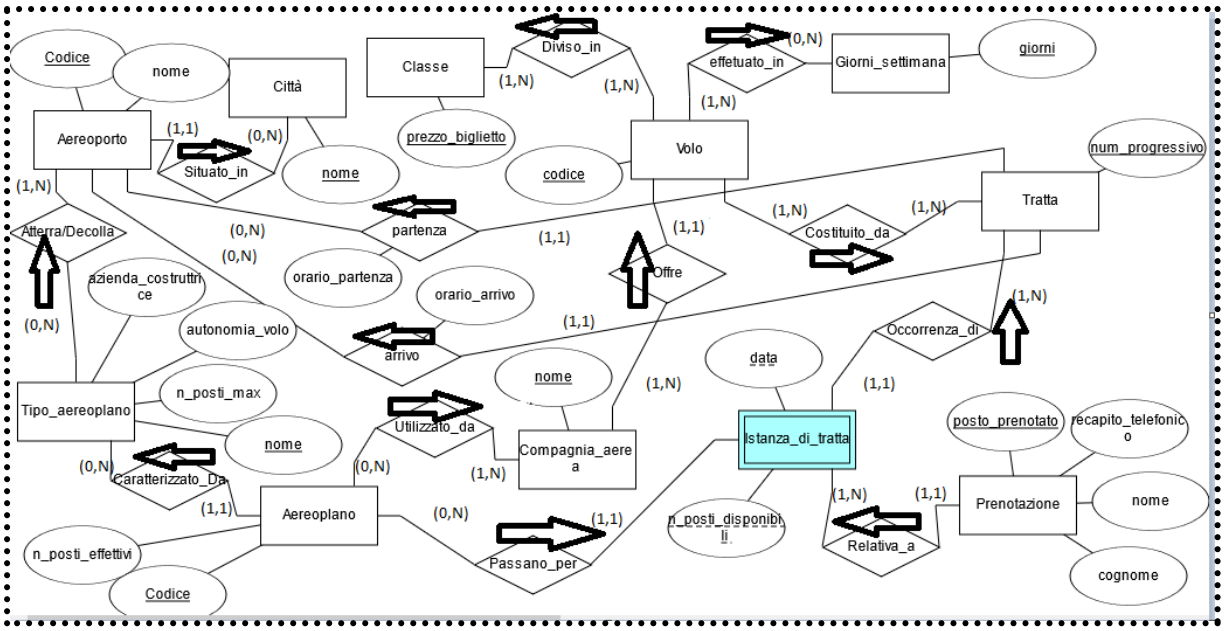


Tabella degli accessi e analisi delle ridondanze.

Il costo di un'operazione, valutata usando la tabella dei volumi e il navigation scheme, può essere riassunto nella tabella degli accessi. Indichiamo per ogni concetto (entità/relazione) il numero di accessi necessari in lettura o scrittura.

Valutiamo il costo di ogni singola operazione richiesta all'interno del nostro sistema in termini di accessi, al fine di decidere se modificare eventuali ridondanze nel nostro schema.

Operazione 1: Inserire un nuovo volo, definendo tratta, prezzo del biglietto per ogni classe. (10 o 15 volte al mese);

Costrutto	Tipo	Accessi	Tipo
Volo	Entità	1	W
Costituito_da	Relazione	1	R
Volo	Entità	1	R
Tratta	Entità	1	R
Diviso_in	Relazione	1	W
Classe	Entità	1	W

Operazione 2: Visualizza, per ogni aeroporto i tipi di aeroplano che vi possono atterrare o decollare. (5000 volte al giorno);

Costrutto	Tipo	Accessi	Tipo
Tipo_aeroplano	Entità	1	R
Atterra/Decolla	Relazione	1	R
Aeroplano	Entità	1	R

Operazione 3: Dato uno specifico volo, il sistema deve essere in grado di riconoscere le tratte intermedie necessarie e per ciascuna assegnare un aeroporto di partenza, uno di arrivo, gli orari per la partenza e per l'arrivo fornendone un numero progressivo. (10 o 15 volte al mese);

Costrutto	Tipo	Accessi	Tipo
Volo	Entità	1	R
Costituito_da	Relazione	1	R
Tratta	Entità	1	R
Partenza	Relazione	1	W
Aeroporto	Entità	1	W
Arrivo	Relazione	1	W
Aeroporto	Entità	1	W

Operazione 4: Per ogni specifica istanza di tratta, è necessario registrare l'aeroplano utilizzato, la data e il numero di posti ancora disponibili. (8000 volte al giorno);

