PizzeriaDB

Progetto di Basi di Dati di Marco Cappellari e Andrea Giurisato

1 Abstract

PizzeriaDB è una rinomata pizzeria con una vasta clientela che ha recentemente rinnovato il proprio database per offrire un'esperienza ancora migliore ai suoi clienti.

PizzeriaDB è dotato di un sistema di prenotazione dei tavoli che assicura un'esperienza piacevole per i clienti. Grazie a questo sistema, è possibile effettuare prenotazioni in anticipo, garantendosi così un posto comodo nel ristorante. Inoltre gestisce accuratamente ogni ordinazione, permettendo un'organizzazione impeccabile del flusso di clienti.

Parlando del personale, tiene traccia dei membri del team, tra cui camerieri e cuochi. Ogni dipendente ha un profilo personale completo di informazioni dettagliate. Ciò consente di pianificare con precisione quali tavoli vengano serviti da parte dei camerieri e quali ordini vengano preparati dai cuochi.

La vera eccellenza di PizzeriaDB risiede nelle sue gustose pizze, ognuna caratterizzata da una combinazione unica di ingredienti. Il database ospita un'ampia selezione di pizze, che spaziano dalle classiche alle creazioni gourmet più innovative. Ogni pizza è accuratamente documentata con il proprio prezzo e un elenco completo degli ingredienti.

Per i clienti fedeli, si offre inoltre un programma di tessera fedeltà esclusivo. I clienti possono registrarsi per ottenere una tessera che accumula punti ad ogni ordine effettuato. Questi punti possono essere successivamente riscattati per sconti speciali o benefici extra.

2 Analisi dei requisiti

2.1 Descrizione testuale

Nella base di dati sono presenti i dati dei **Clienti** registrati all'applicazione, di ogni utente sono noti:

- Mail
- Nome
- Cognome
- Data nascita
- Numero telefono

Ogni cliente può richiedere una sola **Tessera** fedeltà a lui associata per accumulare i punti derivanti dall'acquisto delle pizze. Nella tessera viene specificata inoltre la data di iscrizione.

Ogni cliente ha la possibilità di prenotare un tavolo in una determinata data e ora. Di ogni tavolo si conoscono:

- Numero Tavolo
- Posti
- Sala

Del **Personale** che lavora all'interno della pizzeria si conosce:

- Data assunzione
- Nome
- Cognome
- Sesso
- Telefono
- Data nascita

Il personale è composto esclusivamente da:

- Cuochi
- Camerieri

Un cuoco può preparare più ordinazioni e sarà l'unico che potrà completare quell'ordinazione. Inoltre si conosce la paga per ogni pizza cucinata e il numero di pizze preparate in un'ora.

Un cameriere può servire più tavoli, ma un tavolo può essere servito solamente da un singolo cameriere. Si conosce inoltre la paga oraria.

Di ogni **Pizza** si vuole conoscere:

- Nome pizza
- Se è vegetariana
- Prezzo
- Se ha ingredienti freschi
- Lista degli ingredienti da cui la pizza è composta

Di ogni **Ingrediente** inoltre si è a conoscenza di:

- Nome ingrediente
- Se è surgelato
- Valore nutrizionale
- Se è disponibile

La pizzeria tiene conto di ogni **Ordinazione** effettuata, la quale comprende:

- Data ordinazione
- Ora ordinazione
- Se è per asporto
- Totale ordine
- Il cliente che ha effettuato l'ordinazione

