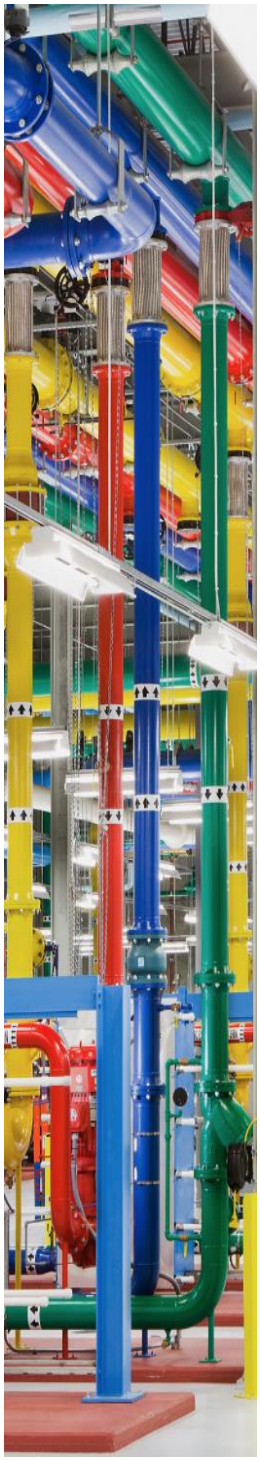
UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II

TESINA BASE DATI

PROGETTAZIONE BASE DI DATI PER UNA SOCIETÁ DI CAR POOLING

PROFESSORE: Vincenzo Moscato

Daniele Dario	N46003511
Di Chiara Guido	N46003206
Fordellone Domenico	N46003350
Gasparo Rippa Salvatore	N46003272



INDICE

CAPITOLO 1: INTRODUZIONE

1.1 Scopo

Si vuole progettare una base dati per una società italiana che consenta il car pooling tra viaggiatori sul territorio nazionale. In particolare la suddetta società intende fornire ai propri utenti un servizio per:

- creazione e condivisione di uno o più viaggi;
- consultazione e prenotazione per uno o più viaggi;

1.2 Specifiche sui dati

1.2.1 Dati generali

La società prevede che i servizi sopraelencati siano accessibili dai propri utenti mediante interfacce di un'applicazione smartphone. In particolare interessano le informazioni relative a:

- **viaggi** con codice, stato, ora partenza, durata, posti disponibili, contributo economico, numero prenotazioni non accettate;
- **utenti** caratterizzati da username, password, email, foto, cellulare e generalità. Questi si dividono in:
  - **autisti** con numero e data di scadenza della patente ed informazioni relative all'auto (marca, modello, colore);
  - **passeggeri** con numero e data di scadenza del documento di identità;
- **giudizio** con codice, numero di stelle e commento;

1.2.2 Previsioni di gestione

Si prevede che, nell'arco del primo anno di esercizio del sistema, la base di dati dovrà gestire la seguente mole di informazioni:

- l'anagrafica di circa 10000 utenti di cui:
- 2000 autisti;
- 8000 passeggeri;
- 8000 comuni;
- 100000 viaggi circa;
- 500000 prenotazioni circa;
- 700000 recensioni circa;

1.2.3 Dati sugli Utenti

Esistono 3 categoria di utenti che dovranno interagire con sistema di Base di Dati:

- il **DBA** che possiede **tutti i privilegi** possibili sullo schema di base dati;
- l'autista che possiede parte dei privilegi sullo schema di base di dati, questo può offrire nuovi viaggi;
- il passeggero che, tramite una web application, può effettuare prenotazioni per un dato viaggio disponibile;

