

UNIVERSITÀ DEGLI STUDI DI NAPOLI
FEDERICO II



DIPARTIMENTO DI INGEGNERIA ELETTRICA
E DELLE TECNOLOGIE DELL'INFORMAZIONE

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

Autori:

Angelo DI MAIO
MATRICOLA N86003699
angelo.dimaio3@studenti.unina.it

Docenti:

Prof. Adriano PERON
Prof. Sergio DI MARTINO
Prof. Luigi Libero Lucio STARACE

Santolo BARRETTA
MATRICOLA N86003666
san.barretta@studenti.unina.it

23 dicembre 2021

Questa pagina è stata lasciata intenzionalmente bianca.

Indice

Indice	3
Capitolo 1	6
 Descrizione del progetto.....	6
1.1 Descrizione sintetica e analisi del problema	6
1.2 L'organizzazione del sistema di tracciamento contatti.....	6
Capitolo 2	8
 Progettazione concettuale	8
Introduzione	8
2.1 Alcune convenzioni per la lettura dei class diagram	8
2.2 Class diagram	9
2.3 Ristrutturazione del class diagram.....	10
2.3.1 Analisi delle informazioni ridondanti.....	10
2.3.2 Analisi degli identificativi	10
2.3.3 Rimozione degli attributi strutturati	10
2.3.4 Rimozione delle gerarchie di specializzazione	11
2.3.5 Class diagram ristrutturato.....	12
2.4 Dizionario dei dati per il class diagram ristrutturato	13
2.4.1 Dizionario delle classi	13
2.4.2 Dizionario delle associazioni	17
2.4.3 Dizionario dei vincoli	19
2.4.4 Dizionario delle interrogazioni	21
Capitolo 3	23
 Progettazione logica	23
Introduzione	23
3.1 Schema logico.....	23
3.1.1 Traduzione in schemi relazionali	23
3.1.2 Traduzione delle associazioni	25
3.1.3 Schema logico generale	26
Capitolo 4	27
 Progettazione fisica	27
Introduzione	27

4.1	Note sull'implementazione.....	27
4.2	Definizione delle tabelle	27
4.2.1	Definizione della tabella PROPRIETARIO	28
4.2.2	Definizione della tabella RISTORANTE	28
4.2.3	Definizione della tabella MANAGERRISTORANTE.....	29
4.2.4	Definizione della tabella SALA	29
4.2.6	Definizione della tabella TAVOLO	30
4.2.7	Definizione della tabella ADIACENZATAVOLO.....	31
4.2.8	Definizione della tabella TAVOLATA	31
4.2.9	Definizione della tabella AVVENTORE	32
4.2.10	Definizione della tabella ACCOGLIENZA	32
4.2.11	Definizione della tabella PARTECIPAZIONETAVOLATA	33
4.2.12	Definizione della tabella CASO	33
4.3	Viste	34
4.3.1	Vista RIEPILOGO_RISTORANTI_PROPRIETARIO	34
4.3.2	Vista RIEPILOGO_TAVOLATE_RISTORANTI_PROPRIETARIO.....	34
4.3.3	Vista RIEPILOGO_AVVENTORI_RISTORANTI	34
4.4	Funzioni, procedure ed altre automazioni.....	35
4.4.1	Stored function IS_NUMBER	35
4.4.2	Stored procedure NUMERO_DI_TELEFONO_LEGALE	35
4.4.3	Stored procedure PASSWORD_LEGALE.....	36
4.5	Implementazione dei vincoli	37
4.5.1	Implementazione del vincolo Password legale	37
4.5.2	Implementazione del vincolo Numero di telefono legale	38
4.5.3	Implementazione del vincolo Cap legale.....	39
4.5.4	Implementazione del vincolo Età cameriere legale.....	39
4.5.5	Implementazione del vincolo MaxAvventori legale	39
4.5.6	Implementazione del vincolo Capienza legale	40
4.5.7	Implementazione del vincolo Temperatura avventore legale	40
4.5.8	Implementazione del vincolo Data nascita legale	41
4.5.9	Implementazione del vincolo Has greenpass.....	41
4.5.10	Implementazione del vincolo Somma avventori a tavolata legale	42
4.5.11	Implementazione del vincolo Data registrazione caso legale	43
Capitolo 5	44	
Manuale d'uso	44	

Introduzione	44
5.1 Popolamento del database con dati di esempio	44
5.1.1 Insert per la tabella PROPRIETARIO.....	44
5.1.2 Insert per la tabella RISTORANTE	44
5.1.3 Insert per la tabella MANAGERRISTORANTE.....	44
5.1.4 Insert per la tabella SALA	45
5.1.5 Insert per la tabella CAMERIERE	45
5.1.6 Insert per la tabella TAVOLO	46
5.1.7 Insert per la tabella ADIACENZATAVOLO.....	48
5.1.8 Insert per la tabella TAVOLATA	50
5.1.9 Insert per la tabella AVVENTORE.....	50
5.1.10 Insert per la tabella ACCOGLIENZA.....	52
5.1.11 Insert per la tabella PARTECIPAZIONETAVOLATA.....	53
5.1.12 Insert per la tabella CASO.....	54
5.2 Esempio di query	55
5.2.1 Numero giornaliero di avventori per ristorante.....	55
5.2.2 Numero mensile di avventori per ristorante	55
5.2.3 Numero giornaliero di avventori per tutti i ristoranti di un proprietario	56
5.2.4 Numero mensile di avventori per tutti i ristoranti di un proprietario	56
5.2.5 Casi positivi di un determinato ristorante per data di arrivo della tavolata	56
5.2.6 Casi positivi di un determinato ristorante per mese di arrivo della tavolata	57
5.2.7 Casi positivi di un determinato ristorante per anno di arrivo della tavolata	57
5.2.8 Casi positivi di tutti i ristoranti di un proprietario per data di arrivo della tavolata.....	57
5.2.9 Casi positivi di tutti i ristoranti di un proprietario per mese di arrivo della tavolata	58
5.2.10 Casi positivi di tutti i ristoranti di un proprietario per anno di arrivo della tavolata	58
5.2.11 Informazioni sugli avventori risultati positivi in un ristorante.....	58
5.2.12 Informazioni sugli avventori risultati positivi in tutti i ristoranti di un proprietario	59
5.2.13 Informazioni sui camerieri risultati positivi in un ristorante.....	59
5.2.14 Informazioni sui camerieri risultati positivi in tutti i ristoranti di un proprietario	60
5.2.15 Avventori positivi con o senza green pass	60
5.2.16 Numero di avventori medio per tavolata di un ristorante.....	61

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:** la Tavolata sarà formata da un tavolo presente in una sala del ristorante ed un singolo o un gruppo di avventori che vi parteciperanno. 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/M6VkXXr>
- **Class diagram ristrutturato:** <https://ibb.co/bQrsSDb>

2.2 Class diagram

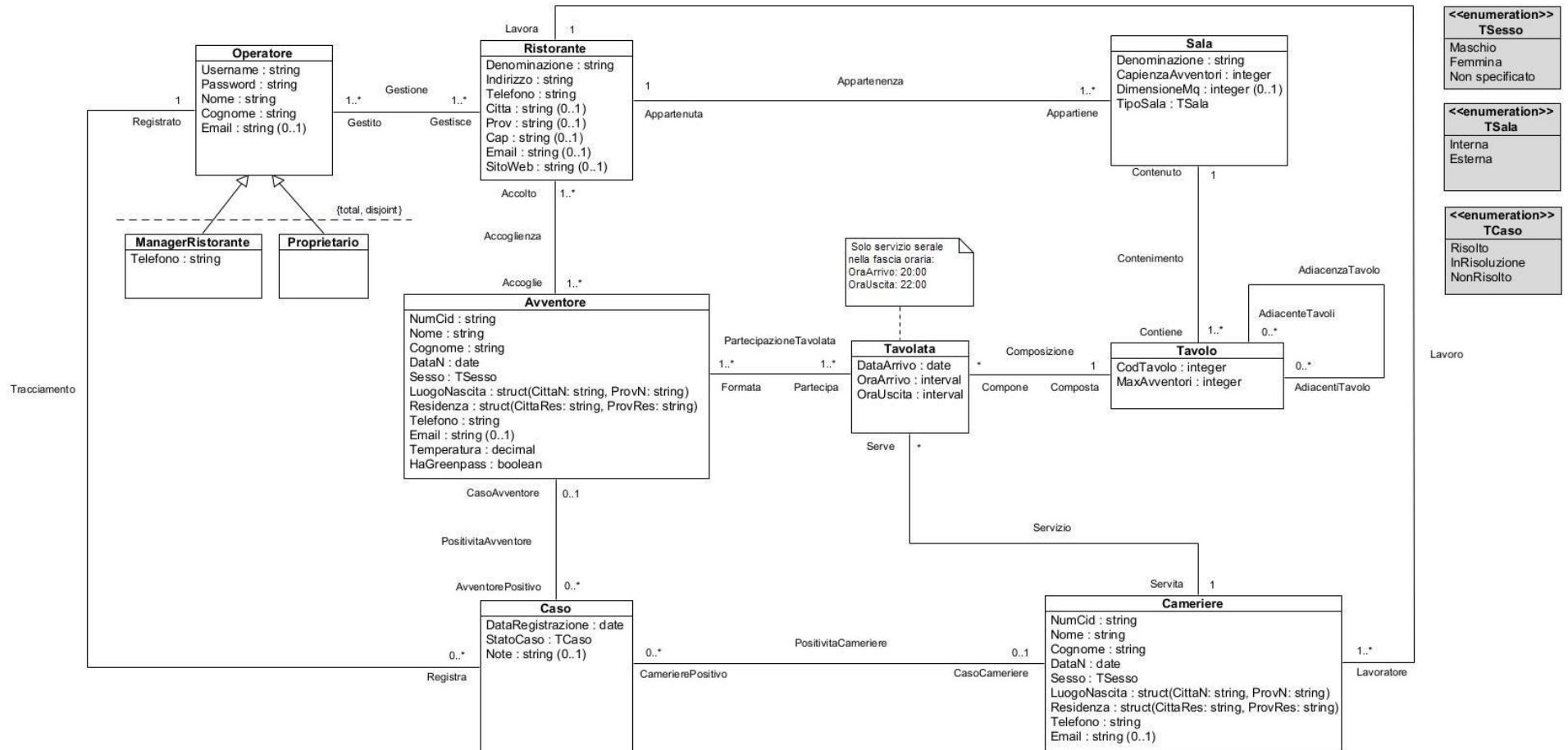


Immagine 2.1: Class diagram della base di dati

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 all'entità 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 presenti in 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 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**.

2.3.5 Class diagram ristrutturato

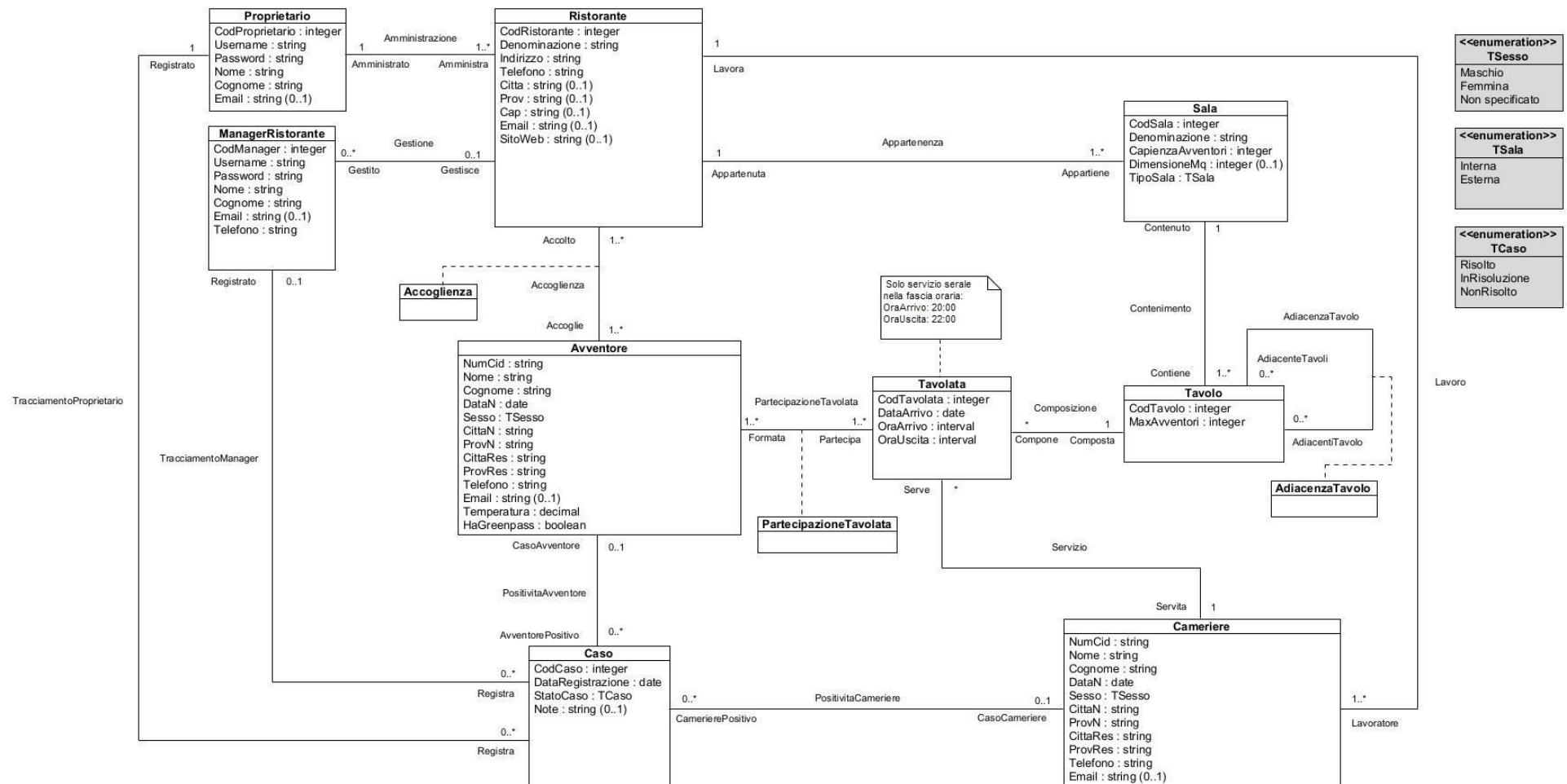


Immagine 2.2: Class diagram ristrutturato della base di dati

2.4 Dizionario dei dati per il class diagram ristrutturato

2.4.1 Dizionario delle classi

<i>Classe</i>	<i>Descrizione</i>	<i>Attributi</i>
Proprietario	Descrittore di un generico proprietario di uno o più ristoranti	<p>CodProprietario (<i>integer</i>): chiave surrogata che serve a identificare un proprietario.</p> <p>Username (<i>string</i>): username associato ad un proprietario.</p> <p>Password (<i>string</i>): password associata ad un proprietario.</p> <p>Nome (<i>string</i>): nome di un proprietario.</p> <p>Cognome (<i>string</i>): cognome di un proprietario.</p> <p>Email (<i>string, opzionale</i>): email associata ad un proprietario.</p>
ManagerRistorante	Descrittore del manager di un ristorante	<p>CodManager (<i>integer</i>): chiave surrogata che serve a identificare univocamente un manager di un ristorante.</p> <p>Username (<i>string</i>): username associato al manager.</p> <p>Password (<i>string</i>): password associata al manager.</p> <p>Nome (<i>string</i>): nome di un manager.</p> <p>Cognome (<i>string</i>): cognome associato di un manager.</p> <p>Email (<i>string, opzionale</i>): email associata al manager.</p> <p>Telefono (<i>string</i>): numero di telefono associato al manager.</p>

