

## 2

## Specifiche dei Requisiti

Il sistema deve permettere ai clienti di effettuare prenotazioni presso i ristoranti iscritti, usufruendo, eventualmente, di promozioni. I ristoratori, invece, possono iscriversi per registrare i propri ristoranti e gestire le loro prenotazioni e le loro promozioni.

Dei clienti interessa conoscere il nome e l'indirizzo e-mail, mentre dei ristoranti interessa il nome, la partita IVA (una stringa numerica), l'indirizzo, la città e l'insieme di tipologie di cucina offerte (scelte da una lista tenuta sotto controllo dallo staff di RistoBook).

I clienti possono prenotare presso un ristorante specificando il giorno, l'ora e il numero di commensali. Le prenotazioni dei clienti devono essere confermate (o rifiutate) dal personale incaricato dei rispettivi ristoranti, che devono poter accedere ad RistoBook tramite una interfaccia dedicata.

Uno dei punti di forza del modello di business di RistoBook è la possibilità per i ristoranti di offrire e pubblicizzare scontistiche ( dette promozioni). In particolare, i ristoratori devono poter definire una promozione specificando una percentuale di sconto sulle prenotazioni consumate in un certo periodo di tempo. Tali promozioni sono valide per al massimo un certo numero di coperti al giorno.

Ad esempio, un ristorante potrebbe definire la seguente promozione: sconto del 20% sulle prenotazioni dalle 20 alle 22 da martedì 15 giugno 2022 a venerdì 18 giugno 2022, valido per al massimo 10 coperti al giorno. Un altro ristorante potrebbe definire invece una promozione del tipo: sconto del 25% sulle prenotazioni dalle 18 alle 19 di tutti i martedì e giovedì dal 1 ottobre al 31 dicembre 2022.

Al momento della prenotazione, il cliente può scegliere una delle promozioni ancora disponibili (anche nessuna, se lo desidera).

Il sistema deve permettere ai ristoratori e ai clienti di gestire lo stato delle prenotazioni. In particolare, quando una prenotazione viene creata, questa è nello stato 'pendente'. Il ristoratore può scegliere se accettarla o rifiutarla. Quando il cliente usufruisce effettivamente della prenotazione, questa viene contrassegnata come 'completata'. Se, invece, il



cliente non dovesse presentarsi al ristorante, il ristoratore contrasseggerà la prenotazione come "non utilizzata". Inoltre, in ogni momento (prima del giorno ed ora prenotati) i clienti possono annullare le proprie prenotazioni, anche se già accettate. Infine, in caso di tutto esaurito (o per altre ragioni, ad es. giorni di chiusura), il responsabile di un ristorante deve poter chiudere le prenotazioni per un certo lasso di tempo (ad es., una certa data e fascia oraria, o un'intera settimana): da quel momento in poi, RistoBook non consentirà più ai clienti di richiedere prenotazioni in quel lasso di tempo (a meno che il ristoratore non le riapra).

Il sistema deve offrire, oltre quelle già descritte, le seguenti funzionalità ai suoi attori:

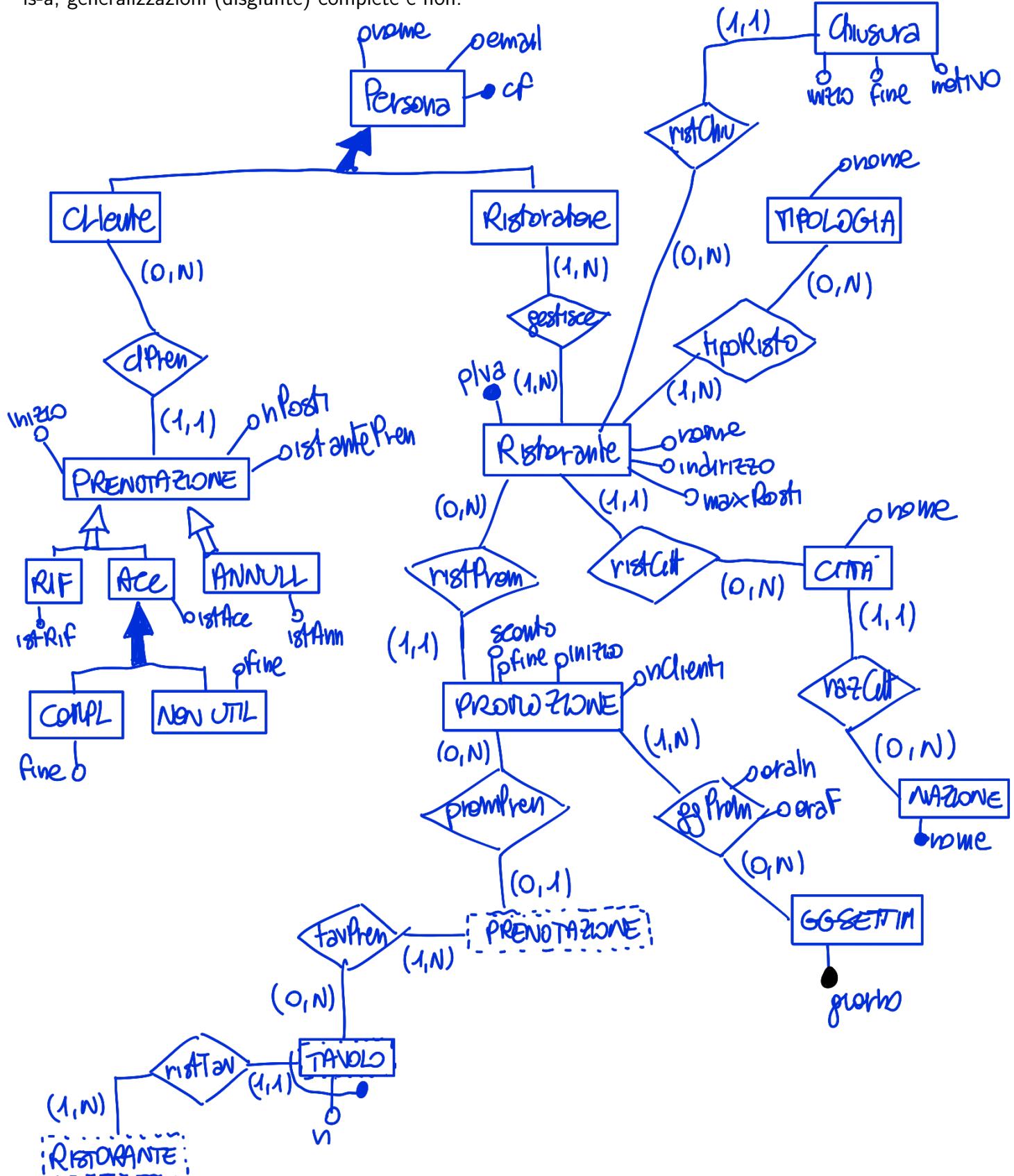
- Il sistema deve permettere ristoratori di RistoBook di calcolare alcune statistiche di utilizzo delle loro promozioni. In particolare, dato un periodo di tempo RistoBook deve permettere calcolare, per ogni promozione del ristorante considerato, il numero medio di clienti al giorno (dove la media è calcolata sui giorni di validità della promozione) che ha utilizzato quella promozione in una prenotazione.
- Data una città  $x$ , un insieme di tipologie di cucina  $C$ , un tasso di sconto minimo  $s$  ed una data  $d$ , i clienti devono poter trovare quali sono i ristoranti nella città  $x$  che offrono almeno una delle tipologie di cucina in  $C$  e prevedono promozioni con sconti di tasso almeno  $s$  nella data  $d$  ancora utilizzabili per il numero di coperti che sono interessati a prenotare.