Costrutto	Tipo	Accessi	Tipo
Aeroplano	Entità	1	R
Passano_per	Relazione	1	R
Istanza_di_tratta	Entità	1000	W

Operazione 5: Il sistema deve essere ora in grado di registrare la prenotazione effettuata dall'utente, tenendo traccia del numero identificativo e acquisendone le caratterizzazioni specifiche quali numero di telefono, nome e cognome. (100000 volte al giorno);

Costrutto	Tipo	Accessi	Tipo
Prenotazione	Entità	1	R
Prenotazione	Entità	1	W

Operazione 6: Visualizzare tutti i passeggeri in una tratta di uno specifico volo (10000 volte al giorno);

Costrutto	Tipo	Accessi	Tipo
Preontazione	Entità	1	R
Relativa_a	Relazione	1	R
Istanza_di_ tratta	Entità	1	R

Operazione 7: Cancellare specifiche tratte di voli a causa di imprevisti (eventualmente notificare i clienti) (2000 volte al giorno).

Costrutto	Tipo	Accessi	Tipo
Istanza_di_ tratta	Entità	1	R
Istanza_di_tratta	Entità	1	W

Dopo un’attenta analisi del navigation scheme e delle tabelle di accesso, non abbiamo riscontrato la presenza di alcuna ridondanza, non richiedendo quindi di stilare un nuovo e aggiornato schema E/R che rimane tale.

Traslazione dello schema E/R nel modello relazionale

Costruiamo ora, a partire da uno schema entità relazioni in cui non persistono generalizzazioni e ridondanze sugli attributi, uno schema di tipo relazionale. Per la traduzione dei due modelli, ci avvaliamo delle norme e consuetudini per mappare le relazioni, distinguendo le relazioni molti-a-molti da quelle uno-a-molti e uno-a-uno.

AEROPORTO (codice, nome, *città*);

CITTA' (nome, nazione);

TIPO_AEROPLANO (nome, azienda_costruttrice, numero_max_posti, autonomia_di_volo);

ATTERRA_DECOLLA (*codice_aeroporto*, *tipo_aeroplano*);

AEROPLANO (codice, *tipo_aeroplano*, n.posti_effettivi);

COMPAGNIA_AEREA (nome);

UTILIZZATO_DA (*codice_aeroplano, compagnia_aerea*);

VOLO (codice, *compagnia_aerea*);

DIVISO_IN (*codice*, *tipo_classe*);

CLASSE (tipo_classe, prezzo_biglietto);

EFFETUATO_IN (*codice*, *giorno*);

GIORNO_SETTIMANA (giorno);

TRATTA (num_progressivo);

ISTANZA_DI_TRATTA (codice, *numero*, n.posti_disponibili, data);

PRENOTAZIONE (posto_prenotato, nome, cognome, recapito telefonico, *data*);

PARTENZA (*codice_aeroporto*, *numero_progressivo*, orario);

ARRIVO (*codice_aeroporto*, *numero_progressivo*, orario);

Assunzioni e chiavi sul modello relazionale proposto.

Assumiamo che le Primary Key delle relazioni proposte siano sottolineate, ed identificano univocamente la relazione a cui appartengono.

Per quanto riguarda le chiavi esterne, in corsivo, assumiamo che *città* nella relazione aeroporto, si riferisca a *nome* della relazione città.
L'attributo *tipo_aeroplano* nella relazione aeroplano si collega al *nome* nella relazione omonima.

La relazione Atterra_Decolla mira a mettere in evidenza quali tipi di aeroplano possono atterrare o decollare all'interno di uno specifico aeroporto. La relazione perciò, si ricollega rispettivamente ad aeroporto con *codice_aeroporto* e a tipo di aeroplano con *tipo_aeroplano*. Anche la relazione Utilizzato_da, ha un funzionamento analogo. Per ogni *codice_aeroplano* (chiave esterna rispetto Aeroplano) indichiamo la compagnia aerea che lo utilizza (chiave esterna verso *nome* di compagnia aerea).

Per quanto riguarda l'attributo *compagnia_aerea* di volo, possiamo facilmente capire che si collega all'attributo *nome* di compagnia_aerea. Nella relazione Diviso in associamo ad ogni *codice* (chiave esterna verso codice di volo) un *tipo_classe* che si riferirà a tipo_classe nella relazione classe. La coppia di attributi precedentemente enunciati andrà a formare la chiave della relazione.

La relazione effettuato in, lega il *codice* (chiave esterna verso codice di volo) al giorno della settimana in cui viene effettuato, tramite *giorno,* foreign key che si riferisce alla primary key giorno nella relazione Giorno_settimana.