- Le pizze ordinate
- Il cuoco che ha preparato l'ordine

2.2 Glossario dei termini

Termine	Descrizione	Collegamenti
Cliente	Utente registrato nell'applicazione	Tavolo, Tessera, Ordinazione
Tavolo	Tavolo prenotabile da un cliente	Cliente, Cameriere
Tessera	Contiene i punti accumulati dall'acquisto delle pizze	Cliente
Personale	Persone che lavorano all'interno della pizzeria	Tavolo, Ordinazione
Cuoco	Personale addetto alla preparazione delle pizze	Entità figlia di Personale
Cameriere	Personale addetto al servizio dei tavoli	Entità figlia di Personale
Pizza	Pizza servita ai clienti	Ingrediente, Ordinazione
Ingrediente	Ingrediente che viene aggiunto sulla pizza	Pizza
Ordinazione	Riepilogo di tutte le pizze che sono state ordinate da un cliente e preparate da un determinato cuoco	Cliente,Pizza, Cuoco
Vegetariana	Indica se una pizza è vegetariana o meno in base agli ingredienti	Attributo di Pizza
Ingredienti freschi	Indica se una pizza non contiene ingredienti surgelati	Attributo di Pizza
Disponibile	Indica se un ingrediente è disponibile o esaurito	Attributo di Ingrediente
Surgelato	Indica se un ingrediente non è "fresco"	Attributo di Ingrediente
Asporto	Indica se il cliente consumerà sul posto o meno	Attributo di Ordinazione
Totale ordine	Indica il totale del prezzo delle pizze ordinate	Attributo di Ordinazione

2.3 Operazioni

OPERAZIONE	TIPO	FREQUENZA
Ricerca del numero di ordinazioni in una certa data	L	50 al giorno
Inserimento di un nuovo cliente	S	5 al giorno
Inserimento di una nuova ordinazione	S	100 al giorno
Annullamento ordine	S	7 al giorno
Aggiornare le pizze	S	1 al mese

Calcolare il guadagno giornaliero della pizzeria	L	1 al giorno
Calcolare il guadagno giornaliero di un cuoco	L	1 al giorno
Numero di tavoli prenotati in una sala	L	20 al giorno
Ingredienti disponibili	L	1 al giorno

3 Progettazione Concettuale

3.1 Lista entità

Se non specificato l'attributo è NOT NULL

Le chiavi primarie sono sottolineate e in grassetto

- Cliente:
  - **Mail**: varchar (50) primary key
  - Nome: varchar(50)
  - Cognome: varchar(50)
  - DataNascita: date
  - NumeroTelefono: varchar(20) unique
- Tessera:
  - **Cliente**: varchar (50) primary key
  - Punti: int, >=0, default 0
  - DataIscrizione: date
- Tavolo:
  - **NumeroTavolo**: int primary key
  - Posti: int, >0
  - Sala: varchar(15)
- Pizza:
  - **NomePizza**: varchar(20) primary key
  - Vegetariana: bool
  - Prezzo: decimal (5,2)
  - IngredientiFreschi: bool
- Ingrediente:
  - **NomeIngrediente**: varchar(20) primary key
  - Surgelato: bool
  - ValoreNutrizionale: int, >=0
  - Disponibile: bool
- Personale:
  - **DataAssunzione**: date primary key
  - **Nome**: varchar(20) primary key
  - **Cognome**: varchar(20) primary key
  - DataNascita: date
  - Sesso: char(1) (Sesso IN ('M', 'F'))
  - Telefono: varchar(20) unique

L'entità ordine si specializza in due sottocategorie con una generalizzazione totale:

- Cuoco:
  - PagaPerPizza: decimal (3,2) > 0
  - PizzaPerOra: int > 0
- Cameriere:
  - PagaOraria: decimal (4,2) >0

- Ordinazione:
  - **Mail_Cliente**: varchar (50) primary key
  - **DataOrdinazione**: date primary key
  - **OraOrdinazione**: time primary key
  - Asporto: bool
  - TotaleOrdine: decimal (5,2) >0

3.2 Tabella delle relazioni

Relazione	Entità coinvolte	Descrizione	Attributi
Possiede	Tessera(1,1) Cliente(0,1)	Un cliente può possedere una sola tessera, la tessera deve essere posseduta da una sola persona	Nessuno
Prenota	Cliente(0,N) Tavolo(0,N)	Un cliente può prenotare più tavoli, un tavolo può essere prenotato da più clienti in giorni e orari diversi	Ora: time DataP: date
Serve	Tavolo(0,1) Cameriere(1,N)	Un cameriere deve servire uno o più tavoli, un tavolo può essere servito da un solo cameriere	Nessuno
Effettua	Cliente(1,N) Ordinazione(1,1)	Un cliente deve effettuare una o più ordinazioni, l'ordinazione deve essere effettuata da un solo cliente	Nessuno
Prepara	Cuoco(0,N) Ordinazione(1,1)	Un cuoco può preparare più ordinazioni, un'ordinazione deve essere effettuata da un solo cuoco	Nessuno
Contiene	Ingrediente(1,N) Pizza(1,N)	Una pizza deve contenere uno o più ingredienti, un ingrediente deve essere contenuto in una o più pizze	Nessuno
Appartiene	Pizza(0,N) Ordinazione(1,N)	Una pizza può appartenere a più ordinazioni, una ordinazione deve contenere una o più pizze	Quantità: int >0

