UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

AGENZIA VIAGGI DALE BOCA S.R.L.

PROGETTAZIONE DI UNA BASE DI DATI PER LA GESTIONE DI UN'AZIENDA DI AGENZIA VIAGGI

CORSO BASI DI DATI 2022/2023 **PROFESSORE: MOSCATO VINCENZO**

PROGETTO REALIZZATO DA:

FIORILLO ANGELO N46005717
GRECO FABIO N46005205

Agenzia viaggi con sistema qualità certificato ISO 9001

INDICE

INTRODUZIONE

1.1 SCOPO

Lo scopo verte nel realizzare una base di dati per l'agenzia di viaggio DALE BOCA S.R.L. al fine di gestire le varie prenotazione, in base ad alcuni pacchetti costituiti da soggiorni e spostamenti, considerando: in primis gli utenti che intendono usufruire dei pacchetti; vari mezzi di traporto necessari a raggiungere la meta desiderata; alberghi con la scelta delle camere; gestire i pagamenti; verificare che non vi siano errori nella prenotazione del viaggio.

1.2 PANORAMA E DATI GENERALI

L'agenzia viaggi deve gestire le varie prenotazioni, in base ai seguenti dati:

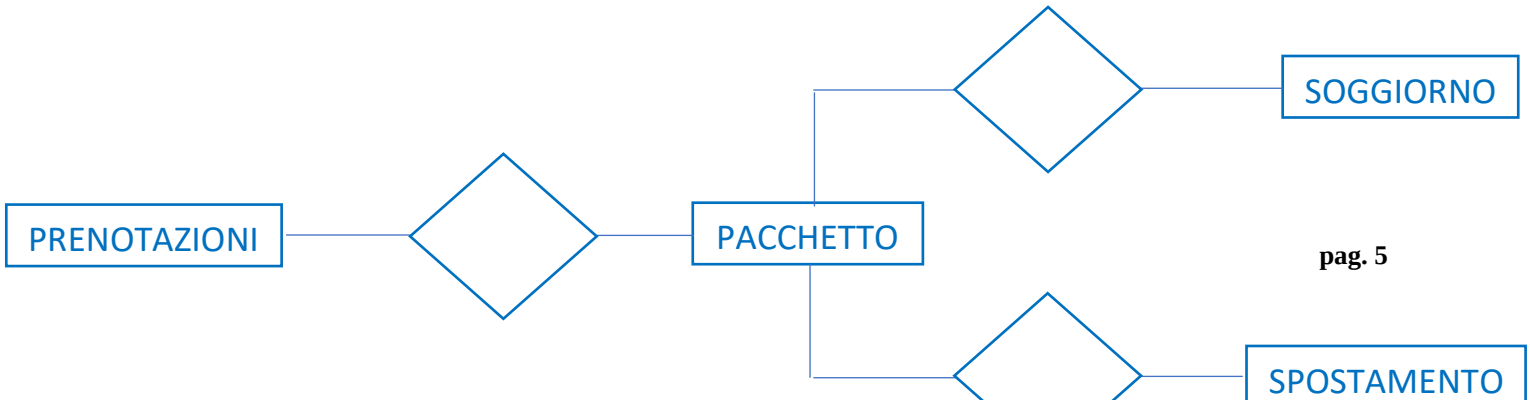
- *Pacchetti* offerti, indentificati da un codice univoco, inoltre essi sono pensati per un numero preciso di persone.
- *Albergo*, identificato da un codice univoco e dal nome, selezionato in base alla tipologia, considerando le date di check-in e check-out ed infine il costo giornaliero.
- le combinazioni possibili tra *soggiorni* e *spostamenti*, dove sono caratterizzati in primis da un codice ambedue e successivamente da: coordinate GPS di partenza ed arrivo; data di arrivo e partenza comprensiva di orari; luogo di partenza e arrivo.
- *Camere* prenotate in base alla tipologia, numero di occupanti, numero di camera e dai servizi che offrono, come wi-fi, smart tv, frigo-bar etc.
- *Prenotazioni* identificate da un codice univoco, numero di ospiti, date di partenza ed arrivo e prezzo complessivo.
- *Utenti* identificati dal proprio codice fiscale, nome e cognome.
- *Mezzi di trasporto* caratterizzati dal tipo, costo, tempo di percorrenza, a l'insieme dei messi necessari al trasporto.
- *Metodo di pagamento* si registra il codice della transazione che può essere effettuato attraverso carta di credito/debito o bonifico bancario (a scelta degli acquirenti):
    - Qualora si scegliesse la carta di credito/debito si devono registrare le informazioni relative all'intestatario della carta, la scadenza, il circuito e il numero;
    - Nel caso, invece, che venga scelto il bonifico, va registrata la data di emissione.

L’agenzia dispone di:

- 20 pacchetti;
- 20 spostamenti possibili;
- 15 mezzi di trasporto;
- 20 alberghi;
- 100 camere libere;
- 10 servizi offerti nelle camere;
- 2000 possibili prenotazioni;
- 2500 possibili utenti;
- 1000 possibili pagamenti con carta di credito/debito;
- 1000 possibili pagamenti con bonifico bancario.

Progettazione relazionale della base di dati

Nel primo step della progettazione di una base di dati, si presenta il problema di traduzione delle varie problematiche del mondo reale in uno **schema concettuale** che abbia il fine di essere maggiormente comprensibile da chiunque, senza considerare l’implementazione nel database. Dunque in questa fase di va a delineare una prima struttura, in modo molto superficiale, attraverso uno schema preciso: **schema E/R (modello Entità Relazione)**. Tale modello è usato per definire le entità (insieme di occorrenze di entità descritte ognuna dalla stessa proprietà comune) da modellare con tutte le relative caratterizzazioni, le varie interazioni che hanno le entità tra esse e le cardinalità fra quest’ultime. Nel modello E/R le entità vengono rappresentate con un rettangolo contenente il suo nome e, ognuna di esse, è caratterizzata da attributi, i quali sono individuati da linee terminate da cerchi con i relativi nomi e sono riferiti ad una singola entità. Le linee possono essere completare con la *cardinalità dell’attributo* e se il cerchio finale è ripieno esso indica che è l’identificatore dell’entità ed è univoco per ogni entità.
Ora, attraverso diverse fasi, si andrà ad arricchire, semplificare e rendere più raffinato lo schema fino ad avere una versione finale.



Come si può vedere, si è partiti dalle entità più semplici da individuare, per iniziare a delineare passo dopo passo quello che sarà lo schema E/R finale. Ora si andrà ad aggiungere le restanti entità con tutti gli attributi ad esse associati.

2.1 Progettazione concettuale base di dati:

SCHEMA E/R COMPLETO

Un utente può scegliere un metodo di pagamento diverso per ogni prenotazione, in particolare sceglierà se optare per un pagamento con carta di credito oppure effettuare un bonifico; quindi, bonifico e carta di credito saranno figlie dell'entità padre "metodo di pagamento". Questa gerarchia risulta essere totale e disgiunta in quanto non è possibile saldare il conto con un ulteriore metodo di pagamento, né farlo pagando in parte con carta di credito e in parte effettuando un bonifico.

Ogni prenotazione sarà effettuata da un solo utente e relativa ad un solo pacchetto, inoltre dovrà avere la registrazione esplicita del nominativo di ciascuno degli altri utenti viaggiatori, quindi l'entità prenotazione avrà un attributo composto multi valore che contenga l'anagrafica degli altri viaggiatori.

Ogni pacchetto può risultare in più prenotazioni, inoltre risulta essere una composizione di spostamenti e soggiorni.

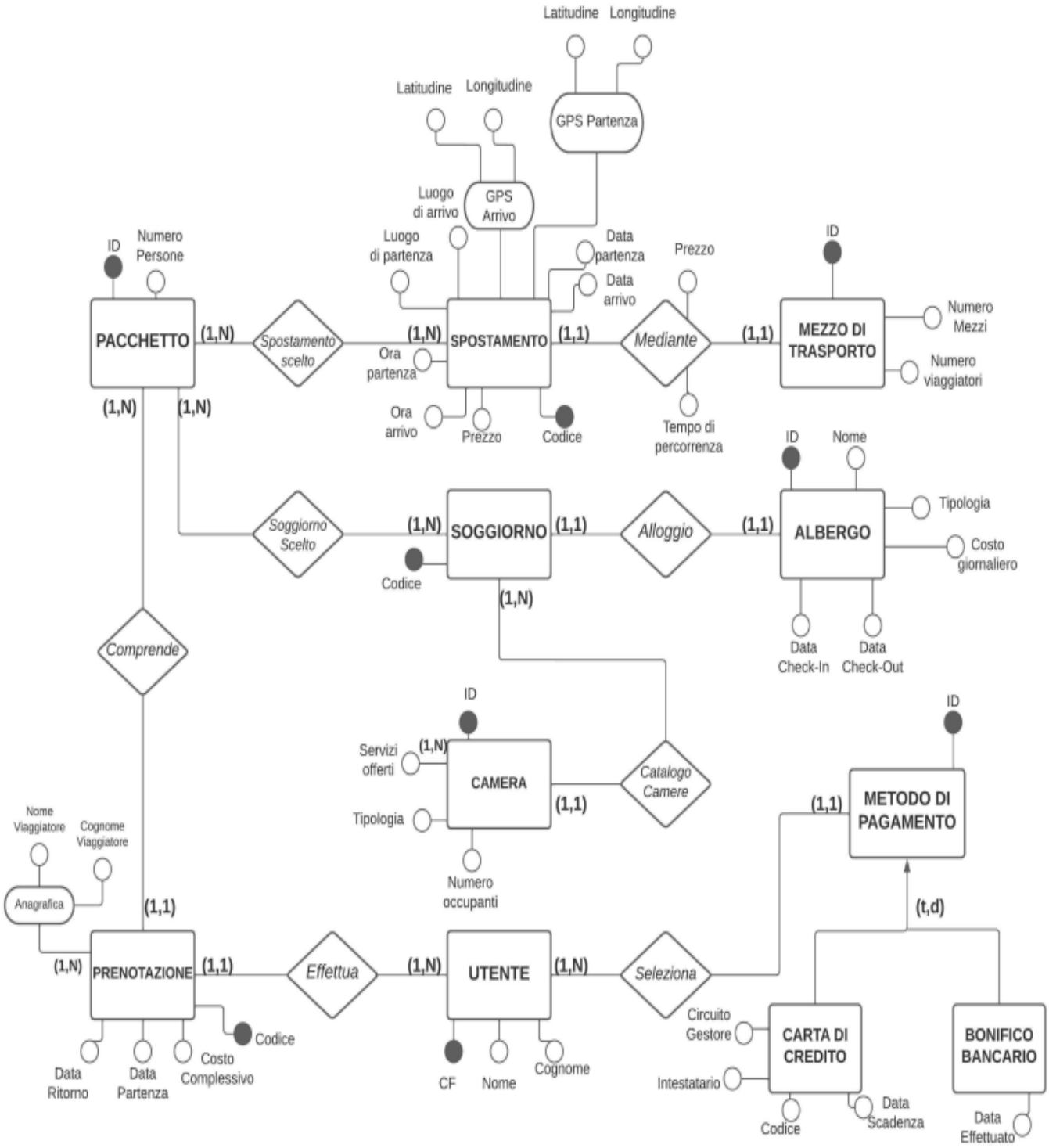
Ogni spostamento può essere relativo a più pacchetti, ma viene effettuato da un solo mezzo di trasporto, quindi fra le due entità avremo un'associazione uno a uno. Fra i suoi attributi uno spostamento tiene traccia delle coordinate GPS del luogo di partenza e di arrivo, cioè si compone di latitudine e longitudine; la scelta del nostro team è stata di trattare questo attributo come composto, quindi di trattare latitudine e longitudine con un tipo stringa. Ogni soggiorno può essere relativo a più pacchetti, ma ad un solo albergo e relativamente a questo albergo possiamo scegliere una o più camere, motivo per il quale l'associazione fra le due entità è un'associazione uno a molti.