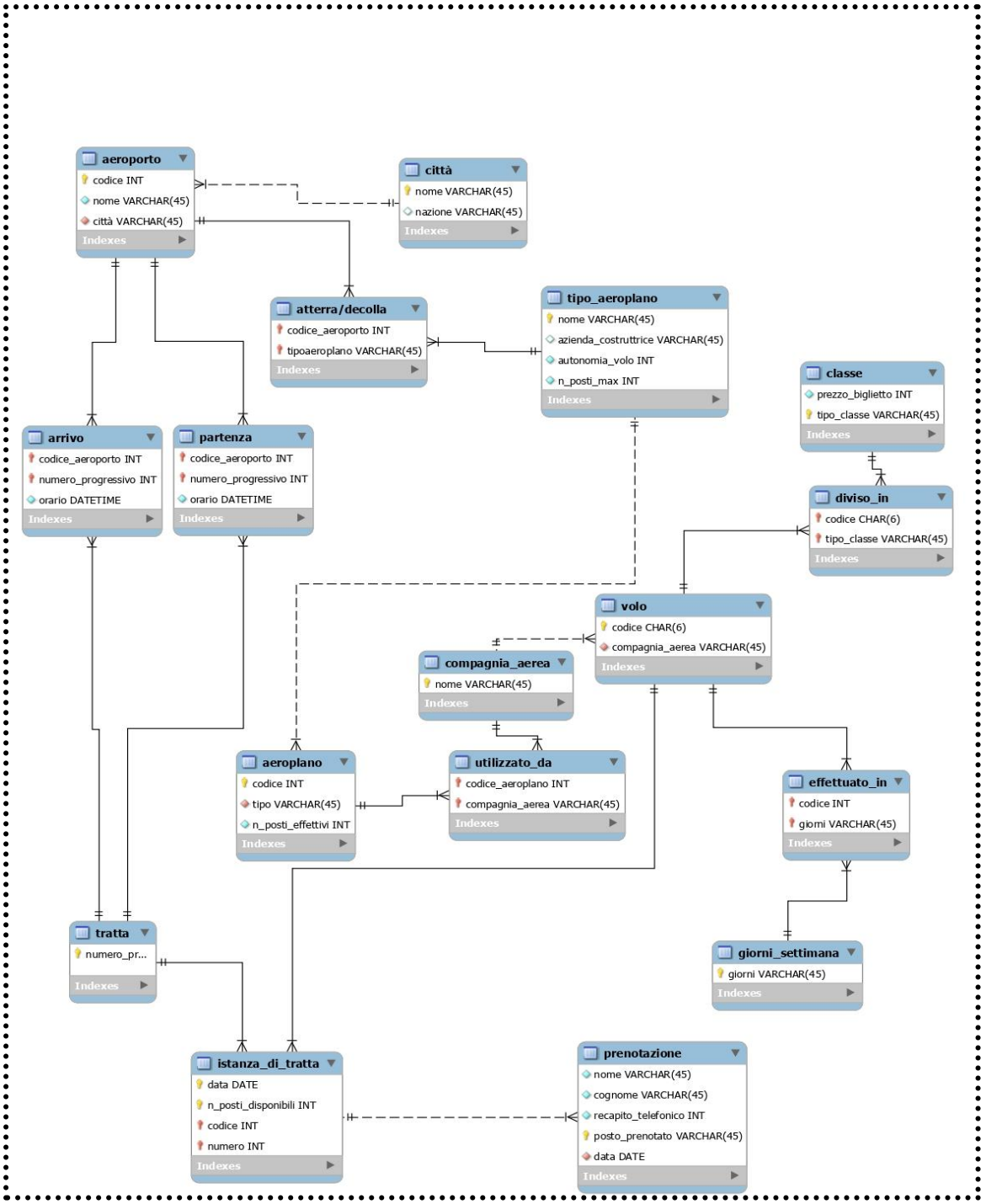
Nella relazione istanza_di_tratta, il *numero* è un riferimento alla chiave primaria num_progressivo di tratta. Nella relazione Prenotazione l'attributo *data*, si riferisce alla data nella relazione istanza di tratta). Nella relazione istanza_di_tratta l'insieme degli attributi andrà a definire la chiave della relazione stessa.

Infine, nelle relazioni Partenza e Arrivo, *codice_aeroporto* è un attributo che si riferisce al codice nella relazione Aeroporto, *numero_progressivo* si riferisce all'omonimo attributo nella relazione tratta.