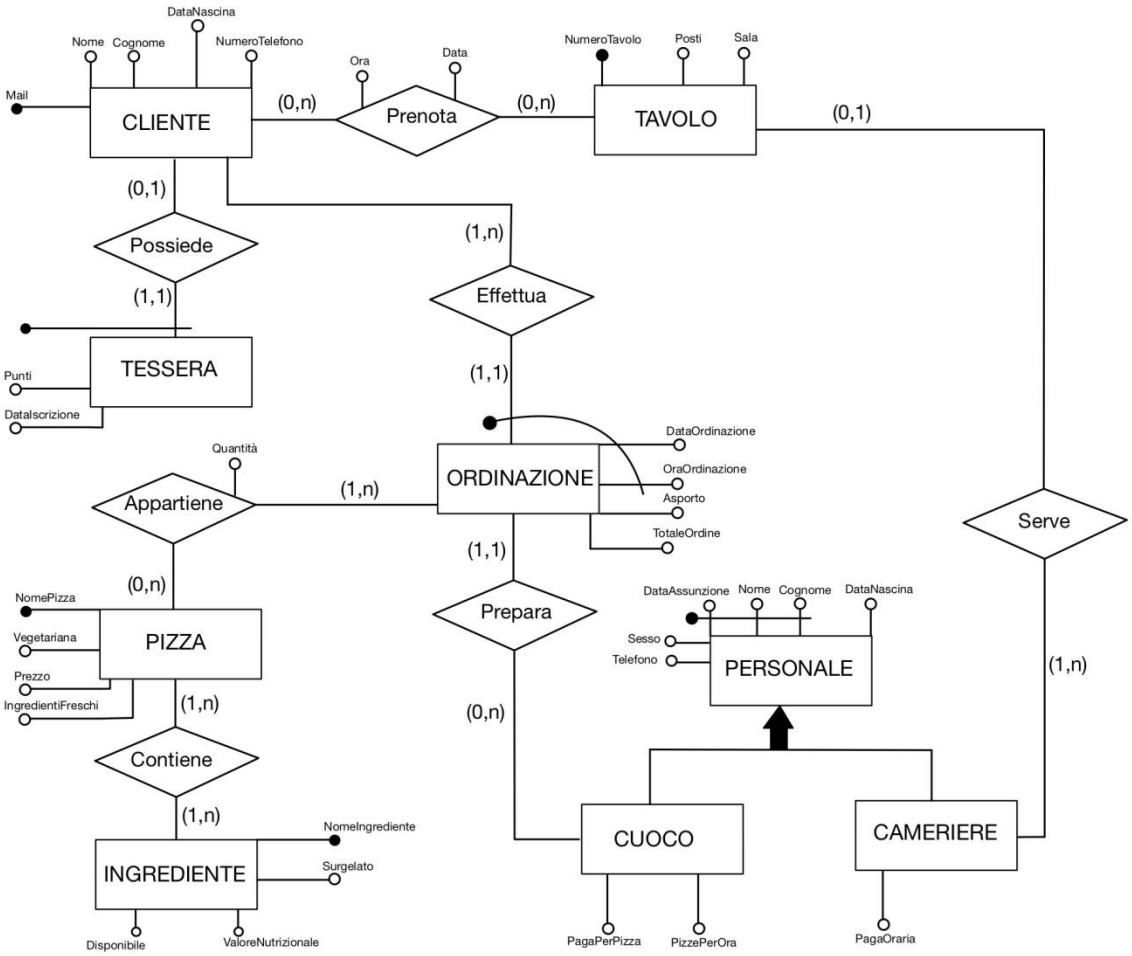
Vincoli non rappresentabili tramite schema E-R:

- La prenotazione di un tavolo dura 1 ora
- Può essere assunta una sola persona al giorno con lo stesso Nome e Cognome
- Il personale per essere assunto deve avere almeno 18 anni

Vincoli di derivazione:

- L’attributo “Asporto” dell’entità Ordinazione si può evincere andando a confrontare la data e l’ora di ordinazione con la data e ora della relazione prenota.
- L’attributo “TotaleOrdine” dell’entità Ordinazione si può ricavare andando a sommare i prezzi di tutte le pizze presenti all’interno dell’ordinazione.
- L’attributo “IngredientiFreschi” è una ripetizione perché si può dedurre andando a controllare se ogni ingrediente all’interno della pizza è surgelato o meno.

Schema Concettuale



4 Progettazione Logica

4.1 Ristrutturazione

4.1.1 Analisi delle ridondanze

- L’attributo “Asporto” dell’entità Ordinazione è ridondante perché si può ricavare andando a confrontare la data e l’ora di ordinazione con la data e ora della relazione prenota.
- L’attributo “TotaleOrdine” dell’entità Ordinazione è ridondante perché si può ricavare andando a sommare i prezzi di tutte le pizze presenti all’interno dell’ordinazione.

- L’attributo “IngredientiFreschi” è una ripetizione perché si può dedurre andando a controllare se ogni ingrediente all’interno della pizza è surgelato o meno.

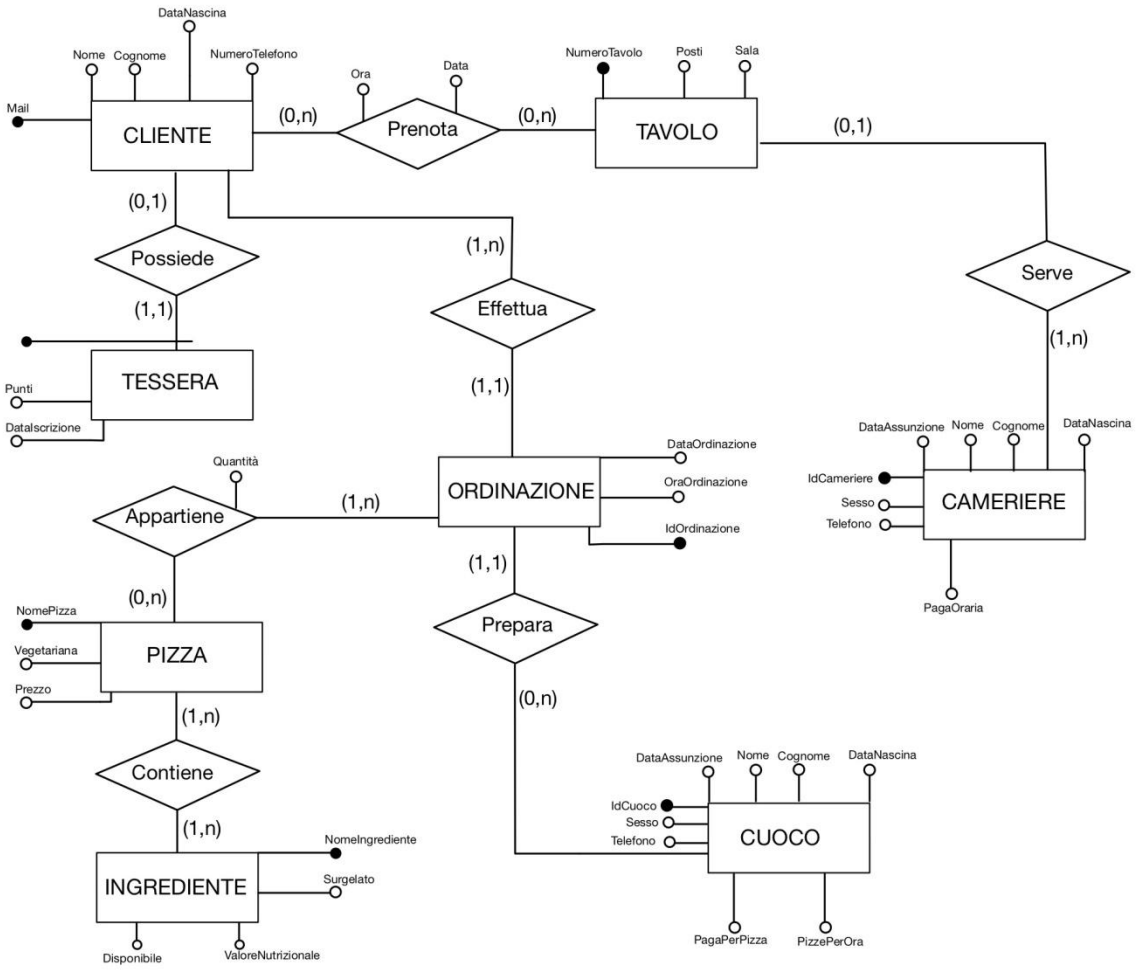
4.1.2 Eliminazione delle generalizzazioni

