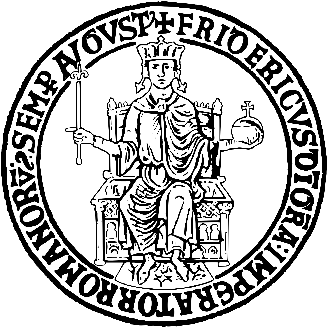
UNIVERSITA DEGLI STUDI DI NAPOLI FEDERICO II

SCUOLA POLITECNICA E DELLE SCIENZE DI BASE

DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL’INFORMAZIONE



Progettazione e sviluppo di una Base di Dati Relazionale per la gestione di un Sistema di Tracciamento Contatti per Ristoranti

Bruno De Vivo

N86003676

Antonio Giordano

N86003842

Indice

[**Descrizione del Progetto** 5](#_Toc108244202)

[1.1 Analisi del Problema: 5](#_Toc108244203)

[1.2 Realizzazione: 5](#_Toc108244204)

[**Progettazione Concettuale** 6](#_Toc108244205)

[2.1 Introduzione 6](#_Toc108244206)

[2.2 Class Diagram 6](#_Toc108244207)

[2.3 Ristrutturazione 6](#_Toc108244208)

[2.4 Class Diagram Ristrutturato 7](#_Toc108244209)

[2.5 Dizionario delle Classi 8](#_Toc108244210)

[2.6 Dizionario delle Associazioni 9](#_Toc108244211)

[2.7 Dizionario dei Vincoli 10](#_Toc108244212)

[**Progettazione Logica** 11](#_Toc108244213)

[3.1 Introduzione 11](#_Toc108244214)

[3.2 Traduzione 11](#_Toc108244215)

[**Progettazione Fisica** 12](#_Toc108244216)

[4.1 Introduzione 12](#_Toc108244217)

[4.2 Definizione delle Tabelle 13](#_Toc108244218)

[4.2.1 Ristorante 13](#_Toc108244219)

[4.2.2 Sala 14](#_Toc108244220)

[4.2.3 Tavolo 15](#_Toc108244221)

[4.2.4 TavoliAdiacenti 16](#_Toc108244222)

[4.2.5 Tavolata 17](#_Toc108244223)

[4.2.6 Cliente 18](#_Toc108244224)

[4.2.7 Cameriere 19](#_Toc108244225)

[4.2.8 Prenotazione 20](#_Toc108244226)

[4.2.9 Servizio 20](#_Toc108244227)

[4.3 Viste 21](#_Toc108244228)

[4.3.1 Prenotazioni Ristorante 21](#_Toc108244229)

[4.4 Vincoli – Triggers 21](#_Toc108244230)

[4.4.1 Check\_Adiacenza\_Stesso\_Tavolo 21](#_Toc108244231)

[4.4.2 Check\_Cameriere\_Ristorante 21](#_Toc108244232)

[4.4.3 Check\_Date 22](#_Toc108244233)

[4.4.4 Check\_Max\_Avventori 22](#_Toc108244234)

[4.4.5 Check\_Tavoli\_Sale 22](#_Toc108244235)

[4.5 Triggers Attributi Calcolati e Trigger semplici 23](#_Toc108244236)

[4.5.1 Capienza (Sala) – Insert 23](#_Toc108244237)

[4.5.2 Capienza (Sala) – Delete 23](#_Toc108244238)

[4.5.3 Capienza (Ristorante) – Insert 24](#_Toc108244239)

[4.5.4 Capienza (Ristorante) – Delete 24](#_Toc108244240)

[4.5.5 Numero Camerieri – Insert 25](#_Toc108244241)

[4.5.6 Numero Camerieri – Delete 25](#_Toc108244242)

[4.5.7 Numero Tavoli – Insert 25](#_Toc108244243)

[4.5.8 Numero Tavoli – Delete 26](#_Toc108244244)

[4.5.9 Numero Clienti – Insert 26](#_Toc108244245)

[4.5.10 Numero Clienti – Delete 26](#_Toc108244246)

Capitolo 1

# Descrizione del Progetto

## 1.1 Analisi del Problema:

Ci si impone di costruire una Base di Dati utile per il tracciamento contatti per vari locali di ristorazione, nello specifico lo scopo è di poter rilevare: i clienti seduti ad un determinato tavolo; i clienti seduti ai tavoli adiacenti nella stessa data; i camerieri che hanno effettuato il servizio per quel tavolo. La necessità nasce dal poter agilmente tenere nota degli individui entrati in contatto per poter effettuare una segnalazione agli interessati e alle autorità sanitarie nel caso di una denuncia di caso di COVID-19.

## 1.2 Realizzazione:

Il sistema permette di contenere informazioni riguardo vari ristoranti, i quali sono caratterizzati dal nome del ristorante, un identificativo e attributi calcolati quali il numero di camerieri e la capienza; questi sono associati a tutte le rispettive sale che contengono, queste sono caratterizzate dal nome della sala, il codice del ristorante associato e attributi calcolati quali il numero di tavoli presenti all’interno della sala e la capienza di clienti; queste a loro volta sono associate a tutti i tavoli contenuti. Ogni tavolo è fornito di un codice identificativo, il nome della sala di cui fa parte e il numero massimo di avventori che possono sedersi al tavolo; il tavolo è associato ad una tavolata e a sé stesso attraverso un’associazione ricorsiva che contiene i codici tutti i tavoli che sono adiacenti ad altri tavoli e a quali. La tavolata è invece considerabile come la raccolta di tutti i clienti che siederanno a un determinato tavolo ad una determinata data, oltre a questi dati questa relazione contiene anche il numero di clienti seduti al tavolo come attributo calcolato; questa è associata alle relazioni di “Prenotazione” e di “Servizio” le quali sono classi di associazione che servono a rappresentare rispettivamente le prenotazioni che i clienti fanno per sedere a una tavolata e l’insieme di camerieri che serve la tavolata. Ogni cliente è identificato dal codice alfanumerico della carta d’identità, e ogni cameriere da un codice identificativo assegnatogli dal sistema. Quest’ultimo è poi associato al ristorante presso il quale presta servizio, in modo da impedire che nel sistema venga assegnato un cameriere a servire presso più tavolate nella stessa data e in ristoranti diversi.

Capitolo 2

# Progettazione Concettuale

## 2.1 Introduzione

La progettazione concettuale si impone di produrre lo schema progettuale nel suo livello di astrazione maggiore, completamente slegata da una realizzazione fisica, e quindi dai mezzi attraverso la quale il progetto diventa concreto.

## 2.2 Class Diagram

Per questo progetto è stato scelto l’UML come linguaggio per rappresentare il progetto, in tutte le sue entità e gli attributi di ognuna. Attraverso questo prezioso strumento si sono potute individuare agilmente le finestre di miglioramento del progetto e la costruzione di vincoli per permettere il corretto funzionamento del sistema.

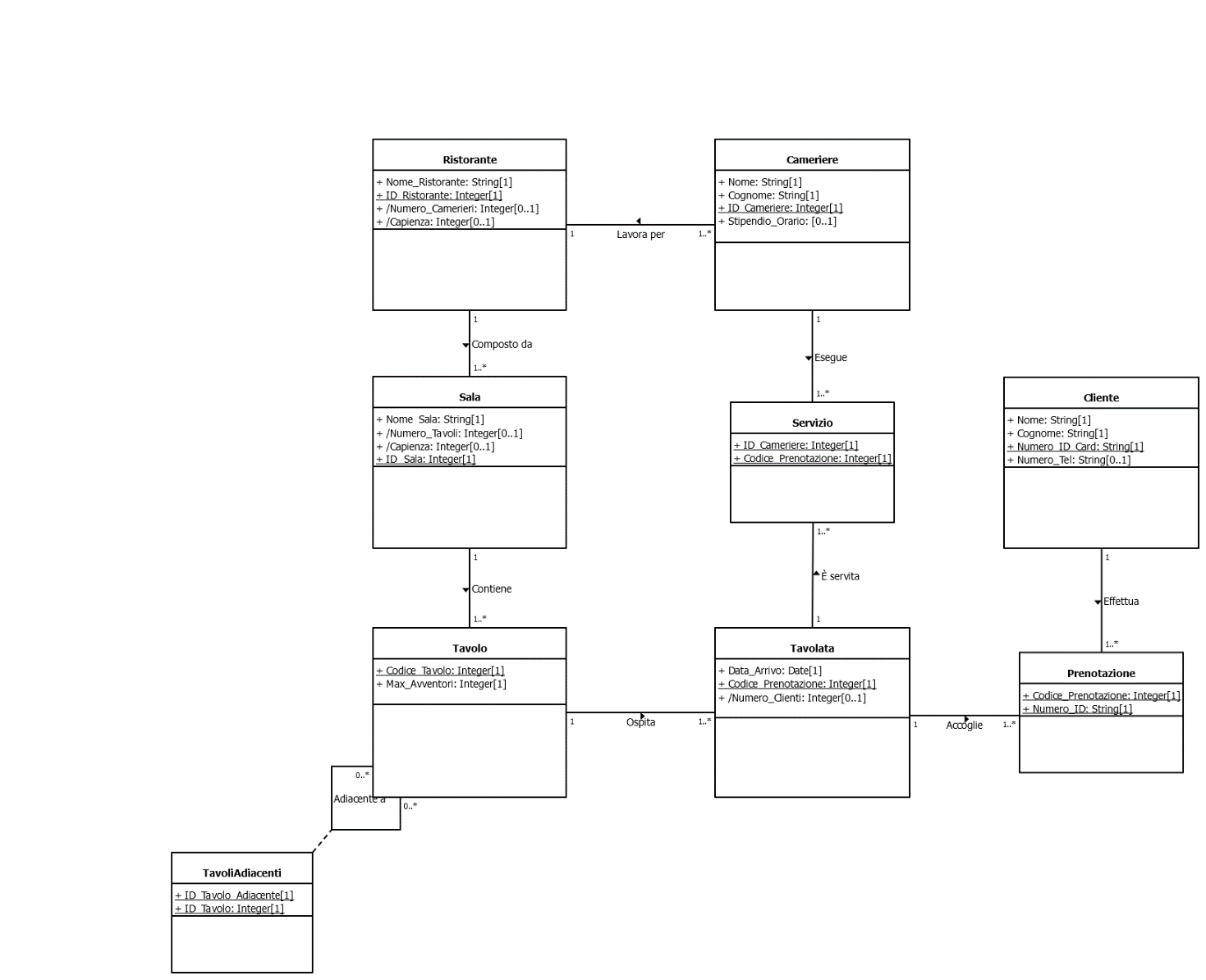
## 2.3 Ristrutturazione

La ristrutturazione del Class Diagram è un passaggio importante che va effettuato per alleggerire il carico del progetto il più possibile, tenendo comunque conto del target per il quale la base di dati sta venendo costruita.

Nel processo di ristrutturazione vengono rimosse le informazioni ridondanti, snelliti gli attributi multipli e le generalizzazioni. Solitamente i Class Diagram vengono utilizzati anche in contesti di Programmazione Orientata agli Oggetti, va quindi tenuto conto del fatto che benché sia possibile, in fase di ristrutturazione, il Class Diagram deve essere leggibile in un contesto relazionale.

Nel nostro caso la ristrutturazione del Class Diagram rispetto all’idea di partenza ha previsto di eliminare una generalizzazione delle entità “Cameriere” e “Cliente”, le quali avevano una classe generalizzata “Persona”, dalla quale ereditavano gli attributi di “Nome” e “Cognome”, mantenuti poi nelle entità ristrutturate.

## 2.4 Class Diagram Ristrutturato



## 2.5 Dizionario delle Classi

|  |  |  |
| --- | --- | --- |
| Classe | Descrizione | Attributi |
| Ristorante | Descrittore di un ristorante | **Nome\_Ristorante** (string)[1]: Nome del ristorante.  **Numero\_Camerieri** (integer) [0..1]: Attributo calcolato che esprime il numero di camerieri che prestano servizio al ristorante.  **Capienza** (integer) [0..1]: Attributo calcolato che esprime il numero massimo di clienti che il ristorante può ospitare contemporaneamente.  **ID\_Ristorante** (integer) [1]: Chiave tecnica che identifica ogni elemento della classe *Ristorante* univocamente. |
| Sala | Descrittore di una sala di un ristorante | **Nome\_Sala** (String) [1]: Nome della sala.  **Numero\_Tavoli** (Integer) [0..1]: Attributo calcolato che esprime il numero di tavoli presenti nella sala.  **Capienza** (Integer) [0..1]: Attributo calcolato che esprime il numero massimo di clienti che la sala può ospitare contemporeaneamente.  **ID\_Sala** (Integer) [1]: Chiave tecnica che identifica ogni elemento della classe *Sala* univocamente. |
| Tavolo | Descrittore di un tavolo in una sala | **Max\_Avventori** (Integer) [1]: Attributo che esprime il numero massimo di clienti che possono sedersi ad un tavolo contemporaneamente.  **Codice\_Tavolo** (Integer) [1]: Chiave tecnica che identifica ogni elemento della classe *Tavolo* univocamente. |
| Tavolata | Descrittore di una tavolata che siede ad un tavolo | **Data\_Arrivo** (Date) [1]: Attributo che esprime la data nella quale la tavolata ha preso posto nel ristorante.  **Numero\_Clienti** (Integer) [0..1]: Attributo calcolato che esprime il numero di clienti presenti alla tavolata.  **Codice\_Prenotazione** (Integer) [1]: Chiave tecnica che identifica ogni elemento della classe *Tavolata* univocamente. |
| Cameriere | Descrittore di un cameriere | **Nome** (String) [1]: Nome del cameriere.  **Cognome** (String) [1]: Cognome del cameriere.  **ID\_Cameriere** (Integer) [1]: Chiave tecnica che identifica ogni elemento della classe *Cameriere* univocamente. |
| Cliente | Descrittore di un cliente | **Nome** (String) [1]: Nome del cliente.  **Cognome** (String) [1]: Cognome del cliente.  **Numero\_Tel** (String) [0..1]: Numero di telefono del cliente.  **Numero\_ID\_Card** (String) [1]: Codice identificativo della carta di identità del cliente. |

## 2.6 Dizionario delle Associazioni

|  |  |  |  |
| --- | --- | --- | --- |
| Associazione | Descrizione | Classi Coinvolte | Classi di Associazione |
| Composto da | Esprime il concetto secondo il quale una o più sale compongono un ristorante | **Ristorante [1..\*]** *ruolo* (**Composto da**): Indica l’elemento di cui è composto il ristorante.  **Sala [1]** *ruolo* (**Compone**): Indica l’elemento di cui Sala è componente. | *Nessuna classe di associazione* |
| Contiene | Esprime il concetto secondo il quale una sala contiene dei tavoli al suo interno | **Sala [1..\*]** *ruolo* (**Contiene**): Indica l’elemento presente all’interno della sala.  **Tavolo [1]** *ruolo* (**Contenuto**): Indica l’elemento in cui Tavolo è contenuto. | *Nessuna classe di associazione* |
| Adiacente a | Esprime il concetto di adiacenza tra tavoli | **Tavolo [0..\*]** *ruolo* (**Adiacente a**): Indica l’elemento a cui Tavolo è adiacente.  **Tavolo [0..\*]** *ruolo* (**Adiacente a**): Indica l’elemento a cui Tavolo è adiacente. | **TavoliAdiacenti:**  ID\_Tavolo: Integer [1]  ID\_Tavolo\_Adiacente: Integer [1] |
| Ospita | Esprime il concetto per cui un tavolo ospita una tavolata | **Tavolo [0..\*]** *ruolo* (**Ospita**): Indica l’elemento che Tavolo ospita.  **Tavolata [1]** *ruolo* (**Ospitato da**): Indica l’elemento che ospita la Tavolata. |  |
| Servizio | Esprime il concetto per cui un cameriere serve una tavolata | **Tavolata [1..\*]** *ruolo* (**È servita** **da**): Indica l’elemento da cui viene servita la tavolata.  **Cameriere** **[1..\*]** *ruolo* (**Serve**): Indica l’elemento che il cameriere serve. | **Servizio:**  ID\_Cameriere: Integer [1]  Codice\_Prenotazione: Integer [1] |
| Prenotazione | Esprime il concetto per cui un cliente si prenota per una tavolata | **Tavolata [1..\*]** *ruolo* (**Accoglie**): Indica l’elemento che la tavolata accoglie.  **Cliente [1..\*]** *ruolo* (**Si prenota per**): Indica l’elemento per cui si prenota il cliente. | **Prenotazione:**  Codice\_Prenotazione: Integer [1]  Numero\_ID: String [1] |
| Lavora per | Esprime il concetto per cui un cameriere lavora in un ristorante | **Cameriere [1]** *ruolo* (**Lavora per**): Indica l’elemento per cui il cameriere lavora.  **Ristorante [1..\*]** *ruolo* (**Fa lavorare**):Indica l’elemento che il ristorante fa lavorare. |  |

## 2.7 Dizionario dei Vincoli

|  |  |  |
| --- | --- | --- |
| Vincolo | Tipo | Descrizione |
| unique\_nome\_ristorante | Intrarelazionale | Garantisce l’unicità del *nome* del *ristorante.* |
| avventori\_positivo | Dominio | Impedisce l’inserimento di un numero minore di 1 come massimo di *avventori* a un tavolo. |
| ck\_only\_numbers | Dominio | Impedisce l’inserimento di valori che non siano numeri per l’attributo *Numero\_Tel* di *Cliente.* |
| ck\_lunghezza\_id | Dominio | La lunghezza del codice della carta d’identità non può essere diversa dal valore stabilito. |
| unique\_numero\_tel | Intrarelazionale | Garantisce l’unicità del *numero di telefono* di un *cliente*. |
| check\_adiacenza\_stesso\_tavolo | N-upla | Impedisce di inserire un *tavolo* come adiacente a sé stesso. |
| check\_cameriere\_ristorante | Interrelazionale | Impedisce di inserire un *cameriere* a servire presso un *ristorante* per il quale non lavora e a più tavoli in ristoranti diversi contemporaneamente. |
| check\_date | Interrelazionale | Impedisce l’inserimento di una *tavolata* per una data in cui il *tavolo* scelto è già occupato. |
| check\_max\_avventori | Interrelazionale | Impedisce l’inserimento di un numero di *clienti* maggiore rispetto a quanti il *tavolo* scelto per la tavolata ne possa ospitare. |
| check\_tavoli\_sale | Interrelazionale | Impedisce che un tavolo venga inserito come adiacente a un tavolo appartenente ad una diversa sala |
| /Numero\_Camerieri | Attributo Calcolato | Si ottiene contando il numero di tutti i *camerieri* che lavorano per un dato *ristorante.* |
| /Capienza (Sala) | Attributo Calcolato | Si ottiene sommando gli attributi *Max\_Avventori* di tutti i *tavoli* presenti nella *sala*. |
| /Capienza (Ristorante) | Attributo Calcolato | Si ottiene sommando gli attributi *Capienza* di tutte le *sale* presenti nel *ristorante*. |
| /Numero\_Tavoli | Attributo Calcolato | Si ottiene contando il numero di tutti i *tavoli* presenti nella *sala*. |
| /Numero\_Clienti | Attributo Calcolato | Si ottiene contando il numero di tutti i *clienti* prenotati per la *tavolata* |

Capitolo 3

# Progettazione Logica

## 3.1 Introduzione

Dopo aver completato la progettazione concettuale si passa ad un livello di astrazione intermedio. Da qui si sceglie il modello di dati da adottare, che sarà quello relazionale, in virtù della conoscenza che si ha di quest’ultimo verranno quindi tradotte le idee ottenute in fase di progettazione concettuale in una forma più concreta e vicina a quella che sarà poi la progettazione fisica che tratteremo più avanti.