Per la maggior parte delle entità presenti il team ha ritenuto opportuno introdurre come chiavi primarie dei codici numerici (ID), in quanto gli attributi delle rispettive entità non presentavano chiavi che ottimizzassero l'accesso ai dati.

Sempre per quanto concerne le chiavi primarie è stata fatta un'ulteriore assunzione, spesso sottovalutata: CF in Utente, che teoricamente può variare nel tempo. Essendo di per sé una pratica abbastanza desueta, quella di cambiare nome (e quindi codice fiscale), abbiamo considerato che in un arco di tempo relativamente breve - data

prenotazione viaggio e relativa data di partenza - ciò non potesse avvenire, a maggior ragione che un'esperienza come un viaggio presuppone, prima della partenza effettiva, sensazioni prettamente positive lontane da problematiche come quella di cambiare nome. Per questo l'abbiamo considerata a tutti gli effetti la miglior candidata per la chiave primaria ai fini della prenotazione.

Di seguito è riportato lo schema concettuale.



2.2 Progettazione logica

A partire dal modello concettuale, la fase successiva è quella della progettazione logica, che si compone di due fasi:

1) *TRASFORMAZIONE*: modificare o eliminare costrutti che non sono traducibili in tabelle.
2) *TRADUZIONE*: passaggio dal modello E/R alla Base Dati relazionale (insieme di relazioni e vincoli).

Fase di Trasformazione

Gli attributi composti e multi valore non sono immediatamente traducibili nel modello logico. Un attributo multi valore deve essere eliminato introducendo una nuova entità, mentre un attributo composto può essere semplificato sciogliendosi in più attributi semplici, quante sono le sue componenti. Nel modello ER ci sono più attributi di questo tipo da trattare:

- L'attributo composto GPS arrivo di Spostamento: si scioglie in nei due attributi semplici Latitudine arrivo, Longitudine arrivo;

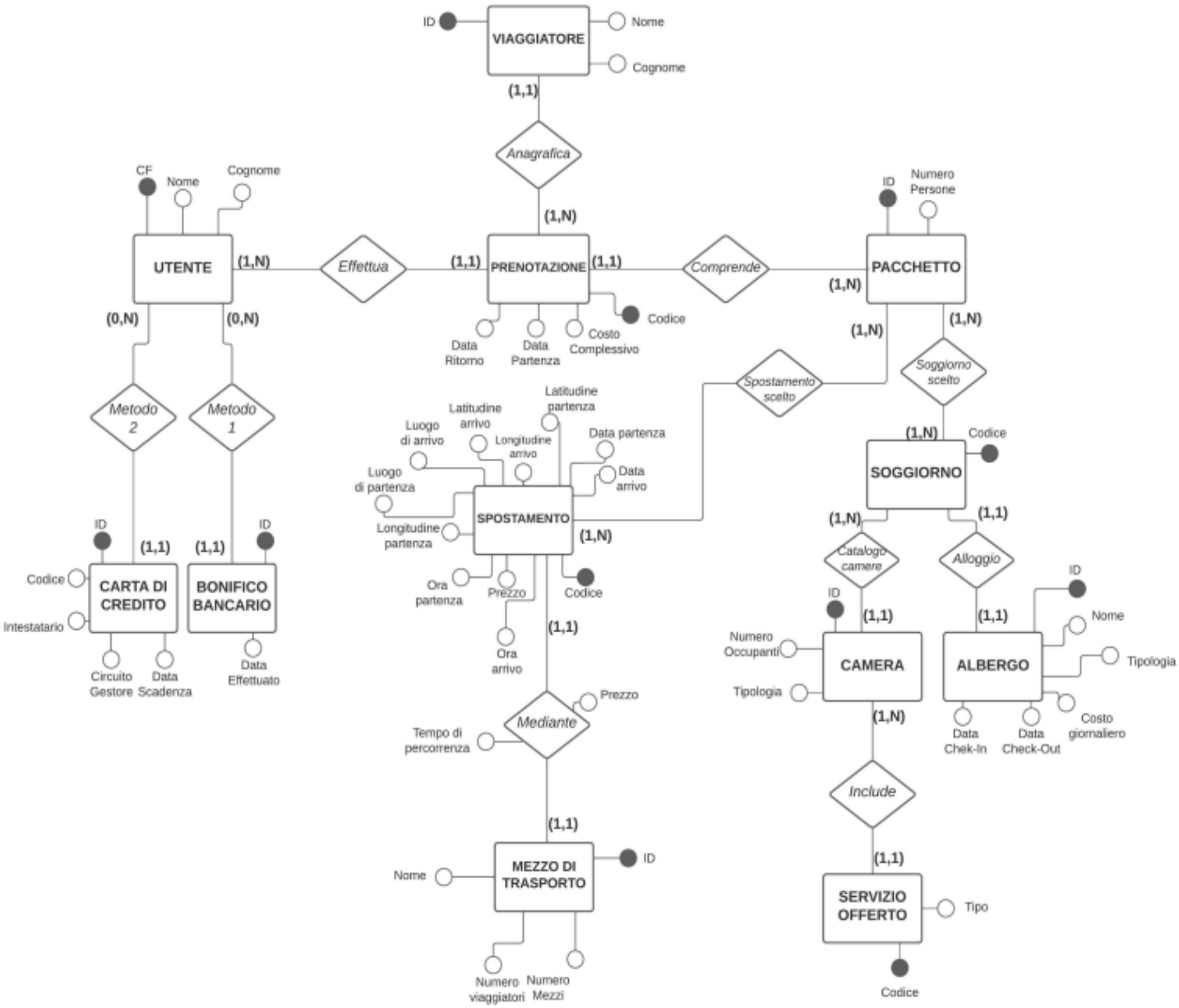
- L'attributo composto GPS partenza di Spostamento: si scioglie in nei due attributi semplici Latitudine partenza, Longitudine partenza;

- L'attributo multivalore Servizi_offerti di Camera: diventa una nuova entità, con un proprio identificativo e un tipo come attributo;

- L'attributo composto multivalore Anagrafica di Viaggiatori: diventa una nuova entità Viaggiatori, con un proprio identificativo e i propri attributi.

Inoltre lo schema presenta la generalizzazione Metodo di pagamento con le sue due specializzazioni Carta di credito e Bonifico Bancario. Questa non può essere rappresentata nel modello logico relazionale, in quanto non esiste un costrutto analogo. Essendo la generalizzazione totale e disgiunta, la scelta consigliata è la seguente: *accorpamento delle superclasse nelle sottoclassi*. Abbiamo quindi eliminato l'entità padre, le figlie quindi ereditano il suo identificatore e la relazione a cui l'entità padre partecipa, cioè quella con l'entità utenti; inoltre abbiamo reso la partecipazione alle entità figlie opzionali, in quanto essendo la generalizzazione disgiunta avrò sempre occorrenze in una delle entità e non nell'altra.

Di seguito lo schema opportunamente trasformato.



2.3 Fase di trasformazioni nello schema E/R finale

Per la fase di traduzione, applichiamo le seguenti regole:

1) Una entità dello schema concettuale si traduce in una relazione dello schema logico avente lo stesso nome (ma al plurale) e gli stessi attributi dell'entità ed avente per chiave primaria il suo identificatore. Quindi trasformiamo tutte le entità.

   Ora traduzione delle associazioni: è possibile se sono binarie e dipende dal rapporto di cardinalità.

2) Relazioni uno a uno:
   Per la relazione *Mediante* e la relazione *Alloggio* il team ha deciso di tradurre aggiungendo alla relazione che traduce una delle due entità (Mezzo di trasporto, Soggiorni) gli attributi dell'associazione e l'identificatore dell'altra entità. Esiste un vincolo di unicità sul nuovo attributo aggiunto (***non vi era presente tale condizione con il libro di Atzeni, il team ringrazia il prof Moscato a riguardo***) ed un vincolo di integrità referenziale fra quest'ultimo e il corrispondente attributo dell'altra entità:

   -MEZZI_DI_TRASPORTO (ID, NumeroViaggiatori, NumeroMezzi, TempoPercorrenza, Prezzo, ID_Spostamento*:SPOSTAMENTI)
   -SOGGIORNI (Codice, ID_Albergo*:ALBERGHI)

3) Relazioni uno a molti:

   Per le relazioni *Anagrafica, Effettua, Comprende, Metodo 1, Metodo 2*, Include si è deciso di aggiungere alla relazione che traduce l'entità dal lato uno gli attributi dell'associazione e l'identificatore dell'entità lato molti, ridenominato. Esiste un vincolo di integrità referenziale tra questo attributo e il corrispondente attributo dell'entità dal lato molti:

   -VIAGGIATORI ID, Nome, Cognome, Prenotazione:PRENOTAZIONI)
   -PRENOTAZIONI(Codice, CostoComplessivo, DataPartenza, DataRitorno, Pacchetto:PACCHETTI, Utente:UTENTI)
   - CARTE_DI_CREDITO(ID, CircuitoGestore, Intestatario, Codice*, DataScadenza, Utente:UTENTI)
   - BONIFICI_BANCARI(ID, DataEffettuata, Utente:UTENTI)
   - SERVIZI_OFFERTI(Codice, Tipo, Camera:CAMERE)

4) Relazioni molti a molti
   Per le relazioni *Soggiorni scelti* e *Spostamenti scelti* si è scelto di tradurre l'associazione in una relazione dello schema logico avente lo stesso nome(ma al plurale) dell'associazione e per attributi gli identificatori delle entità coinvolte. Gli identificatori delle entità coinvolte formano la chiave primaria e, ognuno di essi, ha un vincolo di chiave esterna con il corrispondente attributo dell'entità da cui discende:
   SPOSTAMENTI_SCELTI(ID_Pacchetto:PACCHETTI, ID_Spostamento:SPOSTAMENTI)