Generalizzazione	Risoluzione
Cuoco, Cameriere ←Personale	L’entità Personale viene accorpata nelle entità Cuoco e Cameriere per evitare di avere campi nulli nell’entità padre.

4.1.3 Scelta degli identificatori primari

Per l’entità Ordinazione era stato scelto di utilizzare Mail_Cliente, DataOrdinazione, OraOrdinazione come identificatore primario, ma in questo modo la tabella “Appartiene” avrebbe dovuto memorizzare i 3 campi della tabella Ordinazione come chiave esterna e primaria, quindi è stato deciso di introdurre l’attributo IdOrdinazione come chiave primaria per snellire tale tabella. Lo stesso ragionamento è stato applicato alla tabella “Personale” andando a eliminare il vincolo dove bisognava assumere al massimo una sola persona al giorno con lo stesso nome e cognome.

Schema E-R ristrutturato



4.2 Creazione delle tabelle (A→B indica che B è chiave esterna di A)

Cliente (Mail, Nome, Cognome, DataNascita, NumeroTelefono)

Tessera (Punti, DataIscrizione, Cliente → Cliente.Mail)

Tavolo (NumeroTavolo, Posti, Sala, Cameriere→Cameriere.IdCameriere)

Prenota (Ora, DataP, Mail_Cliente →Cliente.Mail, NumeroTavolo→Tavolo.NumeroTavolo)

Cameriere (IdCameriere, DataAssunzione, Nome, Cognome, DataNascita, Sesso, Telefono, PagaOraria)

Cuoco (IdCuoco, DataAssunzione, Nome, Cognome, DataNascita, Sesso, Telefono, PagaPerPizza, PizzePerOra)

Ordinazione (IdOrdinazione, DataOrdinazione, OraOrdinazione, Mail_Cliente→Cliente.Mail, Cuoco→Cuoco.IdCuoco)

Pizza (NomePizza, Vegetariana, Prezzo)

Appartiene (IdOrdinazione→Ordinazione.IdOrdinazione, NomePizza→Pizza.NomePizza, quantità)

Ingrediente (NomeIngrediente, Surgelato, ValoreNutrizionale, Disponibile)

Contiene (NomePizza→Pizza.NomePizza, NomeIngrediente→Ingrediente.NomeIngrediente)

5 Trigger

Nel database siamo andati ad aggiungere 2 trigger:

1. **Date_check** verifica che per uno stesso tavolo non siano presenti 2 prenotazioni, nello stesso giorno, con meno di un'ora di differenza tra una prenotazione e l'altra

Query | Query History

```
1  CREATE OR REPLACE FUNCTION check_prenotazioni() RETURNS TRIGGER AS $$
2  BEGIN
3      IF EXISTS (
4          SELECT 1
5          FROM Prenota AS p
6          WHERE p.DataP = NEW.DataP
7            AND p.NumeroTavolo = NEW.NumeroTavolo
8            AND ABS(EXTRACT(EPOCH FROM (p.Ora - NEW.Ora))) < 3600
9      ) THEN
10         RAISE EXCEPTION 'Il tavolo è già prenotato per quest'ora';
11     END IF;
12     RETURN NEW;
13 END;
14 $$ LANGUAGE plpgsql;
15
16 CREATE TRIGGER Date_check
17 BEFORE INSERT ON Prenota
18 FOR EACH ROW
19 EXECUTE FUNCTION check_prenotazioni();
20
```