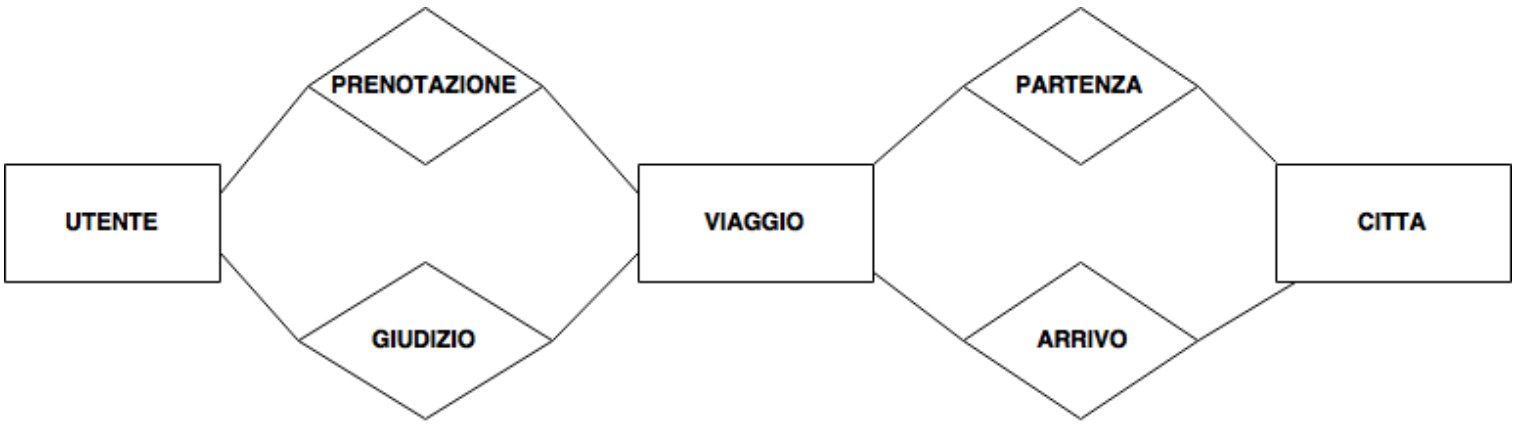
CAPITOLO 2: PROGETTAZIONE CONCETTUALE

2.1 Definizione schema E/R portante

Il primo passo per la progettazione di una base dati è quello di definire la struttura dello schema E/R portante. In questo definiamo le principali entità ed i legami concettuali tra queste. Nella nostra realtà di interesse le entità fondamentali analizzate sono:

1. Utente;
2. Viaggio;
3. Città;

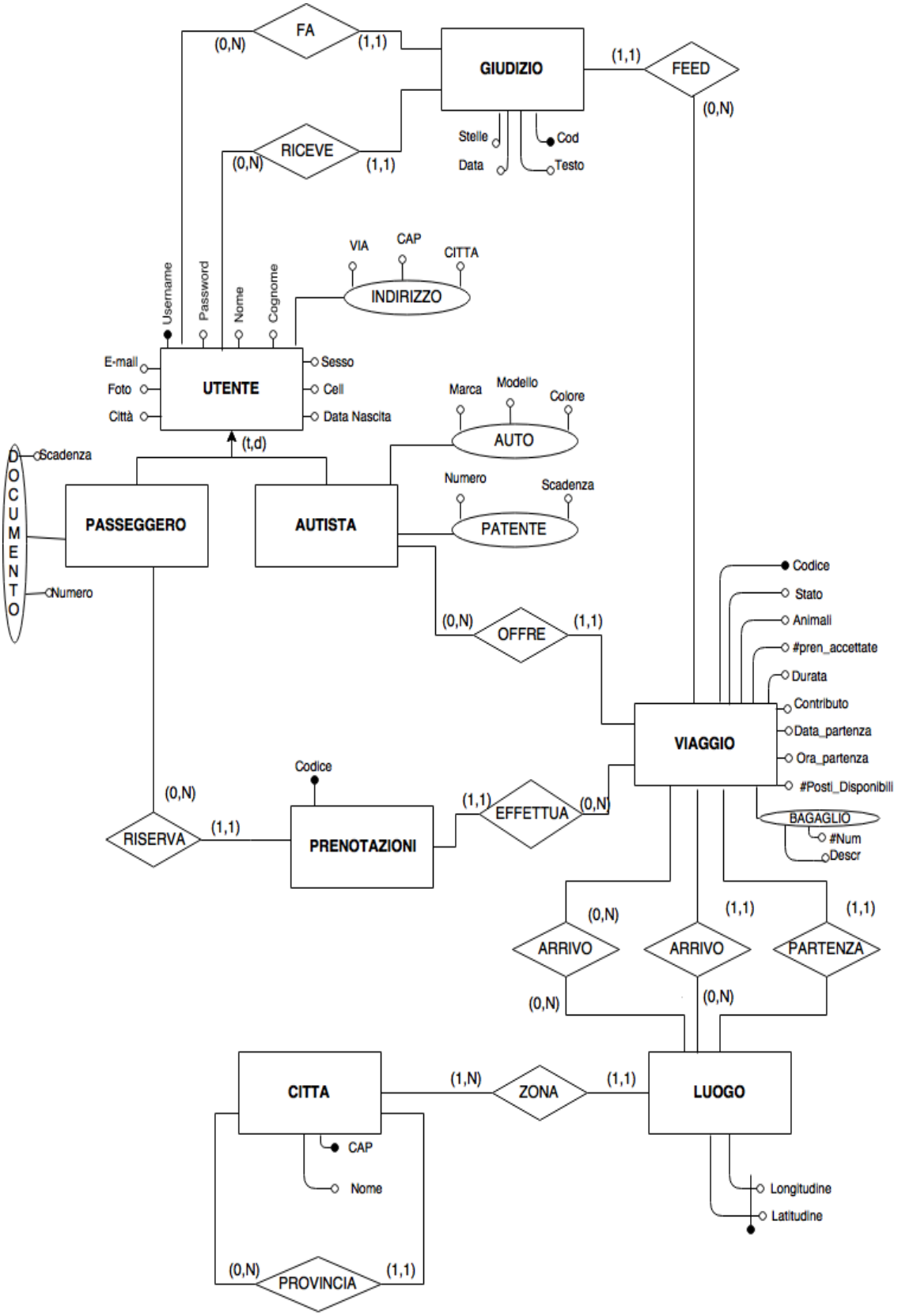
Queste sono collegate tra loro tramite associazioni al fine di offrire agli utenti le varie possibilità della realtà di interesse stessa.