2.4 Fase di traduzione finale

Il risultato finale, a valle di tutte le considerazioni fatte, è il seguente:
PACCHETTI(ID, NumeroPersone)

SPOSTAMENTI(ID, LuogoPartenza, LuogoArrivo, LatitudineArrivo, LongitudineArrivo, LatitudinePartenza, LongitudinePartenza, DataPartenza, DataArrivo, OraArrivo, OraPartenza)
SPOSTAMENTI_SCELTI(ID_Pacchetto:PACCHETTI, ID_Spostamento:SPOSTAMENTI)
SOGGIORNI_SCELTI(ID_Pacchetto: PACCHETTI, Codice_Soggiorno:SOGGIORNI)
MEZZI_DI_TRASPORTO(ID, NumeroViaggiatori, NumeroMezzi, TempoPercorrenza, Prezzo, ID_Spostamento:SPOSTAMENTI)
SOGGIORNI(Codice, ID_Albergo:ALBERGHI)
ALBERGHI(ID, Nome, Tipologia, DataCheckIn, DataCheckOut, CostoGiornaliero)
CAMERE(ID, Tipologia, NumeroOccupanti, Soggiorno:SOGGIORNI)
SERVIZI_OFFERTI(Codice, Tipo, Camera:CAMERE)
PRENOTAZIONI(Codice, CostoComplessivo, DataPartenza, DataRitorno, Pacchetto:PACCHETTI, Utente:UTENTI)
VIAGGIATORI(ID, Nome, Cognome, Prenotazione:PRENOTAZIONI)
UTENTI(CF, Nome, Cognome)
CARTE_DI_CREDITO(ID, CircuitoGestore, Intestatario, Codice, DataScadenza, Utente:UTENTI)
BONIFICI_BANCARI(ID, DataEffettuata, Utente:UTENTI)

DIMENSIONAMENTO

3.1 Informazioni e descrizioni dei dati

Una volta conclusa la fase di progettazione logica e, dopo aver elencato tutte le informazione che saranno presenti all'interno della nostra base di dati, ci andremo a focalizzare sullo specificare i vari tipi di dato che abbiamo utilizzato nellacreazione delle tabelle:

- **VARCHAR2(n)**: utilizzato per gli attributi di tipo testuale a lunghezza variabile chesono più propensi a. Tale vincolo prevede in ORACLE una collocazione di n byte per n caratteri testuali entro la dimensione dei 4000 byte per valore. Rispetto al vincolo VARCHAR(n) in ORACLE è più performante;
- **CHAR**: utilizzato per i codici fiscali con un valore di 16 bit (2 byte), in quanto valori ben definiti, e per specificare l'ora.
- **DATE**: utilizzato per attribuire il valore di tipo data. Esso prevede in ORACLE una collocazione di7 byte.
- **INTEGER**: utilizzato, invece per la gestione dei prezzi, degli ID e codici identificativi, con un valore di 4 byte.
- **FLOAT**: utilizzato per la rappresentazione dei numeri a virgola mobile positivi e negativi, compresi tra 2.23^{-308} e 1.79^{308} per i valori positivi e -2.23^{-308} e -1.79^{308} per i valori negativi. Inoltre ha una precisione di 15 cifre.

- **NUMBER**: utilizzato per attribuire un valore ben definito all'attributo. Se non si specifica il valore, esso sarà associato a DECFLOAT(16), se viene specificato come NUMBER(p), sarà DECIMAL(p), oppure se NUMBER(p,s) sarà DECIMAL(p,s).

3.2 Tabelle illustrative

Date le informazioni sulla mole di dati e successivamente, scelti i tipi degli attributi per l'implementazione della base di dati da noi creata, è possibile effettuare una stima della quantità di memoria che effettivamente sarà occupata dalle tabelle nella memoria centrale.

PACCHETTI			
Attributo	Tipo	Byte	Occorrenza:20
ID	INTEGER	4	0.08KB
Num_Persone	NUMBER(4)	4	0.08KB
STORAGE	-	-	-
Totale spazio occupato			**0.16KB**

SPOSTAMENTI			
Attributo	Tipo	Byte	Occorrenza:20
ID	INTEGER	4	0.08KB
Luogo_partenza	VARCHAR2(100)	100	2KB
Orario_partenza	CHAR (8)	8	0.16KB
Luogo_arrivo	VARCHAR2(100)	100	2KB
Orario_arrivo	CHAR (8)	8	0.16KB
Latitudine_partenza	CHAR (15)	15	0.3KB
Longitudine_partenza	CHAR (15)	15	0.3KB
Latitudine_arrivo	CHAR (15)	15	0.3KB
Longitudine_arrivo	CHAR (15)	15	0.3KB
Data_arrivo	DATE	7	0.14KB
Data_partenza	DATE	7	0.14KB
STORAGE	-	-	-
Totale spazio occupato			**5.88KB**

ALBERGHI			
Attributo	Tipo	Byte	Occorrenza:20
Id_albergo	INTEGER	4	0.08KB
Tipologia	VARCHAR2(100)	100	2KB
Costo	FLOAT	8	0.16KB
Nome	VARCHAR2(100)	100	2KB
Data_checkIn	DATE	7	0.14KB

Data_checkOut	DATE	7	0.14KB
STORAGE	-	-	-
Totale spazio occupato			**4.52KB**

SOGGIORNI			
Attributo	Tipo	Byte	Occorrenza:20
Id_soggiorno	CHAR(5)	5	0.1KB
Albergo	INTEGER	4	0.08KB
STORAGE	-	-	-
Totale spazio occupato			**0.18KB**

SPOSTAMENTI_SCELTI			
Attributo	Tipo	Byte	Occorrenza:20
Id_pacchetto	INTEGER	4	0.08KB
Id_spostamento	INTEGER	4	0.08KB
STORAGE	-	-	-
Totale spazio occupato			**0.16KB**

SOGGIORNI_SCELTI			
Attributo	Tipo	Byte	Occorrenza:20
Id_pacchetto	INTEGER	4	0.08KB
Codice_soggiorno	CHAR(5)	5	0.1KB
STORAGE	-	-	-
Totale spazio occupato			**0.18KB**

MEZZI_DI_TRASPOSRTO			
Attributo	Tipo	Byte	Occorrenza:15
Id_mezzo	INTEGER	4	0.06KB
Num_viaggiatori	NUMBER(5)	4	0.06KB
Num_mezzi	NUMBER(2)	3	0.045KB
Tempo_percorrenza	CHAR(5)	5	0.075KB
Prezzo	FLOAT	8	0.12KB
Id_spostamento	INTEGER	4	0.06KB
STORAGE	-	-	-
Totale spazio occupato			**0.42KB**

CAMERE			
Attributo	Tipo	Byte	Occorrenza:100
ID	INTEGER	4	0.4KB
Tipologia	VARCHAR2(100)	100	10KB

Numero_occupanti	INTEGER	4	0.4KB
Soggiorno	CHAR (5)	5	0.5KB
STORAGE	-	-	-
Totale spazio occupato			**11.3KB**

SERVIZI_OFFERTI			
Attributo	Tipo	Byte	Occorrenza:10
Cod_servizio	INTEGER	4	0.04KB
Tipo	VARCHAR2(100)	100	1KB
Camera	NUMBER(7)	5	0.05KB
STORAGE	-	-	-
Totale spazio occupato			**1.09KB**

UTENTI			
Attributo	Tipo	Byte	Occorrenza:2500
Codice_fiscale	CHAR(16)	16	40KB
Nome	VARCHAR2(100)	100	250KB
Cognome	VARCHAR2(100)	100	250KB
STORAGE	-	-	-
Totale spazio occupato			**540KB**

PRENOTAZIONI			
Attributo	Tipo	Byte	Occorrenza:200
Codice_prenotazione	INTEGER	4	0.8KB
Data_partenza	DATE	7	1.4KB
Data_ritorno	DATE	7	1.4KB
Costo_complessivo	FLOAT	8	1.6KB
Pacchetto	INTEGER	4	0.8KB
Utente	INTEGER	4	0.8KB
STORAGE	-	-	-
Totale spazio occupato			**6.8KB**

ARCHIVIO_PRENOTAZIONI			
Attributo	Tipo	Byte	Occorrenza:2000
Codice	INTEGER	4	8KB
Costo	INTEGER	4	8KB
Data_partenza	DATE	7	14KB
Data_ritorno	DATE	7	14KB
Pacchetto	INTEGER	4	8KB
Utenti	CHAR(16)	16	32KB

STORAGE	-	-	-
Totale spazio occupato			**84KB**

VIAGGIATORI			
Attributo	Tipo	Byte	Occorrenza:2500
Id_viaggiatore	INTEGER	4	10KB
Nome	VARCHAR2(100)	100	250KB
Cognome	VARCHAR2(100)	100	250KB
Prenotazione	INTEGER	4	10KB
STORAGE	-	-	-
Totale spazio occupato			**520KB**

CARTE_DI_CREDITO			
Attributo	Tipo	Byte	Occorrenza:1000
ID	INTEGER	4	4KB
Circuito	VARCHAR2(100)	100	100KB
Intestatario	VARCHAR2(100)	100	100KB
Codice_carta	CHAR(16)	16	16KB
Data_scadenza	DATE	7	7KB
Utente	CHAR(16)	16	16KB
STORAGE	-	-	-
Totale spazio occupato			**243KB**

BONIFICI_BANCARI			
Attributo	Tipo	Byte	Occorrenza:1000
ID	INTEGER	4	4KB
Data_bonifico	DATE	7	7KB
Utente	CHAR(16)	16	16KB
STORAGE	-	-	-
Totale spazio occupato			**27KB**

PROGETTAZIONE FISICA DELLA BASE DI DATI

4.1 Creazione delle tabelle

Ora si procederà alla creazione delle tabelle, andando ad utilizzare il linguaggio di programmazione DDL dell'SQL. Le tabelle saranno generate attraverso la piattaforma ORACLE, il Database più utilizzato al mondo, grazie anche alla sua usabilità in quanto può essere usato su tutti i tipi di computer, dai PC ai Macintosh, dai microcomputer ai mainframe, ovunque si voglia. E' utilizzabile da più categorie di utenti: *utenti per nulla esperti*, per operazioni semplici come l'inserimento di dati e l'esecuzione di report; *utenti esperti in DB che non conosco Oracle*, per la semplicità e per la versatilità di utilizzo; *sviluppatori esperti di Oracle*, per la completezza degli operatori disponibili. Chiaramente per poter accedere come utente, il creatore della base dati deve autorizzare, attraverso la creazione di un username e password, l'accesso agli utenti ed attribuirgli determinate azioni possibili, che possono variare dalla semplice lettura dei dati alla modifica di quest'ultimi.