**Domanda 2 (45 minuti; 75 minuti al massimo)** Proseguire la fase di Analisi Concettuale dei requisiti, producendo un diagramma ER concettuale per l'applicazione, il dizionario dei dati ed eventuali vincoli esterni.

Una risposta soddisfacente a questa domanda è condizione *necessaria* (ma non sufficiente) per superare la prova.

## Diagramma ER

Produrre un diagramma ER concettuale per l'applicazione in termini di entità, relationship, attributi, relazioni is-a, generalizzazioni (disgiunte) complete e non.



**Dizionario dei dati** Per ogni entità e relationship del diagramma ER **con** attributi o vincoli:

- Definire il dominio e la molteplicità degli attributi (se diversa da (1,1))
- Definire eventuali vincoli esterni in logica del primo ordine estesa con teoria degli insiemi e semantica di mondo reale, usando il seguente alfabeto:
  - Un simbolo di predicato  $E/1$  per ogni entità  $E$ .  
Semantica di  $E(x)$ :  $x$  è una istanza di  $E$ .
  - Un simbolo di predicato  $D/1$  per ogni dominio  $D$ .  
Semantica di  $D(x)$ :  $x$  è un valore di  $D$ .
  - Un simbolo di predicato  $r/n$  ( $n > 0$ ) per ogni relationship  $n$ -aria  $r$ .  
Semantica di  $r(x_1, \dots, x_n)$ :  $x_1, \dots, x_n$  è una istanza di  $r$ .
  - Un simbolo di predicato  $a/2$  per ogni attributo  $a$  di entità  
Semantica di  $a(x, v)$ : uno dei valori dell'attributo  $a$  dell'istanza  $x$  è  $v$ .
  - Un simbolo di predicato  $a/(n+1)$  per ogni attributo  $a$  di relationship  $n$ -aria.  
Semantica di  $a(x_1, \dots, x_n, v)$ : uno dei valori dell'attr.  $a$  dell'istanza  $(x_1, \dots, x_n)$  della relat. è  $v$ .
  - Opportuni simboli di predicato (soggetti a *semantica di mondo reale*) per gestire confronti tra valori di domini numerici o comunque ordinati (tra cui  $</2$ ,  $\leq/2$ ,  $>/2$ ,  $\geq/2$ ).
  - Il predicato di uguaglianza  $=/2$  (la cui interpretazione è la relazione che lega ogni elemento del dominio di interpretazione solo con se stesso).
  - Opportuni simboli di costante (soggetti a *semantica di mondo reale*), tra cui *adesso*, interpretato come il valore del dominio DataOra che rappresenta l'istante corrente.

## Risposta

<p>[1] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>Persona</u></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>attributo</th><th>dominio</th><th>moltepl. (*)</th></tr> </thead> <tbody> <tr> <td>name</td><td>str</td><td></td></tr> <tr> <td>email</td><td>Email</td><td></td></tr> <tr> <td>cf</td><td>CodFisc</td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)	name	str		email	Email		cf	CodFisc		<p>[2] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>Chiusura</u></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>attributo</th><th>dominio</th><th>moltepl. (*)</th></tr> </thead> <tbody> <tr> <td>inizio</td><td>DataOra</td><td></td></tr> <tr> <td>fine</td><td>DataOra</td><td></td></tr> <tr> <td>notivo</td><td>str</td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> <p><u>[V.Chiusura.date]</u></p> <p><u><math>\forall c, i, f \ Chiusura(c) \wedge inizio(i, c) \wedge fine(f, c) \rightarrow i &lt; f</math></u></p>	attributo	dominio	moltepl. (*)	inizio	DataOra		fine	DataOra		notivo	str	
attributo	dominio	moltepl. (*)																							
name	str																								
email	Email																								
cf	CodFisc																								
attributo	dominio	moltepl. (*)																							
inizio	DataOra																								
fine	DataOra																								
notivo	str																								