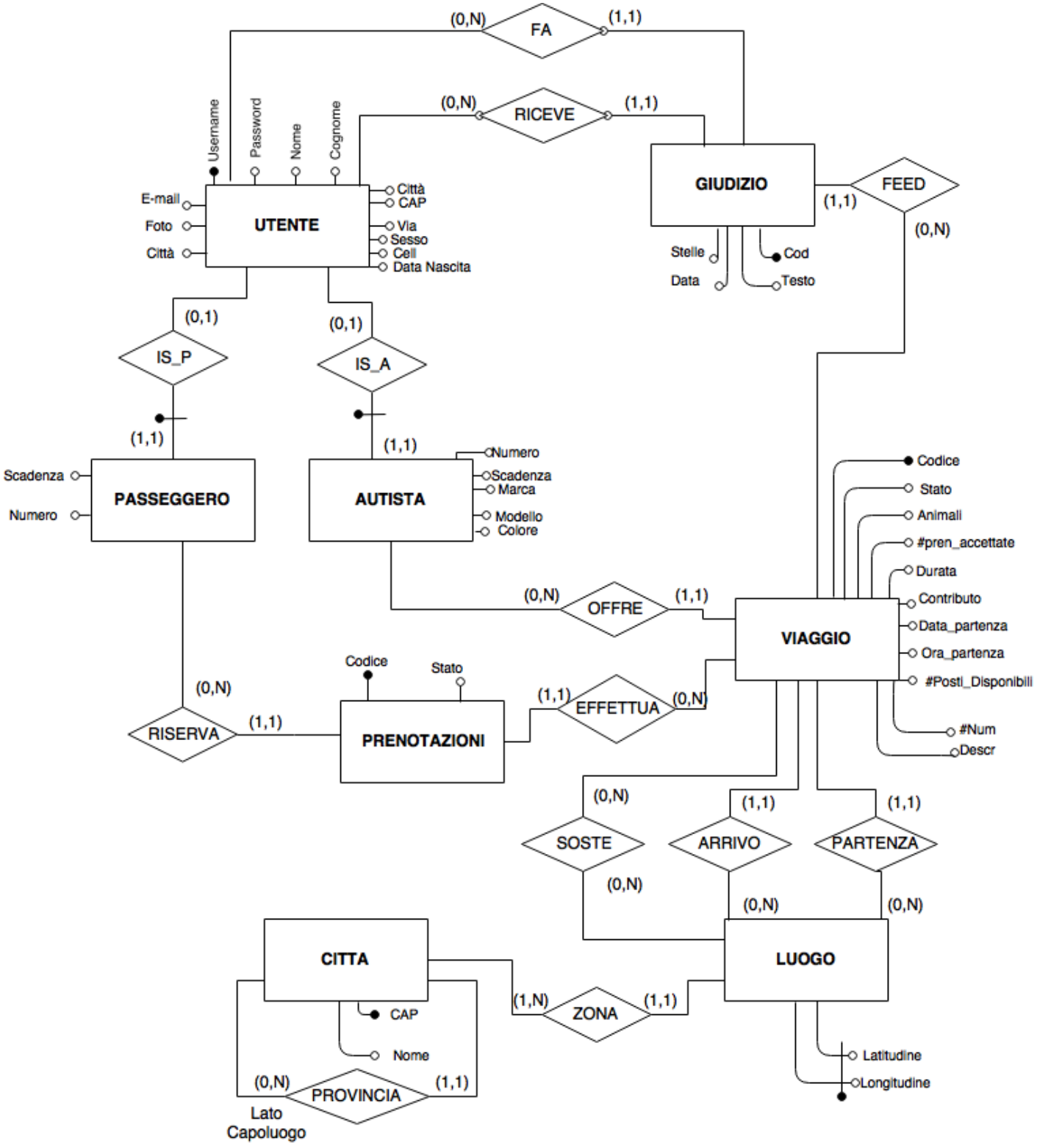


2.2 *Raffinamento*

A seguito di un attento processo di studio, si osservano numerose modifiche ed ampliamenti riguardanti lo schema portante. L'entità *Utente* viene risolta attraverso una gerarchia in cui l'Utente è l'entità Padre e le entità figlie sono *Passeggero* ed *Autista*. Ciascuna di queste è abilitata ad esprimere *Giudizi* tramite un'opportuna associazione.

L'Autista può offrire un *Viaggio* mentre il Passeggero, tramite l'associazione *Prenotazione*, può visualizzare i Viaggi offerti. Si necessità dell'aggiunta dell'entità *Luogo* per identificare in maniera precisa il luogo della partenza e di arrivo del viaggio.




FA
(0,N)
(1,1)
RICEVE
(0,N)
(1,1)
Username
Password
Nome
Cognome
UTENTE
E-mail
Foto
Città
Città
CAP
Via
Sesso
Cell
Data Nascita
GIUDIZIO
FEED
(1,1)
(0,N)
Stelle
Data
Cod
Testo
IS_P
IS_A
(0,1)
(0,1)
(1,1)
(1,1)
PASSEGGERO
Scadenza
Numero
AUTISTA
Numero
Scadenza
Marca
Modello
Colore
OFFRE
(0,N)
(1,1)
VIAGGIO
Codice
Stato
Animali
#pren_accettate
Durata
Contributo
Data_partenza
Ora_partenza
#Posti_Disponibili
#Num
Descr
RISERVA
(0,N)
(1,1)
PRENOTAZIONI
Codice
Stato
EFFETTUA
(1,1)
(0,N)
SOSTE
ARRIVO
PARTENZA
(0,N)
(1,1)
(1,1)
(0,N)
(0,N)
(0,N)
CITTA
CAP
Nome
LUOGO
ZONA
(1,N)
(1,1)
PROVINCIA
(0,N)
(1,1)
Lato
Capoluogo
Latitudine
Longitudine

CAPITOLO 3: PROGETTAZIONE LOGICA

3.1 SCELTE PROGETTUALI

Di seguito sono elencate le scelte progettuali effettuate al fine di fornire una visione quanto più chiara del modello base per il sistema.