Nel nostro caso specifico, sono state create 16 tabelle. Ogni **PRIMARY KEY** è stata definita nella sezione di campo "Altri vincoli", nonostante si poteva definire come tipo dell'attributo. Questo metodo permette di definire con un nome specifico in memoria, grazie al vincolo **CONSTRAINT**, e inoltre, definendo la chiave primaria in tale campo, dà la possibilità di definire chiavi primarie composte, qualora ci fossero.

In alcuni campi è stata fatta la scelta di inserire il vincolo **NOT NULL** ai campi ritenuti necessari per dare maggior consistenza ai dati che si andranno a memorizzare. Prima di procedere con la creazione delle tabelle, bisogna creare il nostro Database sopra al nostro PC, creando l'user e il cliente che vorrà prenotarsi:

```
1  --CREAZIONE TABLESPACE
2  CREATE TABLESPACE daboca_ts DATAFILE 'C:\app\vince\product21\oradata\XE\daboca.dbf';
3
4  --CREAZIONE DBA BASE DI DATI
5  CREATE USER daboca_dba DEFAULT TABLESPACE daboca_ts
6  IDENTIFIED BY BOCA
7  GRANT DBA, UNLIMITED TABLESPACE TO daboca_dba;
8
9  --CREAZIONE SCRIPT OPERATORE
10 CREATE ROLE supporto_clienti;
11 GRANT CONNECT TO supporto_clienti;
12 GRANT SELECT ON daboca_dba.UTENTI TO supporto_clienti;
13 GRANT SELECT ON daboca_dba.VIAGGIATORI TO supporto_clienti;
14 GRANT SELECT ON daboca_dba.SOGGIORNI TO supporto_clienti;
15 GRANT SELECT ON daboca_dba.SPOSTAMENTI TO supporto_clienti;
16 GRANT SELECT ON daboca_dba.MEZZI_DI_TRASPORTO TO supporto_clienti;
17 GRANT SELECT ON daboca_dba.CAMERE TO supporto_clienti;
18 GRANT SELECT ON daboca_dba.ALBERGHI TO supporto_clienti;
19 GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.PRENOTAZIONI TO supporto_clienti;
20 GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.PACCHETTI TO supporto_clienti;
21 GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.SOGGIORNI_SCELTI TO supporto_clienti;
22 GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.SPOSTAMENTI_SCELTI TO supporto_clienti;
23
```

```
--CREAZIONE SCRIPT CLIENTE
CREATE ROLE cliente;
GRANT CONNECT TO cliente;
GRANT SELECT ON daboca_dba.PACCHETTI TO cliente;
GRANT SELECT ON daboca_dba.SPOSTAMENTI TO cliente;
GRANT SELECT ON daboca_dba.SERVIZI_OFFERTI TO cliente;
GRANT SELECT ON daboca_dba.SPOSTAMENTI TO cliente;
GRANT SELECT ON daboca_dba.MEZZI_DI_TRASPORTO TO cliente;
GRANT SELECT ON daboca_dba.CAMERE TO cliente;
GRANT SELECT ON daboca_dba.ALBERGHI TO cliente;
GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.BONIFICI_BANCARI TO cliente;
GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.CARTE_DI_CREDITO TO cliente;
GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.PRENOTAZIONI TO cliente;
GRANT SELECT, INSERT, UPDATE, DELETE ON daboca_dba.VIAGGIATORI TO cliente;

```

Ora, invece, possiamo vedere la creazione delle tabelle con i propri attributi:

```
--CREAZIONE TABELLE

--Tabella pacchetti (ID, Num_persone)
CREATE TABLE PACCHETTI(
    ID INTEGER,
    Num_persone NUMBER(4) NOT NULL,

    CONSTRAINT PK_PACCHETTI PRIMARY KEY (ID)
);


CREATE TABLE SPOSTAMENTI (
    ID INTEGER,
    Luogo_partenza VARCHAR2(100) NOT NULL,
    Orario_partenza CHAR(8) NOT NULL,
    Luogo_arrivo VARCHAR2(100) NOT NULL,
    Orario_arrivo CHAR(8) NOT NULL,
    Latitudine_partenza CHAR(15) NOT NULL,
    Longitudine_partenza CHAR(15) NOT NULL,
    Latitudine_arrivo CHAR(15) NOT NULL,
    Longitudine_arrivo CHAR(15) NOT NULL,
    Data_arrivo DATE NOT NULL,
    Data_partenza DATE NOT NULL,

    CONSTRAINT PK_SPOSTAMENTI PRIMARY KEY (ID),
    CONSTRAINT CK_Orario_partenza CHECK (Orario_partenza LIKE '%:%'),
    CONSTRAINT CK_Orario_arrivo CHECK (Orario_arrivo LIKE '%:%'),
    CONSTRAINT CK_Data_Spostamenti CHECK (NOT(Data_arrivo<Data_partenza))
);

```

```
--SOGGIORNI (Id_soggiorno, Albergo:ALBERGHI)
CREATE TABLE SOGGIORNI(
    Id_soggiorno CHAR(5),
    Albergo INTEGER NOT NULL,

    CONSTRAINT PK_SOGGIORNI PRIMARY KEY (Id_soggiorno),
    CONSTRAINT UQ_SOGGIORNI UNIQUE (Albergo)
);

--ALBERGHI (Id_albergo, Tipologia, Costo, Nome, Data_checkIn,Data_checkOut)
CREATE TABLE ALBERGHI(
    Id_albergo INTEGER,
    Tipologia VARCHAR2(100) NOT NULL,
    Costo FLOAT NOT NULL,
    Nome VARCHAR2(100) NOT NULL,
    Data_checkIn DATE NOT NULL,
    Data_checkOut DATE NOT NULL,

    CONSTRAINT CK_DATA_CHECK_IN CHECK (NOT(Data_checkIn>Data_checkOut)),
    CONSTRAINT PK_ALBERGHI PRIMARY KEY(Id_albergo)
);

--SPOSTAMENTI_SCELTI(Id_pacchetto:PACCHETTI,Id_spostamento:SPOSTAMENTI)
CREATE TABLE SPOSTAMENTI_SCELTI(
    Id_pacchetto INTEGER NOT NULL,
    Id_spostamento INTEGER NOT NULL,

    CONSTRAINT PK_SPOSTAMENTI_SCELTI PRIMARY KEY(Id_pacchetto,Id_spostamento)
);

--SOGGIORNI_SCELTI(Id_pacchetto:PACCHETTI,Codice_Soggiorno:SOGGIORNI)
CREATE TABLE SOGGIORNI_SCELTI(
    Id_pacchetto INTEGER NOT NULL,
    Codice_soggiorno CHAR(5) NOT NULL,

    CONSTRAINT PK_SOGGIORNI_SCELTI PRIMARY KEY(Id_pacchetto,Codice_soggiorno)
);

--CAMERE (ID, Tipologia, Numero_occupanti,Soggiorno:SOGGIORNI)
CREATE TABLE CAMERE(
    ID INTEGER,
    Tipologia VARCHAR2(100) NOT NULL,
    Numero_occupanti INTEGER NOT NULL,
    Soggiorno CHAR(5) NOT NULL,

    CONSTRAINT PK_CAMERE PRIMARY KEY(ID)
);


CREATE TABLE MEZZI_DI_TRASPORTO(
    Id_mezzo INTEGER,
    Num_viaggiatori NUMBER(5) NOT NULL,
    Num_Mezzi NUMBER(2) DEFAULT 1,
    Tempo_percorrenza CHAR(5) NOT NULL,
    Prezzo FLOAT NOT NULL,
    Id_spostamento INTEGER NOT NULL,

    CONSTRAINT PK_MEZZI_DI_TRASPORTO PRIMARY KEY(Id_mezzo)
);


```

```
130 CREATE TABLE PRENOTAZIONI (
131     Codice_prenotazione INTEGER,
132     Data_partenza DATE NOT NULL,
133     Data_ritorno DATE NOT NULL,
134     Costo_complessivo FLOAT NOT NULL,
135     Pacchetto INTEGER NOT NULL,
136     Utente  CHAR(16) NOT NULL,
137 
138     CONSTRAINT PK_PRENOTAZIONI PRIMARY KEY(Codice_prenotazione),
139     CONSTRAINT CK_Data_Prenotazioni CHECK (NOT(Data_ritorno<Data_partenza))
140 );
141 
142 --VIAGGIATORI(Id_viaggiatore, Nome, Cognome, Prenotazione:PRENOTAZIONI)
143 CREATE TABLE VIAGGIATORI(
144     Id_viaggiatore INTEGER,
145     Nome VARCHAR2(100) NOT NULL,
146     Cognome VARCHAR2(100) NOT NULL,
147     Prenotazione INTEGER NOT NULL,
148 
149     CONSTRAINT PK_OSPITI PRIMARY KEY(Id_viaggiatore)
150 );
151 
152 --UTENTI (Codice_fiscale, Nome, Cognome)
153 CREATE TABLE UTENTI(
154     Codice_fiscale CHAR(16),
155     Nome VARCHAR2(100) NOT NULL,
156     Cognome VARCHAR2(100) NOT NULL,
157 
158     CONSTRAINT PK_UTENTI PRIMARY KEY(Codice_fiscale)
159 );
160 
161 
162 --SERVIZI OFFERTI (Cod_servizio, Tipo,Camera:CAMERE)
163 CREATE TABLE SERVIZI_OFFERTI (
164     Cod_servizio INTEGER,
165     Tipo VARCHAR2(100) NOT NULL,
166     Camera NUMBER(7) NOT NULL,
167 
168     CONSTRAINT PK_SERVIZI PRIMARY KEY (Cod_servizio)
169 );
170 
171 CREATE TABLE ARCHIVIO_PRENOTAZIONI(
172     Codice INTEGER,
173     Costo INTEGER NOT NULL,
174     Data_partenza DATE NOT NULL,
175     Data_ritorno DATE NOT NULL,
176     Pacchetto INTEGER NOT NULL,
177     Utenti CHAR(16) NOT NULL,
178 
179     CONSTRAINT PK_ARCHIVIO_PRENOTAZIONI PRIMARY KEY (Codice),
180     CONSTRAINT CK_Data__Archivio_Prenotazioni CHECK (NOT(Data_ritorno<Data_partenza))
181 );
182 
183 --CARTE DI CREDITO (ID, Circuito, Intestatario, Codice_carta, Data_scadenza, Utente:UTENTI)
184 CREATE TABLE CARTE_DI_CREDITO(
185     ID INTEGER NOT NULL,
186     Circuito VARCHAR2(100) NOT NULL,
187     Intestatario VARCHAR2(100) NOT NULL,
188     Codice_carta CHAR(16) NOT NULL,
189     Data_scadenza DATE NOT NULL,
190     Utente CHAR(16) NOT NULL,
191 
192     CONSTRAINT PK_CARTE_DI_CREDITO PRIMARY KEY(ID)
193 );
194 
```

```sql
--BONIFICI BANCARI(ID, Data_bonifico, Utente:UTENTI)
CREATE TABLE BONIFICI_BANCARI (
    ID INTEGER,
    Data_bonifico DATE NOT NULL,
    Utente CHAR(16) NOT NULL,

    CONSTRAINT PK_BONIFICI_BANCARI PRIMARY KEY(ID)
);

```

Siccome il numero di pacchetti, e di conseguenze delle prenotazioni, sono limitate, il team ha scelto di utilizzare le '**SEQUENZE**', in modo che il DBMS gestisca in modo automatico l'assegnazione dei valori nei campi specifici.

```sql
--CREAZIONE SEQUENZE PER I PACCHETTI
CREATE SEQUENCE Contatore_PACCHETTO START WITH 1 INCREMENT BY 1 MAXVALUE 100 NOCYCLE;
CREATE SEQUENCE Contatore_Prenotazioni START WITH 1 INCREMENT BY 1 MAXVALUE 100 NOCYCLE;

```

Si tende a specificare che si è usato il '**NOCYCLE**' poiché non si possono avere valori uguali nelle chiavi primarie, inoltre arrivati al valore massimo '**MAXVALUE**' non si dovrà ripetere il ciclo, poiché l'agenzia viaggi DALE BOCA S.R.L. ha l'intensione di gestire solo le offerte relative ai pacchetti.