4. Progettazione fisica

La fase successiva alla creazione del modello relazionale precedentemente proposto, consiste nella implementazione fisica, andremo perciò a definire i codici che ci permetteranno di definire queste relazioni, nel nostro caso specifico utilizzeremo MySQL.

MySQL è un sistema di gestione di database relazionali (RDBMS) open source molto popolare. È utilizzato per archiviare, gestire e organizzare grandi quantità di dati in modo strutturato. Riportiamo la struttura grafica del nostro modello relazionale, implementata con tale software.



Il codice prodotto per la definizione delle seguenti relazioni è visualizzabile all'interno del file *progettazione_fisica_aeroporto.sql*

Di seguito andremo a descrivere alcune delle assunzioni e delle tecniche implementative che abbiamo utilizzato per la definizione delle relazioni.

Forniamo come esempio la creazione della tabella volo.

```
-- -----------------------------------------------------
-- Table `aeroporti software`.`volo`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `aeroporti software`.`volo` (
  `codice` CHAR(6) NOT NULL,
  CONSTRAINT 'codice_volo_valido'  CHECK ('codice' REGEXP '^[A-Z]{2}[0-9]{4}$'),
  `compagnia_aerea` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`codice`),
  INDEX `compagnia_aerea_idx` (`compagnia_aerea` ASC) VISIBLE,
  CONSTRAINT `compagnia_aerea`
    FOREIGN KEY (`compagnia_aerea`)
    REFERENCES `aeroporti software`.`compagnia_aerea` (`nome`)
    ON DELETE SET NULL
    ON UPDATE CASCADE)
```

Indichiamo nella creazione dei campi, NOT NULL gli attributi che in fase di popolamento non possono risultare nulli. Tale scelta è spiegabile dal fatto che gli attributi che non possono assumere valore nullo sono essenziali per il funzionamento delle operazioni. Attributi con un significato secondario, possono risultare NULL al momento dell'inserimento, facciamo riferimento ad esempio all'attributo "nazione" della relazione città.

Per quanto riguarda la creazione dei domini, ci affidiamo inizialmente al controllo tramite la clausola CONSTRAINT. Su alcuni tipi di dato, tramite un CHECK, siamo in grado di verificare che il dato rispetti certe condizioni, ovvero che mantenga un senso implementativo. Un esempio particolare, è rappresentato dal codice di volo, che lo abbiamo assunto come la combinazione di 2 lettere e 4 numeri (es. RA8765). Attraverso la funzione:

```
`codice` CHAR(6) NOT NULL,
CONSTRAINT 'codice_volo_valido'  CHECK ('codice' REGEXP '^[A-Z]{2}[0-9]{4}$'),
```

Possiamo verificare che il dato inserito rispetti tale vincoli.

Altre scelte riguardanti il dominio, sono descritte all'interno del codice, mentre il tipo di dato di alcuni attributi è stato mantenuto di default, come il tipo intero INT, o stringa di lunghezza variabile VARCHAR(45).

Nella definizione delle tabelle, la clausola PRIMARY KEY, indica la chiave primaria che identifica univocamente la relazione, essa può essere singola, oppure la combinazione di più

attributi. Le scelte di selezione delle chiavi primarie rispecchiano le scelte effettuate durante la progettazione logica.

Per le chiavi esterne, è necessario indicare l'attributo che si riferisce alla tabella esterna e mediante la clausola REFERENCES, la tabella di riferimento con l'attributo prescelto. E' necessario che l'attributo di riferimento sia chiave o parte della chiave nella medesima relazione. Tramite ON DELETE/UPDATE, possiamo scegliere le politiche di aggiornamento e di cancellazione sulle tabelle esterne. Le politiche utilizzate, ricadono su CASCADE (= propaga la cancellazione/aggiornamento), SET NULL (= imposta a NULL il valore).

```
CONSTRAINT `codice`
  FOREIGN KEY (`codice`)
  REFERENCES `aeroporti software`.`volo` (`codice`)
  ON DELETE CASCADE
  ON UPDATE CASCADE,
```

In conclusione, gli indici scelti per la definizione delle relazioni, sono stati implementati nel codice. Definiamo l'indice tramite la clausola INDEX, indicando il nome dell'indice, l'attributo di interesse e la sua visibilità.

```
INDEX `codice_idx` (`codice` ASC) VISIBLE,
INDEX `numero_idx` (`numero` ASC) VISIBLE,
```

5. Implementazione

Il codice relativo alla fase di implementazione è visibile nel file *implementazione_aeroporto.sql*

Definzione di 2 trigger risultanti dalla progettazione.