- Un Autista può offrire nessuno o più Viaggi (0,N) mentre un Viaggio può essere offerto da uno ed un solo Autista (1,1).

  Il rapporto di cardinalità dell'associazione OFFRE è di tipo uno a molti.

- Un Passeggero può inserire nessuna o più Prenotazioni (0,N) mentre una Prenotazione è relativa ad uno ed un solo Passeggero (1,1)

  Il rapporto di cardinalità dell'associazione RISERVA è di tipo uno a molti.

- Un Giudizio può essere fatto/ricevuto da uno ed un solo Autista e/o Passeggero (1,1) mentre un Autista e/o Passeggero possono fare/ricevere nessuno o più Giudizi (0,N).

  Il rapporto di cardinalità delle associazioni FA,RICEVE è di tipo uno a molti.

- Un Viaggio può ricevere nessuna o più Prenotazioni (0,N) ma una Prenotazione è relativa ad uno ed un solo Viaggio (1,1).

  Il rapporto di cardinalità dell'associazione EFFETTUA è di tipo uno a molti.

- Un Viaggio può avere uno ed un solo Luogo di partenza e di arrivo (1,1) mentre questi possono essere scelti per essere associati a nessuno o piu Viaggi (0,N).

  Il rapporto di cardinalità delle associazioni ARRIVO e PARTENZA è di tipo uno a molti.
- Un Viaggio può avere nessuna o più Luoghi di soste (0,N) mentre questi possono essere scelti per essere associati a nessuno o piu Viaggi (0,N).

  Il rapporto di cardinalità dell'associazione SOSTE è di tipo nessuno a molti.

- Una Citta può essere Capoluogo di Provincia di nessuna o più Citta (0,N) mentre una Citta può avere una ed una sola Provincia (1,1).

  PROVINCIA è' una associazione ricorsiva il cui rapporto di cardinalità è di tipo uno a molti.

3.2 TRADUZIONE **Unique** **Chiave Primaria**

- VIAGGI(CODV , STATO, DATA_P , ORA_P, DURATA ,NUM_POSTI_DISP ,NUM_PREN_ATTESA,CONTRIBUTO, ANIMALI, NUMERO_BAGAGLI, DESCRIZIONE_BAGAGLI, CITTA_ARRIVO:LUOGO, CITTA_PARTENZA:LUOGO,SOSTE:LUOGO)

- CITTA(NOME, NOME, PROVINCIA:CITTA)

- LUOGO(LONGITUDINE, LATITUDINE, CITTA:CITTA)

- UTENTI(USERNAME, PW, NOME, COGNOME, SESSO, DATA_NASCITA, EMAIL, FOTO, CELLULARE, VIA, CITTA, CAP)

- PASSEGGERI(USERN:UTENTI,NUM_DOC, SCAD_DOC)

- AUTISTI(USERN:UTENTI,NUM_PAT, SCAD_PAT, MARCA_AUTO, MODELLO_AUTO, COLORE_AUTO)

- PRENOTAZIONI(COD,PASSEGGERO:PASSEGGERI,VIAGGIO:VIAGGI, STATO)

- GIUDIZI(CODICE,STELLE,COMMENTO,DATA,GIUDICANTE:UTENTI,GIUDICATO:UTENTI

3.3 DIMENSIONAMENTO DELLA BASE DI DATI

3.3.1 Operazioni Algebriche

Di seguito viene riportata una stima dello spazio occupato a fronte di un anno di lavoro, eseguendo eventuali supposizioni per informazioni non definite da progetto come per il numero di Giudizi. Per ogni tabella viene riportato il nome dei singoli campi che le vanno a formare, il tipo di dati scelti per essi e la relativa quantità di memoria occupata. Successivamente vengono eseguite le operazioni algebriche che permettono di calcolare con precisione il numero di byte, o suoi multipli, richiesti per l'allocamento di ogni tabella. La dimensione totale della base di dati è uguale alla somma delle dimensioni delle singole tabelle e un valore di tolleranza. Quest'ultimo è un valore adoperato per un'eventuale crescita della base dati.