Ristorante	Descrittore di un ristorante	<p>CodRistorante (<i>integer</i>): chiave surrogata che serve a identificare univocamente un ristorante.</p> <p>Denominazione (<i>string</i>): nome di un ristorante.</p> <p>Indirizzo (<i>string</i>): indirizzo di un ristorante.</p> <p>Telefono (<i>string</i>): numero di telefono di un ristorante.</p> <p>Citta (<i>string, opzionale</i>): città di ubicazione di un ristorante.</p> <p>Prov (<i>string, opzionale</i>): provincia di ubicazione di un ristorante.</p> <p>Cap (<i>string, opzionale</i>): cap della provincia di ubicazione di un ristorante.</p> <p>Email (<i>string, opzionale</i>): email associata ad un ristorante.</p> <p>SitoWeb (<i>string, opzionale</i>): sito web associato ad un ristorante.</p>
Sala	Descrittore della sala di un ristorante	<p>CodSala (<i>integer</i>): chiave surrogata che serve a identificare univocamente una sala.</p> <p>Denominazione (<i>string</i>): nome di una sala.</p> <p>CapienzaAvventori (<i>integer</i>): capienza massima di avventori per una sala.</p> <p>DimensioneMq (<i>string, opzionale</i>): dimensione in metri quadri di una sala.</p> <p>TipoSala (<i>TSala</i>): attributo che indica se la sala è interna oppure esterna.</p>
Tavolo	Descrittore di un tavolo del ristorante	<p>CodTavolo (<i>integer</i>): chiave surrogata che serve a identificare univocamente un tavolo.</p> <p>MaxAvventori (<i>integer</i>): numero massimo di avventori per ogni tavolo.</p>

Tavolata	Descrittore della tavolata di avventori	<p>CodTavolata (<i>integer</i>): chiave surrogata che serve a identificare univocamente ciascuna tavolata.</p> <p>DataArrivo (<i>date</i>): data in cui si svolge la tavolata.</p> <p>OraArrivo (<i>interval</i>): orario di arrivo della tavolata (vale sempre 20:00).</p> <p>OraUscita (<i>interval</i>): orario di uscita della tavolata (vale sempre 22:00).</p>
Avventore	Descrittore di un generico avventore	<p>NumCid (<i>string</i>): numero della carta di identità che serve anche da chiave primaria per identificare univocamente ciascun avventore.</p> <p>Nome (<i>string</i>): nome di un avventore.</p> <p>Cognome (<i>string</i>): cognome di un avventore.</p> <p>DataN (<i>date</i>): data di nascita di un avventore.</p> <p>Sesso (<i>TSesso</i>): sesso di un avventore.</p> <p>CittaN (<i>string</i>): città di nascita di un avventore.</p> <p>ProvN (<i>string</i>): provincia di nascita di un avventore.</p> <p>CittaRes (<i>string</i>): città di residenza di un avventore.</p> <p>ProvRes (<i>string</i>): provincia di residenza di un avventore.</p> <p>Telefono (<i>string</i>): numero di telefono di un avventore.</p> <p>Email (<i>string, opzionale</i>): email associata ad un avventore.</p> <p>Temperatura (<i>decimal</i>): temperatura misurata all'ingresso di un ristorante.</p> <p>HaGreenpass (<i>boolean</i>): indica il possesso o meno del green pass da parte di un avventore.</p>

Cameriere	Descrittore di un cameriere del ristorante	<p>NumCid (<i>string</i>): numero della carta di identità che serve anche da chiave primaria per identificare univocamente ciascun cameriere.</p> <p>Nome (<i>string</i>): nome di un cameriere.</p> <p>Cognome (<i>string</i>): cognome di un cameriere.</p> <p>DataN (<i>date</i>): data di nascita di un cameriere.</p> <p>Sesso (<i>TSesso</i>): sesso di un cameriere.</p> <p>CittaN (<i>string</i>): città di nascita di un cameriere.</p> <p>ProvN (<i>string</i>): provincia di nascita di un cameriere.</p> <p>CittaRes (<i>string</i>): città di residenza di un cameriere.</p> <p>ProvRes (<i>string</i>): provincia di residenza di un cameriere.</p> <p>Telefono (<i>string</i>): numero di telefono di un cameriere.</p> <p>Email (<i>string, opzionale</i>): email associata ad un cameriere.</p>
Caso	Descrittore di un caso Covid	<p>CodCaso (<i>integer</i>): chiave surrogata che serve per identificare univocamente ciascun caso.</p> <p>DataRegistrazione (<i>date</i>): data di registrazione di un caso.</p> <p>StatoCaso (<i>TStato</i>): stato di un caso (Risolto, InRisoluzione, NonRisolto).</p> <p>Note (<i>string, opzionale</i>): note relative al caso.</p>
Accoglienza	Descrittore dell'accoglienza da parte di ristoranti di avventori	
PartecipazioneTavolata	Descrittore della partecipazione da parte di avventori a tavolate	
AdiacenzaTavolo	Descrittore dell'adiacenza fra tavoli	

Tabella 2.1: Dizionario delle classi - fine.

2.4.2 Dizionario delle associazioni

<i>Associazione</i>	<i>Descrizione</i>	<i>Classi coinvolte</i>
Amministrazione	Esprime l'amministrazione da parte di un proprietario di uno o più ristoranti	<p>Proprietario [1] ruolo (Amministrato): indica il proprietario che amministra uno o più ristoranti.</p> <p>Ristorante [1..*] ruolo (Amministra): indica il ristorante che viene amministrato da un proprietario.</p>
Gestione	Esprime la possibile gestione da parte di uno o più manager di un ristorante.	<p>ManagerRistorante [0..*] ruolo (Gestito): indica il manager che può gestire un ristorante.</p> <p>Ristorante [0..1] ruolo (Gestisce): indica il ristorante che può essere gestito da uno o più manager.</p>
Appartenenza	Esprime l'appartenenza di una o più sale ad un ristorante.	<p>Ristorante [1] ruolo (Appartenuta): indica il ristorante a cui appartengono una o più sale.</p> <p>Sala [1..*] ruolo (Appartiene): indica una sala che appartiene ad un ristorante.</p>
Contenimento	Esprime il contenimento di tavoli da parte di una sala.	<p>Sala [1] ruolo (Contenuto): indica una sala contenente uno o più tavoli.</p> <p>Tavolo [1..*] ruolo (Contiene): indica un tavolo contenuto in una sala.</p>
Servizio	Esprime il servizio di un cameriere ad una o più tavolate.	<p>Cameriere [1] ruolo (Servita): indica il cameriere che serve una tavolata</p> <p>Tavolata [*] ruolo (Serve): indica la tavolata servita da un cameriere.</p>
PartecipazioneTavolata	Esprime uno o più avventori che partecipano ad una o più tavolate.	<p>Tavolata [1..*] ruolo (Partecipa): indica la tavolata a cui partecipa uno o più avventori.</p> <p>Avventore [1..*] ruolo (Formata): indica uno o più avventori che partecipano ad una tavolata.</p>

Composizione	Esprime la composizione di una o più tavolate con un tavolo.	<p>Tavolata [*] ruolo (Compone): indica la tavolata composta da un tavolo.</p> <p>Tavolo [1] ruolo (Composta): indica il tavolo che compone una tavolata.</p>
AdiacenzaTavolo	Esprime la possibile adiacenza tra tavoli del ristorante.	<p>Tavolo [0..*] ruolo (AdiacentiTavolo): indica il tavolo a cui possono essere adiacenti altri tavoli.</p> <p>Tavolo [0..*] ruolo (AdiacenteTavoli): indica la possibilità di adiacenza di altri tavoli ad un tavolo.</p>
Lavoro	Esprime il rapporto lavorativo tra uno o più camerieri ed un ristorante.	<p>Cameriere [1..*] ruolo (Lavoratore): indica il cameriere che lavora per un ristorante.</p> <p>Ristorante [1] ruolo (Lavora): indica il ristorante per cui lavorano uno o più camerieri.</p>
Accoglienza	Esprime l'accoglienza da parte di uno o più ristoranti di uno o più avventori.	<p>Ristorante [1..*] ruolo (Accolto): indica il ristorante che accoglie uno o più avventori</p> <p>Avventore [1..*] ruolo (Accoglie): indica uno o più avventori accolti da uno o più ristoranti.</p>
PositivitàAvventore	Esprime la possibilità di esistenza di uno o più casi di positività per un avventore.	<p>Avventore [0..1] ruolo (CasoAvventore): indica l'avventore che può risultare positivo.</p> <p>Caso [0..*] ruolo (AvventorePositivo): indica il possibile caso di un avventore positivo.</p>
PositivitàCameriere	Esprime la possibilità di esistenza di uno o più casi di positività per un cameriere.	<p>Caso [0..*] ruolo (CamerierePositivo): indica il cameriere che può risultare positivo.</p> <p>Cameriere [0..1] ruolo (CasoCameriere): indica il possibile caso di un cameriere positivo.</p>

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 <i>HaGreenPass</i> dell'entità <i>Avventore</i> vale ‘F’ (<i>falso</i>), 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 <i>Email</i> : <i>Proprietario</i> , <i>ManagerRistorante</i> , <i>Ristorante</i> , <i>Avventore</i> , <i>Cameriere</i> . L' <i>Email</i> 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 <i>Password</i> : <i>Proprietario</i> , <i>ManagerRistorante</i> . La <i>Password</i> deve contenere minimo 8 caratteri e rispettare il seguente formato: deve esserci almeno una lettera e un numero.
Sito web legale	Dominio	Il <i>SitoWeb</i> 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 <i>Telefono</i> : <i>ManagerRistorante</i> , <i>Ristorante</i> , <i>Avventore</i> , <i>Cameriere</i> . Il numero di <i>Telefono</i> può contenere solo numeri ad eccezione del carattere iniziale ‘+’ per il prefisso.
Data nascita legale	Interrelazionale	La <i>DataN</i> di un avventore deve essere precedente o uguale alla <i>DataArrivo</i> della tavolata a cui esso partecipa.

Orario tavolata legale	Dominio	Gli attributi <i>OraArrivo</i> e <i>OraUscita</i> di tavolata sono di default sempre uguali a <i>OraArrivo</i> : 20:00, <i>OraUscita</i> : 22:00.
CAP legale	Dominio	Il <i>Cap</i> deve rispettare il formato standard italiano, ossia deve essere formato esattamente da 5 caratteri numerici.
CapienzaAvventori legale	Dominio	La <i>CapienzaAvventori</i> di una sala deve essere maggiore di 0.
DimensioneMq legale	Dominio	La <i>DimensioneMq</i> di una sala deve essere maggiore di 0.
MaxAvventori legale	Interrelazionale	Il valore di <i>MaxAvventori</i> di un tavolo deve essere maggiore di 0 e minore uguale del valore di <i>CapienzaAvventori</i> della sala in cui il tavolo è contenuto.
Capienza legale	Interrelazionale	La somma totale di <i>MaxAvventori</i> per tutti i tavoli di una sala deve essere minore o uguale alla <i>CapienzaAvventori</i> 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 <i>MaxAvventori</i> del tavolo che compone la tavolata.
Data registrazione caso legale	Interrelazionale	La <i>DataRegistrazione</i> di un caso deve essere successiva alla <i>DataN</i> dell'avventore o cameriere a cui il caso è associato.
Temperatura avventore legale	Dominio	La <i>Temperatura</i> 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.
Adiacenza legale	N-upla	Un tavolo non può essere adiacente a sé stesso.
Unica composizione tavolo a tavolata	Interrelazionale	Per ogni <i>DataArrivo</i> , un tavolo può essere associato al più ad una tavolata.
Unico username proprietario	Intrarelazionale	L' <i>Username</i> di un proprietario deve essere unico.
Unico username manager	Intrarelazionale	L' <i>Username</i> di un manager deve essere unico.