4.2 Creazione dei vincoli aggiuntivi

Nella creazione dei vincoli aggiuntivi, o meglio nella gestione dei vincoli d'integrità referenziale, viene utilizzato il comando **ALTER TABLE**, il quale permette di alterare una tabella specifica precedentemente creata (per questo motivo i vincoli aggiuntivi si vanno a delineare alla fine della creazione di tutte le tabelle) e di aggiungere o rimuovere al suo interni dei vincoli o colonne. Nell'utilizzo del suddetto comando, bisogna rispettare le informazioni giù presenti e qualora l'istanza contenesse delle violazioni per il nuovo vincolo, la modifica sarà rifiutata. Inoltre, si andrà ad usare anche:

- **CASCATE**: elimina in maniera autonoma ed automatica tutti gli oggetti che sono associati all'attributo eliminato.

Di seguito è presenta la creazione dei vincoli aggiuntivi:

```
--CREAZIONE DEI VINCOLI D'INTEGRITA' REFERENZIALE (CHIAVI ESTERNE)
ALTER TABLE SPOSTAMENTI_SCELTI ADD CONSTRAINT FK_SPOSTAMENTI_SCELTI_PACCHETTI
FOREIGN KEY (Id_pacchetto) REFERENCES PACCHETTI(ID) ON DELETE CASCADE;

ALTER TABLE SPOSTAMENTI_SCELTI ADD CONSTRAINT FK_SPOSTAMENTI_SCELTI_
FOREIGN KEY (Id_spostamento) REFERENCES SPOSTAMENTI(ID) ON DELETE CASCADE;

ALTER TABLE SOGGIORNI_SCELTI ADD CONSTRAINT FK_SOGGIORNI_SCELTI_PACCHETTI
FOREIGN KEY (Id_pacchetto) REFERENCES PACCHETTI(ID) ON DELETE CASCADE;

ALTER TABLE SOGGIORNI_SCELTI ADD CONSTRAINT FK_SOGGIORNI_SCELTI_SOGGIORNI
FOREIGN KEY (Codice_soggiorno) REFERENCES SOGGIORNI(Id_soggiorno) ON DELETE CASCADE;

ALTER TABLE MEZZI_DI_TRASPORTO  ADD CONSTRAINT FK_MEZZI_DI_TRASPORTO_SPOSTAMENTI
FOREIGN KEY (Id_spostamento) REFERENCES SPOSTAMENTI(ID) ON DELETE CASCADE;

ALTER TABLE SOGGIORNI ADD CONSTRAINT FK_SOGGIORNI_ALBERGHI
FOREIGN KEY (Albergo) REFERENCES ALBERGHI(Id_albergo) ON DELETE CASCADE;

ALTER TABLE CAMERE ADD CONSTRAINT FK_CAMERE_SOGGIORNI
FOREIGN KEY (Soggiorno) REFERENCES SOGGIORNI(Id_soggiorno) ON DELETE CASCADE;

ALTER TABLE SERVIZI_OFFERTI ADD CONSTRAINT FK_SERVIZI_OFFERTI_CAMERE
FOREIGN KEY (Camera) REFERENCES CAMERE(ID) ON DELETE CASCADE;

ALTER TABLE PRENOTAZIONI ADD CONSTRAINT FK_PRENOTAZIONI_PACCHETTO
FOREIGN KEY (Pacchetto) REFERENCES PACCHETTI(ID) ON DELETE CASCADE;

ALTER TABLE PRENOTAZIONI ADD CONSTRAINT FK_PRENOTAZIONI_UTENTI
FOREIGN KEY (Utente) REFERENCES UTENTI(Codice_fiscale) ON DELETE CASCADE;

ALTER TABLE VIAGGIATORI ADD CONSTRAINT FK_VIAGGIATORI_PRENOTAZIONI
FOREIGN KEY (Prenotazione) REFERENCES PRENOTAZIONI(Codice_prenotazione) ON DELETE CASCADE;

ALTER TABLE CARTE_DI_CREDITO ADD CONSTRAINT FK_CARTE_DI_CREDITO_UTENTI
FOREIGN KEY (Utente) REFERENCES UTENTI(Codice_fiscale) ON DELETE CASCADE;

ALTER TABLE BONIFICI_BANCARI ADD CONSTRAINT FK_BONIFICI_BANCARI_UTENTI
FOREIGN KEY (Utente) REFERENCES UTENTI(Codice_fiscale) ON DELETE CASCADE;

```

4.3 Inserimento valori nella base di dati

Il passo successivo alla definizione delle chiavi esterne, è il popolamento della nostra base di dati, mediante l'operazione di **INSERT**. L'inserimento dei dati può provocare la violazione dei vincoli d'integrità vigenti sulla base dati, ossia: *vincoli di dominio*, violati nel caso in cui si inserisce un valore che non appartiene al dominio considerato; *vincoli di chiave primaria*, violati nel caso in cui il valore della chiave primaria inserito è NULL o se esiste già in memoria una tupla con quel valore di chiave, *vincoli d'integrità referenziale*, violati nel caso in cui il valore di una chiave esterna fa riferimento ad una chiave che o non esiste oppure il valore appartiene ad un dominio diverso da quello considerato.