TABELLA UTENTI

CAMPO	TIPO	SPAZIO RICHIESTO (B)
USERNAME	VARCHAR2(50)	50
PW	VARCHAR2(25)	25
NOME	VARCHAR2(30)	30
COGNOME	VARCHAR2(30)	30
SESSO	CHAR(1)	1
DATA_NASCITA	DATE	7
EMAIL	VARCHAR2(70)	70

FOTO	BLOB	4
CELLULARE	NUMBER(11)	11
VIA	VARCHAR2(35)	35
CITTA	VARCHAR2(20)	20
CAP	NUMBER(5)	5

In totale lo spazio richiesto è: 288 Byte

TABELLA PASSEGGERI

CAMPO	TIPO	SPAZIO RICHIESTO (B)
USERN	VARCHAR2(50)	50
NUM_DOC	VARCHAR2(10)	10
SCAD_DOC	DATE	7

In totale lo spazio richiesto è: 67 Byte

TABELLA AUTISTI

CAMPO	TIPO	SPAZIO RICHIESTO (B)
USERN	VARCHAR2(50)	50
MARCA_AUTO	VARCHAR2(20)	20
MODELLO_AUTO	VARCHAR2(20)	20
COLORE_AUTO	VARCHAR2(20)	20
NUM_PAT	VARCHAR2(10)	10
SCAD_PAT	DATE	7

In totale lo spazio richiesto è: 127 Byte

TABELLA PRENOTAZIONI

CAMPO	TIPO	SPAZIO RICHIESTO (B)
CODICE	NUMBER(6)	6
PASSEGGERO	VARCHAR2(50)	50
VIAGGIO	NUMBER(5)	5
STATO	CHAR(9)	9

In totale lo spazio richiesto è: 70 Byte

TABELLA GIUDIZI

CAMPO	TIPO	SPAZIO RICHIESTO (B)

CODICE	NUMBER(6)	6
STELLE	NUMBER(1)	3
COMMENTO	CLOB	4
GIUDICANTE	VARCHAR2(50)	50
GIUDICATO	VARCHAR2(50)	50

In totale lo spazio richiesto è: 113 Byte

TABELLA CITTA

CAMPO	TIPO	SPAZIO RICHIESTO (B)
NOME	VARCHAR2(20)	20
PROVINCIA	VARCHAR2(20)	20

In totale lo spazio richiesto è: 40 Byte

TABELLA LUOGO

CAMPO	TIPO	SPAZIO RICHIESTO (B)
LONGITUDINE	NUMBER(5)	5
LATITUDINE	VARCHAR2(35)	35
CITTA	VARCHAR2(20)	20

In totale lo spazio richiesto è: 60 Byte

TABELLA VIAGGI

CAMPO	TIPO	SPAZIO RICHIESTO (B)
CODV	NUMBER(5)	5
STATO	VARCHAR2(6)	6
DATA_P	DATE	7
ORA_P	DATE	7
DURATA	DATE	7
NUM_POSTI_DISPONIBILI	NUMBER(1)	3
NUM_PREN_ATTESA	NUMBER(2)	3
CONTRIBUTO	NUMBER(3,2)	5
DESCRIZIONE_BAGAGLI	VARVCHAR2(250)	250
NUMERO_BAGAGLI	NUMBER(1)	3
ANIMALI	CHAR(2)	3
DURATA	DATE	7
LUOGO_ARRIVO	VARCHAR2(20)	20
LUOGO_PARTENZA	VARCHAR2(20)	20

SOSTE	CHAR(2)	3

In totale lo spazio richiesto è: 349 Byte

3.3.2 Dimensionamento Finale

La tabella seguente sintetizza i requisiti di memorizzazione della base di dati.

Dimensione Totale Tablespace Dati:

TABELLA	OCCORRENZE	SPAZIO RICHIESTO	TOTALE
UTENTI	10000	288	2,8MB
PASSEGGERI	8000	67	536KB
AUTISTI	2000	127	254KB
PRENOTAZIONI	500000	70	35MB
VIAGGI	100000	349	34,9MB
GIUDIZI	700000	113	79,1MB
CITTA	8000	40	0,32MB
LUOGO	(8000*200)	60	96MB

Spazio totale richiesto: 247,70MB
Viene decisa la tolleranza α=52,30MB

Dimensione totale=247,70MB+α=***300MB***

Dimensione Tablespace Foto Utenti:

TABELLA	OCCORRENZE	SPAZIO RICHIESTO	TOTALE
FOTO	10000	1000	10GB

Dimensione totale=***10GB***

Dimensione Tablespace Commenti:
Supponendo che un commento sia composto da 600 caratteri e quindi che occupi 600B, moltiplicando per il numero dei giudizi, ovvero 700000, otterremo una grandezza richiesta complessiva di 420MB.
Viene decisa la tolleranza α=80MB

Dimensione totale=420MB+ α=**500MB**

Sono state fatte le seguenti ipotesi per le informazioni non definite dal progetto:

- In media vengono scritti 700000 giudizi;
- Vi sono in media 200 luoghi di incontro per comune, quindi le occorrenze di Luogo sono date dal prodotto del numero di comuni con il numero dei luoghi di incontro.
- Ogni foto utente può pesare al massimo 1MB;
- Ogni commento può essere formato da 600 caratteri e quindi pesare 600B;

CAPITOLO 4: PROGETTAZIONE FISICA

4.1 Creazione Tablespace

Come prima operazione è necessario effettuare le creazioni dei tablespace che conterranno il patrimonio dei dati presenti sul nostro Database. Chiameremo le nostre partizioni logiche "*cpooling_dati*" e "*foto_utenti*".

Tali operazioni, analogamente alla creazione dell'utente "cpooling_dba", sono da effettuare tramite le credenziali di SYSTEM, che, si ricorda, dispone di completa libertà operativa sul sistema.