TSesso	Dominio	L'attributo <i>Sesso</i> può assumere solo i valori: ' <i>Maschio</i> ', ' <i>Femmina</i> ', ' <i>Non specificato</i> '.
TSala	Dominio	L'attributo <i>TipoSala</i> può assumere solo i valori: ' <i>Interna</i> ', ' <i>Esterna</i> '.
TCaso	Dominio	L'attributo <i>StatoCaso</i> può assumere solo i valori: ' <i>Risolto</i> ', ' <i>InRisoluzione</i> ', ' <i>NonRisolto</i> '.

Tabella 2.3: Dizionario dei vincoli - fine.

2.4.4 Dizionario delle interrogazioni

<i>Interrogazione</i>	<i>Descrizione</i>	<i>Frequenza</i>	<i>Classi coinvolte</i>
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)

Chiavi esterne: Sala → Sala.CodSala

AdiacenzaTavolo(Tavolo, TavoloAdiacente)

Chiavi esterne: Tavolo → Tavolo.CodTavolo
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)

Chiavi esterne: nessuna.

Accoglienza(Ristorante, Avventore)

Chiavi esterne: Ristorante → Ristorante.CodRistorante
Avventore → Avventore.NumCid

PartecipazioneTavolata(Avventore, Tavolata)

Chiavi esterne: Avventore → Avventore.NumCid
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.

<i>Associazione</i>	<i>Implementazione</i>
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 PartecipazioneTavolata → Tavolata Chiave esterna in PartecipazioneTavolata → Avventore
Composizione	Chiave esterna in Tavolata → Tavolo
AdiacenzaTavolo	Chiave esterna in AdiacenzaTavolo → Tavolo Chiave esterna in AdiacenzaTavolo → Tavolo
Lavoro	Chiave esterna in Cameriere → Ristorante
Accoglienza	Chiave esterna in Accoglienza → Avventore Chiave esterna in Accoglienza → 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	(<u>CodProprietario</u> , Username, Password, Nome, Cognome, Email)
Ristorante	(<u>CodRistorante</u> , Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Email, SitoWeb, <u>Proprietario</u>)
ManagerRistorante	(<u>CodManager</u> , Username, Password, Nome, Cognome, Email, Telefono, <u>RistoranteGestito</u>)
Sala	(<u>CodSala</u> , Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, <u>Ristorante</u>)
Cameriere	(<u>NumCid</u> , Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, <u>Ristorante</u>)
Tavolo	(<u>CodTavolo</u> , MaxAvventori, <u>Sala</u>)
AdiacenzaTavolo	(<u>Tavolo</u> , <u>TavoloAdiacente</u>)
Tavolata	(<u>CodTavolata</u> , DataArrivo, OraArrivo, OraUscita, <u>Tavolo</u> , <u>Cameriere</u>)
Avventore	(<u>NumCid</u> , Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
Accoglienza	(<u>Ristorante</u> , <u>Avventore</u>)
Partecipazione Tavolata	(<u>Avventore</u> , <u>Tavolata</u>)
Caso	(<u>CodCaso</u> , DataRegistrazione, StatoCaso, Note, <u>AvventorePositivo</u> , <u>CamerierePositivo</u> , <u>RegistraProprietario</u> , <u>RegistraManager</u>)

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 contenuto nel file “**DBRistorante.sql**”.

4.2.1 Definizione della tabella PROPRIETARIO

```
-- Creazione della tabella PROPRIETARIO
CREATE TABLE PROPRIETARIO
(
    CodProprietario INTEGER GENERATED BY DEFAULT AS IDENTITY,
    Username        VARCHAR2(64) NOT NULL,
    Password        VARCHAR2(64) NOT NULL,
    Nome            VARCHAR2(64) NOT NULL,
    Cognome         VARCHAR2(64) NOT NULL,
    Email           VARCHAR2(320)
);
/
-- Definizione dei vincoli per la tabella PROPRIETARIO
ALTER TABLE PROPRIETARIO ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_PROPRIETARIO PRIMARY KEY (CodProprietario),
    -- Vincolo Email legale
    CONSTRAINT EMAIL_LEGALE_PROPRIETARIO CHECK (Email LIKE '_%@%.__%' OR Email IS NULL),
    -- Vincolo Unico username proprietario
    CONSTRAINT UNICO_USERNAME_PROPRIETARIO UNIQUE (Username)
);
/
```

4.2.2 Definizione della tabella RISTORANTE

```
-- Creazione della tabella RISTORANTE
CREATE TABLE RISTORANTE
(
    CodRistorante   INTEGER GENERATED BY DEFAULT AS IDENTITY,
    Denominazione   VARCHAR2(64)  NOT NULL,
    Indirizzo       VARCHAR2(64)  NOT NULL,
    Telefono        VARCHAR2(20)   NOT NULL,
    Citta           VARCHAR2(64)   ,
    Prov            VARCHAR2(2)    ,
    Cap             VARCHAR2(5)    ,
    Email           VARCHAR2(320)  ,
    SitoWeb         VARCHAR2(100)  ,
    Proprietario    INTEGER      NOT NULL
);
/
-- Definizione dei vincoli per la tabella RISTORANTE
ALTER TABLE RISTORANTE ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_RISTORANTE PRIMARY KEY (CodRistorante),
    -- Vincolo di chiave esterna
    /*Non sarà possibile eliminare un proprietario a cui sono associati uno o più ristoranti
    ON DELETE NO ACTION è implementato di default da ORACLE*/
    CONSTRAINT FK_PROPRIETARIO_RISTORANTE FOREIGN KEY (Proprietario) REFERENCES PROPRIETARIO(CodProprietario),
    -- Vincolo Email legale
    CONSTRAINT EMAIL_LEGALE_RISTORANTE CHECK (Email LIKE '_%@%.__%' OR Email IS NULL),
    -- Vincolo Sito Web Legale
    CONSTRAINT SITO_WEB_LEGALE CHECK (SitoWeb LIKE 'www.__%.__%' OR SitoWeb IS NULL)
);
/
```

4.2.3 Definizione della tabella MANAGERRISTORANTE

```
-- Creazione della tabella MANAGERRISTORANTE
CREATE TABLE MANAGERRISTORANTE
(
    CodManager      INTEGER GENERATED BY DEFAULT AS IDENTITY,
    Username        VARCHAR2(64) NOT NULL,
    Password        VARCHAR2(64) NOT NULL,
    Nome            VARCHAR2(64) NOT NULL,
    Cognome         VARCHAR2(64) NOT NULL,
    Email           VARCHAR2(320),
    Telefono        VARCHAR2(20) NOT NULL,
    RistoranteGestito  INTEGER
);
/
-- Definizione dei vincoli per la tabella MANAGERRISTORANTE
ALTER TABLE MANAGERRISTORANTE ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_MANAGER_RISTORANTE PRIMARY KEY (CodManager),

    -- Vincolo di chiave esterna
    -- Se si cancella un ristorante vengono eliminati anche i manager che lo gestiscono
    CONSTRAINT FK_MANAGER_RISTORANTE FOREIGN KEY (RistoranteGestito) REFERENCES RISTORANTE(CodRistorante) ON DELETE CASCADE,

    -- Vincolo Email legale
    CONSTRAINT EMAIL_LEGALE_MANAGER_RISTORANTE CHECK (Email LIKE '_%@%.%' OR Email IS NULL),

    -- Vincolo Unico username manager
    CONSTRAINT UNICO_USERNAME_MANAGER_RISTORANTE UNIQUE (Username)
);
/
```

4.2.4 Definizione della tabella SALA

```
-- Creazione della tabella SALA
CREATE TABLE SALA
(
    CodSala          INTEGER GENERATED BY DEFAULT AS IDENTITY,
    Denominazione    VARCHAR2(64) NOT NULL,
    CapienzaAvventori  INTEGER NOT NULL,
    DimensioneMq     INTEGER,
    TipoSala         VARCHAR(10) NOT NULL,
    Ristorante        INTEGER NOT NULL
);
/
-- Definizione dei vincoli per la tabella SALA
ALTER TABLE SALA ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_SALA PRIMARY KEY (CodSala),

    -- Vincolo di chiave esterna
    -- Se si cancella un ristorante vengono eliminate anche le sale che vi appartengono
    CONSTRAINT FK_SALA_RISTORANTE FOREIGN KEY (Ristorante) REFERENCES RISTORANTE(CodRistorante) ON DELETE CASCADE,

    -- Vincolo CapienzaAvventori legale
    CONSTRAINT CAPIENZA_AVVENTORI_LEGALE CHECK (CapienzaAvventori>0),

    -- Vincolo DimensioneMq legale
    CONSTRAINT DIMENSIONE_MQ_LEGALE CHECK (DimensioneMq>0 OR DimensioneMq IS NULL),

    -- Vincolo TSala
    CONSTRAINT TSALA CHECK (TipoSala IN ('Interna','Esterna'))
);
/
```

4.2.5 Definizione della tabella CAMERIERE

```
-- Creazione della tabella CAMERIERE
CREATE TABLE CAMERIERE
(
    NumCid          VARCHAR2(9)      NOT NULL,
    Nome            VARCHAR2(64)     NOT NULL,
    Cognome         VARCHAR2(64)     NOT NULL,
    DataN           DATE            NOT NULL,
    Sesso            VARCHAR2(20)     NOT NULL,
    CittaN          VARCHAR2(64)     NOT NULL,
    ProvN           VARCHAR2(2)      NOT NULL,
    CittaRes        VARCHAR2(64)     NOT NULL,
    ProvRes         VARCHAR2(2)      NOT NULL,
    Telefono         VARCHAR2(20)     NOT NULL,
    Email            VARCHAR2(320)    NOT NULL,
    Ristorante       INTEGER          NOT NULL
);
/
-- Definizione dei vincoli per la tabella CAMERIERE
ALTER TABLE CAMERIERE ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_CAMERIERE PRIMARY KEY (NumCid),
    -- Vincolo di chiave esterna
    -- Se si cancella un ristorante vengono eliminate anche i camerieri che vi lavorano
    CONSTRAINT FK_CAMERIERE_RISTORANTE FOREIGN KEY (Ristorante) REFERENCES RISTORANTE(CodRistorante) ON DELETE CASCADE,
    -- Vincolo Email legale
    CONSTRAINT EMAIL_LEGALE_CAMERIERE CHECK (Email LIKE '_%@%.%' OR Email IS NULL),
    -- Vincolo Sesso
    CONSTRAINT TSesso CHECK (Sesso IN ('Maschio','Femmina','Non specificato'))
);
/
```

4.2.6 Definizione della tabella TAVOLO

```
-- Creazione della tabella TAVOLO
CREATE TABLE TAVOLO
(
    CodTavolo        INTEGER GENERATED BY DEFAULT AS IDENTITY,
    MaxAvventori     INTEGER      NOT NULL,
    Sala             INTEGER      NOT NULL
);
/
-- Definizione dei vincoli per la tabella TAVOLO
ALTER TABLE TAVOLO ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_TAVOLO PRIMARY KEY (CodTavolo),
    -- Vincolo di chiave esterna
    -- Se si cancella una sala vengono eliminati anche i tavoli in essa contenuti
    CONSTRAINT FK_SALA_TAVOLO FOREIGN KEY (Sala) REFERENCES SALA(CodSala) ON DELETE CASCADE
);
/
```

4.2.7 Definizione della tabella ADIACENZATAVOLO

```
-- Creazione della tabella ADIACENZATAVOLO
CREATE TABLE ADIACENZATAVOLO
(
    Tavolo          INTEGER,
    TavoloAdiacente INTEGER
);
/
-- Definizione dei vincoli per la tabella ADIACENZATAVOLO
ALTER TABLE ADIACENZATAVOLO ADD
(
    -- Vincoli di chiave esterna
    CONSTRAINT FK_TAVOLO FOREIGN KEY (Tavolo) REFERENCES TAVOLO(CodTavolo) ON DELETE CASCADE,
    CONSTRAINT FK_TAVOLOADIACENTE FOREIGN KEY (TavoloAdiacente) REFERENCES TAVOLO(CodTavolo) ON DELETE CASCADE,

    -- Vincolo Adiacenza legale
    CONSTRAINT ADIACENZA_LEGALE CHECK (Tavolo <> TavoloAdiacente)
);
/
```

4.2.8 Definizione della tabella TAVOLATA

```
-- Creazione della tabella TAVOLATA
CREATE TABLE TAVOLATA
(
    CodTavolata      INTEGER GENERATED BY DEFAULT AS IDENTITY,
    DataArrivo        DATE                               DEFAULT SYSDATE NOT NULL,
    OraArrivo         INTERVAL DAY(0) TO SECOND(0)      DEFAULT INTERVAL '20:00' HOUR TO MINUTE NOT NULL,
    OraUsicta         INTERVAL DAY(0) TO SECOND(0)      DEFAULT INTERVAL '22:00' HOUR TO MINUTE NOT NULL,
    Tavolo            INTEGER                           NOT NULL,
    Cameriere         VARCHAR2(9)                      NOT NULL
);
/
-- Definizione dei vincoli per la tabella TAVOLATA
ALTER TABLE TAVOLATA ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_TAVOLATA PRIMARY KEY (CodTavolata),

    -- Vincolo di chiave esterna
    -- Se si elimina un tavolo viene eliminata anche la tavolata ad esso associato
    CONSTRAINT FK_TAVOLO_TAVOLATA FOREIGN KEY (Tavolo) REFERENCES TAVOLO(CodTavolo) ON DELETE CASCADE,

    -- Vincolo di chiave esterna
    CONSTRAINT FK_CAMERIERE_TAVOLATA FOREIGN KEY (Cameriere) REFERENCES CAMERIERE(NumCid),

    -- Vincolo Unica composizione tavolo a tavolata
    CONSTRAINT UNICA_COMPOSIZIONE_TAVOLO_A_TAVOLATA UNIQUE(DataArrivo,Tavolo)
);
/
```