Di seguito il popolamento della base di dati:

```
--POPOLAMENTO DELLA BASE DI DATI

--INSERIMENTO PACCHETTI
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 2);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 5);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 3);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 4);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 8);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 1);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 2);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 2);
   INSERT INTO PACCHETTI VALUES (Contatore_PACCHETTO.NEXTVAL, 1);
   SELECT * FROM PACCHETTI ORDER BY ID;

--INSERIMENTO SPOSTAMENTI
    INSERT INTO SPOSTAMENTI VALUES (100,'Napoli','10:00:00','Francoforte','12:05:00','40G50P00S N','14G25P17S E','50G01P58S N','08G34P12S E','28-Mar-2023','28-Mar-2023');
    INSERT INTO SPOSTAMENTI VALUES (101,'Napoli','12:00:00','Parigi','14:25:00','40G50P00S N','14G25P17S E','48G86P31S N','02G33P97S E','08-Apr-2023','08-Apr-2023');
    INSERT INTO SPOSTAMENTI VALUES (102,'Budapest','15:00:00','Roma', '17:30:00','41G90278S N','12G49636S E','47G49791S N','19G04023S E','12-Apr-2023','12-Apr-2023');
    INSERT INTO SPOSTAMENTI VALUES (103,'Londra','19:00:00','Madrid', '22:00:00', '40G41677S N','-3G70379S E','51G50735S N','-0G12775S E' ,'14-Apr-2023','14-Apr-2023');
    INSERT INTO SPOSTAMENTI VALUES (104,'Parigi','19:00:00','Milano', '20:30:00', '45G46542S N','9G18592S E','48G85661S N','2G35222S E' ,'18-Apr-2023','18-Apr-2023');
    INSERT INTO SPOSTAMENTI VALUES (110,'Roma','12:00:00','Francoforte','15:00:00','47G49791S N','19G04023S E','50G01P58S N','08G34P12S E','25-Mar-2023','25-Mar-2023');
    INSERT INTO SPOSTAMENTI VALUES (111,'Milano','10:00:00','Francoforte','11:50:00','48G85661S N','2G35222S E','50G01P58S N','08G34P12S E','22-Mar-2023','22-Mar-2023');
    INSERT INTO SPOSTAMENTI VALUES (120,'Londra','12:00:00','Parigi','13:25:00','40G41677S N','-3G70379S E','48G86P31S N','02G33P97S E','09-Apr-2023','09-Apr-2023');
    INSERT INTO SPOSTAMENTI VALUES (121,'Roma','12:00:00','Parigi','13:25:00','47G49791S N','19G04023S E','48G86P31S N','02G33P97S E','09-Apr-2023','09-Apr-2023');
    SELECT * FROM SPOSTAMENTI;

--INSERIMENTO ALBERGHI
    INSERT INTO ALBERGHI VALUES (200,'3 STELLE',100.0,'HAMPTON FRANKFURT', '28-Mar-2023', '03-Apr-2023');
    INSERT INTO ALBERGHI VALUES (201,'4 STELLE',150.0,'MARITM HOTEL FRANKFURT', '25-Mar-2023', '30-Mar-2023');
    INSERT INTO ALBERGHI VALUES (202,'5 STELLE',300.0,'FRANKFURT MARRIOTT HOTEL', '22-Mar-2023', '28-Mar-2023');
    INSERT INTO ALBERGHI VALUES (400,'3 STELLE',120.0,'HOTEL ELYSEES OPERA DE PARIS', '08-Apr-2023', '15-Apr-2023');
    INSERT INTO ALBERGHI VALUES (401,'4 STELLE',180.0,'HOTEL LE 10 BIS PARIS', '09-Apr-2023', '19-Apr-2023');
    INSERT INTO ALBERGHI VALUES (402,'5 STELLE',400.0,'LA TREMOILLE PARIS', '09-Apr-2023', '12-Apr-2023');
    INSERT INTO ALBERGHI VALUES (500,'4 STELLE',400.0,'TOTTI GOL', '12-Apr-2023', '15-Apr-2023');
    INSERT INTO ALBERGHI VALUES (600,'4 STELLE',300.0,'PLAZA MADRAZO HOTEL', '14-Apr-2023', '20-Apr-2023');
    INSERT INTO ALBERGHI VALUES (700,'3 STELLE',250.0,' POLENTA SQUARE HOTEL', '18-Apr-2023', '21-Apr-2023');
    SELECT * FROM ALBERGHI;

--INSERIMENTO SOGGIORNI
   INSERT INTO SOGGIORNI VALUES(1990,200);
   INSERT INTO SOGGIORNI VALUES(1991,201);
   INSERT INTO SOGGIORNI VALUES(1992, 202);
   INSERT INTO SOGGIORNI VALUES(1993, 400);
   INSERT INTO SOGGIORNI VALUES(1994, 401);
   INSERT INTO SOGGIORNI VALUES(1995, 402);
   INSERT INTO SOGGIORNI VALUES(1996, 500);
   INSERT INTO SOGGIORNI VALUES(1997, 600);
   INSERT INTO SOGGIORNI VALUES(1998, 700);
   SELECT * FROM SOGGIORNI;


--INSERIMENTO SPOSTAMENTI SCELTI
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (1,120);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (2, 100);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (3, 101);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (4, 102);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (5, 103);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (6, 104);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (8, 110);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (9, 111);
   INSERT INTO SPOSTAMENTI_SCELTI VALUES (7, 121);
```

```sql
--INSERIMENTO SOGGIORNI SCELTI
   INSERT INTO SOGGIORNI_SCELTI VALUES (1,1995);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (2, 1990);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (3, 1993);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (4, 1996);
   INSERT INTO SOGGIORNI_SCELTI VALUES (5, 1997);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (6, 1998);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (8, 1991);
   INSERT INTO SOGGIORNI_SCELTI VALUES (9, 1992);
   INSERT INTO SOGGIORNI_SCELTI  VALUES (7, 1994);


--INSERIMENTO CAMERE
   INSERT INTO CAMERE VALUES (45, 'Standard',5 ,1990);
   INSERT INTO CAMERE VALUES (46, 'Camera doppia',2 ,1991);
   INSERT INTO CAMERE VALUES (47, 'Camera singola',1 ,1992);
   INSERT INTO CAMERE VALUES (48, 'Tripla',3 ,1993);
   INSERT INTO CAMERE VALUES (49, 'Suite nunziale',2 ,1994);
   INSERT INTO CAMERE VALUES (50, 'Camera doppia deluxe',2 ,1995);
   INSERT INTO CAMERE VALUES (51, 'Quadrupla standard',4 ,1996);
   INSERT INTO CAMERE VALUES (52, 'Stanza attigua standard',8 ,1997);
   INSERT INTO CAMERE VALUES (53, 'Suite presidenziale',1 ,1998);
SELECT * FROM CAMERE;

--INSERIMENTO MEZZI DI TRASPORTO
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (60,2,1,'1h25m',200 ,120);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (61,2,1,'1h25m',200 ,121);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (62,5,1,'2h05m',280 ,100);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (63,3,1,'1h25m',200 ,101);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (64,4,1,'2h30m',250 ,102);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (65,8,2,'3h00m',400 ,103);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (66,1,1,'1h25m',100 ,104);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (67,2,1,'3h00m',200 ,110);
   INSERT INTO MEZZI_DI_TRASPORTO VALUES (68,1,1,'1h50m',150 ,111);
   SELECT * FROM MEZZI_DI_TRASPORTO ORDER BY Id_mezzo;

--INSERIMENTO UTENTI
    INSERT INTO UTENTI VALUES ('NSCDNR40R23Z602Y', UPPER('Edson Arantes Nascimento'), UPPER ('Pele'));
    INSERT INTO UTENTI VALUES ('VLLGLC64L09D150M', UPPER('Gianluca'), UPPER ('Vialli'));
    INSERT INTO UTENTI VALUES ('NSGLNZ91H04F839X', UPPER('Lorenzo'), UPPER ('Insigne'));
    INSERT INTO UTENTI VALUES ('MRDDRM60R30Z600D', UPPER('Diego Armando'), UPPER ('Maradona'));
    INSERT INTO UTENTI VALUES ('FBNCNNVR0GA8GTAN', UPPER('Fabioo'), UPPER ('Cannavaro'));
    INSERT INTO UTENTI VALUES ('KNYWST89QTY8UZNS', UPPER('Kanye Omari'), UPPER ('West'));
    INSERT INTO UTENTI VALUES ('BRTNSPRS98AEDQ12', UPPER('Britney'), UPPER ('Spears'));
    INSERT INTO UTENTI VALUES ('BRNMRS28NBMNAQS3', UPPER('Bruno'), UPPER ('Mars'));
    INSERT INTO UTENTI VALUES ('REYMSTRO619HHBN5', UPPER('Rey'), UPPER ('Mysterio'));
    SELECT * FROM UTENTI;

```

```sql
--INSERIMENTO PRENOTAZIONI
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '28-Mar-2023','03-Apr-2023',380, 2, 'NSCDNR40R23Z602Y');
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '08-Apr-2023', '15-Apr-2023', 320, 3, 'VLLGLC64L09D150M');
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '12-Apr-2023', '15-Apr-2023', 650, 4, 'NSGLNZ91H04F839X' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '14-Apr-2023', '20-Apr-2023', 700, 5, 'MRDDRM60R30Z600D' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '18-Apr-2023', '21-Apr-2023', 350, 6, 'FBNCNNVR0GA8GTAN' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '25-Mar-2023', '30-Mar-2023', 300, 8, 'KNYWST89QTY8UZNS' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '22-Mar-2023', '28-Mar-2023', 450, 9, 'BRTNSPRS98AEDQ12' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '09-Apr-2023', '12-Apr-2023', 380, 1, 'BRNMRS28NBMNAQS3' );
   INSERT INTO PRENOTAZIONI VALUES (Contatore_Prenotazioni.NEXTVAL, '09-Apr-2023', '19-Apr-2023', 380, 7, 'REYMSTRO619HHBN5' );
   SELECT * FROM PRENOTAZIONI;

--INSERIMENTO VIAGGIATORI
   INSERT INTO VIAGGIATORI VALUES (551, UPPER('Edson Arantes Nascimento'), UPPER ('Pele'), 2);
   INSERT INTO VIAGGIATORI VALUES (552, UPPER('Roberto'), UPPER ('Carlos'), 2);
   INSERT INTO VIAGGIATORI VALUES (553, UPPER('Alexandre'), UPPER ('Pato'), 2);
   INSERT INTO VIAGGIATORI VALUES (554, UPPER('Ricardo'), UPPER ('Kaka'), 2);
   INSERT INTO VIAGGIATORI VALUES (555, UPPER('Neymar'), UPPER ('Jr'), 2);
   INSERT INTO VIAGGIATORI VALUES (331,  UPPER('Gianluca'), UPPER ('Vialli'), 3);
   INSERT INTO VIAGGIATORI VALUES (332,  UPPER('Roberto'), UPPER ('Mancini'), 3);
   INSERT INTO VIAGGIATORI VALUES (333,  UPPER('Franco'), UPPER ('Ricciardi'), 3);
   INSERT INTO VIAGGIATORI VALUES (440,  UPPER('Lorenzo'), UPPER ('Insigne'), 4);
   INSERT INTO VIAGGIATORI VALUES (441,  UPPER('Marco'), UPPER ('Verratti'), 4);
   INSERT INTO VIAGGIATORI VALUES (442,  UPPER('Ciro'), UPPER ('Immobile'), 4);
   INSERT INTO VIAGGIATORI VALUES (443,  UPPER('Zdnek'), UPPER ('Zeman'), 4);
   INSERT INTO VIAGGIATORI VALUES (80,   UPPER('Diego Armando'), UPPER ('Maradona'), 5);
   INSERT INTO VIAGGIATORI VALUES (81,   UPPER('Lionel'), UPPER ('Messi'), 5);
   INSERT INTO VIAGGIATORI VALUES (82,   UPPER('Franco'), UPPER ('Baresi'), 5);
   INSERT INTO VIAGGIATORI VALUES (83,   UPPER('Sergio'), UPPER ('Busquets'), 5);
   INSERT INTO VIAGGIATORI VALUES (84,   UPPER('Bruno'), UPPER ('Giordano'), 5);
   INSERT INTO VIAGGIATORI VALUES (85,   UPPER('Ezequiel'), UPPER ('Lavezzi'), 5);
   INSERT INTO VIAGGIATORI VALUES (86,   UPPER('Edinson'), UPPER ('Cavani'), 5);
   INSERT INTO VIAGGIATORI VALUES (87,   UPPER('German'), UPPER ('Denis'), 5);

   INSERT INTO VIAGGIATORI VALUES (10,  UPPER('Fabioo'), UPPER ('Cannavaro'), 6);
   INSERT INTO VIAGGIATORI VALUES (1,  UPPER('Kanye Omari'), UPPER ('West'), 8);
   INSERT INTO VIAGGIATORI VALUES (2,  UPPER('Kim Kardashian'), UPPER ('West'), 8);
   INSERT INTO VIAGGIATORI VALUES (11,   UPPER('Britney'), UPPER ('Spears'), 9);
   INSERT INTO VIAGGIATORI VALUES (21,   UPPER('Bruno'), UPPER ('Mars'), 1);
   INSERT INTO VIAGGIATORI VALUES (22,   UPPER('Fabio'), UPPER ('Greco'), 1);
   INSERT INTO VIAGGIATORI VALUES (619,  UPPER('Rey'), UPPER ('Mysterio'), 7);
   INSERT INTO VIAGGIATORI VALUES (620,  UPPER('Santino'), UPPER ('Marella'), 7);

SELECT * FROM VIAGGIATORI ORDER BY Prenotazione;

--INSERIMENTO SERVIZI_OFFERTI
   INSERT INTO SERVIZI_OFFERTI VALUES (59, 'Parcheggio gratuito e WI-FI', 45);
   INSERT INTO SERVIZI_OFFERTI VALUES (33, 'Spa e pacchetti benessere', 46);
   INSERT INTO SERVIZI_OFFERTI VALUES (63, 'Servizio in camera', 47);
   INSERT INTO SERVIZI_OFFERTI VALUES (72, 'Colazione gratuita', 48);
   INSERT INTO SERVIZI_OFFERTI VALUES (69, 'Camere insonorizzate', 49);
   INSERT INTO SERVIZI_OFFERTI VALUES (88, 'All inclusive', 50);
   INSERT INTO SERVIZI_OFFERTI VALUES (90, 'Area giochi bambini', 51);
   INSERT INTO SERVIZI_OFFERTI VALUES (92, 'Animali ammessi e purificazione stanze', 52);
   INSERT INTO SERVIZI_OFFERTI VALUES (93, 'Aria condizionata', 53);
```

```
--INSERIMENTO  ARCHIVIO_PRENOTAZIONI
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (11, 380, '28-Mar-2023','03-Apr-2023', 2, 'NSCDNR40R23Z602Y');
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (12, 320, '08-Apr-2023', '15-Apr-2023', 3, 'VLLGLC64L09D150M');
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (13, 650, '12-Apr-2023', '15-Apr-2023', 4, 'NSGLNZ91H04F839X' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (14, 700, '14-Apr-2023', '20-Apr-2023', 5, 'MRDDRM60R30Z600D' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (15, 350, '18-Apr-2023', '21-Apr-2023', 6, 'FBNCNNVR0GA8GTAN' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (16, 300, '25-Mar-2023', '30-Mar-2023', 8, 'KNYWST89QTY8UZNS' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (17, 450, '22-Mar-2023', '28-Mar-2023', 9, 'BRTNSPRS98AEDQ12' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (18, 380, '09-Apr-2023','12-Apr-2023',  1, 'BRNMRS28NBMNAQS3' );
   INSERT INTO  ARCHIVIO_PRENOTAZIONI VALUES (19, 380, '09-Apr-2023', '19-Apr-2023', 7, 'REYMSTRO619HHBN5' );

--INSERIMENTO PAGAMENTI CARTE DI CREDITO
   INSERT INTO CARTE_DI_CREDITO VALUES (300, 'Visa','Edson Arantes Nascimento Pele', 489, '01-Feb-2027', 'NSCDNR40R23Z602Y');
   INSERT INTO CARTE_DI_CREDITO VALUES (301, 'Matercard','Gianluca Vialli', 238, '04-Aug-2028', 'VLLGLC64L09D150M');
   INSERT INTO CARTE_DI_CREDITO VALUES (302, 'Matercard','Lorenzo Insigne', 741, '06-Mar-2028', 'NSGLNZ91H04F839X' );
   INSERT INTO CARTE_DI_CREDITO VALUES (303, 'Visa','Diego Armando Maradona', 854, '19-Mar-2027','MRDDRM60R30Z600D' );

--INSERIMENTO PAGAMENTI BONIFICO
  INSERT INTO BONIFICI_BANCARI VALUES (500,'08-Apr-2023' , 'FBNCNNVR0GA8GTAN' );
  INSERT INTO BONIFICI_BANCARI VALUES (501,'02-Mar-2023' , 'KNYWST89QTY8UZNS' );
  INSERT INTO BONIFICI_BANCARI VALUES (502,'05-Mar-2023' , 'BRTNSPRS98AEDQ12' );
  INSERT INTO BONIFICI_BANCARI VALUES (503,'07-Apr-2023' , 'BRNMRS28NBMNAQS3' );
  INSERT INTO BONIFICI_BANCARI VALUES (504,'01-Apr-2023' , 'REYMSTRO619HHBN5' );