In SQL, un trigger è un tipo di oggetto di database che definisce un insieme di azioni da eseguire automaticamente quando si verifica un certo evento all'interno del database. Questi eventi possono includere modifiche ai dati, come l'inserimento, l'aggiornamento o la cancellazione di record in una tabella.

Creiamo un trigger, che controlla che l'aeroplano abbia effettivamente meno posti posti effettivi del numero massimo dei posti disponibili del tipo di aeroplano che lo caratterizza.

```
CREATE TRIGGER verifica_posti_effettivi
BEFORE INSERT OR UPDATE ON AEROPLANO
FOR EACH ROW
BEGIN
    DECLARE max_posti INT;

    SELECT numero_max_posti INTO max_posti
    FROM TIPO_AEROPLANO
    WHERE nome = NEW.tipo_aeroplano;

    IF NEW.n_posti_effettivi > max_posti THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Errore: Numero posti effettivi supera il numero massimo di posti consentito';
    END IF;
END;//
```

Questo controllo, viene effettuato in seguito all'inserimento o all'aggiornamento sulla tabella aeroplano. Dichiariamo una variabile iniziale max_posti di tipo INT e inseriamo all'interno di essa il risultato di una query sulla relazione tipo di aeroplano. Tramite un IF, possiamo segnalare un errore (== raise notice) se il numero di posti effettivi supera il numero massimo di posti consentito.

Il secondo trigger che implementiamo, si attiene a verificare un requisito della consegna in natural language, ovvero che i biglietti di un dato volo relativi alla stessa classe abbiano tutti lo stesso prezzo.

```
CREATE TRIGGER verifica_prezzo_biglietti
BEFORE INSERT OR UPDATE ON DIVISO_IN
FOR EACH ROW
BEGIN
    DECLARE classe_prezzo DECIMAL(10, 2);

    SELECT prezzo_biglietto INTO classe_prezzo
    FROM CLASSE
    WHERE tipo_classe = NEW.tipo_classe;

    IF EXISTS (
        SELECT *
        FROM DIVISO_IN AS d JOIN CLASSE AS c ON d.tipo_classe = c.tipo_classe
        WHERE d.codice = NEW.codice
          AND c.prezzo_biglietto <> classe_prezzo
    ) THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Errore: I biglietti della stessa classe devono avere lo stesso prezzo';
    END IF;
END;
```

Il funzionamento di questo trigger è analogo al precedente, solamente che interessa le tabelle classe_prezzo e tipo_ di_classe.

Popolamento della base di dati.

Per popolare la nostra base di dati, utilizziamo la clausola INSERT, per ogni tabella ci limitiamo ad inserire alcuni record a mano, seguendo coerentemente il tipo di dato che abbiamo assunto durante la progettazione fisica. La sintassi è la seguente, prendiamo come esempio il popolamento della tabella aeroporto.

```sql
INSERT INTO aeroporto (codice, nome, città)
VALUES ('00001', 'Malpensa airport', 'Milano'),
       ('00002', 'Fiumicino airport', 'Roma'),
       ('00003', 'Jfk airport', 'New York City'),
       ('00004', 'Frankfurt airport', 'Frankfurt'),
       ('00005', 'Ashford airport', 'London'),
       ('00006', 'Linate Airport', 'Milano'),
       ('00007', 'Heathrow Airport', 'London'),
       ('00008', 'Charles de Gaulle Airport', 'Paris'),
       ('00009', 'Dubai International Airport', 'Dubai'),
       ('00010', 'Los Angeles International Airport', 'LA'),
       ('00011', 'Sydney Kingsford Smith Airport', 'Sydney'),
       ('00012', 'Hong Kong Airport', 'Hong Kong');
```

Definizione di 3 query significative (operazioni frequenti).

Nella realizzazione delle query significative, prendiamo come linea guida alcune delle operazioni più frequenti che abbiamo delineato in precedenza. Le query che andremo a definire saranno utili per svolgere un'analisi statistica sulla nostra base dati.