<p>3] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>Tipologia</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td><u>name</u></td> <td><u>str</u></td> <td></td> </tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	<u>name</u>	<u>str</u>		<p>5] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>Ristorante</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td><u>plna</u></td> <td><u>Partita</u></td> <td></td> </tr> <tr> <td><u>name</u></td> <td><u>str</u></td> <td></td> </tr> <tr> <td><u>indirizzo</u></td> <td><u>Indirizzo</u></td> <td></td> </tr> <tr> <td><u>maxPosti</u></td> <td><u>int&gt;0</u></td> <td></td> </tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	<u>plna</u>	<u>Partita</u>		<u>name</u>	<u>str</u>		<u>indirizzo</u>	<u>Indirizzo</u>		<u>maxPosti</u>	<u>int&gt;0</u>	
attributo	dominio	moltep. (*)																				
<u>name</u>	<u>str</u>																					
attributo	dominio	moltep. (*)																				
<u>plna</u>	<u>Partita</u>																					
<u>name</u>	<u>str</u>																					
<u>indirizzo</u>	<u>Indirizzo</u>																					
<u>maxPosti</u>	<u>int&gt;0</u>																					

<p>4] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>home</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td><u>home</u></td> <td><u>str</u></td> <td></td> </tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	<u>home</u>	<u>str</u>		<p>6] Tipo: <b>Entità</b>   Relationship (cerchiare)</p> <p>Nome: <u>Nazione</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td><u>home</u></td> <td><u>str</u></td> <td></td> </tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	<u>home</u>	<u>str</u>	
attributo	dominio	moltep. (*)											
<u>home</u>	<u>str</u>												
attributo	dominio	moltep. (*)											
<u>home</u>	<u>str</u>												

7	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Promozione	
attributo	dominio	moltep. (*)
inizio	dataora	
fine	dataora	
sconto	reale[0,1]	
nClienti	int > 0	

(\*) solo se diversa da (1,1)

Vincoli:

[V. Promozione.date]

$$\forall p, i, f \quad \text{Promozione}(p) \wedge \text{inizio}(i, p) \\ \wedge \text{fine}(f, p) \rightarrow i < f$$

[V. Promozione.posti]

$$\forall pr, p, d \quad \text{Promozione}(pr) \wedge \text{nClienti}(pr, p) \\ \wedge \text{data}(d) \rightarrow \text{ClientiPrem}(pr, d) \leq p$$

9	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	GGSettim	
attributo	dominio	moltep. (*)
giorno	Giorno	

(\*) solo se diversa da (1,1)

Vincoli:

8	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Tavolo	
attributo	dominio	moltep. (*)

$$n \quad int \geq 0$$

(\*) solo se diversa da (1,1)

Vincoli:

10	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Prenotazione	
attributo	dominio	moltep. (*)
nPosti	int > 0	

$$istante \quad dataora$$

$$inizio \quad dataora$$

(\*) solo se diversa da (1,1)

Vincoli:

[V. Prenotazioni.date]

$$\forall p, i, ist \quad \text{Prenotazione}(p) \wedge \\ \text{inizio}(p, i) \wedge \text{istanteP}(ist, p) \\ \rightarrow ist \leq i$$

11	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Annnullata	
attributo	dominio	moltep. (*)
istanteFn	dataora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists f. \text{Annullata}.\text{nonComp}$ $\forall p \text{ Annullata}(p)$ $\rightarrow \neg (\text{Completata}(p) \vee \text{NonUtilizzata}(p))$		

13	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Rifiutata	
attributo	dominio	moltep. (*)
istanteR	dataora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists i. \text{Rifiutata}.\text{date}$ $\forall p, \text{ist}, i \text{ Prenotazione}(p) \wedge$ $\text{inizio}(i, p) \wedge \text{istanteP}(\text{ist}, p)$ $\rightarrow (\forall \text{ist}' \text{ istRif}(\text{ist}', p))$ $\rightarrow \text{ist} \leq \text{ist}' \wedge \text{ist}' < i$		

12	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Accettata	
attributo	dominio	moltep. (*)
istanteFc	dataora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists i. \text{Accettata}.\text{date}$ $\forall p, \text{ist}, i \text{ Prenotazione}(p) \wedge$ $\text{inizio}(i, p) \wedge \text{istanteP}(\text{ist}, p)$ $\rightarrow (\forall \text{ist}' \text{ istRif}(\text{ist}', p))$ $\rightarrow \text{ist} \leq \text{ist}' \wedge \text{ist}' < i$		

14	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Non Utilizzata	
attributo	dominio	moltep. (*)
fine	dataora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists f. \text{NonUtilizzata}.\text{date}$ $\forall p, \text{ist}, f \text{ NonUtilizzata}(p) \wedge$ $\text{istanteP}(\text{ist}, p) \wedge \text{fine}(f, p)$ $\rightarrow \text{ist} < f$		

15	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	Completa	
attributo	dominio	moltep. (*)
fine	dataora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists f. \text{Completa}.date]$		
$\forall p, i, f \text{ Completa}(p) \wedge$		
$\text{inizio}(i, p) \wedge \text{fine}(f, p)$		
$\rightarrow i < f$		

17	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:	ggProm	
attributo	dominio	moltep. (*)
orain	ora	
orafin	ora	
(*) solo se diversa da (1,1)		
Vincoli:		
$\exists V. ggProm. orario]$		
$\forall p, g, i, f \text{ Promozione}(p) \wedge GGSettim(g)$		
$\wedge ggProm(p, g) \wedge orain(p, g, i)$		
$\wedge orafin(p, g, f) \rightarrow i < f$		

16	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:		
attributo	dominio	moltep. (*)
(*) solo se diversa da (1,1)		
Vincoli:		

18	Tipo: <b>Entità</b>   Relationship (cerchiare)	
Nome:		
attributo	dominio	moltep. (*)
(*) solo se diversa da (1,1)		
Vincoli:		

Ulteriori vincoli esterni, specifica di eventuali operazioni ausiliarie invocate da tali vincoli, e specifica dei domini concettuali non di tipo base

**Domino Email:** stringa secondo standard

**Domino CodFis:** stringa di 16 caratteri secondo standard

**Domino Giorno:** { 'LUN', 'MAR', 'MERG', 'GIOV', 'VEN', 'SAB', 'DOM' }

**Domino Piva:** stringa numerica secondo standard

**Dominio Indirizzo:**

Via : str

Civico : int > 0 (0,1)

CAP: str di 5 char

### [V. Prenotazione, chiusura]

$\forall \text{ch}, r, i, f \quad \text{Chiusura}(\text{ch}) \wedge \text{ristChius}(r, \text{ch}) \wedge \text{inizio}(i, \text{ch})$