```

4.4 Definizione degli indici

Dal momento che ORACLE va a creare di default degli indici definiti sulle chiavi primarie, si è ritenuto necessario, per garantire un servizio più efficiente nella base di dati, aggiungere degli indici secondari definiti sui campi di seguito elencati, selezionati in base alla frequenza di utilizzo nelle query che un cliente potrebbe fare:

- ***SPOSTAMENTO.Luogo_partenza***, ***SPOSTAMENTO.Luogo_arrivo*** e ***SPOSTAMENTO_Data_partenza***, poiché la ricerca di un viaggio si basa principalmente sui luoghi di partenza ed arrivo;
- ***ALBERGHI.Tipologia***, poiché gli utenti tendono a dare particolare importanza all'albergo in cui pernottare;
- ***CAMERE.Tipologia***, poiché gli utenti danno importanza, non solo all'albergo, ma anche alla camera di pernottamento, in base alle loro esigenze;
- ***SERVIZI_OFFERTI.Tipo***, anche questo ritenuto molto importante dagli utenti, soprattutto per quelli che hanno determinate esigenze;
- ***PACCHETTI.Num_persone***, poiché la ricerca degli utenti si basa come elemento primario sul numero di persone per cui è pensato il pacchetto.

```

--CREAZIONE INDICI
  CREATE INDEX INDICE_SPOSTAMENTI_Luogo_partenza ON SPOSTAMENTI(Luogo_partenza);
  CREATE INDEX INDICE_SPOSTAMENTI_Luogo_arrivo ON SPOSTAMENTI(Luogo_arrivo);
  CREATE INDEX INDICE_SPOSTAMENTI_Data_partenza ON SPOSTAMENTI(Data_partenza);
  CREATE INDEX INDICE_ALBERGHI_Tipologia ON ALBERGHI(Tipologia);
  CREATE INDEX INDICE_CAMERE_Tipologia ON CAMERE(Tipologia);
  CREATE INDEX INDICE_SERVIZI_OFFERTI_Tipo ON SERVIZI_OFFERTI(Tipo);
  CREATE INDEX INDICE_PACCHETTI_Num_persone ON PACCHETTI(Num_persone);

```

4.5 Gestione della concorrenza

Per rendere un database più efficiente bisogna dapprima evitare le anomalie di ogni tipo, dunque il team ha scelto, come di consueto in questi casi, il protocollo 2PL (Two Phase Locking) stretto, dove tutti i lock possono essere rilasciati solo dopo la fase del commit o l'abort. Inoltre si è scelto di usare il livello REPEATABLE READ per le transazioni in lettura, il che garantiscono che, nel caso in cui un dato venisse letto due volte, il risultato dovrà risultare ovviamente lo stesso, senza modifiche. In tal modo, si garantisce un'esperienza d'utilizzo del servizio migliore. Si è voluto pensare anche di aggiungere un tempo limitato per ogni transazione.

4.6 Controllo dell'affidabilità

Il controllo dell'affidabilità ha come obbiettivo quello di "ripristinare" il corretto stato di un sistema di basi di dati a valle di un possibile malfunzionamento delle sue componenti (hardware o software) dovuto a guasti accidentali o intenzionali. Difatti deve garantire le proprietà di *atomicità* e *persistenza* delle transazioni che rappresentano l'unità base delle attività di recovery. Per tali motivi, la memorizzazione dei dati nel nostro DBMS viene affidata a due tipologie di memorie:

- *memoria centrale,* la quale rappresenterà lo storage primario del nostro sistema di basi di dati con caratteristiche di elevata velocità per le operazioni di lettura/scrittura, capacità ridotto e volatilità delle informazioni;
- *memoria stabile*, grazie al RAID 1, il quale mantiene una copia esatta di tutti i dati su almeno due dischi. I vantaggi di questa tipologia di storage è la persistenza e affidabilità dei dati (poiché in caso di malfunzionamento della macchina o la perdita di uno due dischi, vi è sempre una copia esatta dei dati) ed elevata capacità, però di contro si presenta una velocità ridotta.

Siccome la nostra base di dati sarà creata sul DBMS ORACLE, si hanno delle scelte implicite per quanto riguarda, appunto, il controllo dell'affidabilità del sistema, ovvero:

1. La scrittura sulla base di dati è di tipo differito, in quanto ORACLE possiede dei redo file e non gli undo file;
2. Nel caso in cui ci fossero guasti software, si utilizzerà la tecnica di recovery di ripresa a caldo, la quale garantisce il ripristino e il corretto funzionamento del database a partire dall'ultimo checkpoint;
3. Nel caso in cui ci fossero guasti hardware, si utilizzerà la tecnica di recovery di ripresa a freddo, la quale andrà ad utilizzare le copie di backup che ORACLE ha generato (in questo caso sul disco di copia della memoria stabile e non sul disco di memoria centrale), così da tornare all'ultimo checkpoint ed effettuare, quindi, la ripresa a caldo.

SQL

5.1 Cos'è?

Come già anticipato prima, il linguaggio standardizzato usato per i DBMS relazionali, come stabilito sia da ANSI che da ISO (International Standard Organization), è il linguaggio *SQL* (Structured Query Language), il quale fu originariamente sviluppato presso i laboratori di ricerca dell'IBM con il nome *SEQUEL*. Negli anni sono state fornite agli utenti diverse versioni: dal primo SQL-86 (conosciuto anche come SQL1), fino all'attuale SQL3 o SQL 2003. Esso è un linguaggio di tipo dichiarativo, ovvero permette all'utente di specificare cosa si cerca nel DB, lasciando al sistema il compito di eseguire ed ottimizzare le operazioni. Ciò può sembrare scontato e banale, ma non lo è affatto, in quanto gli altri linguaggi presenti al mondo sono di tipo procedurale, ovvero è l'utente a dover dire come procedere per una determinata azione, dunque anche i tempi di ricerca risulterebbero molto lenti. Inoltre, altro vantaggio del SQL è che permette di interagire con una base di dati qualsiasi sia il DBMS che la gestisca: in questo modo, tutte le applicazioni che operano sulla base di dati sono svincolate dal particolare DBMS che si ha, che sia di tipo commerciale od open-source.

5.2 Viste

Conclusa la fase di creazione delle tabelle, delle chiavi esterne, dell'inserimento die dati e della gestione delle occorrenze e dell'affidabilità, il nostro DB è creato. Ora l'utente ha la possibilità, per esempio, di vedere tutti i pacchetti con i viaggi ad essi associati. Per vedere ciò, ci serviremo delle *viste*. Una vista è una tabella che viene descritta in termini di altre tabelle: è una relazione virtuale, dove le sue tuple non sono effettivamente situate in memoria del nostro DB, ma sono ricavabili da quest'ultimo. Dunque una vista costituisce un'interfaccia da mettere a disposizione di utenti o talvolta applicazioni per le interrogazioni. Di seguito sono riportate delle viste di esempio, delle possibili richieste che l'utente potrebbe avere:

```
450  --CREAZIONE VISTE
451
452  --Una vista in cui è possibile visualizzare tutte le prenotazioni effettuate da ogni utente
453  CREATE OR REPLACE VIEW Report_Prenotazioni AS
454  SELECT
455  U.Codice_fiscale, U.Nome, U.Cognome,
456  P1.Codice_prenotazione, P1.Costo_complessivo AS CostoPrenotazione, P1.Data_partenza, P1.Data_ritorno, P1.Pacchetto AS Pacchetto_Prenotato
457  FROM
458   UTENTI U JOIN PRENOTAZIONI P1 ON U.Codice_fiscale=P1.Utente;
459
460   SELECT * FROM Report_Prenotazioni;
461
462  --Vista che permette di visualizzare tutti i viaggi associati ad un pacchetto
463  CREATE MATERIALIZED VIEW SCHEDE_VIAGGI AS
464  SELECT P.ID AS PACCHETTO_SCELTO, S2.Luogo_partenza, S2. Luogo_arrivo,S2.Data_partenza,S2.Data_arrivo,S2.Orario_partenza,S2.Orario_arrivo
465  FROM (SPOSTAMENTI_SCELTI S1 JOIN PACCHETTI P ON P.ID=S1.Id_pacchetto) JOIN SPOSTAMENTI S2 ON S1.Id_spostamento=S2.ID;
466   SELECT * FROM SCHEDE_VIAGGI;
467
468
469  --Mostra tutte le fatture dei pagamenti effettuati con carta di credito
470   CREATE VIEW Fattura_carta AS
471   SELECT U.Codice_fiscale AS Utente, U.Nome, U.Cognome, CC.Intestatario, CC.Circuito AS Circuito_carta
472   FROM UTENTI U JOIN CARTE_DI_CREDITO CC ON U.Codice_fiscale=CC.Utente;
473
474   SELECT * FROM Fattura_carta;
475
476  --Mostra tutte le fatture dei pagamenti effettuati tramite bonifico con la relativa data
477   CREATE VIEW Fattura_bonifici AS
478   SELECT U.Codice_fiscale AS Utente, U.Nome, U.Cognome, BB.Data_bonifico
479   FROM UTENTI U JOIN BONIFICI_BANCARI BB ON U.Codice_fiscale=BB.Utente;
480
481   SELECT * FROM Fattura_bonifici;
482
```