Visualizziamo, per ogni aeroporto i tipi di aeroplano che possono atterrare o decollare. I dati ottenuti saranno poi plottati con l'ausilio del software *R*.

```sql
SELECT A.codice, A.nome, AD.tipo_aeroplano
FROM aeroporto as A, atterra_decolla as AD
WHERE A.codice = AD.codice_aeroporto
GROUP BY A.codice, A.nome, AD.tipo_aeroplano
ORDER BY A.codice, AD.tipo_aeroplano
```

Determiniamo ora, quali tipi di aeroplani hanno un rapporto migliore di numero massimo di posti e numero effettivi. Questa query, seppur non è una delle operazioni più frequenti, è di grande interesse per l'utilizzatore effettivo della base di dati.

```sql
SELECT TA.nome, AP.codice, TA.n_posti_max/AP.n_posti_effettivi AS rapporto
FROM tipo_aeroplano AS TA, Aeroplano AS AP
WHERE TA.nome = AP.tipo_aeroplano
ORDER BY rapporto DESC
```

Infine, ci chiediamo in quali giorni della settimana vengono effettuati più voli, per potere capire l'affluenza di informazioni effettive a cui il nostro sistema dovrà far fronte.

```
SELECT GS.giorni AS giorni_settimana, COUNT(EI.codice) AS numero_voli
FROM giorni_settimana GS
LEFT JOIN effettuato_in EI ON GS.giorni = EI.giorni
GROUP BY GS.giorni
ORDER BY numero_voli DESC;
```

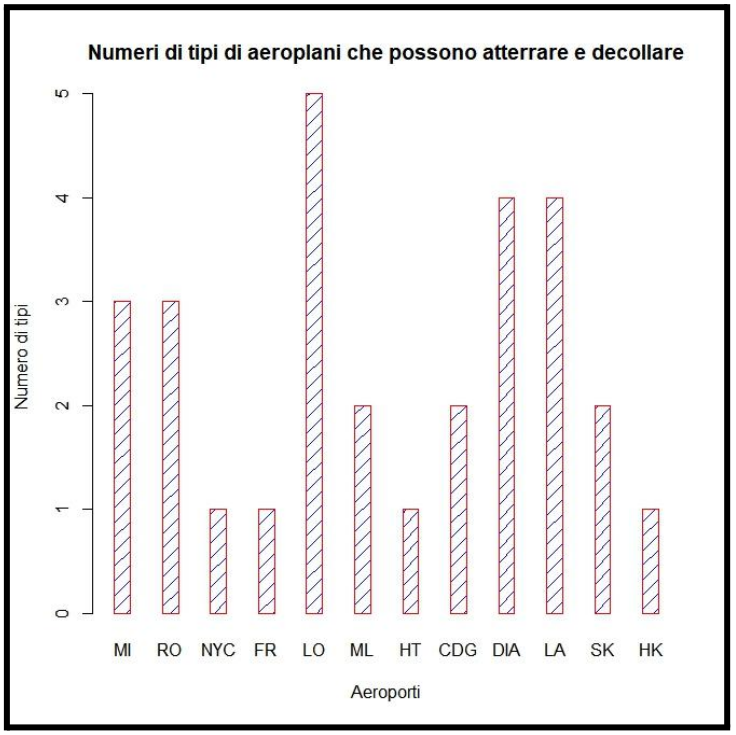
6. Analisi dei dati in R

Lo step finale del nostro progetto, consiste nella presentazione dei dati. Come precedentemente delineato, realizziamo una semplice analisi statistica rivolta alle 3 query.

Per poter fare ciò, è necessario interfacciare *mySQL* con *R*, un ambiente di sviluppo utilizzato principalmente per l'analisi statistica, la visualizzazione dei dati e la creazione di modelli statistici e grafici.