$\wedge \text{fine}(f, \text{ch}) \rightarrow \exists p, t, i' \quad \text{Prenotazione}(p) \wedge \text{tauPre}(p, t)$

$\wedge \text{ristTau}(r, t) \wedge \text{inizio}(i', p) \rightarrow i < i' \wedge i' < f$

### [V. Cliente, prenotazioneMiss]

$\forall c, p, p', i, i' \quad \text{Cliente}(c) \wedge \text{dPre}(c, p) \wedge \text{dPre}(c, p') \wedge p \neq p'$

$\neg \text{Annullata}(p) \wedge \neg \text{Annullata}(p') \wedge \text{Accettata}(p) \wedge \text{Accettata}(p') \wedge$

$\text{inizio}(p, i) \wedge \text{inizio}(p', i') \rightarrow \exists t \quad \text{dataora}(t) \wedge$

$(i \leq t \wedge \forall f \quad \text{fine}(f, p) \rightarrow t \leq f) \wedge$

$(i' \leq t \wedge \forall f \quad \text{fine}(f, p') \rightarrow t \leq f')$

## Risposta alla Domanda 2 (segue)

[V. Tavolo. prenotazioneMsj]

$\forall t, p, p', i, i' \quad \text{Tavolo}(t) \wedge \text{tauPren}(t, p) \wedge \text{tauPren}(t, p') \wedge p \neq p'$

$\rightarrow \text{Annullata}(p) \wedge \neg \text{Annullata}(p') \wedge \text{Accettata}(p) \wedge \text{Accettata}(p') \wedge$

$\text{inizio}(p, i) \wedge \text{inizio}(p', i') \rightarrow \exists t \quad \text{dataora}(t) \wedge$

$(i \leq t \wedge \forall f \quad \text{fine}(f, p) \rightarrow t \leq f) \wedge$

$(i' \leq t \wedge \forall f' \quad \text{fine}(f', p') \rightarrow t \leq f')$

clientProm(pr: Promozione, d: data) : int  $\geq 0$

pre: nessuna

post: No side-effect

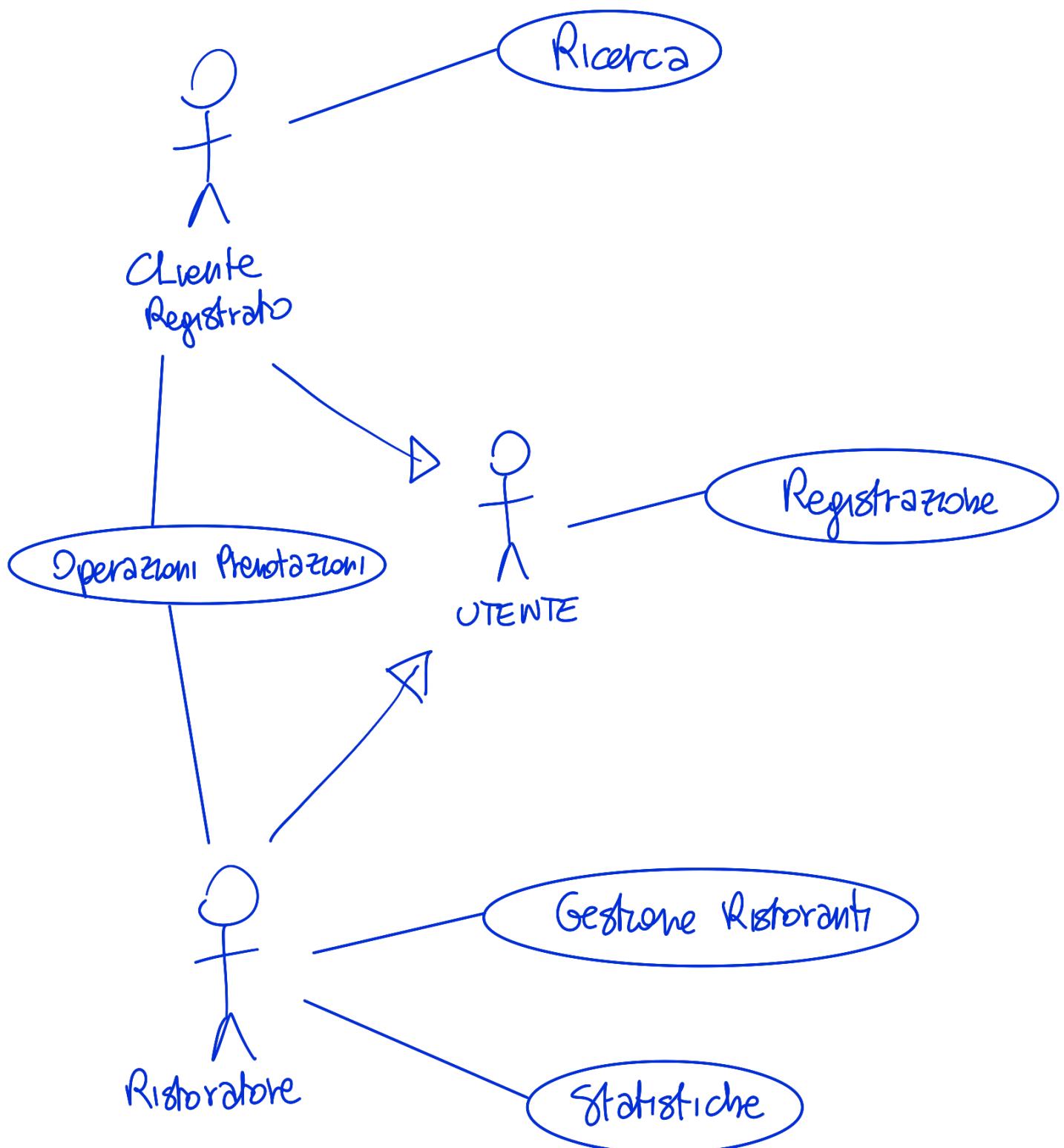
Val ritorno:

$$P = \{(p, n) \mid \text{Prenotazione}(p) \wedge \text{promPren}(pr, p) \wedge$$

$$\text{inPosti}(n, p) \wedge (\exists i \quad \text{inizio}(i, p) \wedge \text{data}(i, d))\}$$

$$\text{result} = \sum_{(p, n) \in P} n$$

**Domanda 3 (5 minuti; 10 minuti al massimo)** Proseguire la fase di Analisi Concettuale dei requisiti, producendo un diagramma UML degli use-case che definisca ad alto livello tutte le funzionalità richieste al sistema.

**Risposta**

**Domanda 4 (10 minuti)** Proseguire la fase di Analisi Concettuale dei requisiti definendo le operazioni degli use-case.

In particolare, per ogni use-case definito nella risposta alla **Domanda 3** definire la **segnatura** di tutte le operazioni che lo compongono, in termini di nome dell'operazione, nomi e dominio concettuale degli argomenti, dominio concettuale dell'eventuale valore di ritorno.

**1 Specifica use-case:** Operazioni Prenotazioni (continua da) (nome use-case)

Operazioni dello use-case:

prenota(n:int > 0, ist: dataora, i: dataora, p: Promozione(0,1),  
c: Cliente, r: Ristorante): Prenotazione

annullaPrenotaz(p: Prenotazione)

**2 Specifica use-case:** Ricerca (nome use-case)

Operazioni dello use-case:

trovaRisto(c: Citta', t: Tipologia(1,N), s: reale[0,1], d: data,  
nP: int > 0): Ristorante(0,N)

**3 Specifica use-case:** Statistiche (nome use-case)

Operazioni dello use-case:

clientiProm(pr: Promozione, d: data): int ≥ 0

mediaClientiProm(istizio: dataora, fine: dataora, r: Ristorante):  
(p: Promozione, n: reale ≥ 0)(0, N)

Specifica use-case: ..... Gestione Ristoranti ..... (nome use-case)

Operazioni dello use-case:

IscrivI Risto (p: Piva, n: str, i: Indirizzo, m: int > 0) : Ristorante

creaPromozione (s: reale [0,1], i: dataora, f: dataora, r: Ristorante,  
G (g: GfSettim, in: ora, fine: ora)(1,N)) : Promozione

nuovaChiusura (i: dataora, f: dataora, m: str, r: Ristorante) : Chiusura

Specifica use-case: ..... Registration ..... (nome use-case)

Operazioni dello use-case:

IscrivI Cliente (n: str, cf: CodFis, e: Email) : Cliente

IscrivI Ristoratore (n: str, cf: CodFis, e: Email) : Ristoratore

Specifica use-case: ..... Operazioni Prenotazioni ..... (nome use-case)

Operazioni dello use-case:

RifiutaPrenotaz (p: Prenotazione)

AccettaPrenotaz (p: Prenotazione)

CompletaPrenotaz (p: Prenotazione)

NonUtilizzataPrenotaz (p: Prenotazione)

Specifica use-case: ..... (nome use-case)

Operazioni dello use-case:

**Domanda 5 (30 minuti; 60 minuti al massimo)** Proseguire la fase di Analisi Concettuale dei requisiti producendo le specifiche concettuali per le operazioni di use-case, **limitandosi** a quelle necessarie a modellare i requisiti contrassegnati dalla barra laterale (come quella qui a sinistra). In particolare, per ogni operazione, definire segnatura, precondizioni e postcondizioni utilizzando il linguaggio della logica del primo ordine. Si assuma lo stesso vocabolario definito alla **Domanda 2**.

Una risposta soddisfacente a questa domanda è condizione *necessaria* (ma non sufficiente) per superare la prova.

### Risposta

$\text{mediaClientiProm}(i: \text{dataora}, f: \text{dataora}, r: \text{Ristorante})$ :

$(p: \text{Promozione}, n: \text{reale} \geq 0)(O, N)$

pre:  $i < f$

post: No side-effect.

Val. ritorno:

$$P = \{(p, n) \mid \text{Promozione}(p) \wedge \text{ristProm}(r, p) \wedge n = \text{numMedPromClien}(p, i, f)\}$$

result = P

$\text{numMedioPromCli}(p: \text{Promozione}, da: \text{dataora}, a: \text{dataora}): \text{reale} \geq 0$

pre: nessuna

post: No side-effect

Val. ritorno:

$$P = \{(pr, n) \mid \text{Prenotazione}(pr) \wedge \text{promPren}(pr, p) \wedge \text{nPosti}(n, pr) \wedge \exists i, f \text{ inizio}(i, p) \wedge \text{fine}(f, p) \wedge (da \leq i \wedge i \leq a \vee da \leq f \wedge f \leq a)\}$$

$$\text{result} = \begin{cases} \text{se } P = \emptyset \rightarrow \text{result} = 0 \\ \text{altrimenti} \rightarrow \text{result} = \frac{\sum n}{(f - i) + 1} \end{cases}$$

## Risposta alla Domanda 5 (segue)

$\text{trovaRisto}(c: \text{Città}, t: \text{Tipologia}(1, N), s: \text{reale}[0, 1], d: \text{data}, nc: \text{int} > 0): \text{Ristorante}(0, N)$

pre: nessuna

post: No side-effect

Val. ritorno:

$$R = \{ r \mid \text{Ristorante}(r) \wedge \text{Citta}(c) \wedge \text{ristCitt}(r, c) \wedge \text{Tipologia}(t) \\ \wedge \text{tipoRisto}(t, r) \wedge \exists p, s', m, i, f, n \\ \wedge \text{Promozione}(p) \wedge \text{seonto}(s, p) \wedge s \leq s' \\ \wedge \text{ristProm}(r, p) \wedge \text{inizio}(i, p) \wedge \\ \wedge \text{fine}(f, p) \wedge i \leq d \wedge d \leq f \\ \wedge \text{nClienti}(p, n) \wedge n - \text{clientiProm}(p, d) \geq nc \}$$

result = R

## 2 Progettazione della base dati e delle funzionalità

**Domanda 6 (20 minuti; 30 minuti al massimo)** Iniziare la fase di progettazione logica della base di dati decidendo il DBMS da utilizzare e ristrutturando lo schema ER concettuale, il dizionario dei dati e i vincoli esterni. In particolare:

- progettare una corrispondenza tra i domini concettuali ed opportuni domini SQL (domini base o utente, oppure realizzati mediante relazioni aggiuntive) supportati dal DBMS scelto
- eliminare attributi multivale o composti
- eliminare relazioni is-a e generalizzazioni
- definire un identificatore primario per ogni entità
- valutare se e come aggiungere ridondanza in maniera controllata
- ristrutturare i vincoli esterni per renderli consistenti con la struttura del nuovo diagramma.

Descrivere brevemente le principali scelte effettuate.

DBMS da utilizzare ..... PostgreSQL

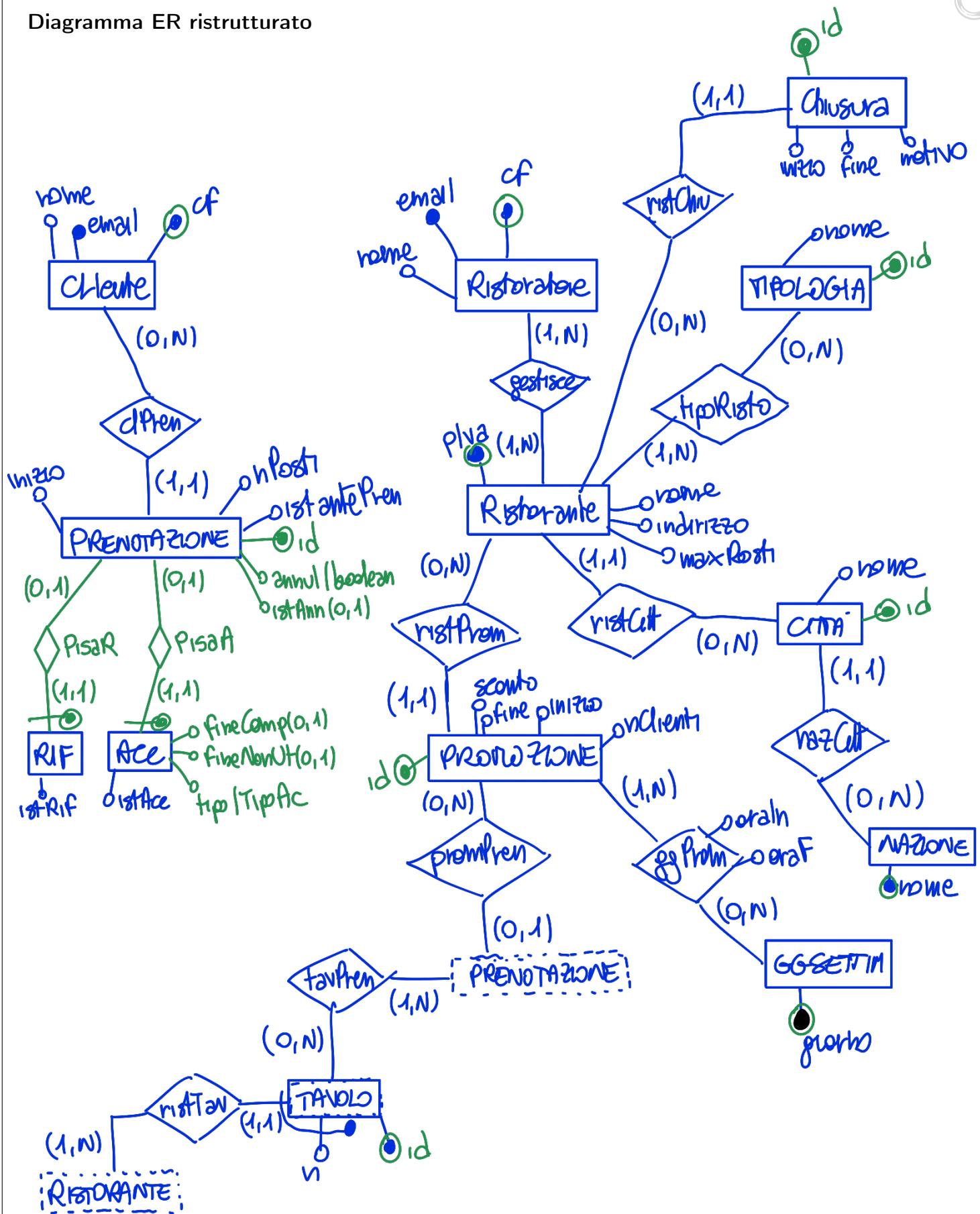
Corrispondenza tra domini concettuali e domini supportati dal DBMS

```

create domain StringS as varchar(50);
create domain StringM as varchar(200);
create domain StringL as varchar(500);
create domain Email as stringM check (isValidEmail(value));
create domain CodFisc as char(16) check (isValidCF(value));
create type Giorno as enum('Lun','...','Dom');
create type Piva as stringM check (isValidPiva(value));
create domain IntegerGZ as integer check (value > 0);
create domain IntegerGEZ as integer check (value ≥ 0);
create domain Int-Sconto as real check (val ≥ 0 and val ≤ 100);
create type Indirizzo as (
    via : StringS
    civico: IntegerGZ*
    cap: char(5)
);
create type Tipofac as enum ('Compl','Non Utilizzata')

```

## Diagramma ER ristrutturato



## Breve descrizione delle scelte effettuate durante la ristrutturazione

divisione di Persona in entità disgiunte  
 trasformazione di 1-sa in Boolean (Annullata)  
 sost. di 1-sa con relationship (Rif, Acc)  
 fusione di entità in Acc

**Vincoli esterni introdotti o modificati durante la fase di ristrutturazione**  
 (si omettano i vincoli esterni la cui formulazione è rimasta identica a seguito della ristrutturazione)

$$\boxed{[V. \text{Pren. disj}]} \quad T \\ \forall p \text{ Prenotazione}(p) \rightarrow [\exists r \text{ PisaR}(r,p)] \rightarrow [\nexists a \text{ PisaA}(a,p)]$$

$$\boxed{[V. \text{Pren. Annullata}]} \\ \forall p \text{ Prenot}(p) \wedge \text{annullata}(p, \text{true}) \rightarrow \exists i \text{ istAnn}(i,p)$$

$$\boxed{[V. \text{Pren. tipAcc}]} \\ \forall p \text{ Prenotaz}(p) \rightarrow [tip(p, \text{compl})] \leftrightarrow [\exists f \text{ fine}(f,p)] \\ \wedge [tip(p, \text{NonUtil})] \leftrightarrow [\exists f \text{ fine}(f,p)]$$

Risposta alla Domanda 6 (segue)

[V. Annullata.nonComp]

$\forall p \text{ Prenotazione}(p) \wedge \text{annullata}(p, \text{true}) \rightarrow \nexists a \text{ PisaA}(p, a)$

[V. Persona.divisiohe] T

$\forall c, r, cf_s, cf_r \text{ Cliente}(c) \wedge \text{Ristoratore}(r) \wedge cf(c, cf_c)$   
 $\wedge cf(r, cf_r) \rightarrow cf_c \neq cf_r$

**Domanda 7 (30 minuti; 60 minuti al massimo)** Proseguire la fase di progettazione logica della base di dati producendo lo schema relazionale della base dati e i relativi vincoli a partire dallo schema ER ristrutturato.

Una risposta soddisfacente a questa domanda è condizione *necessaria* (ma non sufficiente) per superare la prova.

1	Relazione <u>CHIUSURA</u> ... (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi	<u>id</u>   <u>inizio</u>   <u>fine</u>   <u>motivo</u>   <u>ristorante</u>	
Domini	<u>integer</u>   <u>timestamp</u>   <u>timestamp</u>   <u>String</u>   <u>Piva</u>	
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *		
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): <u>seriale:id</u>		
<u>PK: ristorante refer Ristorante(Piva)</u>		
<u>ennupla: inizio &lt; fine</u>		
La relazione accorda le relazioni che implementano le seguenti relationship: <u>ristChiu</u> .....		

2	Relazione <u>RISTORATORE</u> ... (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi	<u>name</u>   <u>email</u>   <u>cf</u>	
Domini	<u>String</u>   <u>Email</u>   <u>CodFisc</u>	
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *		
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):		
<u>inclusione: cf ⊆ gestisce(Ristoratore)</u>		
<u>chiave: email</u>		
La relazione accorda le relazioni che implementano le seguenti relationship: .....		

3	Relazione <u>TIPOLOGIA</u> ... (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi	<u>id</u>   <u>name</u>	
Domini	<u>integer</u>   <u>String</u>	
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *		
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): <u>seriale:id</u>		
La relazione accorda le relazioni che implementano le seguenti relationship: .....		

4	Relazione <u>RISTORANTE</u> ... (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi	<u>piva</u>   <u>name</u>   <u>indirizzo</u>   <u>maxPosti</u>   <u>citta</u>	
Domini	<u>Piva</u>   <u>String</u>   <u>Indirizzo</u>   <u>Integer</u>   <u>Ge</u>   <u>integer</u>	
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *		
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):		
<u>inclusione: piva ⊆ gestisce(Ristorante)</u> <u>PK: citta refer Città(id)</u>		
<u>inclusione: piva ⊆ tipoRisto(Ristorante)</u>		
La relazione accorda le relazioni che implementano le seguenti relationship: <u>ristCitt</u> .....		

5	Relazione <u>CLIENTE</u> ... (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi	<u>cf</u>   <u>name</u>   <u>email</u>	
Domini	<u>CodFis</u>   <u>String</u>   <u>Email</u>	
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *		
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):		
<u>chiave: email</u>		
La relazione accorda le relazioni che implementano le seguenti relationship: .....		

<b>6 Relazione</b>	<u>gestisce</u> .... (nome)	Derivante da:	<b>entità</b>	<b>relationship</b> (cerchiare)
Attributi	<u>Ristoratore</u>   <u>Ristorante</u>			
Domini	<u>Codfis</u>   <u>Piva</u>			

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

PK: ristorante refer Ristorante(piva)

PK: ristoratore refer Ristoratore(cf)

La relazione accorda le relazioni che implementano le seguenti relationship: .....

<b>7 Relazione</b>	<u>tipRisto</u> .... (nome)	Derivante da:	<b>entità</b>	<b>relationship</b> (cerchiare)
Attributi	<u>tipologia</u>   <u>ristorante</u>			
Domini	<u>integer</u>   <u>Piva</u>			

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

PK: ristorante refer Ristorante(piva)

PK: tipologia refer Tipologia(id)

La relazione accorda le relazioni che implementano le seguenti relationship: .....

<b>8 Relazione</b>	<u>PROMOZIONE</u> (nome)	Derivante da:	<b>entità</b>	<b>relationship</b> (cerchiare)
Attributi	<u>id</u>   <u>sconto</u>   <u>inizio</u>   <u>fine</u>   <u>nClienti</u>   <u>ristorante</u>			
Domini	<u>integer</u>   <u>int_sconto</u>   <u>timestamp</u>   <u>timestamp</u>   <u>integer</u>   <u>Piva</u>			

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): seriele::id

PK: ristorante refer Ristorante(Piva)      ennupla::inizio < fine

inclusione: id ⊆ ggProm(Promozione)

La relazione accorda le relazioni che implementano le seguenti relationship: gstProm .....

<b>9 Relazione</b>	<u>GG_SETTIM</u> ... (nome)	Derivante da:	<b>entità</b>	<b>relationship</b> (cerchiare)
Attributi	<u>giorno</u>			
Domini	<u>Giorno</u>			

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

La relazione accorda le relazioni che implementano le seguenti relationship: .....

<b>10 Relazione</b>	<u>ggProm</u> .... (nome)	Derivante da:	<b>entità</b>	<b>relationship</b> (cerchiare)
Attributi	<u>Promozione</u>   <u>GGSettim</u>			
Domini	<u>integer</u>   <u>Giorno</u>			

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

PK: promozione refer Promozione(id)

PK: ggsettim refer GGSettim(giorno)

La relazione accorda le relazioni che implementano le seguenti relationship: .....