5.3 Varie query

Tutte le informazioni sono organizzate in una struttura logica, tutti i dati e le eventuali prenotazioni sono facilmente eseguibili dal cliente. Il modo attraverso il quale il cliente può accedere ai dati è attraverso le query, ossia delle domande che banalmente vengono fatta quotidianamente nei vari motori di ricerca, come Google o Youtube, o, come nel nostro caso, ad un database per cercare le informazioni che si necessitano sapere. Le query vanno poste attraverso il linguaggio pocanzi citato, ossia SQL. Naturalmente, come ogni linguaggio, anche SQL presenta una propria sintassi con delle regole ben precise, attraverso le quali è possibile ricercare ogni dato presente nella nostra base di dati. Di seguito sono riportati delle interrogazioni possibili al nostro database:

```
--QUERY

--Seleziona il numero di viaggiatori relativi ad ogni prenotazione
SELECT V.Prenotazione, COUNT(*) AS numero_viaggiatori
FROM VIAGGIATORI V
GROUP BY V.PRENOTAZIONE;

--Mostra l'anagrafica dei viaggiatori che hanno effettuato prenotazione 5
SELECT NOME, COGNOME
FROM VIAGGIATORI WHERE PRENOTAZIONE=5;

--Mostra la tipologia di albergo,la tipologia di camera e il numero di occupati
SELECT A.TIPOLOGIA,C.TIPOLOGIA,C.NUMERO_OCCUPANTI
FROM (SOGGIORNI S JOIN CAMERE C ON C.SOGGIORNO=S.ID_SOGGIORNO) JOIN ALBERGHI A ON S.ALBERGO=A.ID_ALBERGO;


--Mostra il tempo di percorrenza dovuto allo spostamento verso Roma
SELECT M.TEMPO_PERCORRENZA
FROM MEZZI_DI_TRASPORTO M JOIN SPOSTAMENTI S ON S.ID=M.ID_SPOSTAMENTO
WHERE LUOGO_ARRIVO LIKE 'Roma';



--individuare per ogni utente la spesa totale
SELECT DISTINCT U.Nome, SUM(P.Costo_complessivo) AS SPESA_TOTALE
FROM UTENTI U JOIN PRENOTAZIONI P ON U.Codice_fiscale=P.Utente
GROUP BY U.Nome;

--Individua l'utente che ha speso di più
SELECT DISTINCT UT.Nome, UT.Cognome, SUM(PR.Costo_complessivo) AS SPESA_COMPLESSIVA
FROM UTENTI UT JOIN PRENOTAZIONI PR ON UT.Codice_fiscale=PR.Utente
GROUP BY UT.Nome, UT.Cognome
HAVING SUM(PR.Costo_complessivo) >= ALL(
 SELECT SUM(P.Costo_complessivo)
 FROM UTENTI U JOIN PRENOTAZIONI P ON U.Codice_fiscale=P.Utente
 GROUP BY U.Codice_fiscale);

--Individuare le mete più acquistate--
SELECT S.Luogo_arrivo, COUNT(S.ID) AS NUMERO_PRENOTAZIONI
FROM ((SPOSTAMENTI_SCELTI SS JOIN SPOSTAMENTI S ON SS.ID_spostamento=S.ID)
JOIN PACCHETTI P ON SS.ID_pacchetto=P.ID) JOIN PRENOTAZIONI PR ON P.ID=PR.Pacchetto
GROUP BY S.Luogo_arrivo
HAVING COUNT(*)>=ALL(
SELECT COUNT(S.ID) AS NUMERO_PRENOTAZIONI
FROM ((SPOSTAMENTI_SCELTI SS1 JOIN SPOSTAMENTI S1 ON SS1.ID_spostamento=S1.ID) JOIN
PACCHETTI P1 ON SS1.ID_pacchetto=P1.ID) JOIN PRENOTAZIONI PR1 ON P1.ID=PR1.Pacchetto
GROUP BY S1.Luogo_arrivo);

```

5.4 Creazione Trigger

Infine, si procede alla creazione dei trigger, ovvero una specifica per la quale una data azione o funzione deve attivarsi in maniera del tutto automatica, ogni qual volta un'operazione stabilita viene eseguita su un oggetto specificato della base di dati. I trigger sono basati su tre concetti fondamentali: l'evento, che ha il compito di accedere al trigger; la condizione che deve verificarsi affinché il trigger venga eseguito; l'azione che viene eseguita se la condizione del trigger acceso è soddisfatta. Su tali principi si basa il paradigma E-C-A (Evento, Condizione, Azione). E' anche possibile che i trigger si attivino l'uno con l'altro, ossia che l'azione di un trigger possa essere l'evento innesco per un trigger successivo, in tal caso si ha una vera e propria cascata di due o più trigger. I trigger possono essere:

- *Trigger DML*, ovvero l'evento di innesco è data dalla manipolazione dei seguenti dati: INSERT, UPDATE o DELETE;
- *Trigger DDL*, chiamati anche trigger di sistema, poiché vengono innescati da eventi di sistema, come: l'avvio e la chiusura di un DB; la creazione o cancellazione di tabelle ecc.

Di seguito vengono riportati alcuni trigger:

```
532
533 --CREAZIONE TRIGGER
534
535 --Check di controllo per verificare che la data di arrivo non preceda quella di partenza
536 --nel caso stampare a video l'errore
537 CREATE OR REPLACE TRIGGER CK_SPOSTAMENTI
538 BEFORE INSERT ON SPOSTAMENTI
539 FOR EACH ROW
540 DECLARE ERRORE EXCEPTION;
541 BEGIN
542     IF :NEW.DATA_PARTENZA>:NEW.DATA_ARRIVO
543     THEN RAISE ERRORE;
544     END IF;
545 EXCEPTION
546 WHEN ERRORE
547 THEN RAISE_APPLICATION_ERROR(-20001,'LA DATA DI PARTENZA È SUCCESSIVA ALLA DATA DI ARRIVO');
548 END;
549
550 --Check di controllo dell'orario di arrivo nel verificare che sia succcessivo
551 --a quello di partenza, nel caso non fosse stampare a video che l'orario è errato
552 CREATE OR REPLACE TRIGGER Orapartenza_CK
553 BEFORE INSERT ON SPOSTAMENTI
554 FOR EACH ROW
555 DECLARE ERRORE EXCEPTION;
556 BEGIN
557     IF :NEW.Orario_arrivo<:NEW.Orario_partenza
558     THEN RAISE ERRORE;
559     END IF;
560 EXCEPTION
561 WHEN ERRORE
562 THEN RAISE_APPLICATION_ERROR(-20001,'ORARIO DI ARRIVO ERRATO');
563 END;
564
```

```
-- Check di controllo per il pagamento qualora fosse fatto con PayPal
--nel caso stampare a video che non si accetta tale modalità di pagamento
CREATE TRIGGER Circuito_UP
BEFORE INSERT ON CARTE_DI_CREDITO
FOR EACH ROW
BEGIN
:NEW.Circuito :=UPPER(:NEW.Circuito);
END;
CREATE OR REPLACE TRIGGER NO_PAYPAL
BEFORE INSERT ON CARTE_DI_CREDITO
FOR EACH ROW DECLARE ERRORE EXCEPTION;
BEGIN
    IF :NEW.Circuito ='PAYPAL'
    THEN RAISE ERRORE;
    END IF;
EXCEPTION
WHEN ERRORE THEN RAISE_APPLICATION_ERROR(-20001, 'NON SI ACCETTANO PAGAMENTI CON PAYPAL');
END;
--Check di controllo per verificare che il bonifico non sia stato
--effettuato dopo la data di partenza
CREATE OR REPLACE TRIGGER BonificoBanc_BEF_INS
BEFORE INSERT ON BONIFICI_BANCARI
FOR EACH ROW
DECLARE ERRORE EXCEPTION;
CHECKDATA DATE;
BEGIN
SELECT Data_partenza INTO CHECKDATA
FROM PRENOTAZIONI
WHERE UTENTE=:NEW.UTENTE;
    IF :NEW.Data_bonifico>CHECKDATA
    THEN RAISE ERRORE;
    END IF;
EXCEPTION
WHEN ERRORE
THEN RAISE_APPLICATION_ERROR(-20001, 'DATA BONIFICO NON VALIDA');
END;

--Check di controllo per verificare che la carta di credito non sia
--scaduta, nel caso stamapare a video che la carta è scaduta
CREATE OR REPLACE TRIGGER CartaCredito_BEF_INS
BEFORE INSERT ON CARTE_DI_CREDITO
FOR EACH ROW
DECLARE ERRORE EXCEPTION;
BEGIN
    IF :NEW.Data_scadenza<SYSDATE
    THEN RAISE ERRORE;
    END IF;
EXCEPTION
WHEN ERRORE
THEN RAISE_APPLICATION_ERROR(-20001, 'LA TUA CARTA DI CREDITO È SCADUTA');
END;
--Se il numero dei viaggiatori è maggiore di 10, si applica lo sconto del 20% sul costo complessivo
CREATE OR REPLACE TRIGGER SCONTO
BEFORE INSERT ON VIAGGIATORI
FOR EACH ROW
DECLARE Num_viaggiatori INTEGER;
BEGIN
SELECT COUNT(Id_viaggiatore) INTO Num_viaggiatori
FROM VIAGGIATORI
WHERE Prenotazione = :NEW.Prenotazione;
Num_viaggiatori := Num_viaggiatori+1;
    IF Num_viaggiatori>10
    THEN UPDATE PRENOTAZIONI
    SET Costo_complessivo= Costo_complessivo - Costo_complessivo * 0.20
    WHERE Codice_prenotazione=:NEW.Prenotazione;
    END IF;
END;
```