UTENTE: SYSTEM

```
CREATE TABLESPACE cpooling_dati DATAFILE 'C:\Progetto\cpooling_dati.dbf' SIZE 300M;

CREATE TABLESPACE foto_utenti DATAFILE 'C:\Progetto\foto_utenti.dbf' SIZE 10G;

CREATE TABLESPACE commenti_utenti DATAFILE 'C:\Progetto\commenti_utenti.dbf' SIZE 500M;
```

4.2 Creazione Utenti

E' ora necessario creare i 3 utenti che interagiranno con la base dati.

Il DBA dei tablespace prima definiti, riceverà pieni poteri esclusivamente all'interno di quest'ultimi senza possibilità di operazione al di fuori di essi.

I Passeggeri potranno visualizzare Viaggi ed effettuare Prenotazioni

Gli Autisti potranno inserire nuovi Viaggi e gestire quelli inseriti.

Per assegnare i privilegi nostro DBA utilizzeremo il comando GRANT assegnando …

DA DEFINIRE UTENTI CON RELATIVA GESTIONE DEGLI ACCESSI E PURE I RUOLI SONO DA DEFINIRE

UTENTE: SYSTEM

```
CREATE USER cpooling_dba IDENTIFIED BY 1234 DEFAULT TABLESPACE
cpooling_dati;

GRANT DBA,UNLIMITED TABLESPACE TO cpooling_dba;
```

4.3 Creazione Tabelle

Da questo momento in poi, la creazione delle tabelle prevede l'utilizzo dell'utente DBA appena creato.

Sono stati utilizzati i vincoli *NOT NULL* per determinati campi ,considerati essenziali, per preservare la consistenza dell'intera base dati. Laddove si è presentata la necessità (per la creazione per esempio di chiavi primarie di più campi) è stato usato il campo *CONSTRAINT* nella creazione delle tabelle.

UTENTE: CPOOLING_DBA

```
CREATE TABLE UTENTI(
USERNAME VARCHAR2(50) PRIMARY KEY,
PASSWORD VARCHAR2(25) NOT NULL,
NOME     VARCHAR2(30) NOT NULL,
COGNOME  VARCHAR2(30) NOT NULL,
SESSO    CHAR(1)      NOT NULL,
DATA_NASCITA DATE     NOT NULL,
EMAIL    VARCHAR2(70) NOT NULL UNIQUE,
FOTO             BLOB,
CELLULARE NUMBER(11)  NOT NULL,
VIA              VARCHAR2(35) NOT NULL,
CITTA   VARCHAR2(20) NOT NULL,
CAP              NUMBER(5)       NOT NULL
)LOB(FOTO) STORE AS LOB_FOTO(TABLESPACE foto_utenti)
STORAGE(INITIAL 2800K);

CREATE TABLE PASSEGGERI(
USERN VARCHAR2(50) PRIMARY KEY,
NUM_DOC VARCHAR2(10) NOT NULL,
SCAD_DOC DATE       NOT NULL
)STORAGE (INITIAL 536K);

CREATE TABLE AUTISTI(
USERN VARCHAR2(50) PRIMARY KEY,
NUM_PAT VARCHAR2(10) NOT NULL,
SCAD_PAT DATE          NOT NULL,
MARCA_AUTO VARCHAR2(20) NOT NULL,
MODELLO_AUTO VARCHAR2(20) NOT NULL,
COLORE_AUTO VARCHAR2(20)
) STORAGE (INITIAL 254K);

CREATE TABLE PRENOTAZIONI(
CODICE NUMBER(6) PRIMARY KEY,
STATO CHAR(9),
```

```
PASSEGGERO VARCHAR2(50),
VIAGGIO NUMBER(5) NOT NULL
) STORAGE (INITIAL 30500K)

CREATE TABLE VIAGGI(
CODV NUMBER(5) PRIMARY KEY,
STATO VARCHAR2(6) NOT NULL,
DATA_P DATE NOT NULL,
ORA_P DATE NOT NULL,
DURATA DATE NOT NULL,
NUM_POSTI_DISPONIBILI NUMBER(1),
NUM_ATTESA NUMBER(2),
CONTRIBUTO NUMBER(3,2),
DESCRIZIONE_BAGAGLI VARCHAR2(250),
NUMERO_BAGAGLI NUMBER(1) NOT NULL,
ANIMALI         CHAR(2) NOT NULL,
CITTA_ARRIVO          VARCHAR2(20) NOT NULL,
CITTA_PARTENZA VARCHAR2(20) NOT NULL,
SOSTE CHAR(2) NOT NULL
) STORAGE (INITIAL 34500 K)

CREATE TABLE GIUDIZI(
CODICE NUMBER(6) PRIMARY KEY,
STELLE NUMBER(1) NOT NULL,
COMMENTO CLOB     NOT NULL,
GIUDICANTE VARCHAR2(50),
GIUDICATO VARCHAR2(50),
) STORAGE (INITIAL 79100K)

CREATE TABLE CITTA(
NOME VARCHAR2(20) PRIMARY KEY,
PROVINCIA VARCHAR2(20) NOT NULL
) STORAGE (INITIAL 320K)
```

```
CREATE TABLE LUOGO(
CAP NUMBER(5),
VIA VARCHAR2(35),
CITTA VARCHAR2(20),
CONSTRAINT PK_LUOGO PRIMARY KEY (CAP,VIA)
) STORAGE (INITIAL 60000K)
```

4.4 Creazione Vincoli

Tramite i comandi di ALTER TABLE, tramite il quali possiamo modificare tabelle già esistenti, possiamo aggiungere vincoli di integrità referenziale. Si è preferito tale metodo in modo da non essere condizionati dall'ordine di creazione delle tabelle, data la necessità di dover in caso contrario definire prima le tabelle referenziate poi le referenziati.