2. **Ingredienti_check** mi permette di verificare che prima di inserire un pizza all'interno di una ordinazione, siano presenti tutti gli ingredienti la cui è composta

Query Query History

```
1  CREATE OR REPLACE FUNCTION check_ingredienti_pizza() RETURNS TRIGGER AS $$
2  DECLARE
3      Ingredienti_mancanti INT;
4  BEGIN
5      SELECT COUNT(*) INTO    Ingredienti_mancanti
6      FROM Contiene AS c
7      LEFT JOIN Ingrediente AS i ON c.NomeIngrediente = i.NomeIngrediente
8      WHERE c.NomePizza = NEW.NomePizza AND (i.NomeIngrediente IS NULL OR i.Disponibile = FALSE);
9
10     IF    Ingredienti_mancanti > 0 THEN
11         RAISE EXCEPTION 'Non possiamo aggiungere questa pizza all'ordinazione perchè mancano ingredienti.';
12     END IF;
13
14     RETURN NEW;
15 END;
16 $$ LANGUAGE plpgsql;
17
18 CREATE TRIGGER ingredienti_check
19 BEFORE INSERT ON Appartiene
20 FOR EACH ROW
21 EXECUTE FUNCTION check_ingredienti_pizza ();
22
```

6 Query e indici

6.1 Query

1. Restituire il nome di tutte le pizze che possono essere preparate in base agli ingredienti disponibili

```
SELECT DISTINCT Pizza.NomePizza
FROM Pizza
JOIN Contiene ON Pizza.NomePizza = Contiene.NomePizza
JOIN Ingrediente ON Contiene.NomeIngrediente = Ingrediente.NomeIngrediente
WHERE Ingrediente.Disponibile = true
GROUP BY Pizza.NomePizza
HAVING COUNT(*) = (
    SELECT COUNT(*)
    FROM Contiene AS C
    WHERE C.NomePizza = Pizza.NomePizza
);
```

nomepizza
Asparagi
Bresaola
Carbonara
Chiodini
Crudo
Funghi
Gamberetti
Hawaiian
Leggera
Mais
Margherita
Marinara
Napoli
Pancetta
Patatosa
Pomodorini
Porchetta
Porcini
Prosciutto
Prosciutto e funghi
Prosciutto e mais
Radicchio
Speck
Valtellina
Zucchine

2. Restituire il cameriere e la cameriera con il massimo numero di tavoli assegnati

```sql
SELECT IdCameriere,Sesso, Nome, Cognome, NumeroTavoliServiti
FROM (
    SELECT Cameriere.IdCameriere, Cameriere.Nome, Cameriere.Cognome, COUNT(Tavolo.NumeroTavolo) AS NumeroTavoliServiti, 'M' AS Sesso
    FROM Cameriere
    JOIN Tavolo ON Cameriere.IdCameriere = Tavolo.Cameriere
    WHERE Cameriere.Sesso = 'M'
    GROUP BY Cameriere.IdCameriere
    UNION ALL
    SELECT Cameriere.IdCameriere, Cameriere.Nome, Cameriere.Cognome, COUNT(Tavolo.NumeroTavolo) AS NumeroTavoliServiti, 'F' AS Sesso
    FROM Cameriere
    JOIN Tavolo ON Cameriere.IdCameriere = Tavolo.Cameriere
    WHERE Cameriere.Sesso = 'F'
    GROUP BY Cameriere.IdCameriere
) AS subquery
ORDER BY NumeroTavoliServiti DESC
LIMIT 2;
```

idcameriere	sesso	nome	cognome	numerotavoliserviti
857610	M	Gioele	Testa	19
759791	F	Greta	Rocca	17