4.2.9 Definizione della tabella AVVENTORE

```
-- Creazione della tabella AVVENTORE
CREATE TABLE AVVENTORE
(
    NumCid          VARCHAR2(9)      NOT NULL,
    Nome            VARCHAR2(64)     NOT NULL,
    Cognome         VARCHAR2(64)     NOT NULL,
    DataN           DATE            NOT NULL,
    Sesso            VARCHAR2(15)     NOT NULL,
    CittaN          VARCHAR2(64)     NOT NULL,
    ProvN           VARCHAR2(2)      NOT NULL,
    CittaRes        VARCHAR2(64)     NOT NULL,
    ProvRes         VARCHAR2(2)      NOT NULL,
    Telefono         VARCHAR2(20)     NOT NULL,
    Email            VARCHAR2(320)    NOT NULL,
    Temperatura     DECIMAL(3,1)     NOT NULL,
    HaGreenpass     CHAR             DEFAULT 'F' NOT NULL
);
/
-- Definizione dei vincoli per la tabella AVVENTORE
ALTER TABLE AVVENTORE ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_AVVENTORE PRIMARY KEY (NumCid),

    -- Vincolo Email legale
    CONSTRAINT EMAIL_LEGALE_AVVENTORE CHECK (Email LIKE '_%@___.__%') OR Email IS NULL),

    -- Vincolo TSesso
    CONSTRAINT TSesso_AVVENTORE CHECK (Sesso IN ('Maschio','Femmina','Non specificato')),

    -- Vincolo per il dominio dell'attributo booleano HaGreenPass
    CONSTRAINT HAGREENPASS_VALUES CHECK (HaGreenPass IN('V','F')),

    -- Vincolo Temperatura avventore legale
    CONSTRAINT TEMPERATURA_AVVENTORE_LEGALE CHECK (Temperatura BETWEEN 35.0 AND 37.5)
);
/
```

4.2.10 Definizione della tabella ACCOGLIENZA

```
-- Creazione della tabella ACCOGLIENZA
CREATE TABLE ACCOGLIENZA
(
    Ristorante  INTEGER      NOT NULL,
    Avventore   VARCHAR2(9)   NOT NULL
);
/
-- Definizione dei vincoli per la tabella ACCOGLIENZA
ALTER TABLE ACCOGLIENZA ADD
(
    -- Vincoli di chiave esterna
    CONSTRAINT FK_RISTORANTE FOREIGN KEY (Ristorante) REFERENCES RISTORANTE(CodRistorante) ON DELETE CASCADE,
    CONSTRAINT FK_AVVENTORE FOREIGN KEY (Avventore) REFERENCES AVVENTORE(NumCid) ON DELETE CASCADE
);
/
```

4.2.11 Definizione della tabella PARTECIPAZIONETAVOLATA

```
-- Creazione della tabella PARTECIPAZIONETAVOLATA
CREATE TABLE PARTECIPAZIONETAVOLATA
(
    Avventore  VARCHAR2(9) NOT NULL,
    Tavolata   INTEGER      NOT NULL
);
/
-- Definizione dei vincoli per la tabella PARTECIPAZIONETAVOLATA
ALTER TABLE PARTECIPAZIONETAVOLATA ADD
(
    -- Vincoli di chiave esterna
    CONSTRAINT FK_PARTECIPAZIONE_AVVENTORE FOREIGN KEY (Avventore) REFERENCES AVVENTORE(NumCid) ON DELETE CASCADE,
    CONSTRAINT FK_PARTECIPAZIONE_TAVOLATA FOREIGN KEY (Tavolata) REFERENCES TAVOLATA(CodTavolata) ON DELETE CASCADE
);
/
```

4.2.12 Definizione della tabella CASO

```
-- Creazione della tabella CASO
CREATE TABLE CASO
(
    CodCaso  INTEGER GENERATED BY DEFAULT AS IDENTITY,
    DataRegistrazione DATE          NOT NULL,
    StatoCaso    VARCHAR2(20)     DEFAULT 'NonRisolto' NOT NULL,
    Note        VARCHAR2(100),
    AvventorePositivo VARCHAR2(9),
    CamerierePositivo VARCHAR2(9),
    RegistraProprietario INTEGER,
    RegistraManager  INTEGER
);
/
-- Definizione dei vincoli per la tabella CASO
ALTER TABLE CASO ADD
(
    -- Vincolo di chiave primaria
    CONSTRAINT PK_CASO PRIMARY KEY (CodCaso),

    -- Vincolo di chiave esterna
    CONSTRAINT FK_AVVENTORE_POSITIVO FOREIGN KEY (AvventorePositivo) REFERENCES AVVENTORE(NumCid) ON DELETE SET NULL,

    -- Vincolo di chiave esterna
    CONSTRAINT FK_CAMERIERE_POSITIVO FOREIGN KEY (CamerierePositivo) REFERENCES CAMERIERE(NumCid) ON DELETE SET NULL,

    -- Vincolo di chiave esterna
    CONSTRAINT FK_REGISTRA_PROPRIETARIO FOREIGN KEY (RegistraProprietario) REFERENCES PROPRIETARIO(CodProprietario) ON DELETE SET NULL,

    -- Vincolo di chiave esterna
    CONSTRAINT FK_REGISTRA_MANAGER FOREIGN KEY (RegistraManager) REFERENCES MANAGERRISTORANTE(CodManager) ON DELETE SET NULL,

    -- Vincolo TCaso
    CONSTRAINT TCaso CHECK (StatoCaso IN ('Risolto','InRisoluzione','NonRisolto'))
);
/
```

4.3 Viste

Segue la definizione delle viste utilizzate nel database tratte dal file “DBRistorante.sql”.

4.3.1 Vista RIEPILOGO RISTORANTI PROPRIETARIO

La vista, dato il codice di un proprietario, mostra un riepilogo generale di informazioni su tutti i ristoranti amministrati dal proprietario. Supponiamo che il codice del proprietario sia uguale a 1.

```
CREATE VIEW RIEPILOGO_RISTORANTI_PROPRIETARIO (CodRistorante, Ristorante, Sala, CapienzaSala,
    TipoSala, TavoloInSala, MaxAvventoriTavolo) AS
SELECT R.CodRistorante, R.Denominazione, S.CodSala, S.CapienzaAvventori,
    S.TipoSala, T.CodTavolo, T.MaxAvventori
FROM PROPRIETARIO P JOIN RISTORANTE R ON P.CodProprietario = R.Proprietario
    JOIN SALA S ON R.CodRistorante = S.Ristorante
    JOIN TAVOLO T ON T.Sala = S.CodSala
WHERE P.CodProprietario = 1
ORDER BY R.CodRistorante ASC, S.CodSala ASC, T.CodTavolo ASC;
```

4.3.2 Vista RIEPILOGO TAVOLATE RISTORANTI PROPRIETARIO

La vista, dato il codice di un proprietario, mostra un riepilogo generale di informazioni su tutte le tavolate create nei ristoranti amministrati dal proprietario. Supponiamo che il codice del proprietario sia uguale a 1.

```
CREATE VIEW RIEPILOGO_TAVOLATE_RISTORANTI_PROPRIETARIO (CodRistorante, Ristorante, DataArrivo, CodTavolata, Sala, TipoSala,
    Tavolo, MaxAvventoriTavolo, PartecipantiTavolata) AS
SELECT R.CodRistorante, R.Denominazione, TA.DataArrivo, TA.CodTavolata, S.CodSala, S.TipoSala,
    TA.Tavolo, T.MaxAvventori, COUNT(PT.Avventore)
FROM PROPRIETARIO P JOIN RISTORANTE R ON P.CodProprietario = R.Proprietario
    JOIN SALA S ON R.CodRistorante = S.Ristorante
    JOIN TAVOLO T ON T.Sala = S.CodSala
    JOIN TAVOLATA TA ON TA.Tavolo = T.CodTavolo
    JOIN PARTECIPAZIONETAVOLATA PT ON PT.Tavolata = TA.CodTavolata
WHERE P.CodProprietario = 1
GROUP BY R.CodRistorante, R.Denominazione, TA.DataArrivo, TA.CodTavolata, S.CodSala, S.TipoSala, TA.Tavolo, T.MaxAvventori
ORDER BY R.CodRistorante ASC, TA.DataArrivo ASC, TA.CodTavolata ASC;
```

4.3.3 Vista RIEPILOGO AVVENTORI RISTORANTI

La vista, dato il codice di un proprietario, mostra un riepilogo generale di informazioni su tutti gli avventori accolti nei ristoranti amministrati dal proprietario. Supponiamo che il codice del proprietario sia uguale a 1.

```
CREATE VIEW RIEPILOGO_AVVENTORI_RISTORANTI (CodRistorante, Ristorante, DataArrivo, NumCid, Nome, Cognome, DataN,
    Sesso, Telefono, Tavolata, Tavolo, Sala) AS
SELECT R.CodRistorante, R.Denominazione, T.DataArrivo, AVV.NumCid, AVV.Nome, AVV.Cognome, AVV.DataN,
    AVV.Sesso, AVV.Telefono, PT.Tavolata, T.Tavolo, TA.Sala
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
    JOIN AVVENTORE AVV ON A.Avventore = AVV.NumCid
    JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = AVV.NumCid
    JOIN TAVOLATA T ON T.CodTavolata = PT.Tavolata
    JOIN TAVOLO TA ON TA.CodTavolo = T.Tavolo
WHERE R.Proprietario = 1
ORDER BY R.CodRistorante ASC, T.DataArrivo ASC, PT.Tavolata ASC;
```

4.4 Funzioni, procedure ed altre automazioni

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

4.4.1 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.

```
CREATE FUNCTION IS_NUMBER (stringa IN VARCHAR2) RETURN INT
IS
valorenumerico NUMBER;
BEGIN
    -- Converte da valore numerico a stringa e ritorna 1
    valorenumerico := TO_NUMBER(stringa);
    RETURN 1;

    -- Se fallisce la conversione, errore e ritorna 0
    EXCEPTION WHEN VALUE_ERROR THEN
        RETURN 0;
END;
/
```

4.4.2 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.

```
CREATE PROCEDURE NUMERO_DI_TELEFONO_LEGALE (numerotelefonico IN VARCHAR2)
IS
stringa VARCHAR2(15);
BEGIN

    -- Rimuove inizialmente tutti gli spazi, se presenti, dalla stringa del numero telefonico
    -- per effettuare il successivo controllo
    stringa := REPLACE(numerotelefonico, ' ', '');

    -- Se presente un prefisso nel numero telefonico con il carattere +, rimuove quest'ultimo
    -- per effettuare il successivo controllo
    IF stringa LIKE '+%' THEN
        stringa := TRIM('+ ' FROM stringa);
    END IF;

    -- Controlla che i restanti caratteri associati al numero telefonico siano numeri.
    -- Se il controllo fallisce, è presente un carattere diverso da un numero
    -- quindi il numero di telefono non è valido.
    IF (is_number(stringa)=0) THEN
        RAISE_APPLICATION_ERROR(-20010,'Numero di telefono non valido.');
    END IF;
END;
/
```

4.4.3 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.

```
-- 3. Procedure Password legale (viene riutilizzata più volte nei trigger)
CREATE PROCEDURE PASSWORD_LEGAL (password IN VARCHAR2)
IS
password_okay INTEGER;
BEGIN
    -- Controlla se la password contiene almeno una lettera ed un numero
    IF REGEXP_LIKE(password,'^.*[A-Z].*\$') AND REGEXP_LIKE(password,'^.*[0-9].*\$') THEN
        password_okay := 1;
    ELSE
        password_okay := 0; -- La password non contiene almeno una lettera ed un numero
    END IF;

    -- Se la password inserita è lunga meno di 8 caratteri o
    -- non contiene almeno una lettera ed un numero allora non è valida
    IF (LENGTH(password) < 8) OR (password_okay = 0) THEN
        RAISE_APPLICATION_ERROR(-20011,'Password non valida.
        Deve contenere almeno 8 caratteri, una lettera ed un numero!');
    END IF;
END;
/
```

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_LEGAL.

```
-- Trigger per il vincolo Password legale
CREATE OR REPLACE TRIGGER PASSWORD_PROPRIETARIO_LEGAL
BEFORE INSERT OR UPDATE ON PROPRIETARIO
FOR EACH ROW
BEGIN
    PASSWORD_LEGAL (:NEW.Password);
END;
/

-- Trigger per il vincolo Password legale
CREATE OR REPLACE TRIGGER PASSWORD_MANAGER_RISTORANTE_LEGAL
BEFORE INSERT OR UPDATE ON MANAGERRISTORANTE
FOR EACH ROW
BEGIN
    PASSWORD_LEGAL (:NEW.Password);
END;
/
```

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_LEGAL.

```
-- Trigger per il vincolo Numero di telefono legale
CREATE OR REPLACE TRIGGER NUMERO_DI_TELEFONO_RISTORANTE_LEGAL
BEFORE INSERT OR UPDATE ON RISTORANTE
FOR EACH ROW
BEGIN
    NUMERO_DI_TELEFONO_LEGAL (:NEW.Telefono);
END;
/

-- Trigger per il vincolo Numero di telefono legale
CREATE OR REPLACE TRIGGER NUMERO_DI_TELEFONO_MANAGER_RISTORANTE_LEGAL
BEFORE INSERT OR UPDATE ON MANAGERRISTORANTE
FOR EACH ROW
BEGIN
    NUMERO_DI_TELEFONO_LEGAL (:NEW.Telefono);
END;
/

-- Trigger per il vincolo Numero di telefono legale
CREATE OR REPLACE TRIGGER NUMERO_DI_TELEFONO_CAMERIERE_LEGAL
BEFORE INSERT OR UPDATE ON CAMERIERE
FOR EACH ROW
BEGIN
    NUMERO_DI_TELEFONO_LEGAL (:NEW.Telefono);
END;
/

-- Trigger per il vincolo Numero di telefono legale
CREATE OR REPLACE TRIGGER NUMERO_DI_TELEFONO_AVVENTORE_LEGAL
BEFORE INSERT OR UPDATE ON AVVENTORE
FOR EACH ROW
BEGIN
    NUMERO_DI_TELEFONO_LEGAL (:NEW.Telefono);
END;
/
```