Verranno quindi espresse le entità secondo uno schema logico attinente al modello relazionale puro, e implementati gli elementi importanti per far sì che il modello relazionale possa avere senso.

Per esattezza, le chiavi primarie saranno sottolineate e le chiavi secondarie saranno in corsivo.

## 3.2 Traduzione

**Ristorante:** (ID\_Ristorante, Nome\_Ristorante, Numero\_Camerieri, Capienza)

**Sala:** (ID\_Sala, Nome\_Sala, Numero\_Tavoli, Capienza, *ID\_Ristorante*)

ID\_Ristorante -> Ristorante.ID\_Ristorante

**Tavolo:** (Codice\_Tavolo, Max\_Avventori, *ID\_Sala*)

ID\_Sala -> Sala.ID\_Sala

**TavoliAdiacenti:** (*ID\_Tavolo*, *ID\_Tavolo\_Adiacente*)

ID\_Tavolo -> Tavolo.Codice\_Tavolo

ID\_Tavolo\_Adiacente -> Tavolo.Codice\_Tavolo

**Tavolata:** (Codice\_Prenotazione, Data\_Arrivo, Numero\_Clienti, *Codice\_Tavolo*)

Codice\_Tavolo -> Tavolo.Codice\_Tavolo

**Cliente:** (Nome, Cognome, Numero\_ID\_Card, Numero\_Tel)

**Cameriere:**  (Nome, Cognome, Stipendio\_Orario, ID\_Cameriere, *ID\_Ristorante*)

ID\_Ristorante -> Ristorante.ID\_Ristorante

**Servizio:**  (*ID\_Cameriere*, *Codice\_Prenotazione*)

ID\_Cameriere -> Cameriere.ID\_Cameriere

Codice\_Prenotazione -> Tavolata.Codice\_Prenotazione

**Prenotazione:**  (*Codice\_Prenotazione*, *Numero\_ID*)

Codice\_Prenotazione -> Tavolata.Codice\_Prenotazione

Numero\_ID -> Cliente.Numero\_ID\_Card

Capitolo 4

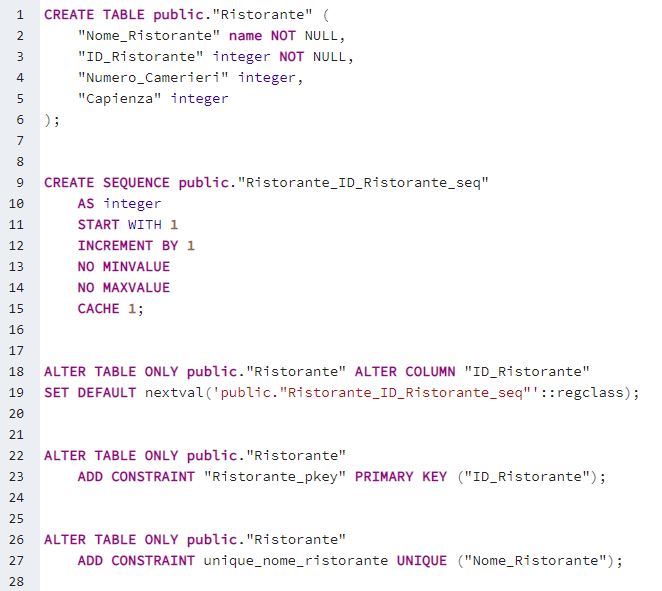
# Progettazione Fisica

## 4.1 Introduzione

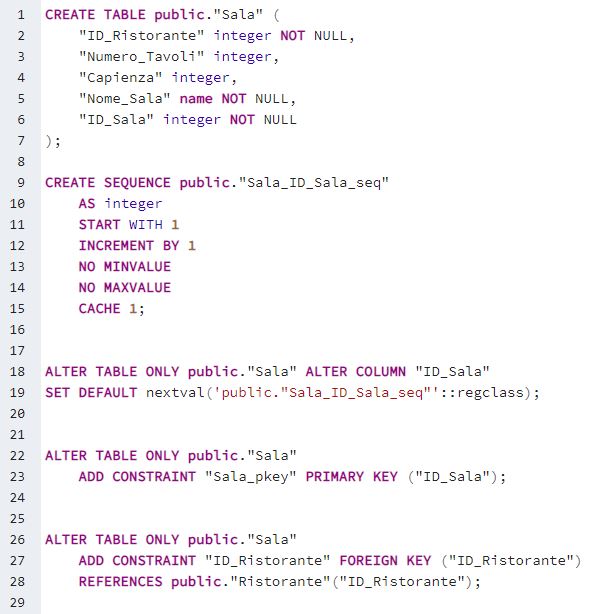
Per la realizzazione fisica del database è stato scelto PostgreSQL come DBMS. Le variazioni rispetto alle differenti progettazioni in quanto a nomenclature e simili sono assenti, in fase di revisione si è proceduto a uniformare tutti i nomi di attributi ed entità in modo da avere una presentazione coerente del progetto dall’inizio alla fine.