3. Restituire i primi 5 clienti che hanno speso in totale più soldi nel ristorante

```sql
SELECT Cliente.Mail, Cliente.Nome, Cliente.Cognome, SUM(Pizza.Prezzo * Appartiene.quantita) AS TotaleSpeso
FROM Cliente
JOIN Ordinazione ON Cliente.Mail = Ordinazione.Mail_Cliente
JOIN Appartiene ON Ordinazione.IdOrdinazione = Appartiene.IdOrdinazione
JOIN Pizza ON Appartiene.NomePizza = Pizza.NomePizza
GROUP BY Cliente.Mail
ORDER BY TotaleSpeso DESC
LIMIT 5;
```

mail	nome	cognome	totalespeso
luca.russo@libero.it	Luca	Russo	521.00
marta.marino@gmail.com	Marta	Marino	444.50
ettore.caruso@hotmail.com	Ettore	Caruso	426.00
ambra.villa@gmail.com	Ambra	Villa	398.50
alessio.caruso@hotmail.com	Alessio	Caruso	373.50

4. Restituire il nome, cognome e ID dei camerieri che hanno gestito più di 5 prenotazioni

```sql
SELECT Cameriere.IdCameriere, Cameriere.Nome, Cameriere.Cognome, COUNT(*) as NumeroPrenotazioni
FROM Cameriere
JOIN Tavolo ON Cameriere.IdCameriere = Tavolo.Cameriere
JOIN Prenota ON Tavolo.NumeroTavolo = Prenota.NumeroTavolo
GROUP BY Cameriere.IdCameriere
HAVING COUNT(*) > 5
ORDER BY NumeroPrenotazioni ASC;
```

idcameriere	nome	cognome	numeroprenotazioni
963027	Greta	Coppola	6
906613	Matteo	Giorgi	7
893535	Mattia	Piazza	8

5. Trovare i cuochi che hanno preparato più di 60 pizze

```
SELECT C.IdCuoco, C.Nome, C.Cognome, SUM(A.quantita) AS NumeroPizza
FROM Cuoco C
JOIN Ordinazione O ON C.IdCuoco = O.Cuoco
JOIN Appartiene A ON O.IdOrdinazione = A.IdOrdinazione
GROUP BY C.IdCuoco
HAVING SUM(A.quantita) > 60
ORDER BY NumeroPizza DESC;
```

idcuoco	nome	cognome	numeropizza
257567	Martina	Silvestri	77
422047	Anna	Piazza	76
778837	Rebecca	Pellegrini	62
153362	Andrea	Monti	61

6. Ottenere i 5 clienti con più punti nella tessera

```
SELECT Cliente.Mail, Cliente.Nome, Cliente.Cognome, Tessera.Punti
FROM Cliente
JOIN Tessera ON Cliente.Mail = Tessera.Cliente
ORDER BY Tessera.Punti DESC
LIMIT 5;
```

mail	nome	cognome	punti
jacopo.barbieri@libero.it	Jacopo	Barbieri	992
giorgio.franco@libero.it	Giorgio	Franco	951
maria.ricci@libero.it	Maria	Ricci	886
margherita.conti@gmail.com	Margherita	Conti	883
nicola.giordano@hotmail.com	Nicola	Giordano	878

7. **Query parametrica**: Restituire gli ingredienti presenti e se sono surgelati di una determinata pizza (nell'esempio ho inserito la pizza 'Contadina'). (NB: mettere la prima lettera maiuscola alla pizza scelta)

```sql
SELECT Ingrediente.NomeIngrediente,
       CASE WHEN Ingrediente.Surgelato THEN 'si' ELSE 'no' END AS Surgelato
FROM Pizza
JOIN Contiene ON Pizza.NomePizza = Contiene.NomePizza
JOIN Ingrediente ON Contiene.NomeIngrediente = Ingrediente.NomeIngrediente
WHERE Pizza.NomePizza = 'Contadina';
```

```
Inserisci il nome della pizza: Contadina
+-----------------+-----------+
| nomeingrediente | surgelato |
+-----------------+-----------+
| Pomodoro        | no        |
| Mozzarella      | no        |
| Radicchio       | si        |
| Spinaci         | si        |
| Piselli         | si        |
+-----------------+-----------+
```