4.5.3 Implementazione del vincolo Cap legale

```
-- Trigger per il vincolo Cap legale
CREATE OR REPLACE TRIGGER CAP_LEGALE
BEFORE INSERT OR UPDATE ON RISTORANTE
FOR EACH ROW
BEGIN
    -- Controlla che i caratteri associati al CAP inserito siano 5 e tutti numeri.
    -- Se il controllo fallisce, quindi è presente un carattere diverso da un numero oppure
    -- ci sono meno di 5 caratteri allora il CAP non è valido.
    IF (is_number(:NEW.Cap)=0) OR (LENGTH(:NEW.Cap)<5) THEN
        RAISE_APPLICATION_ERROR(-20012,'CAP inserito non valido.');
    END IF;
END;
/
```

4.5.4 Implementazione del vincolo Età cameriere legale

```
-- Trigger per il vincolo Età cameriere legale
CREATE OR REPLACE TRIGGER ETA_CAMERIERE_LEGALE
BEFORE INSERT OR UPDATE ON CAMERIERE
FOR EACH ROW
DECLARE etacameriere INTEGER;
BEGIN
    -- Se l'età del cameriere è inferiore a 18 anni,
    -- allora non sarà possibile inserire il cameriere
    etacameriere := TRUNC((TO_NUMBER(SYSDATE - :NEW.DataN))/365.25);

    IF etacameriere < 18 THEN
        RAISE_APPLICATION_ERROR(-20014,'DataN per cameriere non valida. Un cameriere deve essere maggiorenne!');
    END IF;
END;
/
```

4.5.5 Implementazione del vincolo MaxAvventori legale

```
-- Trigger per il vincolo MaxAvventori legale
CREATE OR REPLACE TRIGGER MAXAVVENTORI_LEGALE
AFTER INSERT OR UPDATE ON TAVOLO
FOR EACH ROW
DECLARE
capienza INTEGER;
BEGIN
    -- Calcolo di CapienzaAvventori
    SELECT S.CapienzaAvventori INTO capienza
    FROM SALA S
    WHERE S.CodSala = :NEW.Sala;

    -- Se MaxAvventori è <=0 allora non è valido
    IF :NEW.MaxAvventori <=0 THEN
        RAISE_APPLICATION_ERROR(-20015,'Il valore per MaxAvventori deve essere maggiore di 0!');
    ELSE
        -- Se MaxAvventori è > di CapienzaAvventori della sala in cui il tavolo è contenuto
        -- allora non è valido
        IF :NEW.MaxAvventori > capienza THEN
            RAISE_APPLICATION_ERROR(-20016,'Il valore di MaxAvventori per il tavolo deve essere
minore o uguale alla CapienzaAvventori della sala che contiene il tavolo in questione!');
        END IF;
    END IF;
END;
/
```

4.5.6 Implementazione del vincolo Capienza legale

```
-- Trigger per il vincolo Capienza legale
CREATE OR REPLACE TRIGGER CAPIENZA_LEGALE
BEFORE INSERT ON TAVOLO
FOR EACH ROW
DECLARE
capienza INTEGER;
capienzacorrente INTEGER;
BEGIN
    -- Calcolo di CapienzaAvventori
    SELECT S.CapienzaAvventori INTO capienza
    FROM SALA S
    WHERE S.CodSala = :NEW.Sala;

    -- Calcolo della CapienzaAvventori corrente per la Sala
    SELECT SUM(T.MAXAVVENTORI) INTO capienzacorrente
    FROM TAVOLO T
    WHERE T.Sala = :NEW.Sala;

    -- Se la somma fra la Capienza corrente e il nuovo MaxAvventori supera
    -- la Capienza massima della Sala allora blocca l'inserimento
    IF :NEW.MAXAVVENTORI+capienzacorrente > capienza THEN
        RAISE_APPLICATION_ERROR(-20017,'Errore. Capienza massima della sala di riferimento superata!');
    END IF;

END;
/
```

4.5.7 Implementazione del vincolo Temperatura avventore legale

```
-- Trigger per il vincolo Temperatura avventore legale
CREATE OR REPLACE TRIGGER TEMPERATURA_AVVENTORE
BEFORE INSERT OR UPDATE ON AVVENTORE
FOR EACH ROW
BEGIN
    -- Se la temperatura dell'avventore supera i 37.5 gradi allora comunica che l'avventore
    -- puo' essere un potenziale caso da registrare nella tabella CASO.
    IF :NEW.Temperatura > 37.5 THEN
        RAISE_APPLICATION_ERROR( -20020, 'Temperatura avventore illegale!
        Potrebbe essere un potenziale CASO da registrare.');
    END IF;
END;
/
```

4.5.8 Implementazione del vincolo Data nascita legale

```
-- Trigger per il vincolo Data nascita legale
CREATE OR REPLACE TRIGGER DATA_NASCITA_LEGALE
BEFORE INSERT OR UPDATE ON PARTECIPAZIONETAVOLATA
FOR EACH ROW
DECLARE
datatavolata TAVOLATA.DataArrivo%TYPE;
datanascita AVVENTORE.DataN%TYPE;
BEGIN
    -- Recupera la data della tavolata a cui partecipa l'avventore
    SELECT T.DataArrivo INTO datatavolata
    FROM TAVOLATA T
    WHERE T.CodTavolata = :NEW.Tavolata;

    -- Recupera la data di nascita dell'avventore che partecipa alla tavolata
    SELECT A.DataN INTO datanascita
    FROM AVVENTORE A
    WHERE A.NumCid = :NEW.Avventore;

    -- Se la data di nascita dell'avventore è successiva alla data della tavolata allora
    -- blocca l'inserimento dell'avventore
    IF datanascita > datatavolata THEN
        RAISE_APPLICATION_ERROR( -20018, 'L avventore non puo partecipare alla tavolata.
        Controllare la sua data di nascita!');
    END IF;
END;
/
```

4.5.9 Implementazione del vincolo Has greenpass

```
-- Trigger per il vincolo Has greenpass
CREATE OR REPLACE TRIGGER HAS_GREENPASS
BEFORE INSERT OR UPDATE ON PARTECIPAZIONETAVOLATA
FOR EACH ROW
DECLARE
tipologiasala SALA.TipoSala%TYPE;
hagreenpass AVVENTORE.HaGreenpass%TYPE;
BEGIN
    -- Recupera la tipologia della sala in cui è ubicato il tavolo della tavolata a cui partecipa l'avventore
    SELECT S.TipoSala INTO tipologiasala
    FROM TAVOLATA T JOIN TAVOLO TA ON T.Tavolo = TA.CodTavolo JOIN SALA S ON S.CodSala = TA.Sala
    WHERE T.CodTavolata = :NEW.Tavolata;

    -- Recupera informazione sul green pass per l'avventore che partecipa alla tavolata
    SELECT A.HaGreenpass INTO hagreenpass
    FROM AVVENTORE A
    WHERE A.NumCid = :NEW.Avventore;

    -- Se l'avventore è sprovvisto di green pass ed è stato associato ad una tavolata il cui tavolo
    -- si trova in una sala interna allora non può partecipare alla tavolata.
    IF hagreenpass='F' AND tipologiasala='Interna' THEN
        RAISE_APPLICATION_ERROR( -20019, 'L avventore è sprovvisto di green pass.
        Puo partecipare unicamente ad una tavolata composta da un tavolo ubicato in una sala esterna!');
    END IF;
END;
/
```

4.5.10 Implementazione del vincolo Somma avventori a tavolata legale

```
-- Trigger per il vincolo Somma avventori a tavolata legale
CREATE OR REPLACE TRIGGER SOMMA_AVVENTORI_A_TAVOLATA_LEGALE
BEFORE INSERT ON PARTECIPAZIONETAVOLATA
FOR EACH ROW
DECLARE
maxavventoritavolo TAVOLO.MaxAvventori%TYPE;
numavventoricorrente INTEGER;
BEGIN
    -- Recupera il massimo numero dei posti del tavolo associato alla tavolata
    SELECT TA.MaxAvventori INTO maxavventoritavolo
    FROM TAVOLATA T JOIN TAVOLO TA ON T.Tavolo = TA.CodTavolo
    WHERE T.CodTavolata = :NEW.Tavolata;

    -- Conta gli avventori correnti che partecipano alla tavolata
    SELECT COUNT(P.Avventore) INTO numavventoricorrente
    FROM PARTECIPAZIONETAVOLATA P
    WHERE P.Tavolata = :NEW.Tavolata;

    -- Se il numero di avventori correnti che partecipano alla tavolata
    -- più il nuovo avventore che si sta per registrare
    -- superano complessivamente il numero dei posti del tavolo
    -- allora non è possibile registrare il nuovo avventore
    IF numavventoricorrente+1 > maxavventoritavolo THEN
        RAISE_APPLICATION_ERROR( -20021, 'Impossibile registrare avventore alla tavolata.
        Il tavolo di riferimento ha tutti i posti occupati!');
    END IF;
END;
/
```

4.5.11 Implementazione del vincolo Data registrazione caso legale

```
-- Trigger per il vincolo Data registrazione caso
CREATE OR REPLACE TRIGGER DATA_REGISTRAZIONE_CASO
BEFORE INSERT OR UPDATE ON CASO
FOR EACH ROW
DECLARE
datanascita_avventore AVVENTORE.DataN%TYPE;
datanascita_cameriere CAMERIERE.DataN%TYPE;
BEGIN
IF :NEW.AvventorePositivo IS NOT NULL THEN
    -- Recupera la data di nascita dell'avventore positivo
    SELECT A.DataN INTO datanascita_avventore
    FROM AVVENTORE A
    WHERE A.NumCid = :NEW.AvventorePositivo;

    -- Se la data di registrazione del caso precede la data di nascita
    -- allora non e' possibile registrare il caso
    IF :NEW.DataRegistrazione < datanascita_avventore THEN
        RAISE_APPLICATION_ERROR( -20022, 'Impossibile registrare caso: data registrazione non valida.' );
    END IF;
END IF;

IF :NEW.CamerierePositivo IS NOT NULL THEN
    -- Recupera la data di nascita del cameriere positivo
    SELECT C.DataN INTO datanascita_cameriere
    FROM CAMERIERE C
    WHERE C.NumCid = :NEW.CamerierePositivo;

    -- Se la data di registrazione del caso precede la data di nascita
    -- allora non e' possibile registrare il caso
    IF :NEW.DataRegistrazione < datanascita_cameriere THEN
        RAISE_APPLICATION_ERROR( -20022, 'Impossibile registrare caso: data registrazione non valida.' );
    END IF;
END IF;
END;
/
```

Capitolo 5

Manuale 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.

5.1 Popolamento del database con dati di esempio

5.1.1 Insert per la tabella PROPRIETARIO

```
-- Insert per la tabella PROPRIETARIO: aggiunge il proprietario dei ristoranti.  
INSERT INTO PROPRIETARIO (CodProprietario, Username, Password, Nome, Cognome, Email)  
VALUES (1,'sandro05','password00','Santolo','Barretta','santolobarretta05@gmail.com');  
COMMIT;
```

5.1.2 Insert per la tabella RISTORANTE

```
-- Insert per la tabella RISTORANTE: aggiunge i ristoranti amministrati da un proprietario.  
INSERT INTO RISTORANTE (CodRistorante, Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Email, SitoWeb, Proprietario)  
VALUES (1,'Bella Napoli','Via Francesco Caracciolo, 1','0813509900','Napoli','NA','80122','bellanapoli@gmail.com','www.ristorantebellanapoli.it',1);  
  
INSERT INTO RISTORANTE (CodRistorante, Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Email, SitoWeb, Proprietario)  
VALUES (2,'Taverna Napoletana','Via Toledo, 30','0813339900','Napoli','NA','80134','bellanapoli@gmail.com','www.ristorantebellanapoli.it',1);  
  
INSERT INTO RISTORANTE (CodRistorante, Denominazione, Indirizzo, Telefono, Citta, Prov, Cap, Proprietario)  
VALUES (3,'Trattoria Milanese','Via Alessandro Manzoni, 5','+390289091122','Milano','MI','20121',1);  
COMMIT;
```

5.1.3 Insert per la tabella MANAGERRISTORANTE

```
-- Insert per la tabella MANAGERRISTORANTE: aggiunge i manager che gestiscono un ristorante, se previsti.  
INSERT INTO MANAGERRISTORANTE (CodManager, Username, Password, Nome, Cognome, Email, Telefono, RistoranteGestito)  
VALUES (1,'angelo02','qwert0123','Angelo','Di Maio','angelodimaio@gmail.com','+393337060999',2);  
  
INSERT INTO MANAGERRISTORANTE (CodManager, Username, Password, Nome, Cognome, Telefono, RistoranteGestito)  
VALUES (2,'mariorossi33','mr123456','Mario','Rossi','+393517799111',3);  
COMMIT;
```

5.1.4 Insert per la tabella SALA

```
-- Insert per la tabella SALA: aggiunge le sale ai relativi ristoranti cui appartengono.  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (1,'Sala Pulcinella',50,100,'Interna',1);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (2,'Sala Vesuvio',20,60,'Interna',1);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (3,'Sala Garden',10,30,'Esterna',1);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (4,'Sala Maradona',50,100,'Interna',2);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (5,'Sala Partenope',18,25,'Interna',2);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (6,'Sala Duomo',30,80,'Interna',3);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (7,'Sala Meneghino',30,80,'Interna',3);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (8,'Sala Biscione',20,60,'Interna',3);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (9,'Sala Exclusive',10,20,'Esterna',3);  
  
INSERT INTO SALA (CodSala, Denominazione, CapienzaAvventori, DimensioneMq, TipoSala, Ristorante)  
VALUES (10,'Sala Vip',10,NULL,'Esterna',3);  
COMMIT;
```