11 Relazione ...TAVOLO.... (nome)	Derivante da: entità   relationship (cerchiare)
Attributi   <u>n</u>   <u>id</u>   Ristorante	
Domini   IntegerEZ integer   Piva	

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): seriale : id

PK : ristorante refer Ristorante(Piva)  
chiave : (n, ristorante)

La relazione accorda le relazioni che implementano le seguenti relationship: ...rightar.....

12 Relazione ...PRENOTAZIONE.... (nome)	Derivante da: entità   relationship (cerchiare)
Attributi   <u>id</u>   inizio   nPosti   annullata   istPren   istAnnullata   cliente   Promozione*	
Domini   integer   timestamp   integer   boolean   timestamp   timestamp   codice   integer	

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): seriale : id

inclusione : id ≤ favPren (Prenotazione) PK : cliente..... , FK : promozione....

ennupla : istPren ≤ inizio ennupla : annullata = TRUE → istAnnullata ≠ null

La relazione accorda le relazioni che implementano le seguenti relationship: ...prolPren, clPren.....

13 Relazione ...RIF.... (nome)	Derivante da: entità   relationship (cerchiare)
Attributi   Prendaz   istRIF	
Domini   integer   timestamp	

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

PK : prenotazione refer Prenotazione(id)

La relazione accorda le relazioni che implementano le seguenti relationship: ...PISAR.....

14 Relazione ...ACC.... (nome)	Derivante da: entità   relationship (cerchiare)
Attributi   Prendaz   istAcc   fineComp   fineNonUtl   tipoAc	
Domini   integer   timestamp   timestamp   timestamp   TipoAc	

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

PK : prenotazione refer Prenotazione(id) ennupla : tipoAc = NonUtl → fineNonUtl ≠ null

ennupla : tipoAc = Completata → fineComp ≠ null

La relazione accorda le relazioni che implementano le seguenti relationship: ...Pisaf.....

15 Relazione ..... (nome)	Derivante da: entità   relationship (cerchiare)
Attributi	
Domini	

Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con \*

Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):

La relazione accorda le relazioni che implementano le seguenti relationship: .....

**Ulteriori vincoli esterni**

Per ogni ulteriore vincolo esterno (non ancora espresso perché non definibile mediante vincoli di chiave, foreign key, ennupla, dominio, inclusione), progettare un trigger che lo implementi, definendo: (a) gli eventi da intercettare (inserimento, modifica, eliminazione di ennuple); (b) quando intercettare tali eventi (appena prima o subito dopo l'evento intercettato); (c) la relativa funzione in pseudo-codice con SQL immerso che implementa il controllo del vincolo.

**[V. Prez. diss]**

Operaz: insert o modify in Rif e Acc

Istante: before

Funzione:

- 1 isError: select exists( select \*  
from Rif r, Acc a  
where r.id = newr.prenotazione  
and a.id = newr.prenotazione )

- 2 if isError: genera errore

- 3 else: permetti l'operazione

**[V. Prez. diss]**

Operaz: insert o modify in Cliente e Ristoratore

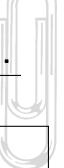
Istante: before

Funzione:

- 1 isError: select exists( select \*  
from Cliente c, Ristoratore r  
where c.cf = newr.cliente  
and r.cf = newr.ristoratore  
and c.cf <> r.cf )

- 2 if isError: genera errore

- 3 else: permetti l'operazione



**Risposta alla Domanda 7 (segue)**

**Domanda 8 (30 minuti; 45 minuti al massimo)** Proseguire la fase di progettazione dell'applicazione producendo le specifiche realizzative delle operazioni di use-case definite per modellare i requisiti contrassegnati dalla barra laterale della specifica dei requisiti.

In particolare, per ogni operazione definire la segnatura, in termini di nome dell'operazione, nomi e dominio SQL degli argomenti, dominio SQL dell'eventuale valore di ritorno, e un algoritmo in pseudo-codice con SQL immerso che verifichi le precondizioni e garantisca il raggiungimento delle postcondizioni definite in fase di Analisi.

Una risposta soddisfacente a questa domanda è condizione *necessaria* (ma non sufficiente) per superare la prova.

### Risposta

mediaClientiProm(*i*:Timestamp, *f*:Timestamp, *r*:Plva): (<*p*:integer, *n*:RealGEZ>)

if *f* ≤ *i*: genera errore

*Q* = select *pr.id*, (sum(*pr.nPosti* / Extract(day from *p.fine* - *p.inizio*) + 1))  
 from Promozione *p*, Prenotazione *pr*  
 where *p.ristorante* = *r* and *pr.promozione* = *p.id*  
 and (:*i* ≤ *p.inizio* and *p.inizio* <= :*f*  
 or :*i* ≤ *p.fine* and *p.fine* <= :*f*)  
 group by *pr.id*

return *Q*

## Risposta alla Domanda 8 (segue)

`trovaRistoro(c:integer, <t:integer>, s:Int-Sconto,  
d:date, nc:IntegerGZ):(<r:Piva>)`

`Q = select r.id`

`from Ristorante r, tipoRistoro tr, Promozione p, Prenotazione pr,`

`(select sum(p.nPosti) as tot`

`from Prenotazione pr`

`where pr.inizio = :d ) PostiProm pp`

`where p.ristorante = r.id and r.citta = :c`

`and tr.ristorante = r.id and tr.tipologia = :t`

`and p.inizio <= :d and :d <= p.fine`

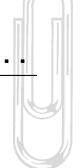
`and p.sconto <= :s and pr.promozione = p.id`

`and Extract(date from pr.inizio) = :d`

`and ((pr.nClienti - pp.tot) >= :nc)`

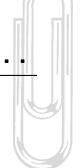
`return Q`

Tempo totale stimato per svolgere questa prova: 180 minuti (tempo totale concesso: 300 minuti).  
[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]



[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]

[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]



[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]