Si noti che dove non è stata definita nessuna politica di cancellazione ORACLE per default imposta il NO ACTION non compiendo nessuna azione sulla base dati. Le altre politiche di reazione usate sono:

- • SET NULL: che setta automaticamente a NULL tutti gli attributi che sono collegati, come chiave esterna, a quello eliminato;
- • SET DEFAULT: che imposta gli attributi collegati a quello cancellato a 0;
- • CASCADE: elimina automaticamente tutti gli oggetti associati all'attributo eliminato;

*UTENTE: **CPOOLING_DBA***

```
ALTER TABLE PASSEGGERI ADD CONSTRAINT FK_UTENTI_PASSEGGERI FOREIGN KEY (Usern) REFERENCES UTENTI (Username) ON DELETE CASCADE;

ALTER TABLE AUTISTI ADD CONSTRAINT FK_UTENTI_AUTISTI FOREIGN KEY (Usern) REFERENCES UTENTI(Username) ON DELETE CASCADE;
```

```
ALTER TABLE PRENOTAZIONI ADD CONSTRAINT FK_PRENOTAZIONI_PASSEGGERO
FOREIGN KEY (Passeggero) REFERENCES PASSEGGERI(Usern) ON DELETE SET NULL;

ALTER TABLE PRENOTAZIONI ADD CONSTRAINT FK_PRENOTAZIONI_VIAGGIO
FOREIGN KEY (Viaggio) REFERENCES VIAGGI(Codv) ON DELETE CASCADE;

ALTER TABLE VIAGGI ADD CONSTRAINT FK_VIAGGI_ARRIVO FOREIGN KEY
(Citta_Arrivo) REFERENCES CITTA(Nome);

ALTER TABLE VIAGGI ADD CONSTRAINT FK_VIAGGI_PARTENZA FOREIGN KEY
(Citta_Partenza) REFERENCES CITTA(Nome);

ALTER TABLE GIUDIZI ADD CONSTRAINT FK_GIUDIZI_GIUDICANTE FOREIGN KEY
(Giudicante) REFERENCES UTENTI(Username) ON DELETE SET NULL;

ALTER TABLE GIUDIZI ADD CONSTRAINT FK_GIUDIZI_GIUDICATO FOREIGN KEY
(Giudicato) REFERENCES UTENTI(Username) ON DELETE SET NULL;

ALTER TABLE CITTA ADD CONSTRAINT FK_CITTA FOREIGN KEY (Provincia)
REFERENCES CITTA(Nome);

ALTER TABLE LUOGHI ADD CONSTRAINT FK_LUOGHI FOREIGN KEY (Citta)
REFERENCES CITTA(Nome);
```

Osservazioni:

- Sulla cancellazione di un Viaggio segue la cancellazione di tutte le Prenotazioni relative a quel viaggio;
- Sulla cancellazione di un Utente segue la sostituzione in Prenotazioni del nome utente con il valore NULL;

4.5 Creazione Indici

Per velocizzare la ricerca da parte degli Utenti sulle città di Partenza ed Arrivo dei Viaggi si creano due Indici sui campi Citta_Arrivo e Citta_Partenza della tabella Viaggi. Inoltre si crea anche un Indice sul campo Giudicato della tabella Giudizi al fine di velocizzare le ricerche delle recensioni ricevute dall'Utente.

UTENTE: ***CPOOLING_DBA***

```
CREATE INDEX index_citta_arrivo ON VIAGGI(Citta_Arrivo ASC);

CREATE INDEX index_citta_partenza ON VIAGGI(Citta_Partenza ASC);

CREATE INDEX index_giudicato ON GIUDIZI(Giudicato ASC);
```

4.6 Creazione Trigger

Un trigger è una stored procedure che viene eseguita quando si verifica un particolare evento sulla nostra Base Dati. Nel caso in esame viene sfruttata la potenza del DBMS Oracle 11g grazie al quale possiamo integrare una serie di trigger in PL/SQL volte a rendere le informazioni quanto più indipendenti possibili dai livelli superiori. Di seguito sono riportati alcuni dei Trigger utilizzati per rendere automatiche le regole di business applicate.

UTENTE: ***CPOOLING_DBA***

```
CREATE OR REPLACE TRIGGER Chiusura_Viaggio
AFTER UPDATE ON PRENOTAZIONI
FOR EACH ROW
DECLARE
```

```
Conteggio INTEGER;
Numero          INTEGER;
BEGIN
SELECT COUNT(*) INTO Conteggio
FROM Prenotazioni
WHERE Viaggio=:OLD.Viaggio AND
Stato='Accettata';
SELECT Num_Posti_Disponibili INTO Numero
FROM Viaggi
WHERE Codv=:NEW.Viaggio;
IF Numero=Conteggio THEN
UPDATE Viaggi SET Stato='Chiuso'
WHERE Codv=:NEW.Viaggio;
END IF;
END;


CREATE OR REPLACE TRIGGER Controllo_Data_Viaggio
BEFORE INSERT ON VIAGGI
FOR EACH ROW
DECLARE
Errore EXCEPTION;
BEGIN
IF :NEW.DATA_P<TO_CHAR(SYSDATE,'DD-Mon-YYYY') THEN RAISE Errore;
END IF;
EXCEPTION WHEN Errore
        THEN RAISE_APPLICATION_ERROR (-20001,'Data non valida!');
END;


CREATE OR REPLACE TRIGGER Controllo_Data_Documento
BEFORE INSERT ON PASSEGGERI
FOR EACH ROW
DECLARE
```

```
Errore EXCEPTION;
BEGIN
IF :NEW.SCAD_DOC<TO_CHAR(SYSDATE,'DD-Mon-YYYY')
        THEN RAISE Errore;
END IF;
EXCEPTION
WHEN Errore THEN RAISE_APPLICATION_ERROR (-20002,'Documento scaduto!');
END;

CREATE OR REPLACE TRIGGER Controllo_Data_Patante
BEFORE INSERT ON AUTISTI
FOR EACH ROW
DECLARE
Errore EXCEPTION;
BEGIN
IF :NEW.SCAD_PAT<TO_CHAR(SYSDATE,'DD-Mon-YYYY')
        THEN RAISE Errore;
END IF;
EXCEPTION
WHEN Errore THEN RAISE_APPLICATION_ERROR (-20003,'Patente scaduta!');
END;


CREATE OR REPLACE TRIGGER Agg_Prenotazioni_Attesa_Plus
AFTER INSERT ON PRENOTAZIONI
FOR EACH ROW
BEGIN
UPDATE VIAGGI
SET Num_Attesa = Num_Attesa+1
WHERE CODV=:NEW.Viaggio;
END;


CREATE OR REPLACE TRIGGER Agg_Prenotazioni_Attesa_Minus
```

```
AFTER UPDATE ON PRENOTAZIONI
FOR EACH ROW
BEGIN
IF (:OLD.STATO='IN_ATTESA' AND :NEW.STATO='ACCETTATA' ) OR
  (:OLD.STATO='IN_ATTESA' AND :NEW.STATO='RIFIUTATA' )  THEN
     UPDATE VIAGGI
     SET Num_Attesa = Num_Attesa-1
     WHERE CODV=:NEW.Viaggio;
END IF;
END;
```

- Chiusura_Viaggio:

Chiusura del Viaggio al fronte del raggiungimento massimo di prenotazioni accettate a seconda del numero posti disponibili per quel Viaggio.

- Controllo_Data_Viaggio :

All'atto dell'inserimento di un Viaggio controlla se la Data impostata è successiva a quella corrente di Sistema.

- Controllo_Data_Documento e Controllo_Data_Patente:

Controllano che le date di scadenza dei rispettivi documenti siano valide;

- Agg_Prenotazioni_Attesa:

A valle di un inserimento in Prenotazioni incrementa il numero di Prenotazioni in Attesa per quel Viaggio;

- Agg_Prenotazioni_Attesa:

A seguito di una modifica dello stato di una Prenotazione viene decrementato il numero di Prenotazioni in Attesa del Viaggio corrispondente;

Osservazioni:

- Num_Attesa è il totale delle prenotazioni in attesa di accettazione/rifiuto e quelle già rifiutate;

- Num_Posti_Disponibili è un valore statico utile per la corretta gestione dello stato del Viaggio;

4.7 Creazione Sequenze

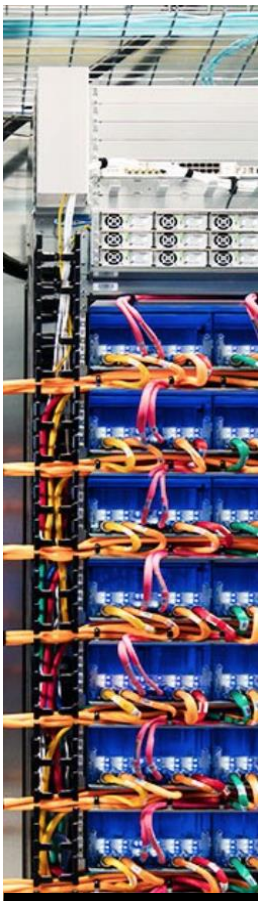
Per l'inserimento di alcuni campi abbiamo scelto di utilizzare la struttura "sequenza", messa a disposizione da Oracle. Le sequenze permettono di assegnare, in automatico, i valori in alcuni campi come codici, facilitando le operazioni di inserimento.

UTENTE: CPOOLING_DBA

```
CREATE SEQUENCE CODV_SEQ
START WITH 1
INCREMENT BY 1
NOCYCLE;


CREATE SEQUENCE CODICE_PREN_SEQ
START WITH 1
INCREMENT BY 1
NOCYCLE;


CREATE SEQUENCE CODICE_GIUD_SEQ
START WITH 1
INCREMENT BY 1
NOCYCLE;
```



“La vita è un percorso di conoscenza. Vivere è imparare!”

Lorenz Konrad

Bibliografia

[1] A. Chianese, V.Moscato, A.Picariello, *Sistemi di basi di dati e applicazioni*, Santarcangelo Romagna, Maggiolini Editore, 2015