## 4.2 Definizione delle Tabelle

### 4.2.1 Ristorante



### 4.2.2 Sala



### 4.2.3 Tavolo



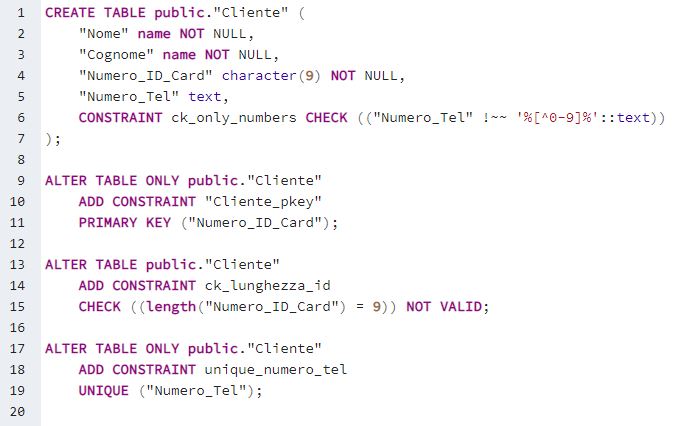
### 4.2.4 TavoliAdiacenti



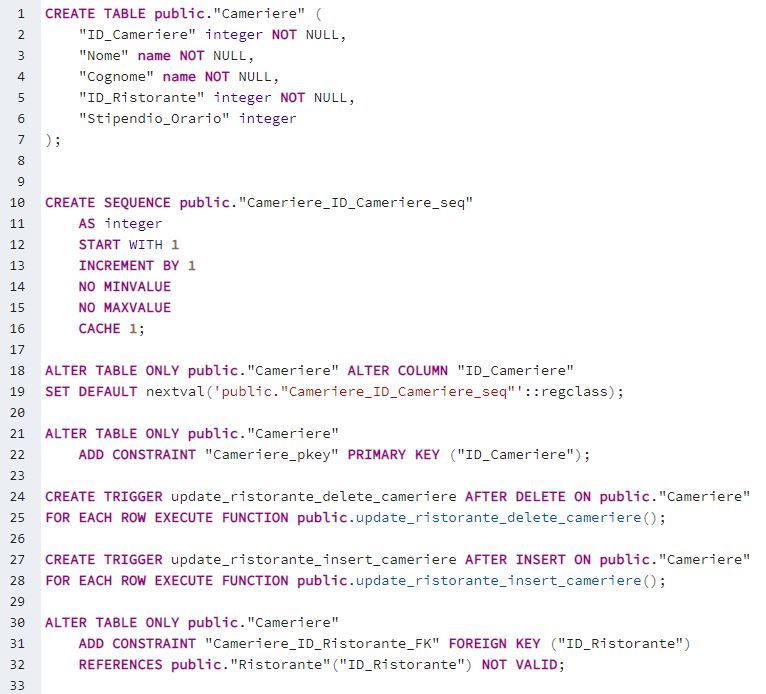
### 4.2.5 Tavolata



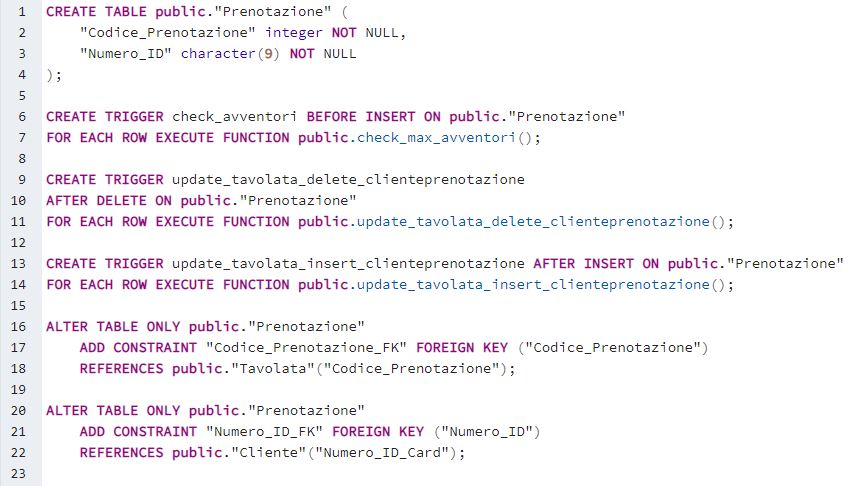
### 4.2.6 Cliente



### 4.2.7 Cameriere



### 4.2.8 Prenotazione

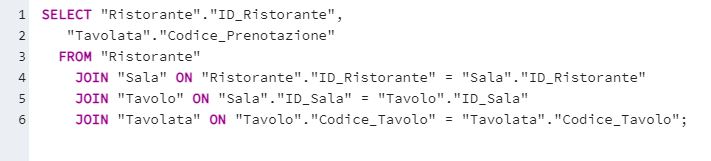


### 4.2.9 Servizio



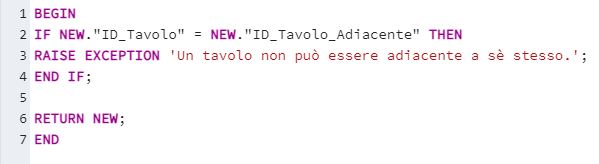
## 4.3 Viste

### 4.3.1 Prenotazioni Ristorante

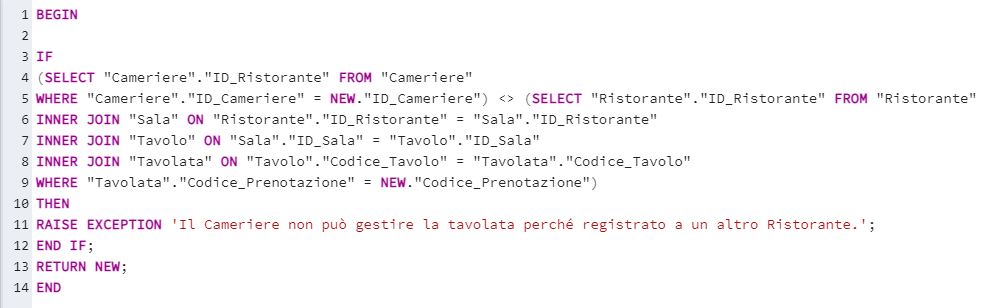


## 4.4 Vincoli – Triggers

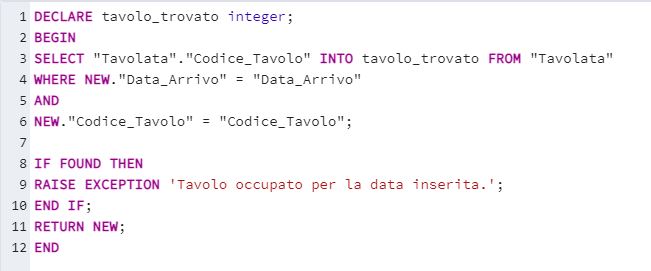
### 4.4.1 Check\_Adiacenza\_Stesso\_Tavolo



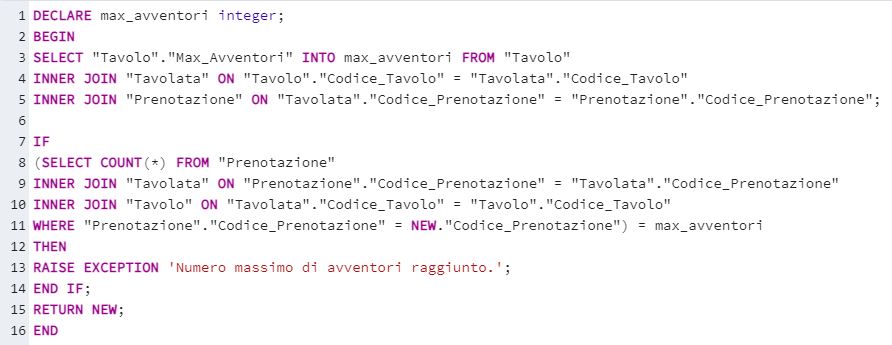
### 4.4.2 Check\_Cameriere\_Ristorante



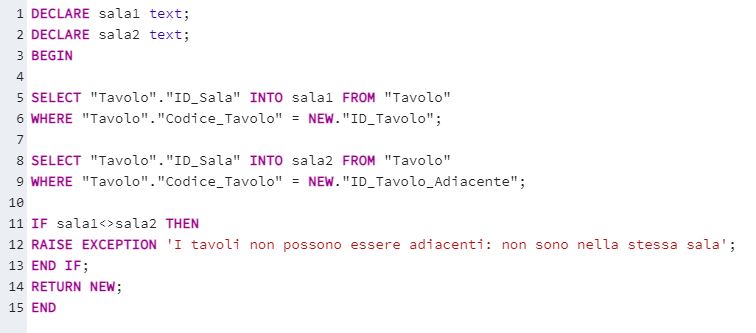
### 4.4.3 Check\_Date



### 4.4.4 Check\_Max\_Avventori

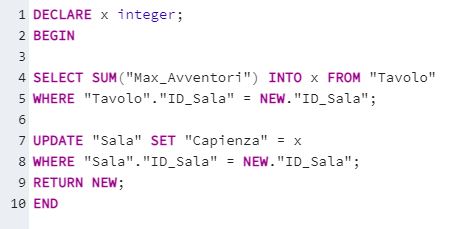


### 4.4.5 Check\_Tavoli\_Sale

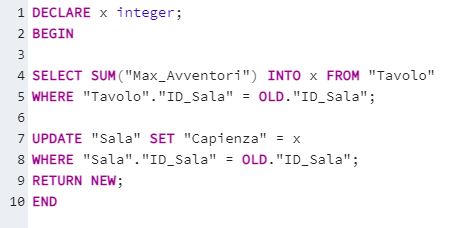


## 4.5 Triggers Attributi Calcolati e Trigger semplici

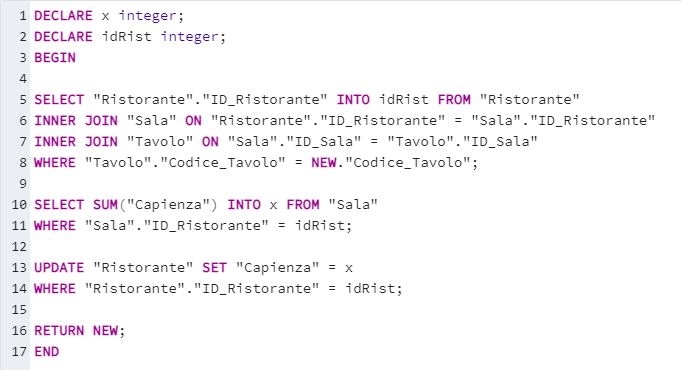
### 4.5.1 Capienza (Sala) – Insert



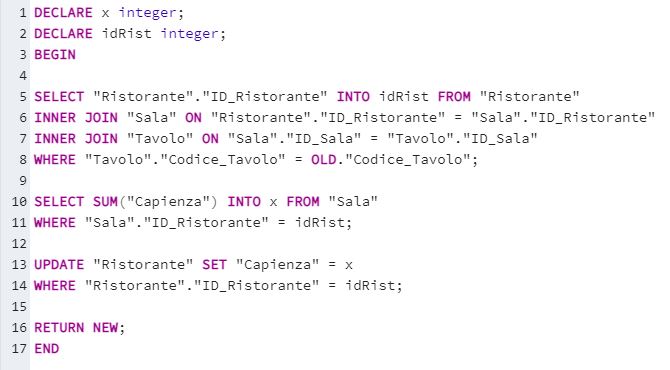
### 4.5.2 Capienza (Sala) – Delete



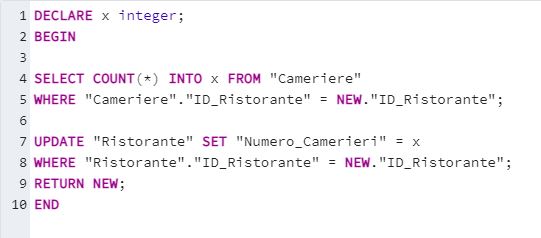
### 4.5.3 Capienza (Ristorante) – Insert



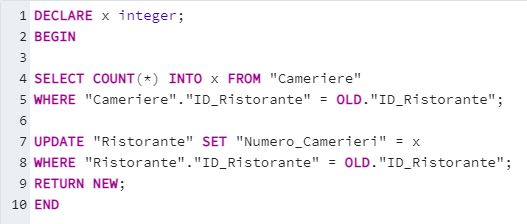
### 4.5.4 Capienza (Ristorante) – Delete



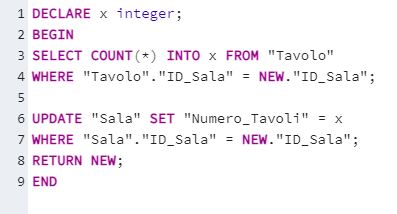
### 4.5.5 Numero Camerieri – Insert



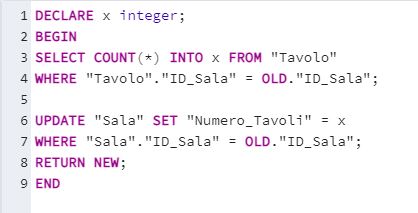
### 4.5.6 Numero Camerieri – Delete



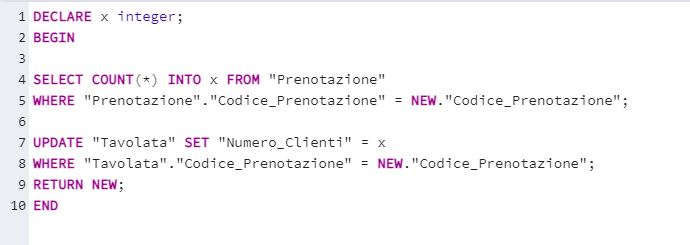
### 4.5.7 Numero Tavoli – Insert



### 4.5.8 Numero Tavoli – Delete



### 4.5.9 Numero Clienti – Insert



### 4.5.10 Numero Clienti – Delete