5.1.5 Insert per la tabella CAMERIERE

```
-- Insert per la tabella CAMERIERE: aggiunge i camerieri ai rispettivi ristoranti per cui lavorano.  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, Cittakes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA78432DB','Ciro','Esposito',TO_DATE('08/01/2000','dd/mm/yyyy'),'Maschio','Napoli','NA','Qualiano','NA','3517486042','ciroespo@gmail.com',1);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, Cittakes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA66421DA','Andrea','Russo',TO_DATE('10/10/1990','dd/mm/yyyy'),'Maschio','Villaricca','NA','Quarto','NA','3591676343','andrearusso@outlook.it',1);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('AU0199811','Maria','De Rosa',TO_DATE('22/03/1989','dd/mm/yyyy'),'Femmina','Pozzuoli','NA','Napoli','NA','3722282194','mariaderosa@gmail.com',1);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA64321CE','Rosaria','Romano',TO_DATE('30/08/1993','dd/mm/yyyy'),'Femmina','Napoli','NA','Napoli','NA','3218505681','romano01@gmail.com',1);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('AU3425442','Pasquale','Riccio',TO_DATE('12/04/1988','dd/mm/yyyy'),'Maschio','Caserta','CE','Casoria','NA','3273256337','ricciopas22@outlook.it',1);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA53216BE','Gabriele','Marino',TO_DATE('21/02/1993','dd/mm/yyyy'),'Maschio','Napoli','NA','Pozzuoli','NA','3553493696','marino0@gmail.com',2);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA73121FE','Salvatore','Neri',TO_DATE('12/03/1996','dd/mm/yyyy'),'Maschio','Napoli','NA','Bacoli','NA','3215642156','sasineri11@gmail.com',2);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('AD0567231','Luisa','Esposito',TO_DATE('08/01/2000','dd/mm/yyyy'),'Femmina','Napoli','NA','Quarto','NA','3445768683','espolui0@outlook.com',2);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, Cittakes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA52131PV','Carmen','Granata',TO_DATE('28/05/1995','dd/mm/yyyy'),'Femmina','Marcianise','CE','Pozzuoli','NA','3830046533','carmen233@gmail.com',2);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA87452TB','Laura','Galli',TO_DATE('17/11/1999','dd/mm/yyyy'),'Femmina','Milano','MI','Legnano','MI','3746409289','lauragalli56@gmail.com',3);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA33145OD','Sara','Sala',TO_DATE('13/12/2000','dd/mm/yyyy'),'Femmina','Milano','MI','Sesto San Giovanni','MI','3227317200','sara123@gmail.com',3);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('AU0236754','Fiorella','Bianchi',TO_DATE('25/10/1990','dd/mm/yyyy'),'Femmina','Milano','MI','Rho','MI','3229731501','bianchi0@outlook.com',3);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA10946DE','Raffaele','Fontana',TO_DATE('04/09/1994','dd/mm/yyyy'),'Maschio','Milano','MI','Legnano','MI','3619439924','raffaele444@gmail.com',3);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA332410A','Achille','Rinaldi',TO_DATE('21/05/1987','dd/mm/yyyy'),'Maschio','Milano','MI','Milano','MI','3283664283','achillerinaldi2@gmail.com',3);  
  
INSERT INTO CAMERIERE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Ristorante)  
VALUES ('CA13264UI','Morena','Costa',TO_DATE('19/03/1999','dd/mm/yyyy'),'Femmina','Milano','MI','Assago','MI','3194130484','morena43@outlook.com',3);  
COMMIT;
```

5.1.6 Insert per la tabella TAVOLO

```
-- Insert per la tabella TAVOLO: aggiunge i tavoli alle relative sale dei ristoranti.  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (1, 10, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (2, 5, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (3, 5, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (4, 2, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (5, 2, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (6, 2, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (7, 2, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (8, 2, 1);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (9, 20, 1);  
  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (10, 5, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (11, 5, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (12, 2, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (13, 2, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (14, 2, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (15, 2, 2);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (16, 2, 2);  
  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (17, 2, 3);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (18, 2, 3);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (19, 2, 3);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (20, 2, 3);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (21, 2, 3);  
  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (22, 10, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (23, 5, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (24, 5, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (25, 2, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (26, 2, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (27, 2, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (28, 2, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (29, 2, 4);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (30, 20, 4);  
  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (31, 2, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (32, 2, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (33, 4, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (34, 2, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (35, 2, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (36, 2, 5);  
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (37, 4, 5);
```

```
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (38, 10, 6);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (39, 5, 6);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (40, 5, 6);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (41, 10, 6);

INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (42, 10, 7);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (43, 5, 7);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (44, 5, 7);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (45, 10, 7);

INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (46, 5, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (47, 5, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (48, 2, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (49, 2, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (50, 2, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (51, 2, 8);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (52, 2, 8);

INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (53, 2, 9);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (54, 2, 9);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (55, 2, 9);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (56, 2, 9);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (57, 2, 9);

INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (58, 2, 10);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (59, 2, 10);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (60, 2, 10);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (61, 2, 10);
INSERT INTO TAVOLO (CodTavolo, MaxAvventori, Sala) VALUES (62, 2, 10);
COMMIT;
```

5.1.7 Insert per la tabella ADIACENZATAVOLO

```
-- Insert per la tabella ADIACENZATAVOLO: aggiunge la lista di tavoli adiacenti per ogni tavolo, se esiste adiacenza
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (1, 2);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (1, 4);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (2, 1);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (2, 5);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (2, 3);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (3, 2);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (3, 6);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (4, 1);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (4, 5);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (4, 7);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (5, 4);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (5, 2);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (5, 6);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (5, 8);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (6, 3);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (6, 5);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (6, 9);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (7, 4);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (7, 8);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (8, 7);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (8, 5);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (8, 9);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (9, 8);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (9, 6);

INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (10, 13);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (10, 11);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (11, 14);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (11, 12);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (12, 11);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (12, 15);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (13, 10);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (13, 14);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (14, 11);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (14, 15);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (15, 12);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (15, 16);
INSERT INTO ADIACENZATAVOLO (Tavolo, TavoloAdiacente) VALUES (16, 15);
```


5.1.8 Insert per la tabella TAVOLATA

```
-- Insert per la tabella TAVOLATA: inserisce le tavolate create per i rispettivi ristoranti
INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (1, TO_DATE('17/11/2021','dd/mm/yyyy'), 4, 'CA78432DB');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (2, TO_DATE('17/11/2021','dd/mm/yyyy'), 5, 'CA66421DA');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (3, TO_DATE('18/11/2021','dd/mm/yyyy'), 12, 'AU0199811');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (4, TO_DATE('18/11/2021','dd/mm/yyyy'), 13, 'CA64321CE');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (5, TO_DATE('19/11/2021','dd/mm/yyyy'), 18, 'AU3425442');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (6, TO_DATE('20/11/2021','dd/mm/yyyy'), 8, 'CA78432DB');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (7, TO_DATE('20/11/2021','dd/mm/yyyy'), 25, 'CA53216BE');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (8, TO_DATE('22/11/2021','dd/mm/yyyy'), 26, 'CA73121FE');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (9, TO_DATE('22/11/2021','dd/mm/yyyy'), 27, 'AU0567231');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (10, TO_DATE('25/11/2021','dd/mm/yyyy'), 37, 'CA52131PV');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (11, TO_DATE('19/12/2021','dd/mm/yyyy'), 26, 'CA53216BE');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (12, TO_DATE('20/12/2021','dd/mm/yyyy'), 39, 'CA87452TB');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (13, TO_DATE('20/12/2021','dd/mm/yyyy'), 49, 'CA33145OD');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (14, TO_DATE('24/12/2021','dd/mm/yyyy'), 50, 'AU0236754');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (15, TO_DATE('25/12/2021','dd/mm/yyyy'), 53, 'CA10946DE');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (16, TO_DATE('25/01/2022','dd/mm/yyyy'), 55, 'CA33241OA');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (17, TO_DATE('25/01/2022','dd/mm/yyyy'), 62, 'CA13264UI');

INSERT INTO TAVOLATA (CodTavolata, DataArrivo, Tavolo, Cameriere) VALUES (18, TO_DATE('25/01/2022','dd/mm/yyyy'), 50, 'CA13264UI');
COMMIT;
```

5.1.9 Insert per la tabella AVVENTORE

```
-- Insert per la tabella AVVENTORE: inserisce una lista di avventori generici con le loro generalita
INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('IC3159530','Baldassarre','Pinto',TO_DATE('04/06/1953','dd/mm/yyyy'),'Maschio','N',N,'+393493302038','baldassareepinto@gmail.com',35.8,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('FW824173','Filiberto','Milano',TO_DATE('19/11/1940','dd/mm/yyyy'),'Maschio','N',N,'+393598741558','filibertomilano@outlook.com',36.8,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('WF1996739','Felicità','Piazza',TO_DATE('16/06/1973','dd/mm/yyyy'),'Femmina','V',N,'+393658942018','felicitapiazza3@libero.it',36.9,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('SU2810531','Lea','Barrese',TO_DATE('26/07/1985','dd/mm/yyyy'),'Femmina','C',S,'+393259847862','leaBarrese26@gmail.com',36.5,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('BD0518049','Nazzarena','Onio',TO_DATE('07/12/2001','dd/mm/yyyy'),'Maschio','V','Marcanise','C','+393517788436','nazaon2001@outlook.it',37.0,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('K25802431','Edardo','Ricci',TO_DATE('29/01/2002','dd/mm/yyyy'),'Maschio','P','Benevento','BN','+393254159687','edoRich@libero.it',36.4,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('W10726870','Prospero','Padovesi',TO_DATE('24/04/1963','dd/mm/yyyy'),'Maschio','D','PV','Caserta','C','+393659852014','prfa2404@gmail.com',36.4,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('CR1881570','Alice','Cocci',TO_DATE('24/05/1951','dd/mm/yyyy'),'Femmina','Calvignasco','MI','Roma','RM','+393654002598','AliceCC@outlook.com',36.2,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('RT1739919','Quinzio','Pugliesi',TO_DATE('18/05/1985','dd/mm/yyyy'),'Maschio','Ascensione','RA','Recala','C','+393398947052','QuinzioPugliesi0589@gmail.com',37.0,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('YS3432289','Ida','Pagnotto',TO_DATE('01/03/1966','dd/mm/yyyy'),'Femmina','Martano','LE','Martano','C','+393698521655','IdaPagnotto@gmail.com',36.8,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('GU1160846','Geronima','Endrizzi',TO_DATE('09/11/2000','dd/mm/yyyy'),'Non specificato','Nago','TN','Vico Equense','NA','+393854759624','GeronimaEnd@gmail.com',35.9,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('AD8695321','Carolina','Pugliese',TO_DATE('07/04/1953','dd/mm/yyyy'),'Femmina','Casavatore','N','Casoria','N','+393275273639','CarolinaPugliese@gmail.com',35.4,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('BG2548759','Marcella','Pezzali',TO_DATE('28/12/1977','dd/mm/yyyy'),'Femmina','Arzano','N','Arzano','N','+393732068075','MarcellaPezzali@gmail.com',36.1,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('GH5632104','Dolores','Zanzi',TO_DATE('18/01/1955','dd/mm/yyyy'),'Femmina','Volta','N','Cetara','SA','+393380722982','DoloresDePanza@gmail.com',36.4,'v');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('FR5849625','Fabiola','Mengolo',TO_DATE('19/09/1956','dd/mm/yyyy'),'Femmina','Giffoni Sei Casali','SA','San Giorgio a Cremano','N','+393416755810','FabiaMengolo@gmail.com',36.5,'v');
```

```

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('TR5625487','Daniele','Bernardi',TO_DATE('03/12/1959','dd/mm/yyyy'), 'Maschio','Cercola','NA','Portici','NA','+393562319517','DaanBernardi@gmail.com',37.1,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('BR2646058','Carolina','Lombardi',TO_DATE('07/11/1961','dd/mm/yyyy'), 'Femmina','Mellito di Napoli','NA','+393516530360','CaroLombardi@gmail.com',37.0,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('LE2615481','Rossana','Filzi',TO_DATE('23/09/1965','dd/mm/yyyy'), 'Femmina','Afragola','NA','Montecorvino Pugliano','SA','+393316825631','RossanaFilzi@gmail.com',36.8,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('ML6215403','Pierluigi','Trapanese',TO_DATE('30/09/1965','dd/mm/yyyy'), 'Maschio','Casandrino','NA','Fisciano','SA','+393303659368','LuigiTrapanese@gmail.com',36.8,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('RS4697315','silvia','Golgi',TO_DATE('22/04/1968','dd/mm/yyyy'), 'Femmina','Casalnuovo di Napoli','NA','Salernitano','SA','+393869408344','SilviGolgi@gmail.com',36.6,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('GC9674582','Nicola','Ricci',TO_DATE('21/08/1969','dd/mm/yyyy'), 'Maschio','Bellizzi','SA','San Sebastiano al Vesuvio','NA','+393362582808','NicoRic@gmail.com',36.5,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('AE6164595','Antonio','Bettoni',TO_DATE('20/08/1971','dd/mm/yyyy'), 'Maschio','Mugnano di Napoli','NA','Mercato San Severino','SA','+393410197804','AntoBet@gmail.com',36.3,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('PO3031697','Amleto','Virgilic',TO_DATE('05/01/1975','dd/mm/yyyy'), 'Maschio','Marano di Napoli','NA','Maiori','SA','+393468869310','AmletoVirgilic@gmail.com',36.2,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('MR9748683','Mauro','Priuli',TO_DATE('15/04/1988','dd/mm/yyyy'), 'Maschio','Marano di Napoli','NA','Marano di Napoli','NA','+393394922230','MauroPriuli@gmail.com',36.2,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('DR0301264','Rosaria','Argurio',TO_DATE('25/05/1976','dd/mm/yyyy'), 'Femmina','Dugenta','BN','Durazzano','BN','+393250404211','RosariArgurio@gmail.com',36.5,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('L29575684','Mauro','Zola',TO_DATE('17/10/1979','dd/mm/yyyy'), 'Maschio','Bellona','CE','Arieno','CE','+393411767562','MauroZola@gmail.com',36.5,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('IC2558319','Donaatello','Pellegrini',TO_DATE('19/03/1980','dd/mm/yyyy'), 'Maschio','Vitulazio','CE','Vitulazio','CE','+393397508638','DonaPellegrini@gmail.com',37.0,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('TE0864370','Giancarlo','Boitani',TO_DATE('05/04/1982','dd/mm/yyyy'), 'Maschio','Cervino','CE','San Felice a Cancello','CE','+393416061365','GianBoitani@gmail.com',37.0,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('Z06283162','Melania','Pecorino',TO_DATE('23/04/1984','dd/mm/yyyy'), 'Femmina','Santa Maria a Vico','CE','Santa Maria la Fossa','CE','+39353370100','MelaniaPecorino@gmail.com',36.7,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('IR2648039','Renata','Pizzetti',TO_DATE('11/11/1985','dd/mm/yyyy'), 'Femmina','Piana di Monte Verna','CE','Piana di Monte Verna','CE','+393516025201','RenataPizz@gmail.com',36.7,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('FU2645786','Giuseppina','Giannuzzi',TO_DATE('20/07/1990','dd/mm/yyyy'), 'Femmina','Grazzanise','CE','Grazzanise','CE','+393520418674','GeppyGianni@gmail.com',36.3,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('UR6437280','Angelica','Barbarigo',TO_DATE('02/07/1991','dd/mm/yyyy'), 'Femmina','San Tammaro','CE','Santa Maria Capua Vetere','CE','+393510895615','LikaBarbagio@gmail.com',36.3,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('OG6407351','Melissa','Tasso',TO_DATE('16/08/1992','dd/mm/yyyy'), 'Femmina','Santa Maria Capua Vetere','CE','Caiazzo','CE','+393309496253','MeliTasso@gmail.com',36.4,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('OG6427318','Gianluca','Murri',TO_DATE('19/05/1995','dd/mm/yyyy'), 'Maschio','Marcianise','CE','Marcianise','CE','+393528831547','GianMurri@gmail.com',36.5,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('LR9137648','Jolanda','Adinolfi',TO_DATE('06/05/1996','dd/mm/yyyy'), 'Femmina','Portico di Caserta','CE','Macerata Campania','CE','+39351947992','JoleAdi@gmail.com',37.0,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('HE5467974','Fernando','Cilibresi',TO_DATE('19/07/1997','dd/mm/yyyy'), 'Maschio','Pozzuoli','NA','Pozzuoli','NA','+393342696918','FernandoCilibresi@gmail.com',36.5,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('TR9746286','Roberto','Camanni',TO_DATE('24/11/1998','dd/mm/yyyy'), 'Maschio','Quarto','NA','Quarto','NA','+393438255310','RobiCamanni@gmail.com',36.4,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('IR3146975','Carmelo','Mastriani',TO_DATE('12/02/1999','dd/mm/yyyy'), 'Maschio','Cardito','NA','Curti','CE','+393309290526','CarmeloMastriani@gmail.com',36.5,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('HI9746580','Ezio','Ammariti',TO_DATE('11/12/2001','dd/mm/yyyy'), 'Maschio','Frattamaggiore','NN','Frattamaggiore','NN','+393353722101','EzioAmm@gmail.com',36.8,'F');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('LM9137684','Aria','Moschino',TO_DATE('02/05/2002','dd/mm/yyyy'), 'Femmina','Capodistri','CE','Massa di Somma','NA','+393301539510','AriMoschino@gmail.com',36.7,'V');

INSERT INTO AVVENTORE (NumCid, Nome, Cognome, DataN, Sesso, CittaN, ProvN, CittaRes, ProvRes, Telefono, Email, Temperatura, HaGreenpass)
VALUES ('GE9197735','Vittorio','Pulci',TO_DATE('10/12/2002','dd/mm/yyyy'), 'Maschio','Grumo Nevano','NA','Casapulla','CE','+393273531946','VikPulci@gmail.com',36.5,'V');

COMMIT;

```

5.1.10 Insert per la tabella ACCOGLIENZA

```
-- Insert per la tabella ACCOGLIENZA: collega ogni avventore al ristorante che lo accoglie
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'IC3159530');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'PW824173');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'WF1996739');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'SU2810531');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'BD8518049');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'KT5802431');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'WI0726870');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'CY1881570');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'RT1739919');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'YS3432289');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'GU1160846');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (1,'AD8695321');

INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'BG2548759');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'GH5632104');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'FR5849625');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'TR5625487');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'BR2648058');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'LE2615481');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'ML6215403');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'RS4697315');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'GC9674582');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'AE6164595');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'PO3031697');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (2,'MR9748683');

INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'DR0301264');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'LZ9575684');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'IC2558319');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'TE0864370');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'ZO6283162');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'IH2648039');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'FU2645786');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'UR6437280');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'OG6407351');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'OG6427318');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'LR9137648');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'HE5467974');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'TR9746286');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'IT3146975');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'HY9746580');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'LH9137684');
INSERT INTO ACCOGLIENZA (Ristorante, Avventore) VALUES (3,'GE9197735');

COMMIT;
```

5.1.11 Insert per la tabella PARTECIPAZIONETAVOLATA

```
-- Insert per la tabella PARTECIPAZIONETAVOLATA: collega ogni avventore alla tavolata a cui partecipa
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('IC3159530',1);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('PW824173',1);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('WF1996739',2);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('SU2810531',2);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('BD8518049',3);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('KT5802431',3);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('WI0726870',4);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('CY1881570',4);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('RT1739919',5);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('YS3432289',5);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('GU1160846',6);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('AD8695321',6);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('BG2548759',7);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('GH5632104',7);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('FR5849625',8);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('TR5625487',8);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('BR2648058',9);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('LE2615481',9);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('ML6215403',10);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('RS4697315',10);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('GC9674582',10);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('AE6164595',10);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('PO3031697',11);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('MR9748683',11);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('DR0301264',12);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('LZ9575684',12);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('IC2558319',12);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('TE0864370',12);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('Z06283162',12);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('IH2648039',13);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('FU2645786',13);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('UR6437280',14);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('OG6407351',14);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('OG6427318',15);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('LR9137648',15);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('HE5467974',16);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('TR9746286',16);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('IT3146975',17);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('HY9746580',17);

INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('LH9137684',18);
INSERT INTO PARTECIPAZIONETAVOLATA (Avventore, Tavolata) VALUES ('GE9197735',18);
COMMIT;
```

5.1.12 Insert per la tabella CASO

```
-- Insert per la tabella CASO: inserisce i casi covid registrati nei ristoranti
INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (1, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'IC3159530', NULL, 1, NULL);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (2, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'SU2810531', NULL, 1, NULL);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (3, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'RT1739919', NULL, 1, NULL);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (4, TO_DATE('25/01/2022','dd/mm/yyyy'), 'InRisoluzione', NULL, 'LE2615481', NULL, NULL, 1);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (5, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'L29575684', NULL, NULL, 2);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (6, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'OG6427318', NULL, NULL, 2);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (7, TO_DATE('25/01/2022','dd/mm/yyyy'), 'Risolto', NULL, 'GE9197735', NULL, NULL, 2);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (8, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'CA78432DB', 1, NULL);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (9, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, 'CA33241OA', NULL, 2);

INSERT INTO CASO (CodCaso, DataRegistrazione, StatoCaso, Note, AvventorePositivo, CamerierePositivo, RegistraProprietario, RegistraManager)
VALUES (10, TO_DATE('25/01/2022','dd/mm/yyyy'), 'NonRisolto', NULL, NULL, 'CA66421DA', 1, NULL);

COMMIT;
```

5.2 Esempio di query

Seguono le definizioni di esempi di query in SQL tratte dal file “DBQuery.sql”, mostrando successivamente l’output ottenuto dall’esecuzione sul database implementato. Per la descrizione del funzionamento delle query si rimanda al dizionario delle interrogazioni.

5.2.1 Numero giornaliero di avventori per ristorante

```
-- 1.Numero giornaliero di avventori per ristorante
-- Supponiamo che il ristorante dato abbia CodRistorante = 1 e la DataArrivo = '17/11/2021'.
SELECT R.Denominazione AS RISTORANTE, T.DataArrivo AS DATA, COUNT(ACC.Avventore) AS TOT_GIORNALIERO_AVVENTORI
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
                     JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = A.NumCid
                     JOIN TAVOLATA T ON T.CodTavolata = PT.Tavolata
WHERE R.CodRistorante = 1 AND T.DataArrivo = TO_DATE('17/11/2021', 'dd/mm/yyyy')
GROUP BY R.Denominazione, T.DataArrivo;
```

Output:

RISTORANTE	DATA	TOT_GIORNALIERO_AVVENTORI
Bella Napoli	17-NOV-21	4

5.2.2 Numero mensile di avventori per ristorante

```
-- 2.Numero mensile di avventori per ristorante
-- Supponiamo che il ristorante dato abbia CodRistorante = 1 e il mese di riferimento sia 11/2021
SELECT R.Denominazione AS RISTORANTE, TO_CHAR(T.DataArrivo, 'mm ') AS MESE, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(ACC.Avventore) AS TOT_MENSILE_AVVENTORI
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
                     JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = A.NumCid
                     JOIN TAVOLATA T ON T.CodTavolata = PT.Tavolata
WHERE R.CodRistorante = 1 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021 AND TO_CHAR(T.DataArrivo, 'mm') = 11
GROUP BY R.Denominazione, TO_CHAR(T.DataArrivo, 'mm '), TO_CHAR(T.DataArrivo, 'yyyy');
```

Output:

RISTORANTE	MESE	ANNO	TOT_MENSILE_AVVENTORI
Bella Napoli	11	2021	12

5.2.3 Numero giornaliero di avventori per tutti i ristoranti di un proprietario

```
-- 3.Numero giornaliero di avventori per tutti i ristoranti di un proprietario
-- Supponiamo che il proprietario abbia CodProprietario = 1 e la DataArrivo = '17/11/2021'
SELECT R.Proprietario, T.DataArrivo AS DATA, COUNT(ACC.Avventore) AS TOT_GIORNALIERO_AVVENTORI_RISTORANTI
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = A.NumCid
JOIN TAVOLATA T ON T.CodTavolata = PT.Tavolata
WHERE R.Proprietario = 1 AND T.DataArrivo = TO_DATE('17/11/2021','dd/mm/yyyy')
GROUP BY R.Proprietario, T.DataArrivo;
```

Output:

PROPRIETARIO DATA	TOT_GIORNALIERO_AVVENTORI_RISTORANTI
1 17-NOV-21	4

5.2.4 Numero mensile di avventori per tutti i ristoranti di un proprietario

```
-- 4.Numero mensile di avventori per tutti i ristoranti di un proprietario
-- Supponiamo che il proprietario abbia CodProprietario = 1 e il mese di riferimento sia 11/2021
SELECT R.Proprietario, TO_CHAR(T.DataArrivo, 'mm') AS MESE, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(ACC.Avventore) AS TOT_MENSILE_AVVENTORI_RISTORANTI
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = A.NumCid
JOIN TAVOLATA T ON T.CodTavolata = PT.Tavolata
WHERE R.Proprietario = 1 AND TO_CHAR(T.DataArrivo,'mm') = 11 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021
GROUP BY R.Proprietario, TO_CHAR(T.DataArrivo,'mm'), TO_CHAR(T.DataArrivo,'yyyy');
```

Output:

PROPRIETARIO MESE ANNO	TOT_MENSILE_AVVENTORI_RISTORANTI
1 11 2021	22

5.2.5 Casi positivi di un determinato ristorante per data di arrivo della tavolata

```
-- 5.Casi positivi di un determinato ristorante per data di arrivo della tavolata
-- Supponiamo che il codice del ristorante sia CodRistorante = 1 e che la data di arrivo della tavolata sia il: '17/11/2021'
SELECT R.Denominazione AS RISTORANTE, T.DataArrivo AS DATA_ARV,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
JOIN CASO C ON C.AvventorePositivo = A.Avventore
JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.CodRistorante = 1 AND T.DataArrivo = TO_DATE('17/11/2021','dd/mm/yyyy')
GROUP BY R.Denominazione, T.DataArrivo;
```

Output:

RISTORANTE	DATA_ARV	TOT_RISULTATI_POSITIVI
Bella Napoli	17-NOV-21	2

5.2.6 Casi positivi di un determinato ristorante per mese di arrivo della tavolata

```

-- 6.Casi positivi di un determinato ristorante per mese di arrivo della tavolata
-- Supponiamo che il codice del ristorante sia CodRistorante = 1 e che il mese di arrivo della tavolata sia 11/2021
SELECT R.Denominazione AS RISTORANTE, TO_CHAR(T.DataArrivo, 'mm ') AS MESE, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
                     JOIN CASO C ON C.AvventorePositivo = A.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.CodRistorante = 1 AND TO_CHAR(T.DataArrivo, 'mm') = 11 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021
GROUP BY R.Denominazione, TO_CHAR(T.DataArrivo, 'mm '), TO_CHAR(T.DataArrivo, 'yyyy');

```

Output:

RISTORANTE	MESE	ANNO	TOT_RISULTATI_POSITIVI
Bella Napoli	11	2021	3

5.2.7 Casi positivi di un determinato ristorante per anno di arrivo della tavolata

```

-- 7.Casi positivi di un determinato ristorante per anno di arrivo della tavolata
-- Supponiamo che il codice del ristorante sia CodRistorante = 1 e che l'anno di arrivo della tavolata sia il 2021
SELECT R.Denominazione AS RISTORANTE, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
                     JOIN CASO C ON C.AvventorePositivo = A.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.CodRistorante = 1 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021
GROUP BY R.Denominazione, TO_CHAR(T.DataArrivo, 'yyyy');

```

Output:

RISTORANTE	ANNO	TOT_RISULTATI_POSITIVI
Bella Napoli	2021	3

5.2.8 Casi positivi di tutti i ristoranti di un proprietario per data di arrivo della tavolata

```

-- 8.Casi positivi di tutti i ristoranti di un proprietario per data di arrivo della tavolata
-- Supponiamo che il codice del proprietario sia CodProprietario = 1 e che la data di arrivo della tavolata sia il: '17/11/2021'
SELECT R.Proprietario, T.DataArrivo AS DATA_ARV,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI_RISTORANTI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
                     JOIN CASO C ON C.AvventorePositivo = A.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.Proprietario = 1 AND T.DataArrivo = TO_DATE('17/11/2021', 'dd/mm/yyyy')
GROUP BY R.Proprietario, T.DataArrivo;

```

Output:

PROPRIETARIO	DATA_ARV	TOT_RISULTATI_POSITIVI_RISTORANTI
1	17-NOV-21	2

5.2.9 Casi positivi di tutti i ristoranti di un proprietario per mese di arrivo della tavolata

```
-- 9.Casi positivi di tutti i ristoranti di un proprietario per mese di arrivo della tavolata
-- Supponiamo che il codice del proprietario sia CodProprietario = 1 e che il mese di arrivo della tavolata sia 11/2021
SELECT R.Proprietario, TO_CHAR(T.DataArrivo, 'mm ') AS MESE, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI_RISTORANTI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
                     JOIN CASO C ON C.AvventorePositivo = A.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.Proprietario = 1 AND TO_CHAR(T.DataArrivo, 'mm') = 11 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021
GROUP BY R.Proprietario, TO_CHAR(T.DataArrivo, 'mm '), TO_CHAR(T.DataArrivo, 'yyyy');
```

Output:

PROPRIETARIO	MESE	ANNO	TOT_RISULTATI_POSITIVI_RISTORANTI
1	11	2021	4

5.2.10 Casi positivi di tutti i ristoranti di un proprietario per anno di arrivo della tavolata

```
-- 10.Casi positivi di tutti i ristoranti di un proprietario per anno di arrivo della tavolata
-- Supponiamo che il codice del proprietario sia CodProprietario = 1 e che l'anno di arrivo della tavolata sia il 2021
SELECT R.Proprietario, TO_CHAR(T.DataArrivo, 'yyyy') AS ANNO,
       COUNT(C.AvventorePositivo) TOT_RISULTATI_POSITIVI_RISTORANTI
FROM RISTORANTE R JOIN ACCOGLIENZA A ON R.CodRistorante = A.Ristorante
                     JOIN CASO C ON C.AvventorePositivo = A.Avventore
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
WHERE R.Proprietario = 1 AND TO_CHAR(T.DataArrivo, 'yyyy') = 2021
GROUP BY R.Proprietario, TO_CHAR(T.DataArrivo, 'yyyy');
```

Output:

PROPRIETARIO	ANNO	TOT_RISULTATI_POSITIVI_RISTORANTI
1	2021	6

5.2.11 Informazioni sugli avventori risultati positivi in un ristorante

```
-- 11.Informazioni sugli avventori risultati positivi in un ristorante
-- Supponiamo che il codice del ristorante che ha accolto gli avventori risultati positivi sia CodRistorante = 1
SELECT C.CodCaso, C.DataRegistrazione AS DATAR, CAST(R.Denominazione AS VARCHAR2(20)) AS RISTORANTE,
       A.NumCid, CAST(A.Nome AS VARCHAR2(30)) AS Nome, CAST(A.Cognome AS VARCHAR2(30)) AS Cognome, A.DataN,
       A.Telefono, A.HaGreenpass AS G, T.CodTavolata, T.DataArrivo, T.Cameriere, T.Tavolo, TAV.Sala
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
                     JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
                     JOIN CASO C ON C.AvventorePositivo = A.NumCid
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLO TAV ON TAV.CodTavolo = T.CodTavolata
                     JOIN TAVOLO TAV ON TAV.CodTavolo = T.Tavolo
WHERE R.CodRistorante = 1
ORDER BY C.CodCaso;
```

Output:

CODCASO	DATAR	RISTORANTE	NUMCID	NOME	COGNOME	DATAN	TELEFONO	G CODTAVOLATA	DATAARRIV	CAMERIERE	TAVOLO	TAVOLOADIACENTE	
1	25-GEN-22	Bella Napoli	IC3159530	Baldassarre	Pinto	04-GIU-53	+393493302038	V	1	17-NOV-21	CA78432DB	4	3
2	25-GEN-22	Bella Napoli	SU2810531	Lea	Barrese	26-LUG-85	+393259847862	V	2	17-NOV-21	CA66421DA	5	4
3	25-GEN-22	Bella Napoli	RT1739919	Quinzio	Fugliesi	18-MAG-99	+393398847852	F	5	19-NOV-21	AU3425442	18	17

5.2.12 Informazioni sugli avventori risultati positivi in tutti i ristoranti di un proprietario

```

-- 12.Informazioni sugli avventori risultati positivi in tutti i ristoranti di un proprietario
-- Supponiamo che il codice del proprietario dei ristoranti sia CodProprietario = 1
SELECT C.CodCaso, C.DataRegistrazione AS DATAR, CAST(R.Denominazione AS VARCHAR2(20)) AS RISTORANTE,
       A.NumCid, CAST(A.Nome AS VARCHAR2(30)) AS Nome, CAST(A.Cognome AS VARCHAR2(30)) AS Cognome, A.DataN,
       A.Telefono, A.HaGreenpass AS G, T.CodTavolata, T.DataArrivo, T.Cameriere, T.Tavolo, TAV.Sala
FROM RISTORANTE R JOIN ACCOGLIENZA ACC ON R.CodRistorante = ACC.Ristorante
                     JOIN AVVENTORE A ON A.NumCid = ACC.Avventore
                     JOIN CASO C ON C.AvventorePositivo = A.NumCid
                     JOIN PARTECIPAZIONETAVOLATA PT ON PT.Avventore = C.AvventorePositivo
                     JOIN TAVOLATA T ON PT.Tavolata = T.CodTavolata
                     JOIN TAVOLO TAV ON TAV.CodTavolo = T.Tavolo
WHERE R.Proprietario = 1
ORDER BY C.CodCaso;

```

Output:

CODCASO	DATAR	RISTORANTE	NUMCID	NOME	COGNOME	DATAN	TELEFONO	G	CODTAVOLATA	DATAARRIV	CAMERIERE	TAVOLO	SALA
1	25-GEN-22	Bella Napoli	IC3159530	Baldassarre	Pinto	04-GIU-53	+393493302038	V	1	17-NOV-21	CA78432DB	4	1
2	25-GEN-22	Bella Napoli	SU2810531	Lea	Barese	26-LUG-85	+393259847862	V	2	17-NOV-21	CA66421DA	5	1
3	25-GEN-22	Bella Napoli	RT1739919	Quinzio	Pugliesi	18-MAG-89	+393398947852	F	5	19-NOV-21	AU3425442	18	3
4	25-GEN-22	Taverna Napoletana	LE2615481	Rossana	Filzi	23-SET-65	+393316825631	V	9	22-NOV-21	AU0567231	27	4
5	25-GEN-22	Trattoria Milanese	LZ9575684	Mauro	Zola	17-OTT-79	+393411767562	V	12	20-DIC-21	CA87452TB	39	6
6	25-GEN-22	Trattoria Milanese	OG6427318	Gianluca	Murri	19-MAG-95	+393528831547	F	15	25-DIC-21	CA10946DE	53	9
7	25-GEN-22	Trattoria Milanese	GE5197735	Vittorio	Pulci	10-DIC-02	+393273531946	V	18	25-GEN-22	CA13264UI	50	8

5.2.13 Informazioni sui camerieri risultati positivi in un ristorante

```

-- 13.Informazioni sui camerieri risultati positivi in un ristorante
-- Supponiamo che il codice del ristorante per cui lavorano i camerieri risultati positivi sia CodRistorante = 1
SELECT CA.CodCaso, CA.DataRegistrazione AS DATAR,
       CAST(R.Denominazione AS VARCHAR2(20)) AS RISTORANTE,
       C.NumCid, CAST(C.Nome AS VARCHAR2(30)) AS Nome,
       CAST(C.Cognome AS VARCHAR2(30)) AS Cognome,
       C.DataN, C.Telefono
FROM RISTORANTE R JOIN CAMERIERE C ON R.CodRistorante = C.Ristorante
                     JOIN CASO CA ON CA.CamerierePositivo = C.NumCid
WHERE C.Ristorante = 1
ORDER BY CA.CodCaso;

```

Output:

CODCASO	DATAR	RISTORANTE	NUMCID	NOME	COGNOME	DATAN	TELEFONO
8	25-GEN-22	Bella Napoli	CA78432DB	Ciro	Esposito	08-GEN-00	3517486042
10	25-GEN-22	Bella Napoli	CA66421DA	Andrea	Russo	10-OTT-90	3591676343

5.2.14 Informazioni sui camerieri risultati positivi in tutti i ristoranti di un proprietario

```

-- 14.Informazioni sui camerieri risultati positivi in tutti i ristoranti di un proprietario
-- Supponiamo che il codice del proprietario dei ristoranti sia CodProprietario = 1
SELECT CA.CodCaso, CA.DataRegistrazione AS DATAR,
       CAST(R.Denominazione AS VARCHAR2(20)) AS RISTORANTE,
       C.NumCid, CAST(C.Nome AS VARCHAR2(30)) AS Nome,
       CAST(C.Cognome AS VARCHAR2(30)) AS Cognome,
       C.DataN, C.Telefono
FROM RISTORANTE R JOIN CAMERIERE C ON R.CodRistorante = C.Ristorante
                     JOIN CASO CA ON CA.CamerierePositivo = C.NumCid
WHERE R.Proprietario = 1
ORDER BY CA.CodCaso;

```

Output:

CODCASO	DATAR	RISTORANTE	NUMCID	NOME	COGNOME	DATAN	TELEFONO
8	25-GEN-22	Bella Napoli	CA78432DB	Ciro	Esposito	08-GEN-00	3517486042
9	25-GEN-22	Trattoria Milanese	CA33241OA	Achille	Rinaldi	21-MAG-87	3283664283
10	25-GEN-22	Bella Napoli	CA66421DA	Andrea	Russo	10-OTT-90	3591676343

5.2.15 Avventori positivi con o senza green pass

```

-- 15.Avventori positivi con o senza green pass
-- Per ristorante: supponiamo che CodRistorante = 1
SELECT COUNT(C.CodCaso) AS AVVENTORI_POSITIVI_CON_GREENPASS
FROM CASO C JOIN AVVENTORE A ON C.AvventorePositivo = A.NumCid
               JOIN ACCOGLIENZA ACC ON ACC.Avvventore = A.NumCid
WHERE A.HaGreenpass='V' AND ACC.Ristorante = 1;

SELECT COUNT(C.CodCaso) AS AVVENTORI_POSITIVI_SENZA_GREENPASS
FROM CASO C JOIN AVVENTORE A ON C.AvventorePositivo = A.NumCid
               JOIN ACCOGLIENZA ACC ON ACC.Avvventore = A.NumCid
WHERE A.HaGreenpass='F' AND ACC.Ristorante = 1;

-- Per proprietario: supponiamo che CodProprietario = 1
SELECT COUNT(C.CodCaso) AS AVVENTORI_POSITIVI_CON_GREENPASS
FROM CASO C JOIN AVVENTORE A ON C.AvventorePositivo = A.NumCid
               JOIN ACCOGLIENZA ACC ON ACC.Avvventore = A.NumCid
               JOIN RISTORANTE R ON R.CodRistorante = ACC.Ristorante
WHERE A.HaGreenpass='V' AND R.Proprietario = 1;

SELECT COUNT(C.CodCaso) AS AVVENTORI_POSITIVI_SENZA_GREENPASS
FROM CASO C JOIN AVVENTORE A ON C.AvventorePositivo = A.NumCid
               JOIN ACCOGLIENZA ACC ON ACC.Avvventore = A.NumCid
               JOIN RISTORANTE R ON R.CodRistorante = ACC.Ristorante
WHERE A.HaGreenpass='F' AND R.Proprietario = 1;

```

Output:

AVVENTORI_POSITIVI_CON_GREENPASS	AVVENTORI_POSITIVI_SENZA_GREENPASS
2	5

AVVENTORI_POSITIVI_SENZA_GREENPASS	AVVENTORI_POSITIVI_SENZA_GREENPASS
1	2

5.2.16 Numero di avventori medio per tavolata di un ristorante

```
-- 16.Numero di avventori medio per tavolata di un ristorante  
-- Supponendo di voler conoscere la media di avventori per tavolata del ristorante di CodRistorante = 2  
-- Utilizziamo la vista RIEPILOGO_TAVOLATE_RISTORANTI_PROPRIETARIO  
SELECT T.CodRistorante, T.Ristorante, AVG(T.PartecipantiTavolata) AS MEDIA_AVVENTORI_PER_TAVOLATA  
FROM RIEPILOGO_TAVOLATE_RISTORANTI_PROPRIETARIO T  
WHERE T.CodRistorante = 2  
GROUP BY T.CodRistorante, T.Ristorante;
```

Output:

RISTORANTE	MEDIA_AVVENTORI_PER_TAVOLATA
2	2,4



DIPARTIMENTO DI INGEGNERIA ELETTRICA
E DELLE TECNOLOGIE DELL'INFORMAZIONE

Angelo Di MAIO
MATRICOLA N86003699
angelo.dimai3@studenti.unina.it

Santolo BARRETTA
MATRICOLA N86003666
santolo.barretta@studenti.unina.it