





**Corso di laurea in Informatica  
Insegnamento di Basi di Dati I  
Anno accademico 2021/2022**

**Progettazione e sviluppo di una base di dati relazionale per la gestione di un sistema di tracciamento contatti per ristoranti**

*Docenti:*

Prof. Adriano Peron

Prof. Sergio Di Martino

Prof. Luigi Libero Lucio Starace

*Autori:*

Angelo Di Maio

Matricola N86003699

*angelo.dimaio3@studenti.unina.it*

Santolo Barretta

Matricola N86003666

*san.barretta@studenti.unina.it*

23 dicembre 2021

*Questa pagina è stata lasciata intenzionalmente bianca.*

# **Indice**

[Indice 3](#_Toc94188511)

[Capitolo 1 6](#_Toc94188512)

[Descrizione del progetto 6](#_Toc94188513)

[1.1 Descrizione sintetica e analisi del problema 6](#_Toc94188514)

[1.2 L’organizzazione del sistema di tracciamento contatti 6](#_Toc94188515)

[Capitolo 2 8](#_Toc94188516)

[Progettazione concettuale 8](#_Toc94188517)

[Introduzione 8](#_Toc94188518)

[2.1 Alcune convenzioni per la lettura dei class diagram 8](#_Toc94188519)

[2.2 Class diagram 9](#_Toc94188520)

[2.3 Ristrutturazione del class diagram 10](#_Toc94188521)

[2.3.1 Analisi delle informazioni ridondanti 10](#_Toc94188522)

[2.3.2 Analisi degli identificativi 10](#_Toc94188523)

[2.3.3 Rimozione degli attributi strutturati 10](#_Toc94188524)

[2.3.4 Rimozione delle classi di associazione 11](#_Toc94188525)

[2.3.5 Rimozione delle gerarchie di specializzazione 11](#_Toc94188526)

[2.3.6 Class diagram ristrutturato 12](#_Toc94188527)

[2.4 Dizionario dei dati per il class diagram ristrutturato 13](#_Toc94188528)

[2.4.1 Dizionario delle classi 13](#_Toc94188529)

[2.4.2 Dizionario delle associazioni 17](#_Toc94188530)

[2.4.3 Dizionario dei vincoli 19](#_Toc94188531)

[2.4.4 Dizionario delle interrogazioni 21](#_Toc94188532)

[Capitolo 3 23](#_Toc94188533)

[Progettazione logica 23](#_Toc94188534)

[Introduzione 23](#_Toc94188535)

[3.1 Schema logico 23](#_Toc94188536)

[3.1.1 Traduzione in schemi relazionali 23](#_Toc94188537)

[3.1.2 Traduzione delle associazioni 25](#_Toc94188538)

[3.1.3 Schema logico generale 26](#_Toc94188539)

[Capitolo 4 27](#_Toc94188540)

[Progettazione fisica 27](#_Toc94188541)

[Introduzione 27](#_Toc94188542)

[4.1 Note sull’implementazione 27](#_Toc94188543)

[4.2 Definizione delle tabelle 27](#_Toc94188544)

[4.2.1 Definizione della tabella PROPRIETARIO 28](#_Toc94188545)

[4.2.2 Definizione della tabella RISTORANTE 28](#_Toc94188546)

[4.2.3 Definizione della tabella MANAGERRISTORANTE 29](#_Toc94188547)

[4.2.4 Definizione della tabella SALA 29](#_Toc94188548)

[4.2.5 Definizione della tabella CAMERIERE 30](#_Toc94188549)

[4.2.6 Definizione della tabella TAVOLO 30](#_Toc94188550)

[4.2.7 Definizione della tabella TAVOLATA 31](#_Toc94188551)

[4.2.8 Definizione della tabella AVVENTORE 32](#_Toc94188552)

[4.2.9 Definizione della tabella CASO 33](#_Toc94188553)

[4.3 Funzioni, procedure ed altre automazioni 34](#_Toc94188554)

[4.3.1 Stored function IS\_NUMBER 34](#_Toc94188555)

[4.3.2 Stored procedure NUMERO\_DI\_TELEFONO\_LEGALE 34](#_Toc94188556)

[4.3.3 Stored procedure PASSWORD\_LEGALE 35](#_Toc94188557)

[4.4 Viste 36](#_Toc94188558)

[4.4.1 Vista TAVOLATE 36](#_Toc94188559)

[4.5 Implementazione dei vincoli 37](#_Toc94188560)

[4.5.1 Implementazione del vincolo Password legale 37](#_Toc94188561)

[4.5.2 Implementazione del vincolo Numero di telefono legale 38](#_Toc94188562)

[4.5.3 Implementazione del vincolo Cap legale 39](#_Toc94188563)

[4.5.4 Implementazione del vincolo Età cameriere legale 39](#_Toc94188564)

[4.5.5 Implementazione del vincolo MaxAvventori legale 40](#_Toc94188565)

[4.5.6 Implementazione del vincolo Capienza legale 40](#_Toc94188566)

[4.5.7 Implementazione del vincolo Data nascita legale 41](#_Toc94188567)

[4.5.8 Implementazione del vincolo Has greenpass 41](#_Toc94188568)

[4.5.9 Implementazione del vincolo Temperatura avventore legale 42](#_Toc94188569)

[4.5.10 Implementazione del vincolo Somma avventori a tavolata legale 42](#_Toc94188570)

[4.5.11 Implementazione del vincolo Data registrazione caso 43](#_Toc94188571)

[Capitolo 5 44](#_Toc94188572)

[Esempio d’uso 44](#_Toc94188573)

[Introduzione 44](#_Toc94188574)

[5.1 Popolamento del database 44](#_Toc94188575)

# **Capitolo 1**

# **Descrizione del progetto**

## **1.1 Descrizione sintetica e analisi del problema**

Si provvederà alla progettazione e allo sviluppo di una base di dati relazionale per la gestione del tracciamento di contatti COVID-19 in ristoranti. Il sistema permetterà di tenere traccia dei contatti di un avventore in un ristorante, fornendo ulteriori informazioni per il tracciamento degli eventuali avventori risultati positivi al virus SARS-CoV-2. In particolare, il sistema deve permettere di gestire il tracciamento dei contatti di uno o più ristoranti. Ciascun ristorante è organizzato in una o più sale, identificate da un nome, le quali contengono uno o più tavoli. Ciascun tavolo presente all’interno di una sala sarà identificato da un codice univoco, avrà l’indicazione del numero massimo di avventori che possono sedersi a quel tavolo e dei tavoli adiacenti ad esso. Il sistema viene gestito da operatori che tengono traccia delle tavolate formate dagli avventori nei ristoranti. Un avventore sarà identificato da un nome, un cognome, un numero di carta d’identità e un numero di telefono. La tavolata viene identificata dalla data di arrivo degli avventori, i camerieri che l’hanno servita e il tavolo ad essa assegnata. I ristoranti effettueranno soltanto servizio serale nella fascia oraria 20 – 22, quindi in ciascuna data, al più una tavolata deve essere associata ad un dato tavolo, poiché il servizio è unico. Inoltre, il numero di avventori che partecipano ad una tavolata non deve superare il numero massimo di avventori che possono sedersi al tavolo a cui essa è assegnata. Infine, il sistema dovrà permettere la visualizzazione di alcune statistiche su base mensile e giornaliera, tra cui il numero totale di avventori accolti da ciascun ristorante.

## **1.2 L’organizzazione del sistema di tracciamento contatti**

Di seguito verrà descritta brevemente la **chiave di lettura** data al sistema da implementare, sulla quale si basa la progettazione della base di dati relazionale.

Si suppone che il sistema di tracciamento contatti COVID-19 venga utilizzato da uno o più ristoranti appartenenti ad una stessa catena, ubicati fisicamente nel territorio italiano e con gruppi di avventori accolti prevalentemente italiani. Il sistema verrà gestito dalla figura dell’Operatore il quale, una volta autenticato nella piattaforma sviluppata, con un Username e una Password, si occuperà di alcuni aspetti rilevanti, fra cui:

* **la gestione e l’inserimento nella piattaforma di tavolate di avventori**: si intende come Tavolata l’atto di associare i tavoli presenti in una sala del ristorante ad un singolo o ad un gruppo di avventori. La tavolata verrà identificata quindi da una data di arrivo degli avventori facenti parte della stessa e da un orario di arrivo e uno di uscita, che coincideranno sempre con quelli dell’unico servizio serale effettuato (nella fascia oraria 20-22). L’Operatore che gestisce la piattaforma si occuperà poi di registrare anche le informazioni sugli avventori facenti parte di una determinata tavolata, tra cui: le informazioni anagrafiche, la temperatura registrata dall’avventore al suo arrivo al ristorante (se oltre i 37.5° non sarà possibile registrare l’Avventore nel sistema) ed inoltre se possiede o meno green pass (se un Avventore non possiede green pass non sarà possibile associarlo ad un tavolo presente in una sala Interna ma solo ad un tavolo presente in una sala Esterna, se il ristorante ne è dotato). In questo modo gli operatori che gestiscono il sistema terranno traccia di tutte le tavolate formate in un ristorante;
* **la gestione e l’inserimento nella piattaforma di casi COVID-19 avvenuti nel ristorante**: nel caso in cui un avventore registrato ad una tavolata in un determinato ristorante risultasse positivo nei giorni seguenti al virus SARS-CoV-2, dopo sua preventiva segnalazione al ristorante (ad esempio tramite form online o telefonata), sarà compito degli operatori aprire un nuovo Caso nella piattaforma e gestirlo. La gestione di un Caso comprende l’inserimento in piattaforma dei dati dell’avventore risultato positivo, quindi la sua anagrafica, la data di registrazione del caso, il suo stato e delle eventuali note associate al caso. Lo stesso vale anche per eventuali camerieri risultati positivi. Si suppone che lo stato di un caso possa essere “NonRisolto” quando viene solamente inserito in piattaforma, “InRisoluzione” quando un operatore prenderà in carico il caso e avviserà tutti gli avventori che hanno avuto contatti con l’avventore risultato positivo e “Risolto” quando saranno stati avvisati tutti gli avventori venuti in contatto con l’avventore positivo. In questo modo il sistema permetterà di tenere traccia dei contatti di un avventore in un ristorante.

La figura dell’Operatore che gestisce l’intero sistema descritto può essere svolta da:

* uno o più Manager che gestiscono un determinato ristorante, i quali avranno dei privilegi limitati nel sistema relativi solo alla gestione di casi e all’inserimento di tavolate;
* direttamente dal Proprietario che amministra il ristorante o la catena formata da più ristoranti ed il quale avrà privilegi “*ad hoc*” nel sistema come la possibilità di aggiungere, modificare o rimuovere ristoranti, sale, tavoli presenti in piattaforma, oltre che la gestione del personale presente per quanto riguarda le figure del Manager e del Cameriere.

Per ulteriori chiarimenti generali si rimanda al successivo capitolo riguardante la **progettazione concettuale**, in cui è definito il **class diagram** della base di dati e viene riportato il **dizionario dei dati**.

# **Capitolo 2**

# **Progettazione concettuale**

## **Introduzione**

In questo capitolo viene affrontata la progettazione della base di dati al livello di astrazione più alto, ossia quello della progettazione concettuale. Dall’analisi iniziale del problema e dei requisiti che devono essere soddisfatti si arriverà ad uno schema concettuale indipendente dalla struttura dei dati e dall’implementazione fisica della base di dati. Lo schema concettuale verrà rappresentato usando un class diagram UML, in cui saranno presenti le entità rilevanti individuate ai fini della rappresentazione dei dati e le relazioni che intercorrono tra esse.

## **2.1 Alcune convenzioni per la lettura dei class diagram**

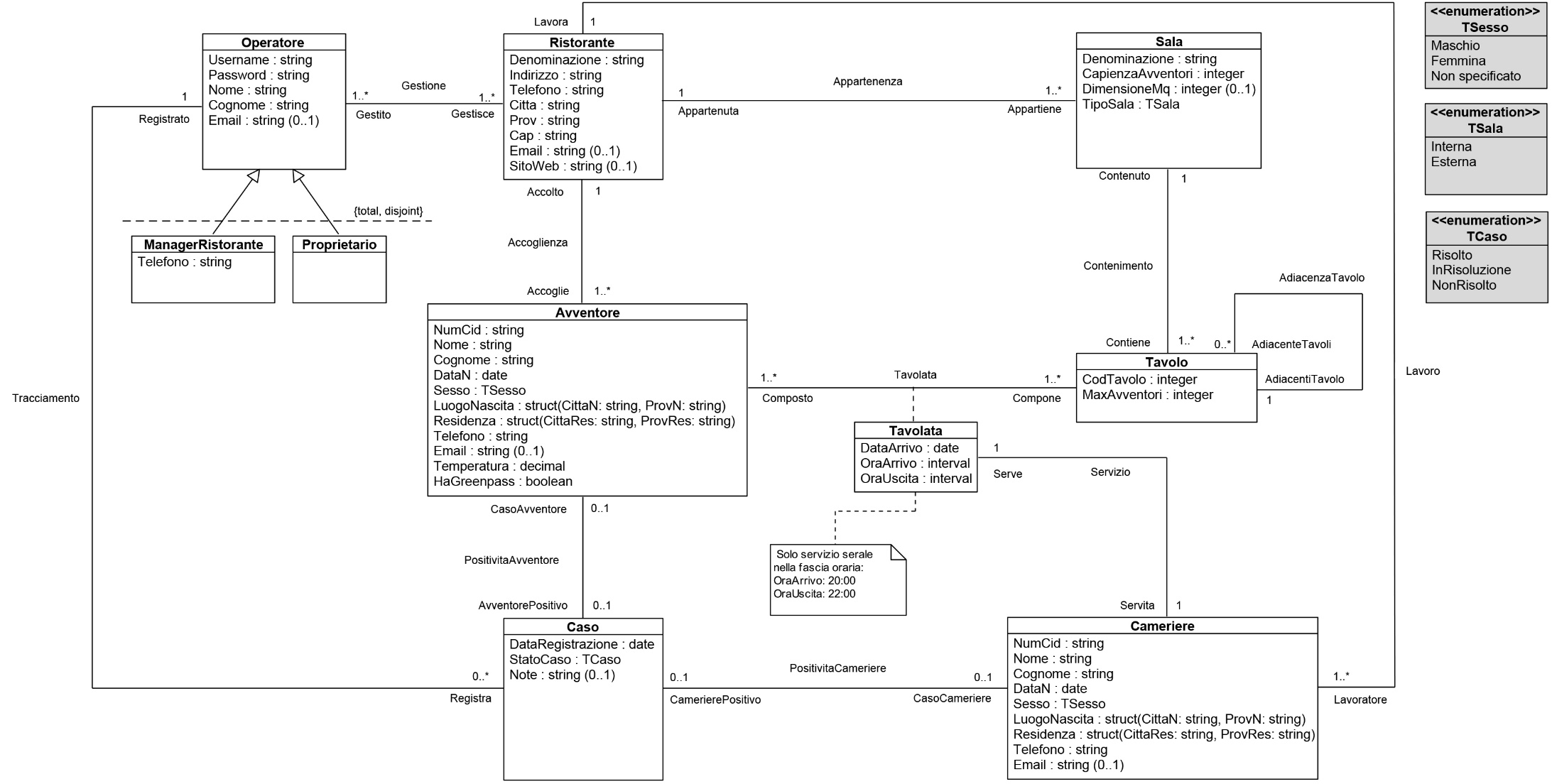
Al fine di semplificare la lettura dei class diagram UML che seguono si è scelto di adottare le seguenti convenzioni:

* tutti gli attributi, salvo quelli in cui è specificata, hanno cardinalità (1,1) o brevemente 1, ossia sono attributi a valore singolo totali;
* per le associazioni: la cardinalità di una partecipazione è situata a destra se la linea di collegamento è verticale, in alto se la linea di collegamento è orizzontale;
* le enumerazioni sono identificabili dal colore grigio.

**NOTA -** Nel caso in cui i class diagram non risultassero essere abbastanza leggibili si riportano di seguito i link alle rispettive versioni digitali esterne:

* **Class diagram:** <https://ibb.co/7WC7zWC>
* **Class diagram ristrutturato:** <https://ibb.co/VpHcWvx>

## **2.2 Class diagram**



**Immagine 2.1**: Class diagram della base di dati

## **2.3 Ristrutturazione del class diagram**

Al fine di rendere il class diagram conforme al modello relazionale che sarà adottato successivamente nella fase di progettazione logica e per migliorare l’efficienza dell’implementazione fisica si procede alla ristrutturazione dello stesso. Al termine del procedimento di ristrutturazione il class diagram non conterrà attributi strutturati e gerarchie di specializzazione.

## **2.3.1 Analisi delle informazioni ridondanti**

Sono presenti due informazioni ridondanti nel class diagram di partenza. La prima riguarda gli attributi OraArrivo e OraUscita appartenenti alla classe di associazione Tavolata. Tali attributi avranno sempre lo stesso valore per ogni istanza di Tavolata, poiché dall’analisi dei requisiti sappiamo che ogni singolo ristorante effettuerà soltanto servizio serale nella fascia oraria OraArrivo: 20:00, OraUscita: 22:00. Per completezza si è deciso di non rimuovere i suddetti attributi nella ristrutturazione del class diagram. La seconda informazione ridondante riguarda l’attributo CapienzaAvventori appartenente all’entità Sala. È infatti possibile ottenere la CapienzaAvventori di una determinata Sala sommando l’attributo MaxAvventori dell’entità Tavolo, per tutti i tavoli appartenenti alla Sala presa in considerazione. Anche in questo caso, per completezza, si è deciso di non rimuovere l’attributo nella fase di ristrutturazione poiché si è pensato che in ottica pandemica, la capienza di avventori per una determinata sala rimanga fissa per un lungo periodo di tempo nello schema della base di dati e non variabile a seconda dei tavoli che vengono associati ad una sala. In conclusione, per ogni Sala avremo una capienza fissa ed un massimo numero di tavoli che possono essere ad essa associati, la cui somma totale di MaxAvventori ovviamente dovrà essere minore o uguale alla CapienzaAvventori della Sala.

## **2.3.2 Analisi degli identificativi**

Per alcune entità del class diagram è stata decisa l’introduzione di chiavi primarie “surrogate” in modo da evitare l’impiego di chiavi candidate composte da più attributi. Tali chiavi primarie saranno degli identificativi numerici che permetteranno di discriminare con maggiore facilità le istanze.  
Le chiavi primarie surrogate introdotte sono le seguenti: CodProprietario, CodManager, CodRistorante, CodSala, CodTavolo, CodTavolata. Le entità Avventore e Cameriere saranno identificate dalla chiave primaria NumCid ossia una stringa che corrisponde al numero di carta d’identità, univoco per ogni persona.

## **2.3.3 Rimozione degli attributi strutturati**

È necessario gestire gli attributi strutturati LuogoNascita e Residenza presenti rispettivamente nelle entità Avventore e Cameriere. È stato deciso di estendere gli attributi che formano gli strutturati LuogoNascita e Residenza nell’entità cui appartengono, rimuovendo questi ultimi. Dunque, alle entità Avventore e Cameriere verranno aggiunti i seguenti attributi: CittaN, ProvN (che costituiscono l’attributo strutturato LuogoNascita) e CittaRes, ProvRes (che costituiscono l’attributo strutturato Residenza).

È stata scelta questa gestione poiché tali attributi potrebbero essere utili per alcune interrogazioni riguardanti statistiche sul tracciamento dei contatti (ad esempio per stabilire quante persone di una determinata CittaRes siano risultate positive in un determinato ristorante, su base giornaliera o mensile).

## **2.3.4 Rimozione delle classi di associazione**

Si procede con l’eliminazione della classe di associazione Tavolata e la reintroduzione della stessa come entità, contenente gli stessi attributi della classe di associazione con l’aggiunta di un identificativo numerico, CodTavolata. Sono state introdotte dunque due nuove associazioni:

* Partecipazione fra le entità Avventore e Tavolata;
* Composizione fra le entità Tavolo e Tavolata.

Ciò faciliterà la creazione e la gestione delle tavolate di avventori.

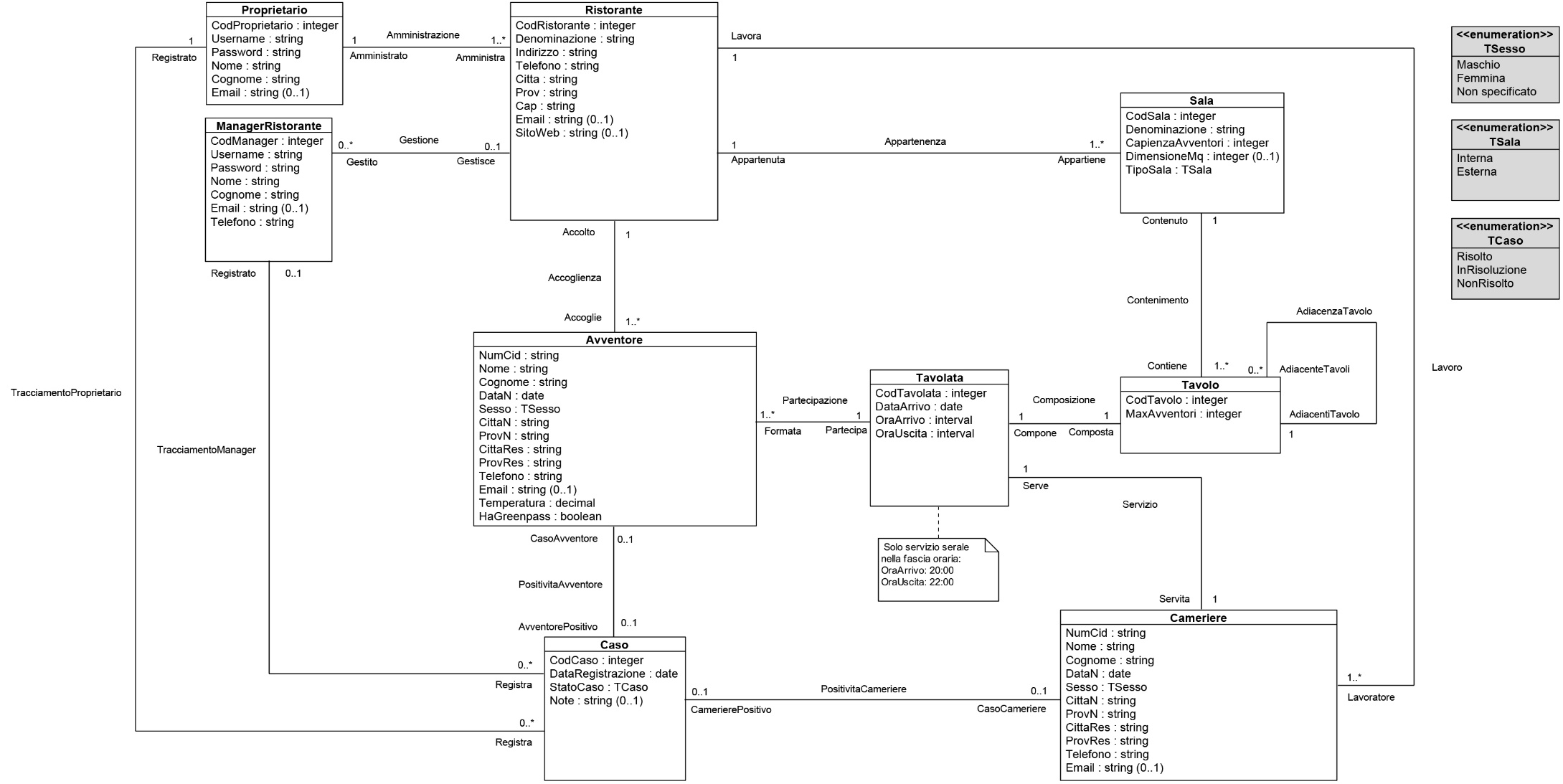
2.3.5 Rimozione delle gerarchie di specializzazione

Si procede con l’eliminazione della specializzazione riguardante l’entità Operatore che può essere specializzata in Proprietario oppure ManagerRistorante. Si tratta di una specializzazione totale e disgiunta; per tanto si procede alla sua eliminazione attraverso la “compressione” della superclasse nelle sottoclassi, le quali ereditano gli attributi e le associazioni della superclasse Operatore. Sono quindi state aggiunte le associazioni:

* Amministrazione e Gestione (in sostituzione della precedente, Gestione);
* TracciamentoProprietario e TracciamentoManager (in sostituzione della precedente, Tracciamento).

In questo modo la figura dell’operatore generico che gestirà la piattaforma per il tracciamento verrà dettagliata maggiormente, così come descritto al punto [**1.2**](#_1.2_L’organizzazione_del).

## **2.3.6 Class diagram ristrutturato**

****

**Immagine 2.2**: Class diagram ristrutturato della base di dati

2.4 Dizionario dei dati per il class diagram ristrutturato

## **2.4.1 Dizionario delle classi**

|  |  |  |
| --- | --- | --- |
| *Classe* | *Descrizione* | *Attributi* |
| Proprietario | Descrittore di un generico proprietario di uno o più ristoranti | **CodProprietario** (*integer*):chiave surrogata che serve a identificare un proprietario.  **Username** (*string*): username associato ad un proprietario.  **Password** (*string*): password associata ad un proprietario.  **Nome** (*string*): nome di un proprietario.  **Cognome** (*string*): cognome di un proprietario.  **Email** (*string, opzionale*): email associata ad un proprietario. |
| ManagerRistorante | Descrittore del manager di un ristorante | **CodManager** (*integer*): chiave surrogata che serve a identificare univocamente un manager di un ristorante.  **Username** (*string*): username associato al manager.  **Password** (*string*): password associata al manager.  **Nome** (*string*): nome di un manager.  **Cognome** (*string*): cognome associato di un manager.  **Email** (*string, opzionale*): email associata al manager.  **Telefono** (*string*): numero di telefono associato al manager. |
| Ristorante | Descrittore di un ristorante | **CodRistorante** (*integer*): chiave surrogata che serve a identificare univocamente un ristorante.  **Denominazione** (*string*): nome di un ristorante.  **Indirizzo** (*string*): indirizzo di un ristorante.  **Telefono** (*string*): numero di telefono di un ristorante.  **Citta** (*string*): città di ubicazione di un ristorante.  **Prov** (*string*): provincia di ubicazione di un ristorante.  **Cap** (*string*): cap della provincia di ubicazione di un ristorante.  **Email** (*string, opzionale*): email associata ad un ristorante.  **SitoWeb** (*string, opzionale*): sito web associato ad un ristorante. |
| Sala | Descrittore della sala di un ristorante | **CodSala** (*integer*): chiave surrogata che serve a identificare univocamente una sala.  **Denominazione** (*string*): nome di una sala.  **CapienzaAvventori** (*integer*): capienza massima di avventori per una sala.  **DimensioneMq** (*string, opzionale*): dimensione in metri quadri di una sala.  **TipoSala** (*TSala*): attributo che indica se la sala è interna oppure esterna. |
| Tavolo | Descrittore di un tavolo del ristorante | **CodTavolo** (*integer*): chiave surrogata che serve a identificare univocamente un tavolo.  **MaxAvventori** (*integer*): numero massimo di avventori per ogni tavolo. |
| Tavolata | Descrittore della tavolata di avventori | **CodTavolata** (*integer*): chiave surrogata che serve a identificare univocamente ciascuna tavolata.  **DataArrivo** (*date*): data in cui si svolge la tavolata.  **OraArrivo** (*interval*): orario di arrivo della tavolata (vale sempre 20:00).  **OraUscita** (*interval*): orario di uscita della tavolata (vale sempre 22:00). |
| Avventore | Descrittore di un generico avventore | **NumCid** (*string*): numero della carta di identità che serve anche da chiave primaria per identificare univocamente ciascun avventore.  **Nome** (*string*): nome di un avventore.  **Cognome** (*string*): cognome di un avventore.  **DataN** (*date*): data di nascita di un avventore.  **Sesso** (*TSesso*): sesso di un avventore.  **CittaN** (*string*): città di nascita di un avventore.  **ProvN** (*string*): provincia di nascita di un avventore.  **CittaRes** (*string*): città di residenza di un avventore.  **ProvRes** (*string*): provincia di residenza di un avventore.  **Telefono** (*string*): numero di telefono di un avventore.  **Email** (*string, opzionale*): email associata ad un avventore.  **Temperatura** (*decimal*): temperatura misurata all’ingresso di un ristorante.  **HaGreenpass** (*boolean*): indica il possesso o meno del green pass da parte di un avventore. |
| Cameriere | Descrittore di un cameriere del ristorante | **NumCid** (*string*): numero della carta di identità che serve anche da chiave primaria per identificare univocamente ciascun cameriere.  **Nome** (*string*): nome di un cameriere.  **Cognome** (*string*): cognome di un cameriere.  **DataN** (*date*): data di nascita di un cameriere.  **Sesso** (*TSesso*): sesso di un cameriere.  **CittaN** (*string*): città di nascita di un cameriere.  **ProvN** (*string*): provincia di nascita di un cameriere.  **CittaRes** (*string*): città di residenza di un cameriere.  **ProvRes** (*string*): provincia di residenza di un cameriere.  **Telefono** (*string*): numero di telefono di un cameriere.  **Email** (*string, opzionale*): email associata ad un cameriere. |
| Caso | Descrittore di un caso covid | **CodCaso** (*integer*): chiave surrogata che serve per identificare univocamente ciascun caso.  **DataRegistrazione** (*date*): data di registrazione di un caso.  **StatoCaso** (*TStato*): stato di un caso (Risolto, InRisoluzione, NonRisolto).  **Note** (*string, opzionale*): note relative al caso. |

**Tabella 2.1**: Dizionario delle classi - fine.

## **2.4.2 Dizionario delle associazioni**

|  |  |  |
| --- | --- | --- |
| *Associazione* | *Descrizione* | *Classi coinvolte* |
| Amministrazione | Esprime l’amministrazione da parte di un proprietario di uno o più ristoranti | **Proprietario [1]** *ruolo* (**Amministrato**): indica il proprietario che amministra uno o più ristoranti.  **Ristorante [1..\*]** *ruolo* (**Amministra**): indica il ristorante che viene amministrato da un proprietario. |
| Gestione | Esprime la possibile gestione da parte di uno o più manager di un ristorante. | **ManagerRistorante [0..\*]** *ruolo* (**Gestito**):  indica il manager che può gestire un ristorante.  **Ristorante [0..1]** *ruolo* (**Gestisce**): indica il ristorante che può essere gestito da uno o più manager. |
| Appartenenza | Esprime l’appartenenza di una o più sale ad un ristorante. | **Ristorante [1]** *ruolo* (**Appartenuta**): indica il ristorante a cui appartengono una o più sale.  **Sala [1..\*]** *ruolo* (**Appartiene**): indica una sala che appartiene ad un ristorante. |
| Contenimento | Esprime il contenimento di tavoli da parte di una sala. | **Sala [1]** *ruolo* (**Contenuto**): indica una sala contenente uno o più tavoli.  **Tavolo [1..\*]** *ruolo* (**Contiene**): indica un tavolo contenuto in una sala. |
| Servizio | Esprime il servizio di un cameriere ad una tavolata. | **Cameriere [1]** *ruolo* (**Servita**): indica il cameriere che serve una tavolata  **Tavolata [1]** *ruolo* (**Serve**): indica la tavolata servita da un cameriere. |
| Partecipazione | Esprime uno o più avventori che partecipano ad una tavolata. | **Tavolata [1..\*]** *ruolo* (**Partecipa**): indica la tavolata a cui partecipa uno o più avventori.  **Avventore [1]** *ruolo* (**Formata**): indica uno o più avventori che partecipano ad una tavolata. |
| Composizione | Esprime la composizione di una tavolata con un tavolo. | **Tavolata [1]** *ruolo* (**Compone**): indica la tavolata composta da un tavolo.  **Tavolo [1]** *ruolo* (**Composta**): indica il tavolo che compone una tavolata. |
| AdiacenzaTavolo | Esprime la possibile adiacenza di altri tavoli ad un tavolo. | **Tavolo [1]** *ruolo* (**AdiacentiTavolo**): indica il tavolo a cui possono essere adiacenti altri tavoli.  **Tavolo [0..\*]** *ruolo* (**AdiacenteTavoli**): indica la possibilità di adiacenza di altri tavoli ad un tavolo. |
| Lavoro | Esprime il rapporto lavorativo tra uno o più camerieri ed un ristorante. | **Cameriere [1..\*]** *ruolo* (**Lavoratore**): indica il cameriere che lavora per un ristorante.  **Ristorante [1]** *ruolo* (**Lavora**): indica il ristorante per cui lavorano uno o più camerieri. |
| Accoglienza | Esprime l’accoglienza da parte di un ristorante di uno o più avventori. | **Ristorante [1]** *ruolo* (**Accolto**): indica il ristorante che accoglie uno o più avventori  **Avventore [1..\*]** *ruolo* (**Accoglie**): indica uno o più avventori accolti da un ristorante. |
| PositivitaAvventore | Esprime la possibilità di esistenza di un caso di positività per un avventore. | **Avventore [0..1]** *ruolo* (**CasoAvventore**): indica l’avventore che può risultare positivo.  **Caso [0..1]** *ruolo* (**AvventorePositivo**): indica il possibile caso di un avventore positivo. |
| PositivitaCameriere | Esprime la possibilità di esistenza di un caso di positività per un cameriere. | **Caso [0..1]** *ruolo* (**CamerierePositivo**): indica il cameriere che può risultare positivo.  **Cameriere [0..1]** *ruolo* (**CasoCameriere**): indica il possibile caso di un cameriere positivo. |
| TracciamentoProprietario | Esprime il tracciamento da parte di un proprietario di uno o più possibili casi. | **Proprietario [1]** *ruolo* (**Registrato**): indica il proprietario che registra uno o più possibili casi.  **Caso [0..\*]** *ruolo* (**Registra**): indica uno o più possibili casi registrati da un proprietario. |
| TracciamentoManager | Esprime il possibile tracciamento da parte di un manager di uno o più possibili casi. | **Manager [0..1]** *ruolo* (**Registrato**): indica il manager che può registrare uno o più possibili casi.  **Caso [0..\*]** *ruolo* (**Registra**): indica uno o più possibili casi registrati da un manager. |

**Tabella 2.2**: Dizionario delle associazioni - fine.

## **2.4.3 Dizionario dei vincoli**

|  |  |  |
| --- | --- | --- |
| *Vincolo* | *Tipo* | *Descrizione* |
| Has greenpass | Interrelazionale | Se l’attributo *HaGreenPass* dell’entità *Avventore* vale ‘F’ (*falso*), allora un avventore può partecipare solo ad una tavolata composta da un tavolo la cui ubicazione è in una sala di tipo ‘Esterna’. |
| Email legale | Dominio | Vale per tutte le entità che possiedono l’attributo *Email*: *Proprietario, ManagerRistorante, Ristorante, Avventore, Cameriere*. L’*Email* deve rispettare la forma standard ovvero: contenere almeno un carattere prima della @, almeno un carattere tra essa e il punto e almeno due caratteri nella parte finale dopo il punto. |
| Password legale | Dominio | Vale per tutte le entità che possiedono l’attributo *Password*: *Proprietario, ManagerRistorante*. La *Password* deve contenere minimo 8 caratteri e rispettare il seguente formato: deve esserci almeno una lettera e un numero. |
| Sito web legale | Dominio | Il *SitoWeb* deve rispettare la forma standard ovvero: contenere tre caratteri iniziali corrispondenti a “www” seguiti da un punto, successivamente almeno due caratteri ed infine un punto seguito da almeno due caratteri. |
| Numero di telefono legale | Dominio | Vale per tutte le entità che possiedono l’attributo *Telefono*: *ManagerRistorante, Ristorante, Avventore, Cameriere*. Il numero di *Telefono* può contenere solo numeri ad eccezione del carattere iniziale ‘*+*’ per il prefisso. |
| Data nascita legale | Interrelazionale | La *DataN* di un avventore deve essere precedente o uguale alla *DataArrivo* della tavolata a cui esso partecipa. |
| Orario tavolata legale | Dominio | Gli attributi *OraArrivo* e *OraUscita* di tavolata sono di default sempre uguali a *OraArrivo: 20:00*, *OraUscita: 22:00.* |
| CAP legale | Dominio | Il *Cap* deve rispettare il formato standard italiano, ossia deve essere formato esattamente da 5 caratteri numerici. |
| CapienzaAvventori legale | Dominio | La *CapienzaAvventori* di una sala deve essere maggiore di 0. |
| DimensioneMq legale | Dominio | La *DimensioneMq* di una sala deve essere maggiore di 0. |
| MaxAvventori legale | Interrelazionale | Il valore di *MaxAvventori* di un tavolo deve essere maggiore di 0 e minore uguale del valore di *CapienzaAvventori* della sala in cui il tavolo è contenuto. |
| Capienza legale | Interrelazionale | La somma totale di *MaxAvventori* per tutti i tavoli di una sala deve essere minore o uguale alla *CapienzaAvventori* della sala che contiene i tavoli in questione. |
| Somma avventori a tavolata legale | Interrelazionale | La somma totale di tutti gli avventori partecipanti ad una stessa tavolata deve essere minore o uguale al *MaxAvventori* del tavolo che compone la tavolata. |
| Data registrazione caso legale | Interrelazionale | La *DataRegistrazione* di un caso deve essere successiva alla *DataN* dell’avventore o cameriere a cui il caso è associato. |
| Temperatura avventore legale | Dominio | La *Temperatura* registrata di un avventore deve essere maggiore o uguale a 35 e minore uguale a 37,5. |
| Età cameriere legale | Dominio | L’età di un cameriere deve essere maggiore o uguale a 18 anni. |
| Unica composizione tavolo a tavolata | Interrelazionale | Per ogni *DataArrivo*, un tavolo può essere associato al più ad una tavolata. |
| Unico username proprietario | Intrarelazionale | L’*Username* di un proprietario deve essere unico. |
| Unico username  manager | Intrarelazionale | L’*Username* di un manager deve essere unico. |
| TSesso | Dominio | L’attributo *Sesso* può assumere solo i valori:  ‘*Maschio*’, ‘*Femmina*’, ‘*Non specificato*’. |
| TSala | Dominio | L’attributo *TipoSala* può assumere solo i valori:  ‘*Interna*’, ‘*Esterna*’. |
| TCaso | Dominio | L’attributo *StatoCaso* può assumere solo i valori:  ‘*Risolto*’, ‘*InRisoluzione*’, ‘*NonRisolto*’. |

**Tabella 2.3**: Dizionario dei vincoli - fine.

## **2.4.4 Dizionario delle interrogazioni**

|  |  |  |  |
| --- | --- | --- | --- |
| *Interrogazione* | *Descrizione* | *Frequenza* | *Classi coinvolte* |
| Numero giornaliero di avventori per ristorante | La query, dato il codice di un ristorante e specificata una data, calcola il numero totale di avventori che sono stati accolti dal ristorante in quella data. | Può essere eseguita giornalmente | Ristorante,  Avventore, Tavolata |
| Numero mensile  di avventori per ristorante | La query, dato il codice di un ristorante e specificato il mese di un determinato anno, calcola il numero totale di avventori che sono stati accolti dal ristorante in quel mese dell’anno specificato. | Può essere eseguita mensilmente ma anche giornalmente | Ristorante,  Avventore, Tavolata |
| Numero giornaliero di avventori per tutti i ristoranti di un proprietario | La query, dato il codice di un proprietario e specificata una data, calcola il numero totale di avventori che sono stati accolti da tutti i ristoranti appartenenti al proprietario, in quella data. | Può essere eseguita giornalmente | Ristorante, Avventore,  Tavolata |
| Numero mensile  di avventori per tutti i ristoranti di un proprietario | La query, dato il codice di un proprietario e specificato il mese di un determinato anno, calcola il numero totale di avventori che sono stati accolti da tutti i ristoranti appartenenti al proprietario, in quel mese dell’anno. | Può essere eseguita mensilmente ma anche giornalmente | Ristorante, Avventore,  Tavolata |
| Casi positivi di un determinato ristorante per data di arrivo della tavolata | La query, dato il codice di un ristorante e specificata una data di arrivo, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti dal ristorante in quella data. | Può essere eseguita giornalmente | Ristorante,  Tavolata, Avventore, Caso |
| Casi positivi di un determinato ristorante per mese di arrivo della tavolata | La query, dato il codice di un ristorante e specificato il mese di un anno in cui si è svolta la tavolata, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti dal ristorante in quel mese dell’anno. | Può essere eseguita mensilmente ma anche giornalmente | Ristorante,  Tavolata, Avventore, Caso |
| Casi positivi di un determinato ristorante per anno di arrivo della tavolata | La query, dato il codice di un ristorante e specificato un anno in cui si è svolta la tavolata, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti dal ristorante in quell’anno. | Può essere eseguita annualmente ma anche mensilmente | Ristorante,  Tavolata, Avventore, Caso |
| Casi positivi di tutti i ristoranti di un proprietario per data di arrivo della tavolata | La query, dato il codice di un proprietario e specificata una data di arrivo, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti da tutti i ristoranti appartenenti al proprietario in quella data. | Può essere eseguita giornalmente | Ristorante, Avventore,  Tavolata,  Caso |
| Casi positivi di tutti i ristoranti di un proprietario per mese di arrivo della tavolata | La query, dato il codice di un proprietario e specificato il mese di un anno in cui si è svolta la tavolata, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti nei ristoranti appartenenti al proprietario in quel mese dell’anno. | Può essere eseguita mensilmente ma anche giornalmente | Ristorante, Avventore,  Tavolata,  Caso |
| Casi positivi di tutti i ristoranti di un proprietario per anno di arrivo della tavolata | La query, dato il codice di un proprietario e specificato un anno in cui si è svolta la tavolata, calcola il numero totale di avventori risultati successivamente positivi che sono stati accolti nei ristoranti appartenenti al proprietario in quell’anno. | Può essere eseguita annualmente ma anche mensilmente | Ristorante, Avventore,  Tavolata,  Caso |
| Informazioni sugli avventori risultati positivi in un ristorante | La query, dato un ristorante, permette di visualizzare l’anagrafica e alcune informazioni sul relativo caso di tutti gli avventori risultati positivi nel ristorante. | Può essere eseguita giornalmente | Ristorante, Avventore,  Caso,  Tavolata,  Tavolo |
| Informazioni sugli avventori risultati positivi in tutti i ristoranti di un proprietario | La query, dato un determinato proprietario, permette di visualizzare l’anagrafica e alcune informazioni sul relativo caso di tutti gli avventori risultati positivi nei ristoranti da lui amministrati. | Può essere eseguita giornalmente | Ristorante, Avventore,  Caso,  Tavolata,  Tavolo |
| Informazioni sui camerieri risultati positivi in un ristorante | La query, dato un ristorante, permette di visualizzare l’anagrafica e alcune informazioni sul relativo caso di tutti i camerieri risultati positivi nel ristorante. | Può essere eseguita giornalmente | Ristorante, Cameriere,  Caso |
| Informazioni sui camerieri risultati positivi in tutti i ristoranti di un proprietario | La query, dato un determinato proprietario, permette di visualizzare l’anagrafica e alcune informazioni sul relativo caso di tutti i camerieri risultati positivi nei ristoranti da lui amministrati. | Può essere eseguita giornalmente | Ristorante, Cameriere,  Caso |
| Avventori positivi con o senza  green pass | La query mostra il numero complessivo totale di casi di avventori risultati positivi che possiedono green pass. Lo stesso per gli avventori risultati positivi sprovvisti di green pass. | Può essere eseguita giornalmente | Avventore,  Caso |
| Numero di avventori medio per tavolata di un ristorante | La query, dato un determinato ristorante, mostra il numero di avventori medio per tavolata. | Può essere eseguita giornalmente ma anche mensilmente | Avventore,  Tavolata |

**Tabella 2.4**: Dizionario delle interrogazioni - fine.

# **Capitolo 3**

**Progettazione logica**

## **Introduzione**

In questo capitolo viene affrontata la fase successiva alla progettazione concettuale della base di dati, scendendo ad un livello di astrazione più basso e vicino all’implementazione vera e propria della base di dati. Si tradurrà lo schema concettuale ristrutturato in uno **schema logico**, dipendente dal **modello dei dati** scelto ossia quello **relazionale**. Negli schemi relazionali che seguiranno le **chiavi primarie** sono indicate con una singola *sottolineatura* mentre le **chiavi esterne** con una *doppia sottolineatura*.

3.1 Schema logico

## **3.1.1 Traduzione in schemi relazionali**

**Proprietario**(CodProprietario, Username, Password, Nome, Cognome, Email)

**Chiavi esterne**: nessuna.

**Ristorante**(CodRistorante, Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Email, SitoWeb, Proprietario)

**Chiavi esterne**: Proprietario → Proprietario.CodProprietario

**ManagerRistorante**(CodManager, Username, Password, Nome, Cognome, Email, Telefono, RistoranteGestito)

**Chiavi esterne**: RistoranteGestito → Ristorante.CodRistorante

**Sala**(CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)

**Chiavi esterne**: Ristorante → Ristorante.CodRistorante

**Cameriere**(NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)

**Chiavi esterne**: Ristorante → Ristorante.CodRistorante

**Tavolo**(CodTavolo, MaxAvventori, Sala, TavoloAdiacente)

**Chiavi esterne**: Sala → Sala.CodSala  
**………………:** TavoloAdiacente → Tavolo.CodTavolo

**Tavolata**(CodTavolata, DataArrivo, OraArrivo, OraUscita, Tavolo, Cameriere)

**Chiavi esterne**: Tavolo → Tavolo.CodTavolo  
**………………:** Cameriere → Cameriere.NumCid

**Avventore**(NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass, Ristorante, Tavolata)

**Chiavi esterne**: Ristorante → Ristorante.CodRistorante  
**………………:** Tavolata → Tavolata.CodTavolata

**Caso**(CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)

**Chiavi esterne**: AvventorePositivo → Avventore.NumCid  
**………………:** CamerierePositivo → Cameriere.NumCid  
**………………:** RegistraProprietario → Proprietario.CodProprietario  
**………………:** RegistraManager → ManagerRistorante.CodManager

## **3.1.2 Traduzione delle associazioni**

Nella seguente tabella vengono riportate le traduzioni delle associazioni individuate nello schema logico.

|  |  |
| --- | --- |
| *Associazione* | *Implementazione* |
| Amministrazione | Chiave esterna in **Ristorante** → **Proprietario** |
| Gestione | Chiave esterna in **ManagerRistorante** → **Ristorante** |
| Appartenenza | Chiave esterna in **Sala** → **Ristorante** |
| Contenimento | Chiave esterna in **Tavolo** → **Sala** |
| Servizio | Chiave esterna in **Tavolata** → **Cameriere** |
| Partecipazione | Chiave esterna in **Avventore** → **Tavolata** |
| Composizione | Chiave esterna in **Tavolata** → **Tavolo** |
| AdiacenzaTavolo | Chiave esterna in **Tavolo** → **Tavolo** |
| Lavoro | Chiave esterna in **Cameriere** → **Ristorante** |
| Accoglienza | Chiave esterna in **Avventore** → **Ristorante** |
| PositivitaAvventore | Chiave esterna in **Caso** → **Avventore** |
| PositivitaCameriere | Chiave esterna in **Caso** → **Cameriere** |
| TracciamentoProprietario | Chiave esterna in **Caso** → **Proprietario** |
| TracciamentoManager | Chiave esterna in **Caso** → **ManagerRistorante** |

**Tabella 3.1**: Traduzione delle associazioni

## **3.1.3 Schema logico generale**

Alla fine del procedimento di traduzione, in conclusione, perveniamo al seguente schema logico:

|  |  |
| --- | --- |
| Proprietario | (CodProprietario, Username, Password, Nome, Cognome, Email) |
| Ristorante | (CodRistorante, Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Email, SitoWeb, Proprietario) |
| ManagerRistorante | (CodManager, Username, Password, Nome, Cognome, Email, Telefono, RistoranteGestito) |
| Sala | (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante) |
| Cameriere | (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante) |
| Tavolo | (CodTavolo, MaxAvventori, Sala, TavoloAdiacente) |
| Tavolata | (CodTavolata, DataArrivo, OraArrivo, OraUscita, Tavolo, Cameriere) |
| Avventore | (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass, Ristorante, Tavolata) |
| Caso | (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager) |

**Tabella 3.2**: Schema logico

# **Capitolo 4**

**Progettazione fisica**

## **Introduzione**

In questo capitolo viene trattata l’ultima fase relativa alla progettazione della base di dati, ossia la progettazione fisica. Dopo aver scelto il **DBMS** su cui implementare la base di dati, si passerà alla traduzione dello schema logico in uno **schema fisico** dei dati, attraverso la definizione delle tabelle, l’implementazione dei relativi vincoli di integrità, dei trigger e delle procedure individuate in **SQL**.

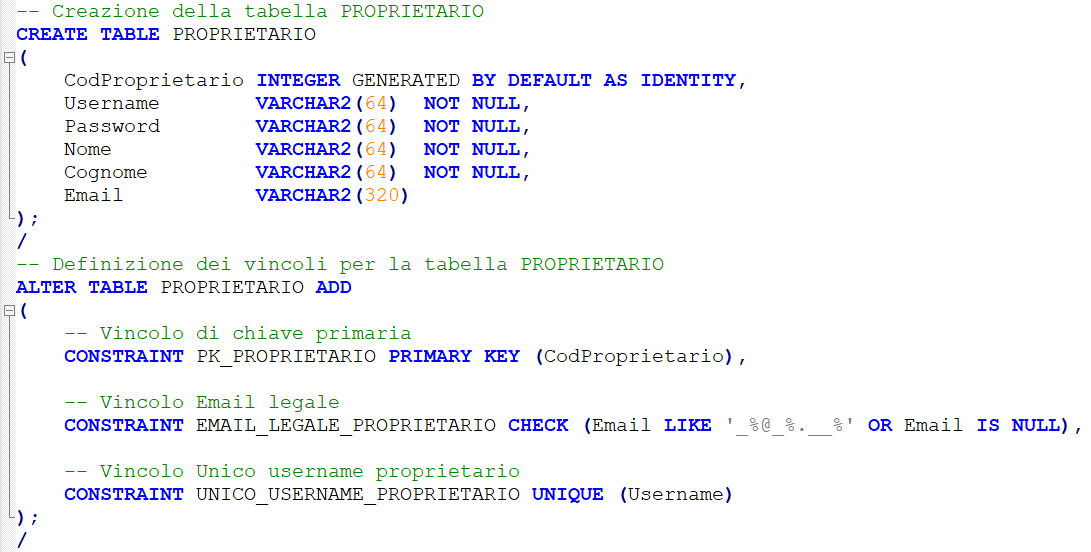
4.1 Note sull’implementazione

Il DBMS scelto per l’implementazione della base di dati è **Oracle Database** nella versione **19C**. Siccome nel DBMS Oracle non è implementato il tipo di dato **boolean**, esso è stato simulato con un carattere (CHAR) che può assumere solo i valori in {‘V’, ‘F’} che rappresentano rispettivamente VERO e FALSO.

4.2 Definizione delle tabelle

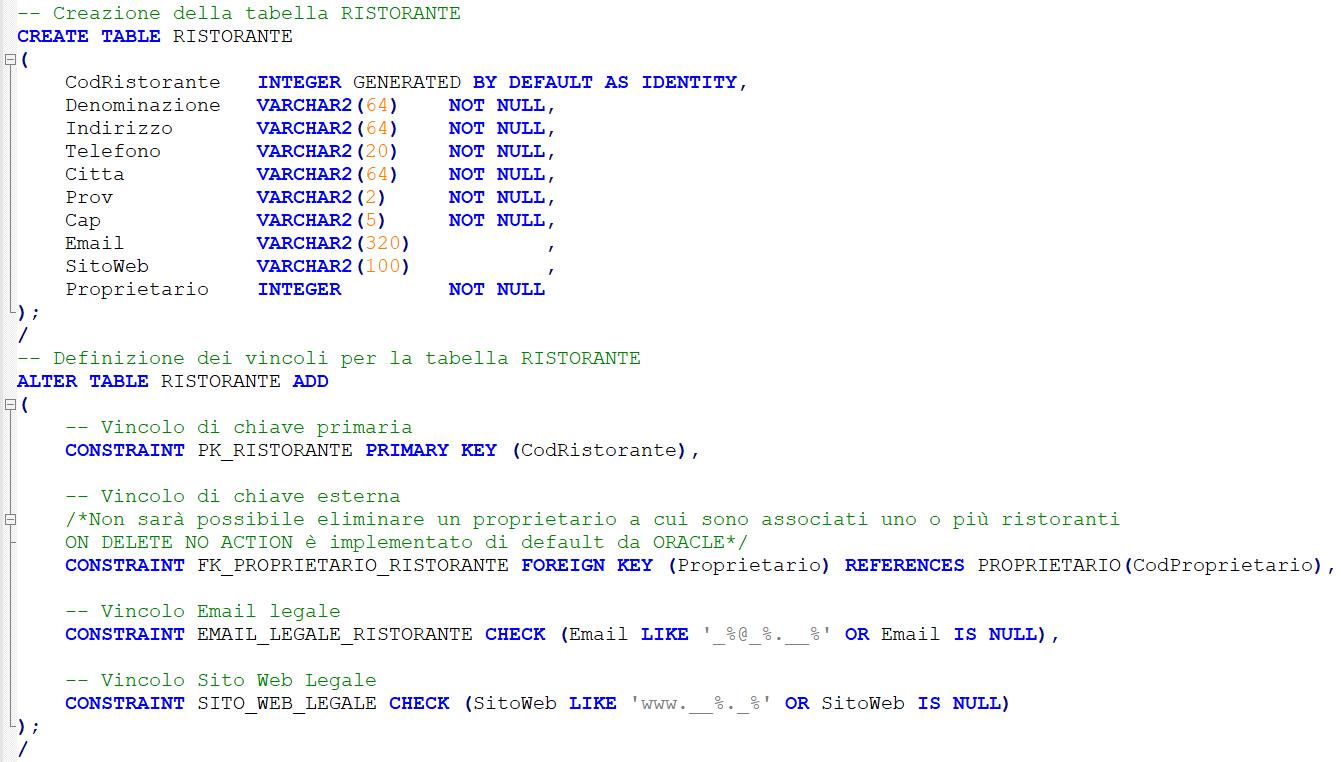
Seguono le definizioni delle tabelle estratte dallo script di creazione del database tratte dal file “DBRistorante.sql”.

## **4.2.1 Definizione della tabella PROPRIETARIO**

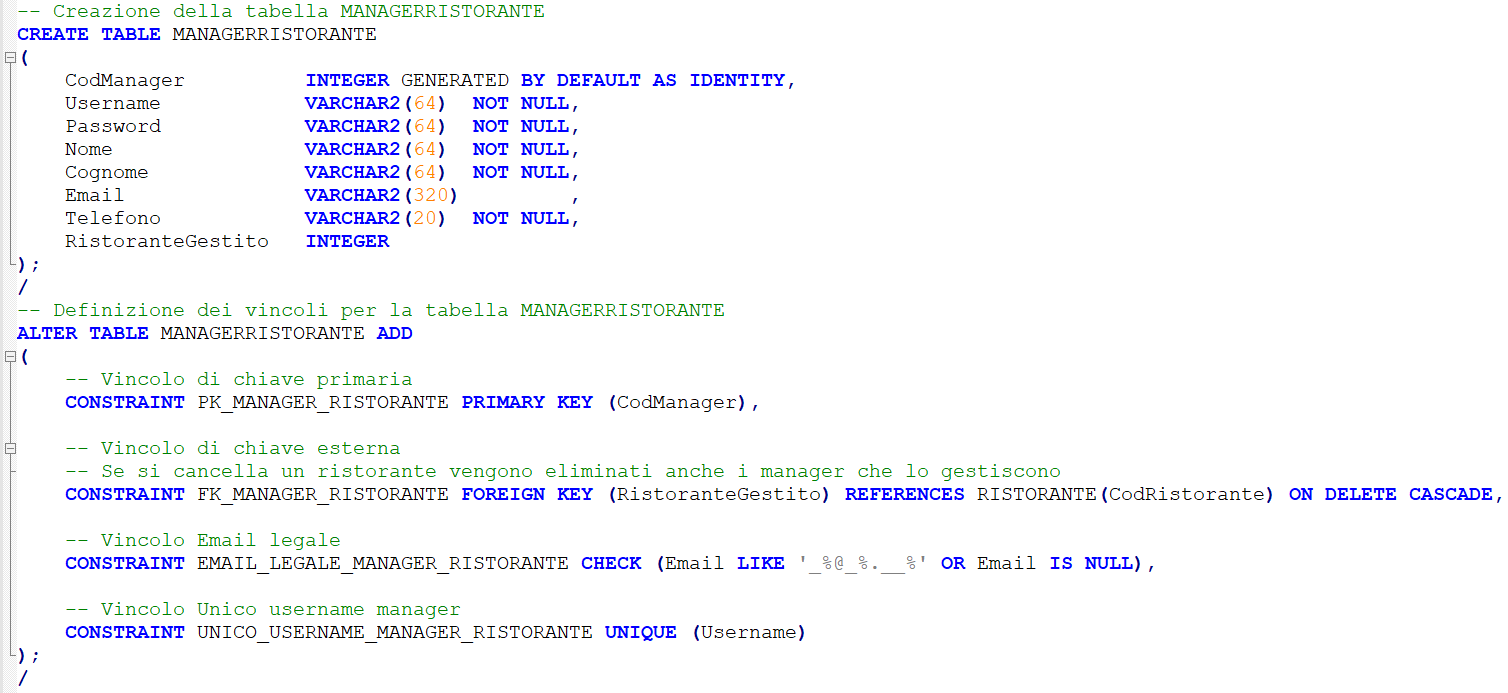


Estratto da **DBRistorante.sql** - Definizione della tabella **PROPRIETARIO**

## **4.2.2 Definizione della tabella RISTORANTE**

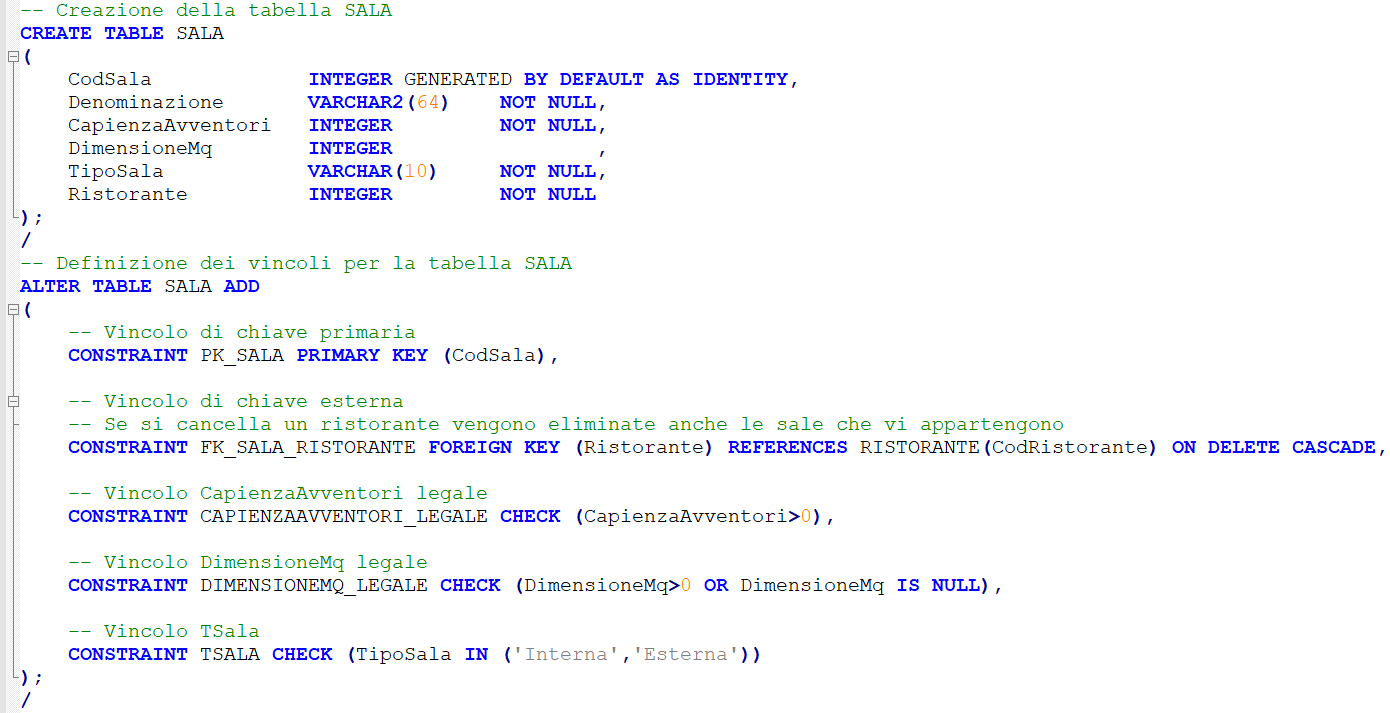
****Estratto da **DBRistorante.sql** - Definizione della tabella **RISTORANTE**

## **4.2.3 Definizione della tabella MANAGERRISTORANTE**



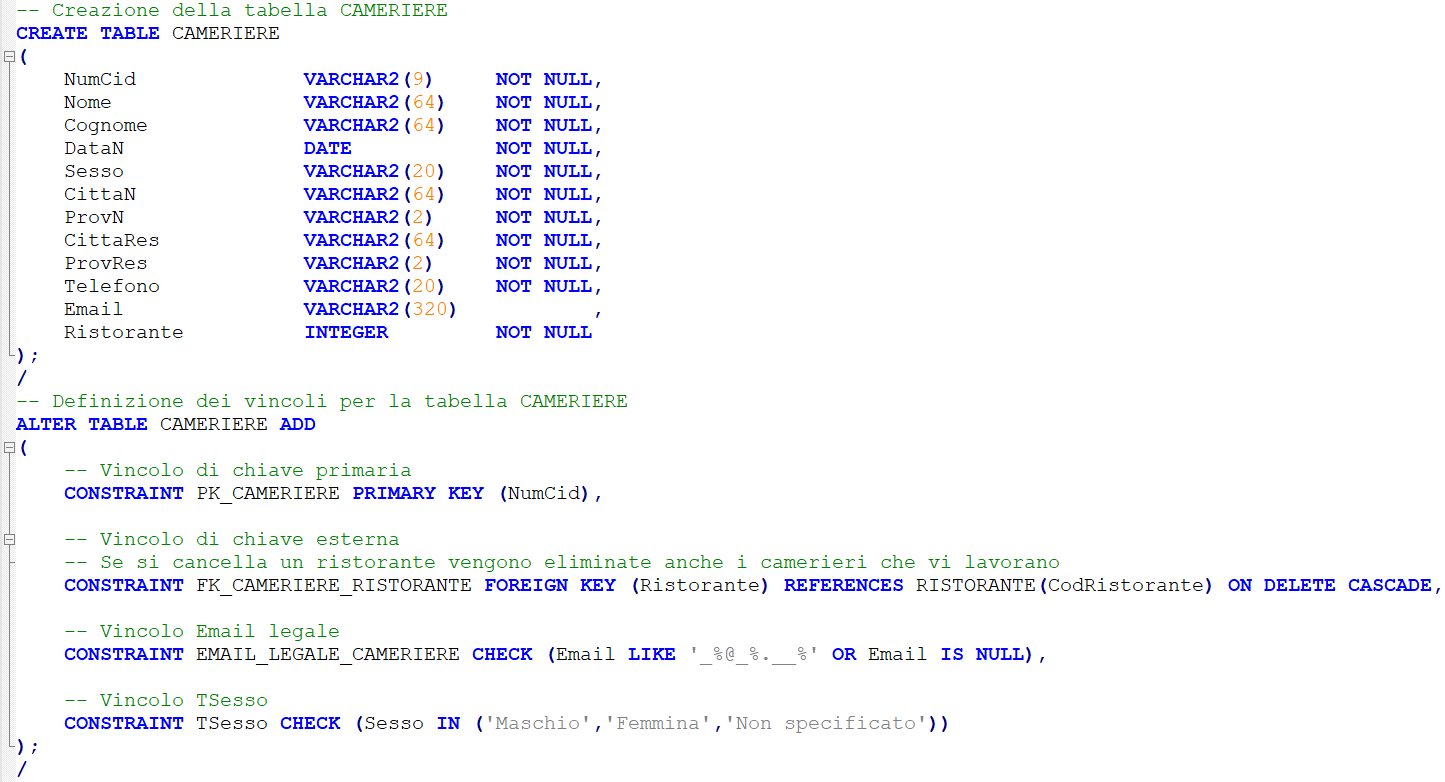
Estratto da **DBRistorante.sql** - Definizione della tabella **MANAGERRISTORANTE**

## **4.2.4 Definizione della tabella SALA**



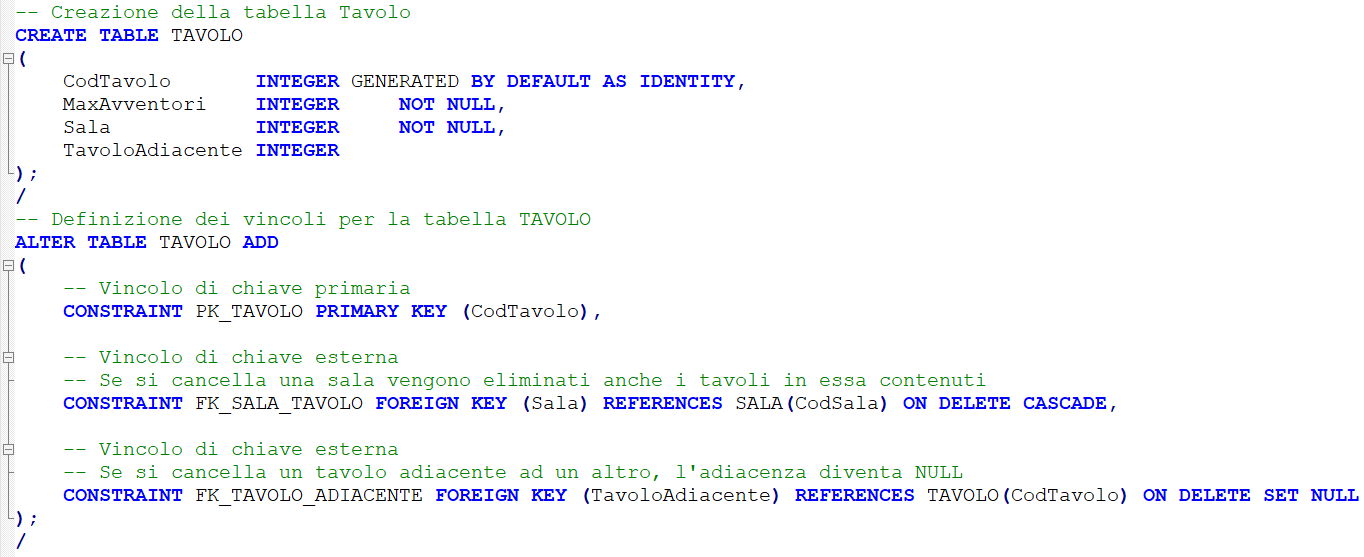
Estratto da **DBRistorante.sql** - Definizione della tabella **SALA**

4.2.5 Definizione della tabella CAMERIERE



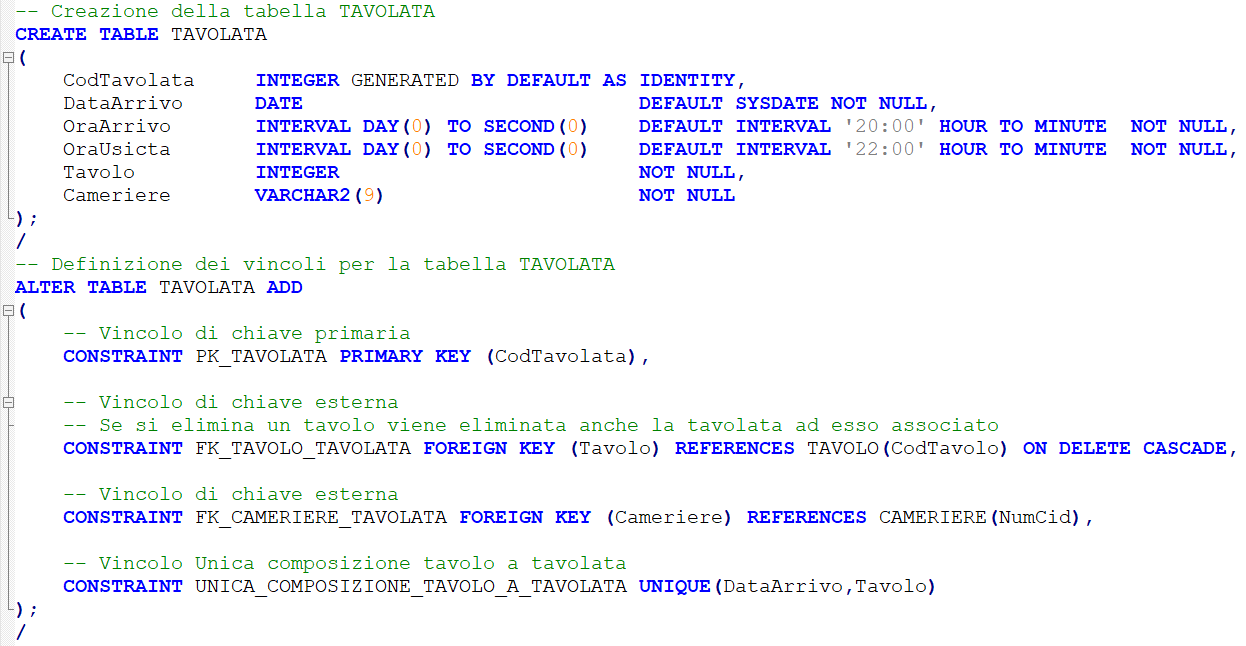
Estratto da **DBRistorante.sql** - Definizione della tabella **CAMERIERE**

## **4.2.6 Definizione della tabella TAVOLO**



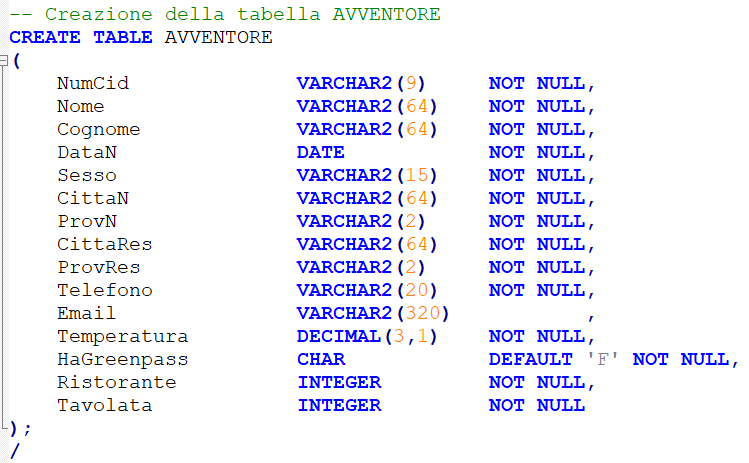
Estratto da **DBRistorante.sql** - Definizione della tabella **TAVOLO**

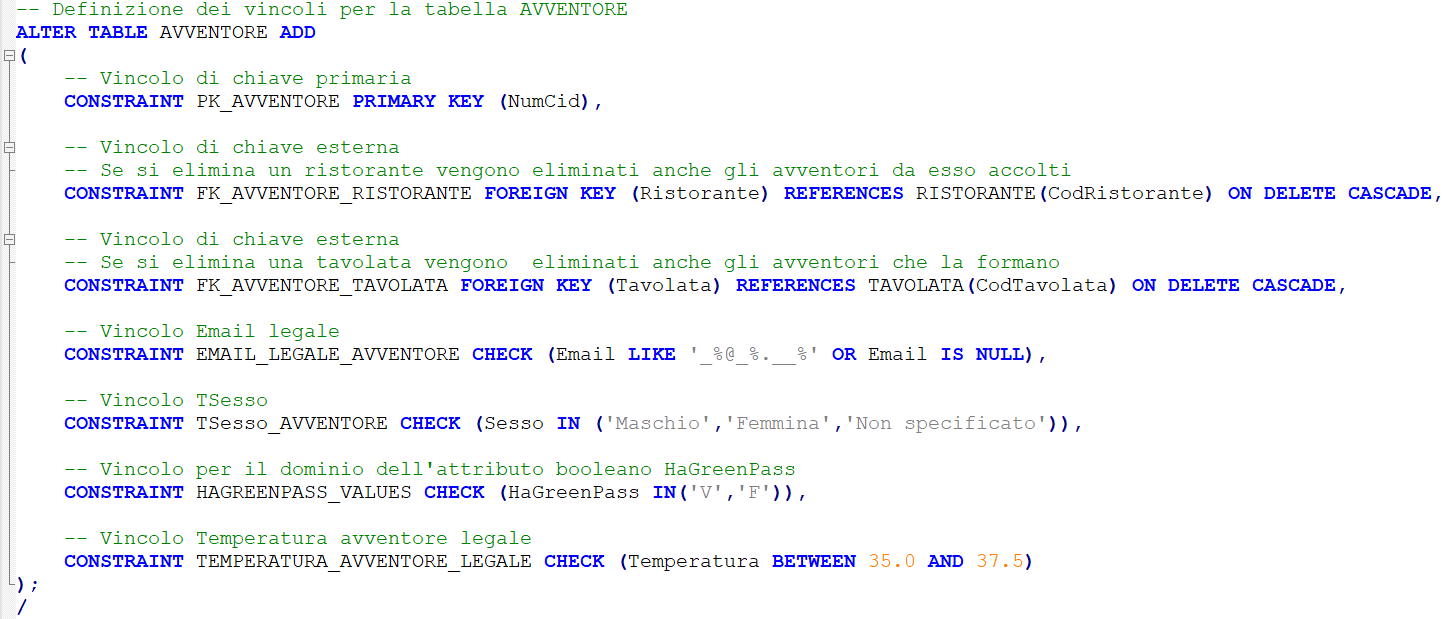
## **4.2.7 Definizione della tabella TAVOLATA**



Estratto da **DBRistorante.sql** - Definizione della tabella **TAVOLATA**

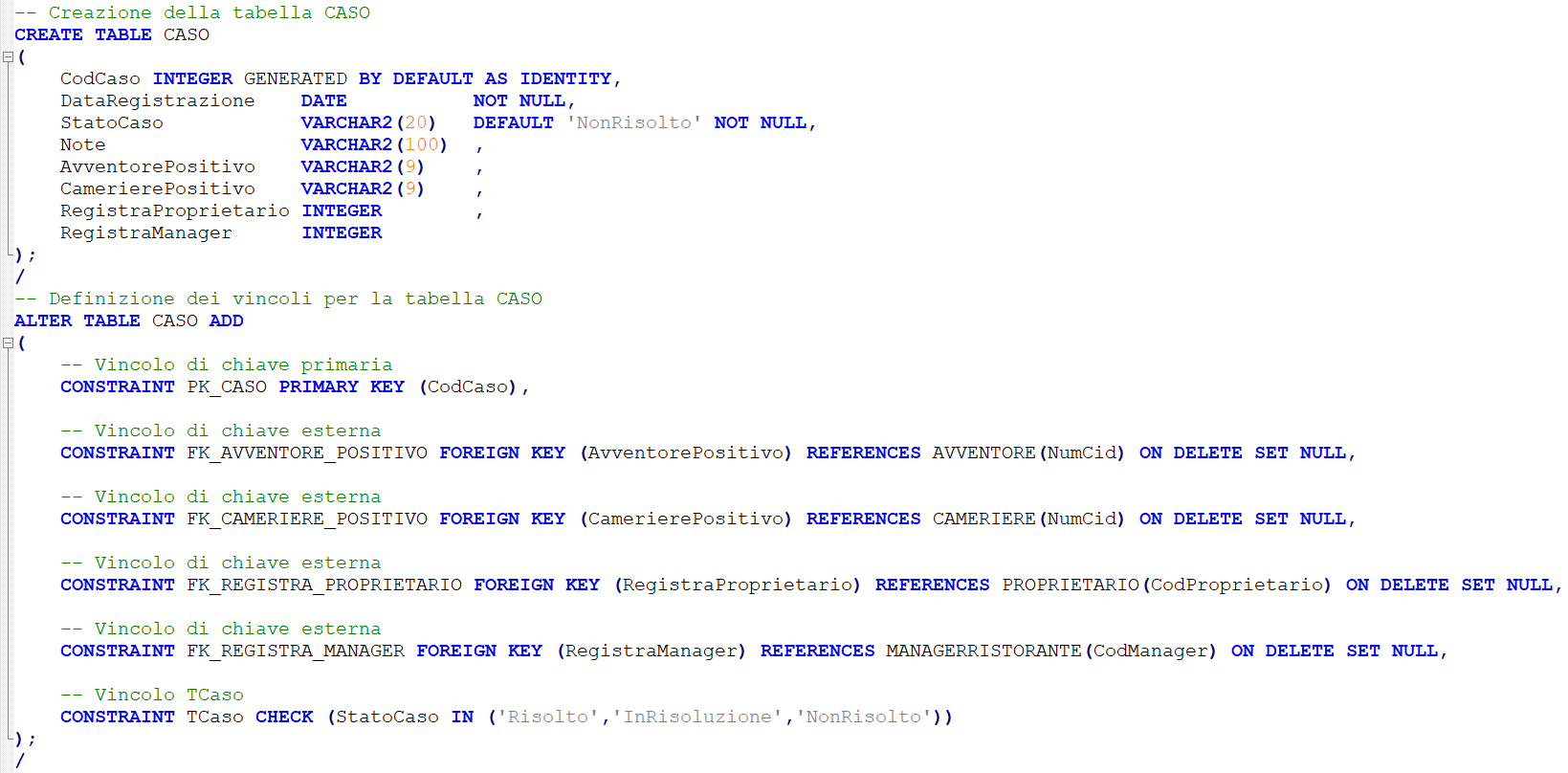
## **4.2.8 Definizione della tabella AVVENTORE**





Estratto da **DBRistorante.sql** - Definizione della tabella **AVVENTORE**

## **4.2.9 Definizione della tabella CASO**



Estratto da **DBRistorante.sql** - Definizione della tabella **CASO**

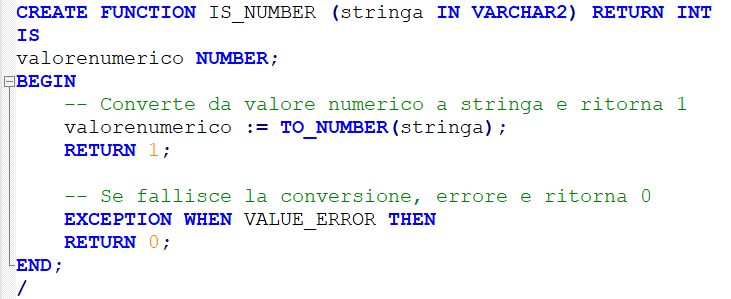
- fine definizione delle tabelle -

4.3 Funzioni, procedure ed altre automazioni

Seguono le definizioni delle procedure e funzioni utilizzate nel database tratte dal file “DBProcedureFunzioni.sql”.

4.3.1 Stored function IS\_NUMBER

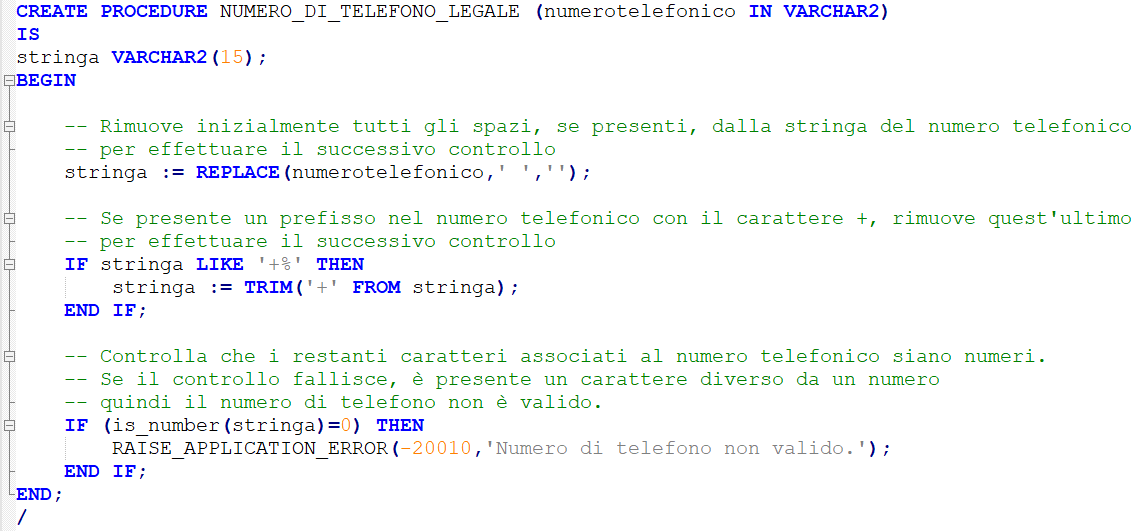
Lo scopo della stored function IS\_NUMBER è quello di verificare che la stringa data in input contenga solo caratteri numerici. Verrà utilizzata successivamente nella definizione di alcuni trigger.



Estratto da **DBProcedureFunzioni.sql** - Definizione della stored function **IS\_NUMBER**

4.3.2 Stored procedure NUMERO\_DI\_TELEFONO\_LEGALE

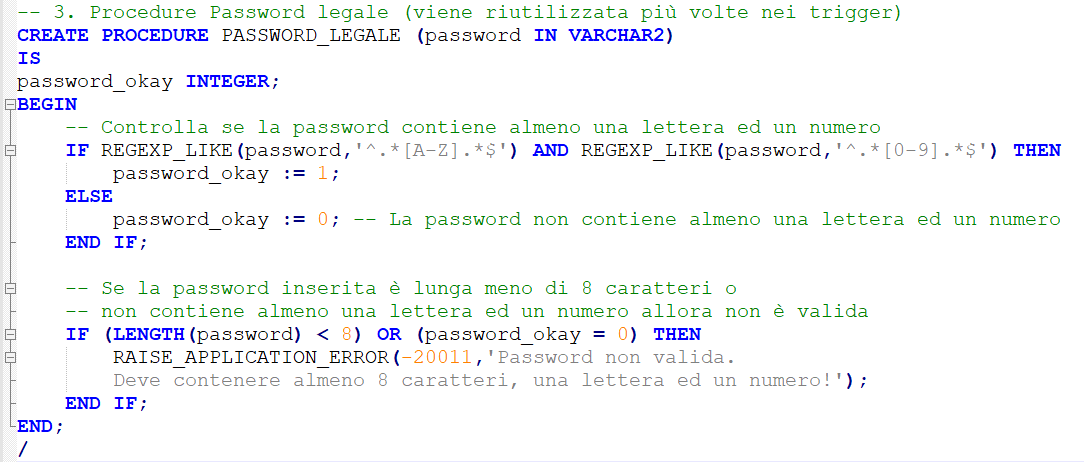
Lo scopo della stored procedure NUMERO\_DI\_TELEFONO\_LEGALE è quello di verificare che una stringa data in input rispetti il formato standard di un numero telefonico. Il controllo esclude il numero di cifre del numero telefonico, che possono essere variabili. Verrà utilizzata successivamente nella definizione di alcuni trigger.



Estratto da **DBProcedureFunzioni.sql** - Definizione della stored procedure **NUMERO\_DI\_TELEFONO\_LEGALE**

4.3.3 Stored procedure PASSWORD\_LEGALE

Lo scopo della stored procedure PASSWORD\_LEGALE è quello di verificare che una stringa data in input rispetti il formato stabilito per una password. Verrà utilizzata successivamente nella definizione di alcuni trigger.



Estratto da **DBProcedureFunzioni.sql** - Definizione della stored procedure **PASSWORD\_LEGALE**

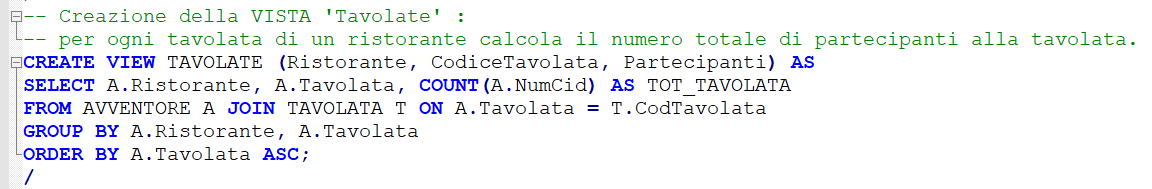
- fine definizione delle procedure e funzioni -

4.4 Viste

Segue la definizione dell’unica vista utilizzata nel database tratta dal file “DBRistorante.sql”.

4.4.1 Vista TAVOLATE

Lo scopo della vista TAVOLATE di calcolare il numero totale di partecipanti alla tavolata, per ogni tavolata presente in ciascun ristorante. Verrà utilizzata successivamente per l’implementazione di una query.



Estratto da **DBRistorante.sql** - Definizione della vista **TAVOLATE**

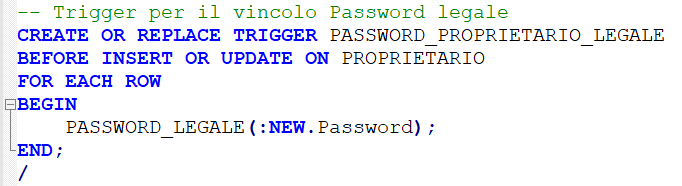
- fine definizione delle viste -

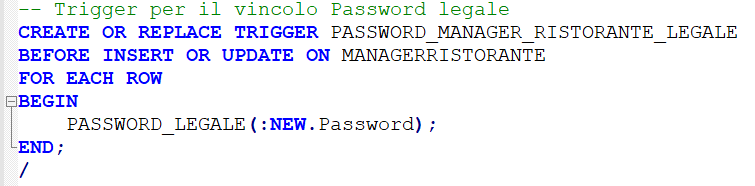
4.5 Implementazione dei vincoli

Seguono le implementazioni dei vari vincoli che non sono già stati mostrati nella definizione delle tabelle. Tutti i vincoli sono tratti dal file “DBRistorante.sql”.

4.5.1 Implementazione del vincolo Password legale

Il vincolo si ripete nella definizione di diverse tabelle. Per questo scopo è stata definita e riutilizzata più volte la stored procedure PASSWORD\_LEGALE.

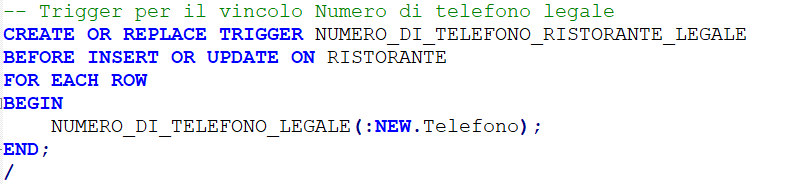


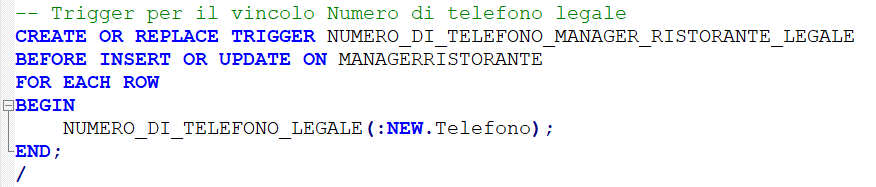


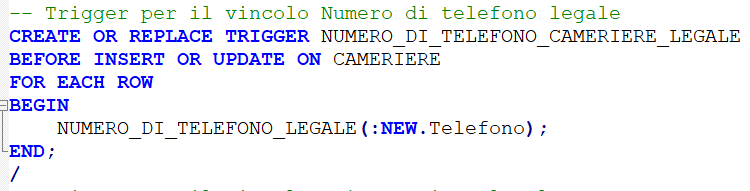
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Password legale**

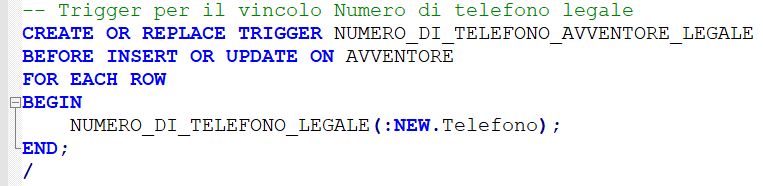
4.5.2 Implementazione del vincolo Numero di telefono legale

Il vincolo si ripete nella definizione di diverse tabelle. Per questo scopo è stata definita e riutilizzata più volte la stored procedure NUMERO\_DI\_TELEFONO\_LEGALE.



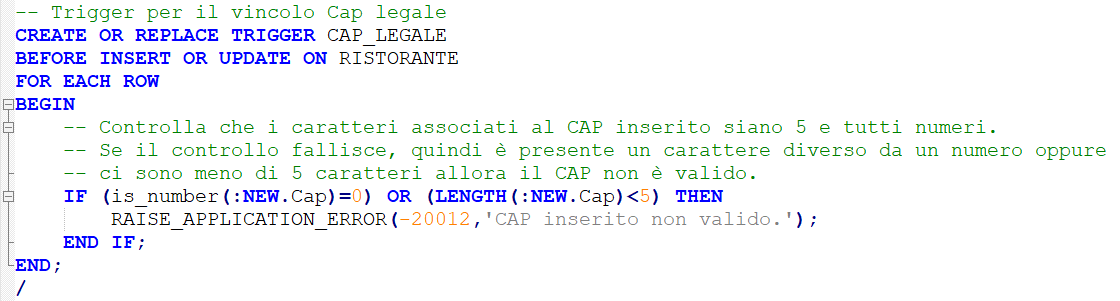






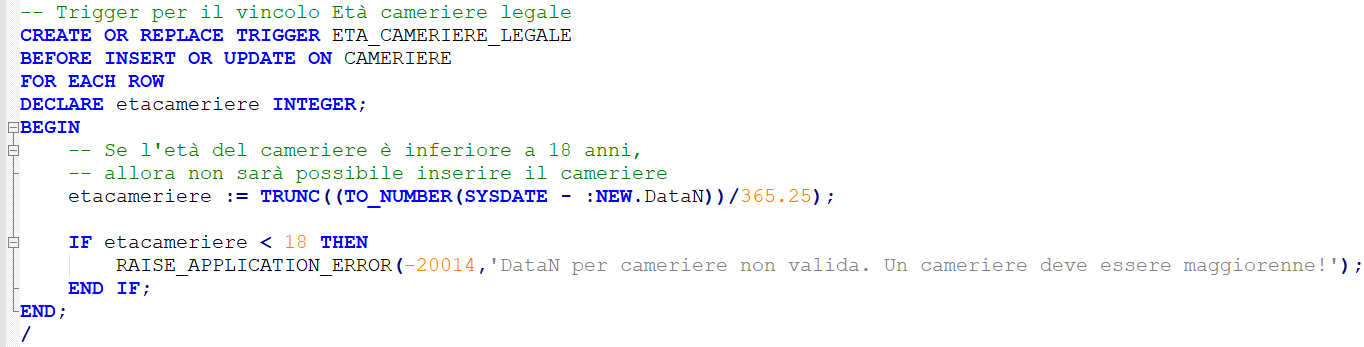
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Numero di telefono legale**

4.5.3 Implementazione del vincolo Cap legale



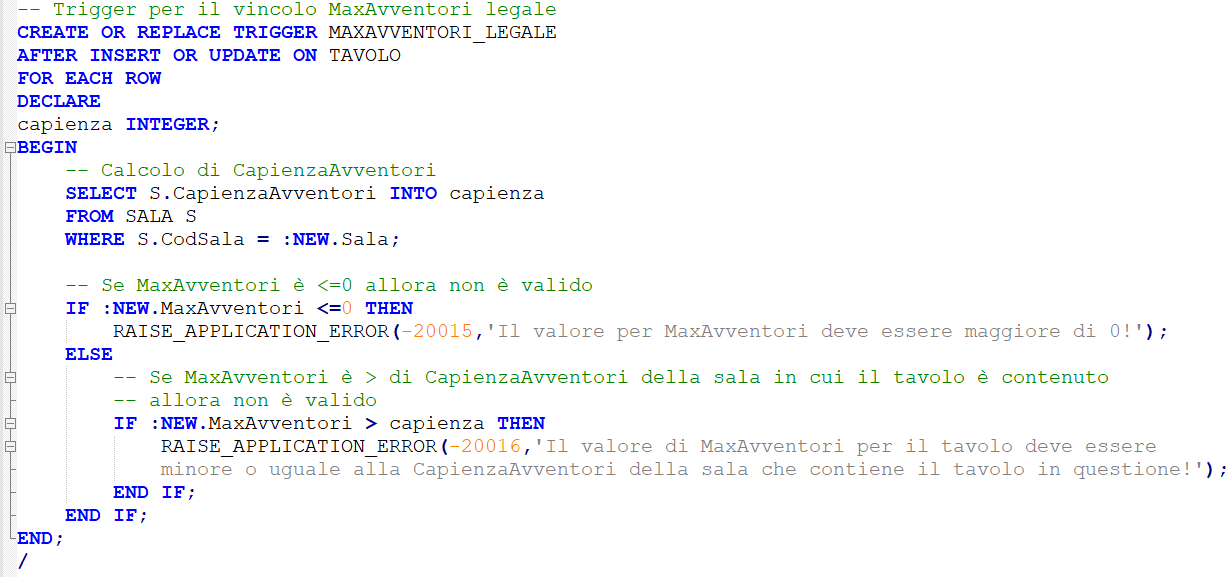
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Cap legale**

4.5.4 Implementazione del vincolo Età cameriere legale



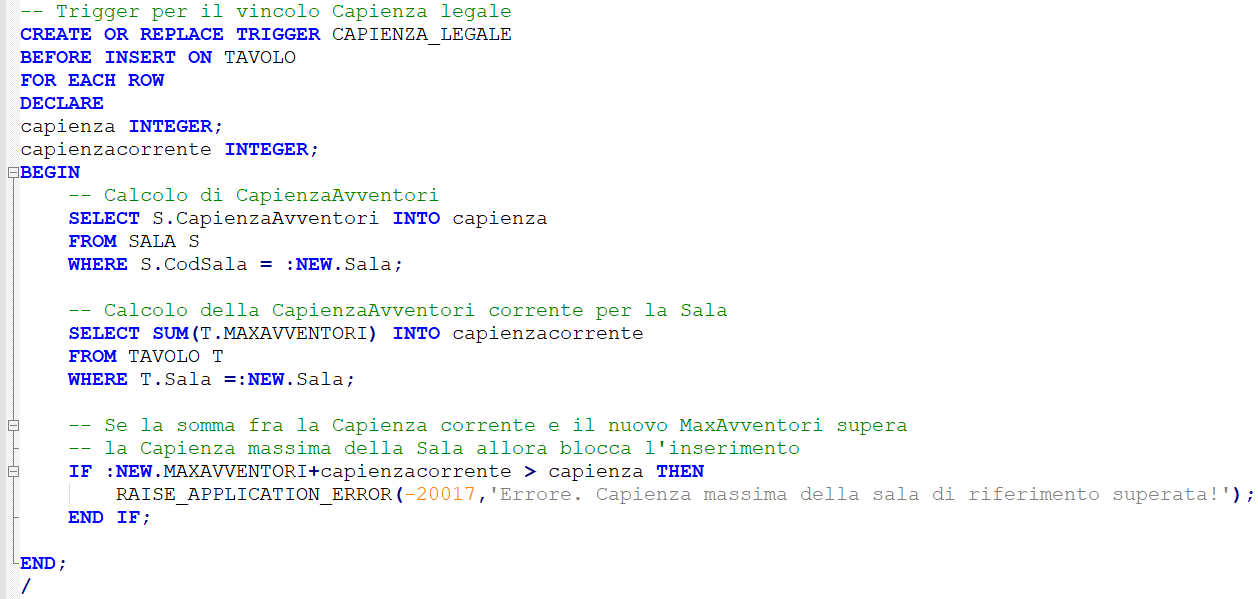
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Età cameriere legale**

4.5.5 Implementazione del vincolo MaxAvventori legale



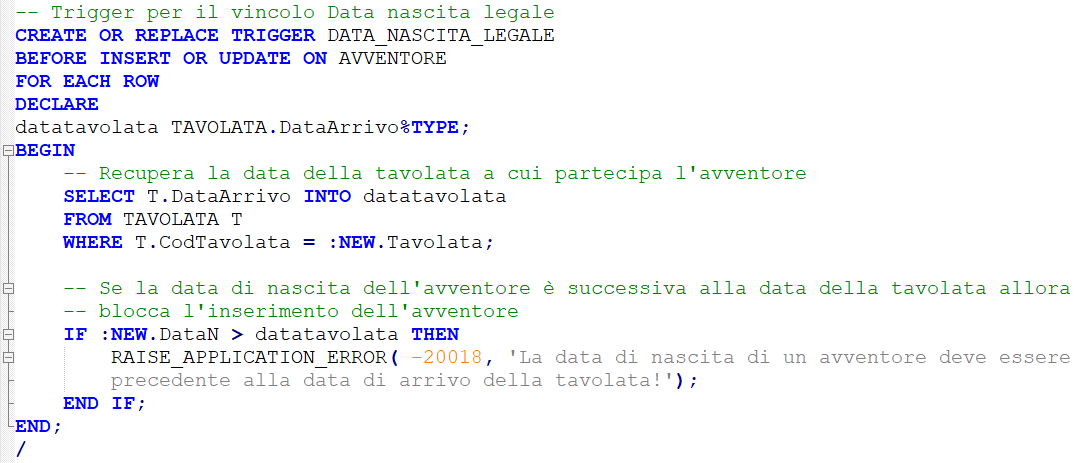
Estratto da **DBRistorante.sql** - Implementazione del vincolo **MaxAvventori legale**

4.5.6 Implementazione del vincolo Capienza legale



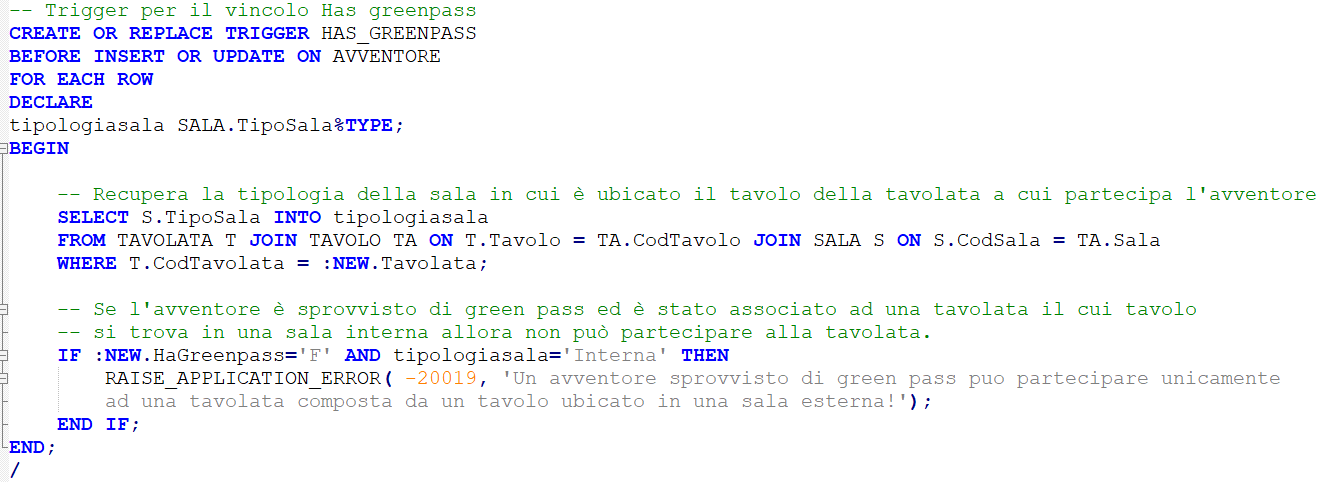
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Capienza legale**

4.5.7 Implementazione del vincolo Data nascita legale



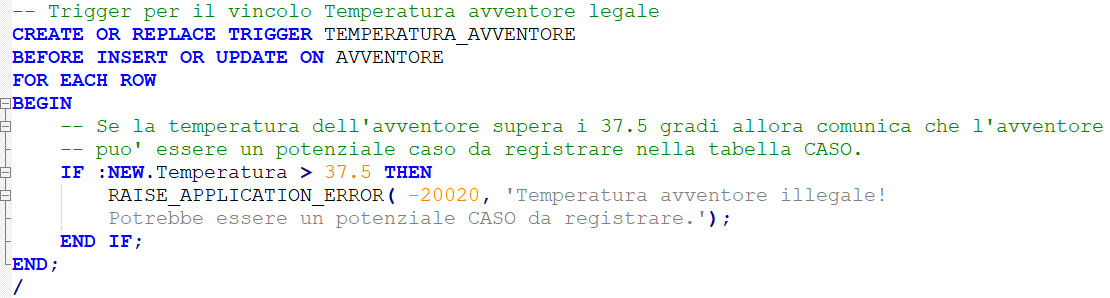
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Data nascita legale**

4.5.8 Implementazione del vincolo Has greenpass



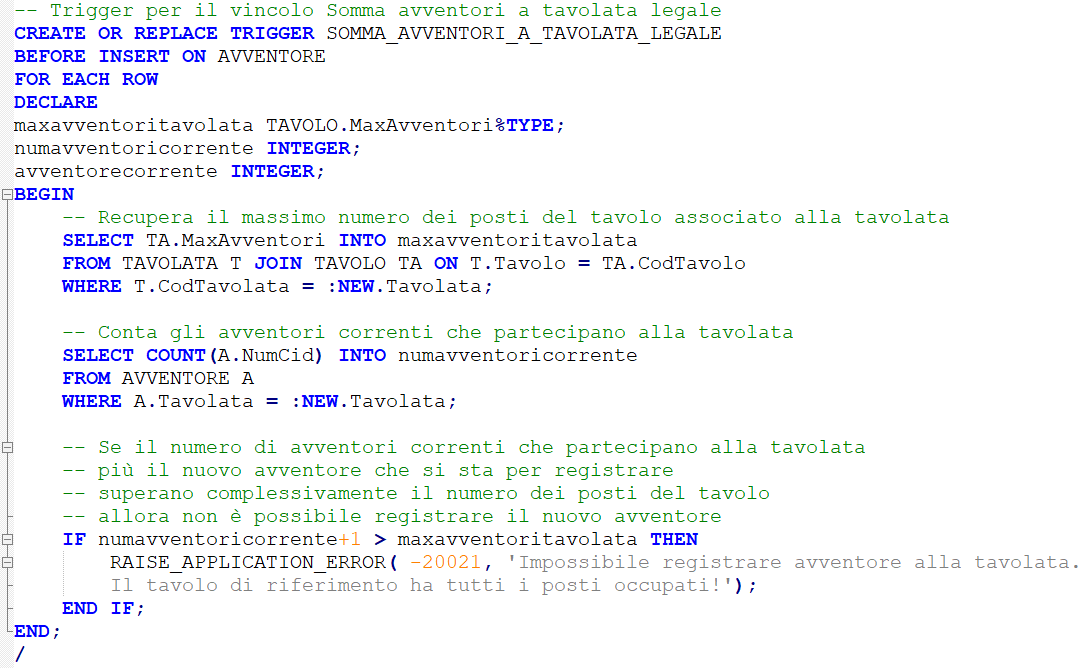
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Has greenpass**

4.5.9 Implementazione del vincolo Temperatura avventore legale



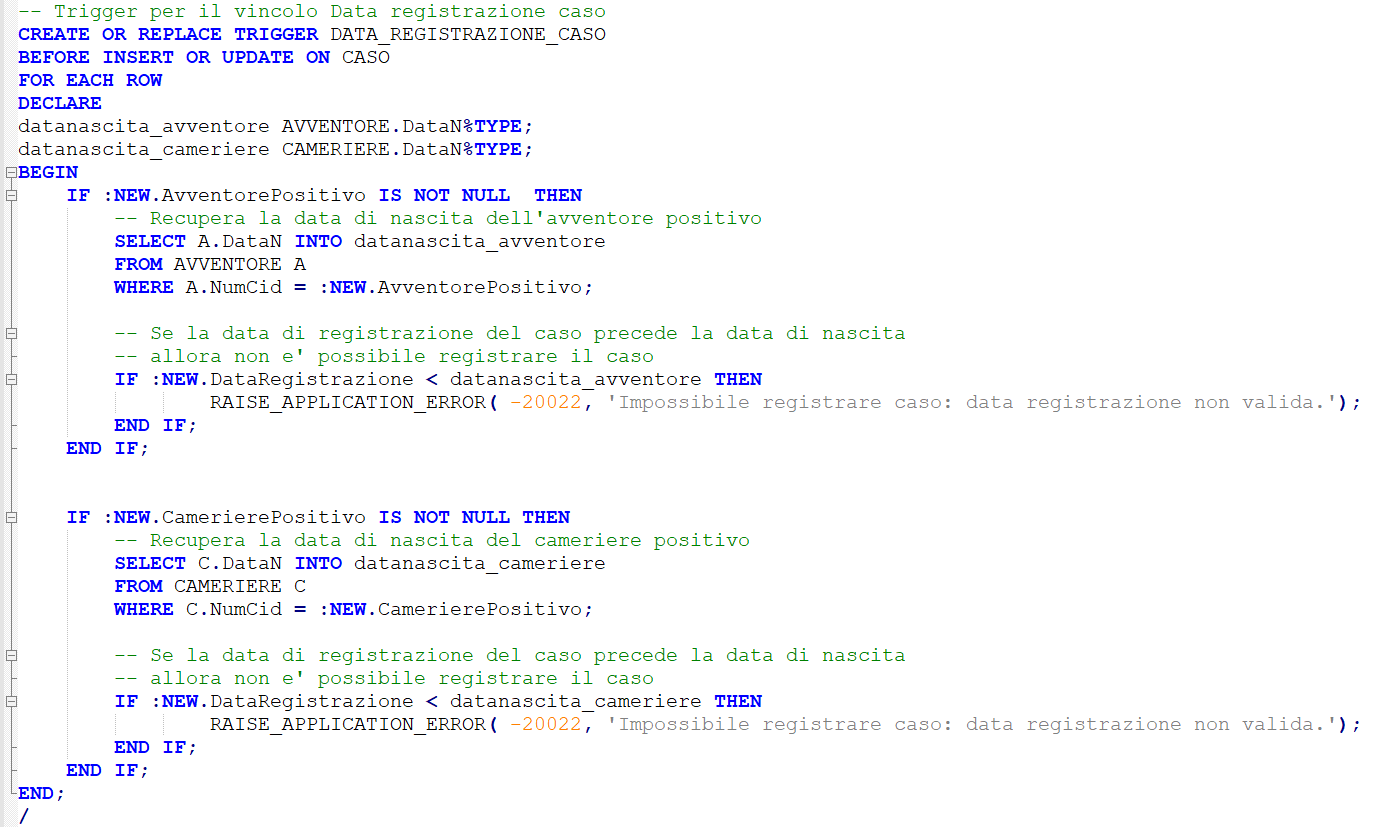
Estratto da **DBRistorante.sql** - Implementazione del vincolo **Temperatura avventore legale**

4.5.10 Implementazione del vincolo Somma avventori a tavolata legale



Estratto da **DBRistorante.sql** - Implementazione del vincolo **Somma avventori a tavolata legale**

4.5.11 Implementazione del vincolo Data registrazione caso



Estratto da **DBRistorante.sql** - Implementazione del vincolo **Data registrazione caso**

- fine definizione dei vincoli -

# **Capitolo 5**

**Esempio d’uso**

## **Introduzione**

In questo capitolo viene mostrato un esempio d’uso del database implementato. Si inizierà mostrando lo script definito per il popolamento del db contenuto nel file “DBInsert.sql”. Successivamente viene mostrata l’implementazione delle possibili query da effettuare sul database, tratte dal file “DBQuery.sql” e definite precedentemente nel [dizionario delle interrogazioni](#_2.4.4_Dizionario_delle).

5.1 Popolamento del database