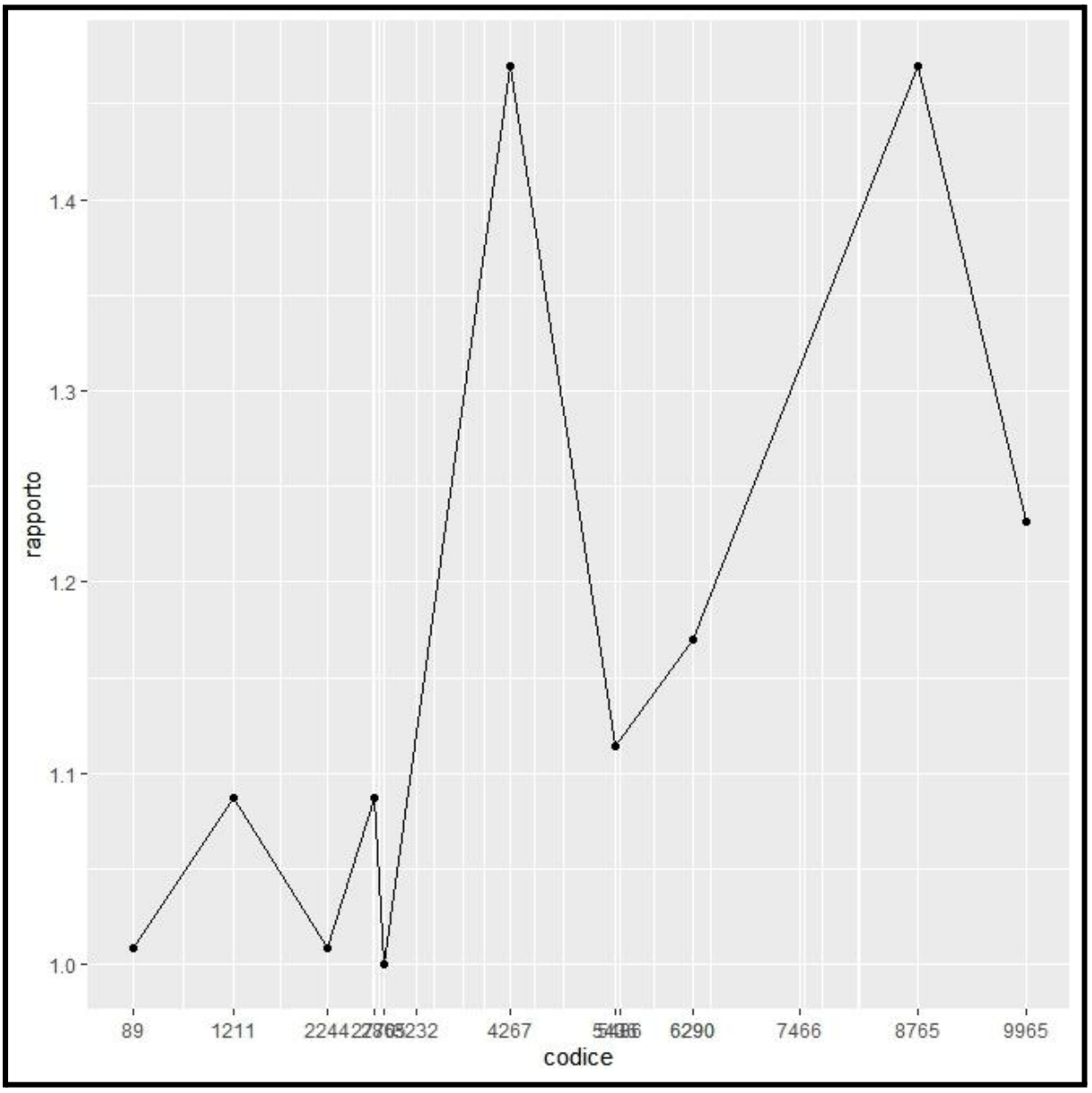
La parte di codice relativa a questa analisi statistica è visualizzabile all'interno del file *Presentazione_dati_aeroporto.R*

Visualizza, per ogni aeroporto i tipi di aeroplano che vi possono decollare.



Il grafico sopra riportato, rappresenta un barplot (grafico a barre). Indichiamo nell'asse x l'aeroporto (città), mentre nell'asse y, i numeri di tipi di aeroplani che possono atterrare o decollare. Notiamo che Londra può accogliere più tipi di aeroplani diversi rispetto ad altre città come Milano, Roma, LA o Dubai.

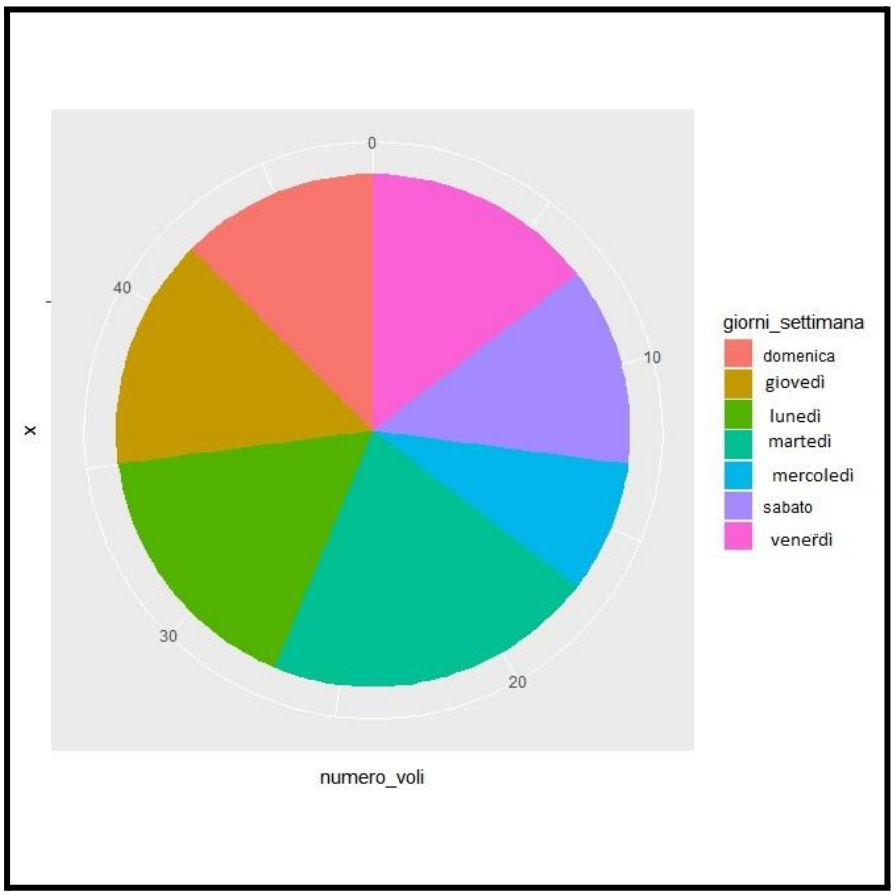
Determinare quali tipi di aeroplani hanno un rapporto di numero massimo di posti e posti effettivi migliore.



Rappresentiamo l'oggetto di questa analisi in un grafico di tipo scatterplot. Nell'asse x, per comodità di ordinamento, indichiamo il codice dell'aeroplano, mentre nell'asse y indichiamo il rapporto tra il numero massimo di posti messi a disposizione dal tipo di aeroplano e il numero di posti effettivi che l'aeroplano utilizza.

Il rapporto ottenuto, è un numero positivo maggiore o uguale a 1 (caso limite in cui i posti massimi siano effettivamente quelli messi a disposizione).
Perciò, l'aeroplano con codice 2865, con 78 posti effettivi e 78 posti messi a disposizione (avente rapporto di 1, picco minimo del grafico) è quello che gode di maggior efficienza.

Determinare quali sono i giorni della settimana dove vengono effettuati più voli.



Riportiamo, attraverso un grafico a torta, i giorni della settimana in cui vengono offerti più voli.
I 7 settori che rappresentano i giorni, sono accompagnati dalla distribuzione percentuale approssimativa.

Possiamo notare che la distribuzione è apparentemente equa, con un lieve calo rappresentato dal mercoledì, giorno infrasettimanale in cui vengono offerti mediamente meno voli. Il martedì offre più voli rispetto agli altri giorni.

Conclusioni

In questa relazione, abbiamo esaminato il processo di sviluppo di un database, affrontando diverse fasi chiave: dalla fase iniziale di pianificazione alla progettazione concettuale e logica, per poi passare allo sviluppo in SQL e all'analisi dei dati utilizzando R.

Nella fase iniziale, abbiamo identificato l'obiettivo principale del nostro database e definito i requisiti fondamentali. Questa fase ci ha fornito una panoramica chiara delle esigenze e degli scopi del database, aprendo la strada per le fasi successive.

Successivamente, nella progettazione concettuale, abbiamo creato un modello concettuale che rappresenta le entità, le relazioni e gli attributi chiave del nostro dominio di dati. Questo modello ha catturato l'essenza del sistema e ci ha guidato nella progettazione logica.

La progettazione logica ha comportato la trasformazione del modello concettuale in uno schema relazionale, dove ogni entità e relazione è stata rappresentata come una tabella con attributi appropriati. Abbiamo definito chiavi primarie, chiavi esterne e vincoli per garantire l'integrità dei dati.

Successivamente, ci siamo dedicati allo sviluppo del database utilizzando *SQL*, un linguaggio di interrogazione e gestione dei dati. Abbiamo creato tabelle, popolato il database con dati di esempio e implementato query complesse per illustrare le capacità di recupero dei dati.

Infine, abbiamo utilizzato *R* per condurre un'analisi dei dati sul nostro database. Abbiamo importato i dati nel nostro ambiente, creato visualizzazioni significative come grafici e diagrammi, e applicato tecniche statistiche per rivelare tendenze e relazioni nei dati.

In conclusione, il processo di sviluppo di un database è un percorso articolato che abbraccia la pianificazione, la progettazione, l'implementazione e l'analisi. Ogni fase contribuisce alla costruzione di un sistema robusto e significativo. Attraverso la combinazione di abilità di progettazione concettuale e logica, conoscenza di *SQL* e capacità di analisi dei dati con *R*, siamo stati in grado di creare un database funzionale e condurre un'analisi approfondita dei dati in esso contenuti.