8. Determinare il numero di clienti che hanno consumato per asporto o al ristorante

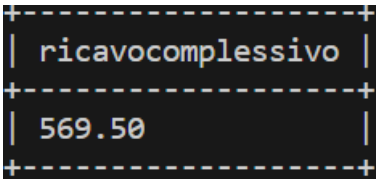
```sql
SELECT
    CASE
        WHEN EXISTS (
            SELECT 1
            FROM Prenota T
            WHERE O.OraOrdinazione = T.Ora AND O.DataOrdinazione = T.DataP AND O.Mail_Cliente = T.Mail_Cliente
        ) THEN 'Ristorante'
        ELSE 'Asporto'
    END AS Modalita,
    COUNT(*) AS Quantita
FROM Ordinazione O
JOIN Cliente C ON O.Mail_Cliente = C.Mail
GROUP BY Modalita;
```

```
+------------+----------+
| modalita   | quantita |
+------------+----------+
| Ristorante | 102      |
| Asporto    | 500      |
+------------+----------+
```

9. Mostrare il ricavo complessivo nell'anno 2022

La query può essere ottimizzata attraverso l'utilizzo dell'indice **IndiceOrdinazione**.

```sql
SELECT SUM(OrdinazioneImporto.ImportoTotale) AS RicavoComplessivo
FROM (
    SELECT O.IdOrdinazione, SUM(P.Prezzo * AO.quantita) AS ImportoTotale
    FROM Ordinazione AS O
    JOIN Appartiene AS AO ON O.IdOrdinazione = AO.IdOrdinazione
    JOIN Pizza AS P ON AO.NomePizza = P.NomePizza
    WHERE EXTRACT(YEAR FROM O.DataOrdinazione) = 2022
    GROUP BY O.IdOrdinazione
) AS OrdinazioneImporto;
```



6.2 Indici

La tabella Ordinazione viene utilizzata molto spesso in lettura per calcolare il ricavato giornaliero del ristorante. Aggiungendo un indice nella suddetta tabella nell'attributo DataOrdinazione si riesce ad ottimizzare il calcolo del ricavo giornaliero (mensile, annuo) perché si evita di scorrere tutti i record della tabella grazie all'indice creato.

```
create index IndiceOrdinazione on Ordinazione (DataOrdinazione);
```

7 Codice C++

7.1 Descrizione dell'utilizzo del codice

Il codice C++ per l'esecuzione delle query consiste in un unico file .cpp, che va compilato attraverso il comando g++ tabelle.cpp -o Queries -Idependencies/include -Ldependencies/lib -lpq.

Prima di poter compilare il file è necessario copiare i file libpq.dll e libpq.lib in "./dependencies/lib" e copiare libpq-fe.h, pg_consig_ext.h e postgres_ext.h in "./dependencies/include", dove "./" è il percorso della directory che contiene il file .cpp.

Per eseguire il codice, basta avviare l'eseguibile "Queries", mostrerà a schermo la lista delle query, identificate da un numero da 1 a 9, eseguibili all'interno del programma. Per eseguire una query bisogna inserire da tastiera il numero della query scelta, mentre per terminare l'esecuzione del programma va inserito '0'.

La query **7** richiede l'inserimento di un parametro da parte dell'utente: in base al nome della pizza scelta, verranno mostrati a schermo gli ingredienti che la compongono e il metodo di conservazione (se surgelato o meno).

7.2 Documentazione del codice

Funzioni utilizzate dal codice:

```
PGresult* execute(PGconn* conn, const char* query)
```

Esegue una query passata come stringa ritornandone il risultato. Se l'esecuzione non va a buon fine mostra un messaggio di errore e termina il programma.

```
void printQuery(PGresult* res)
```

Stampa a schermo sotto forma di tabella il risultato res di una query. La funzione è in grado di gestire la dimensione delle colonne automaticamente.

```
void printLine(int campi, int* maxChar)
```

Funzione ausiliaria di printQuery che stampa una riga di separazione per la tabella.

```
char* chooseParam(PGconn* conn, const char* query, const char* table)
```

Viene utilizzata per mostrare un elenco di valori da una query per poter sceglierne uno (viene usata per scegliere la pizza nella query 7).

Le pizze presenti all'interno del menù sono: