

2

Specifica dei Requisiti

Il sistema *xFit* deve consentire la gestione dei clienti del centro, delle attività offerte, e del personale. In particolare, *xFit* deve rappresentare i dati anagrafici (cognome, nome e data di nascita) del personale amministrativo e dei clienti. Queste categorie possono essere considerate disgiunte. Il personale amministrativo è formato da dipendenti della società. Di ognuno è di interesse conoscere la tipologia di contratto (a tempo indeterminato o determinato, con date di inizio e –per i contratti a tempo determinato– fine) con il quale sono assunti, il livello contrattuale (un intero positivo) e il ruolo ricoperto (ad es., membro di segreteria). Di ogni dipendente è necessario mantenere anche i contratti passati.

Ogni ruolo per il personale amministrativo definisce un insieme di mansioni che i suoi membri possono svolgere utilizzando il sistema. Ad esempio, il personale avente come ruolo ‘membro di segreteria’ deve poter vendere/rinnovare contratti a clienti o visualizzare le informazioni che li riguardano.

Gli istruttori non sono considerati personale stabile della società, in quanto sono inquadrati come liberi professionisti che prestano la loro opera in diversi centri. Di ogni istruttore interessa rappresentare il contratto (solo a tempo determinato) che ha in essere con la società che gestisce il centro e le attività sportive (ad es., fitness, spinning, aerobica, sala pesi) che è titolato a supervisionare.

Il sistema deve poter rappresentare anche informazioni sui clienti. Di ogni cliente interessa (oltre ai dati anagrafici comuni a tutte le tipologie di persone di interesse nel sistema) mantenere l’insieme degli abbonamenti che questi ha sottoscritto con il centro nel tempo. La tipologia degli abbonamenti segue un preciso modello di business: attualmente il centro offre abbonamenti trimestrali, semestrali ed annuali, ma è importante progettare *xFit* di modo che riesca a rappresentare abbonamenti di qualsivoglia durata, se definita dal management (ad esempio, il management potrebbe introdurre un nuovo abbonamento a 15 mesi). Un abbonamento non consente l’accesso indiscriminato alla struttura, ma esclusivamente ad alcune aree (fisiche della struttura, ad es., le diverse

sale) dove vengono praticate le attività sportive comprese nell'abbonamento. Il sistema deve poter rappresentare tali aree; di ogni area è di interesse il nome e la capienza ovvero il numero massimo di clienti che possono ospitare contemporaneamente).

Le attività sportive erogate dal centro vengono definite dal management, insieme alle aree fisiche della struttura (ad es., le diverse sale) e agli orari (giorno della settimana e intervallo orario) in cui vengono erogate. In un'area possono essere praticate più attività (ad es., in giorni e/o orari diversi). Le tipologie di abbonamento che possono essere offerte ai clienti possono variare nel tempo e nel prezzo, e vengono definite dal management. Ad esempio, in questa stagione viene offerta una tipologia di abbonamento semestrale a EUR 150 che comprende le attività di sala pesi e fitness.

VINCO 2 M.
in per blog

Il sistema deve consentire al management della società di definire una nuova tipologia di abbonamento (decidendone prezzo, durata, attività comprese, e periodo nel quale la nuova tipologia di abbonamento può essere venduta ai clienti) e al personale amministrativo di registrare la stipula di un nuovo abbonamento (tra quelli attivi in quel momento) da parte di un cliente; se il cliente è nuovo, deve permettere l'inserimento dei suoi dati anagrafici.

I clienti utilizzano direttamente il sistema interagendo con i varchi di accesso alle diverse aree (ad es., le diverse sale) della struttura. Di ogni varco il sistema deve rappresentare l'insieme delle aree a cui consente l'accesso. Le informazioni sulle aree alle quali un varco consente l'accesso sono definite dal management. Quando un cliente (riconoscibile attraverso un codice¹) passa attraverso un varco, xFit deve permettere al varco di aprirsi se e solo se il cliente ha un abbonamento attivo per almeno una delle aree protette dal varco dove si sta (in quel momento) tenendo almeno un'attività compresa nel suo abbonamento.

Tutti i varchi fisici sono bidirezionali. Tracciano quindi sia l'ingresso che l'uscita dei clienti dalle diverse aree. Tra le aree della struttura ne esiste una, chiamata area comune che rappresenta zone della struttura dove non si tengono attività (ad es., corridoi, spogliatoi, bar) e quindi sono accessibili a tutti i clienti. (Si veda la figura alla pagina seguente per un esempio di disposizione dei varchi.)

Il sistema deve offrire al management le seguenti ulteriori funzionalità statistiche:

1. Data una fascia oraria O (ad es., dalle 17:00 alle 20:00), calcolare per ogni area A della struttura (ad es., piscina) il numero medio, minimo e massimo (anche se zero) di clienti che hanno acceduto all'area A nella fascia oraria O nell'ultimo mese (le statistiche si intendono su base giornaliera).
2. Dato un cliente C , calcolare il numero di volte in cui C ha frequentato ogni area A del centro nell'ultimo mese. Il risultato deve essere ordinato fornendo prima le aree maggiormente frequentate, terminando con quelle mai frequentate (se esistono).

La Figura 2.1 mostra una possibile piantina di un centro sportivo, con la disposizione dei varchi di accesso. Nell'esempio:

¹Tale codice può consistere in una opportuna codifica della impronta digitale del cliente che viene letta al varco o nel codice trasmesso da un dispositivo di identificazione di prossimità. Si può ignorare questo punto, assumendo, per semplicità, che sia un sistema esterno a produrre tale codice.

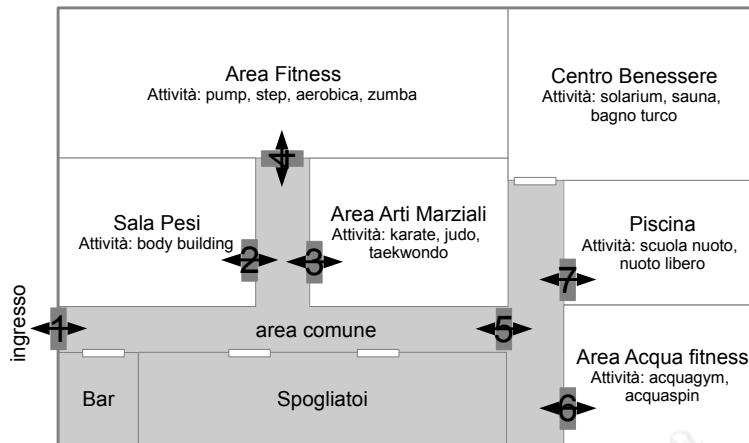


Figura 2.1: Esempio di disposizione dei vanchi.

- il varco 1 può essere acceduto:
 - verso destra da tutti i clienti che hanno un abbonamento attivo
 - verso sinistra da tutti
- il varco 2 può essere acceduto:
 - verso sinistra da tutti i clienti che hanno un abbonamento attivo che comprende l'attività di body building nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso destra da tutti
- il varco 3 può essere acceduto:
 - verso destra da tutti i clienti che hanno un abbonamento attivo che comprende almeno una tra le seguenti attività: karate, judo, taekwondo nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso sinistra da tutti
- il varco 4 può essere acceduto:
 - verso l'alto da tutti i clienti che hanno un abbonamento attivo che comprende almeno una tra le seguenti attività: pump, step, aerobica, zumba nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso il basso da tutti
- il varco 5 può essere acceduto:



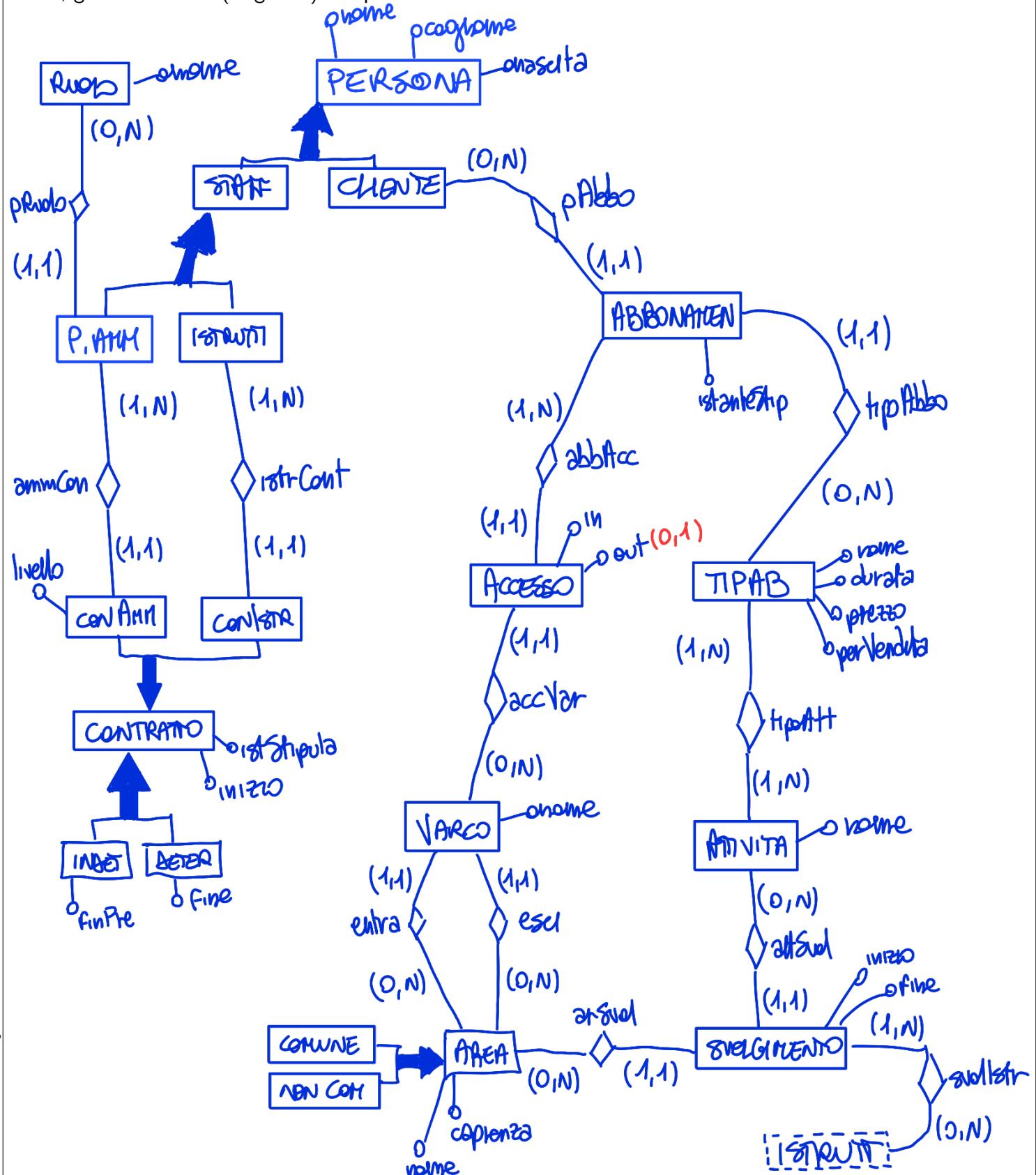
- verso destra da tutti i clienti che hanno un abbonamento attivo che comprende almeno una tra le seguenti attività: solarium, sauna, bagno turco, scuola nuoto, nuoto libero, acquagym, acquaspin nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso sinistra da tutti
- il varco 6 può essere acceduto:
 - verso destra da tutti i clienti che hanno un abbonamento attivo che comprende almeno una tra le seguenti attività: acquagym, acquaspin nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso sinistra da tutti
 - il varco 7 può essere acceduto:
 - verso destra da tutti i clienti che hanno un abbonamento attivo che comprende almeno una tra le seguenti attività: scuola nuoto, nuoto libero nei giorni ed orari in cui almeno una tra tali attività comprese nell'abbonamento è offerta
 - verso sinistra da tutti

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: Entità Relationship (cerchiare)</p> <p>Nome: <i>Persona</i></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">attributo</th><th style="text-align: left;">dominio</th><th style="text-align: left;">moltepl. (*)</th></tr> </thead> <tbody> <tr> <td><i>nome</i></td><td><i>str</i></td><td></td></tr> <tr> <td><i>Cognome</i></td><td><i>str</i></td><td></td></tr> <tr> <td><i>nasita</i></td><td><i>data</i></td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)	<i>nome</i>	<i>str</i>		<i>Cognome</i>	<i>str</i>		<i>nasita</i>	<i>data</i>		<p>[2] Tipo: Entità Relationship (cerchiare)</p> <p>Nome: <i>Ruolo</i></p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left;">attributo</th><th style="text-align: left;">dominio</th><th style="text-align: left;">moltepl. (*)</th></tr> </thead> <tbody> <tr> <td><i>name</i></td><td><i>str</i></td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)	<i>name</i>	<i>str</i>	
attributo	dominio	moltepl. (*)																	
<i>nome</i>	<i>str</i>																		
<i>Cognome</i>	<i>str</i>																		
<i>nasita</i>	<i>data</i>																		
attributo	dominio	moltepl. (*)																	
<i>name</i>	<i>str</i>																		

<input checked="" type="checkbox"/> Tipo: Entità Relationship (cerchiare)		
Nome: <u>Contratto</u>		
attributo	dominio	moltep. (*)
livello	int > 0	

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

Vincoli:

<input checked="" type="checkbox"/> Tipo: Entità Relationship (cerchiare)		
Nome: <u>Contratto</u>		
attributo	dominio	moltep. (*)
inizC	dataora	

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

Vincoli:

$$\begin{aligned} & [\text{V}. \text{Contratto}. \text{date}] \\ & \forall c, i, \text{ist} \quad \text{Contratto}(c) \wedge \text{inizC}(i, c) \\ & \wedge \text{istanteSipC}(\text{ist}, c) \rightarrow i \leq \text{ist} \end{aligned}$$

<input checked="" type="checkbox"/> Tipo: Entità Relationship (cerchiare)		
Nome: <u>TipoAbbonamento</u>		
attributo	dominio	moltep. (*)
nome	str	
durata	Periodo	
prezzo	Numero	
perVendita	Periodo	

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

Vincoli:

$$\begin{aligned} & \forall t, d_a, d_a' \quad \text{TipoAbbo}(t) \wedge \\ & \text{durata}(d_a, t) \wedge d_a(d_a, x) \\ & \text{perVendita}(p, t) \wedge d_a(p, y) \\ & \rightarrow x \leq y \end{aligned}$$

<input checked="" type="checkbox"/> Tipo: Entità Relationship (cerchiare)		
Nome: <u>Varco</u>		
attributo	dominio	moltep. (*)
nome	str	
in	dataora	
out	dataora	

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

Vincoli:

$$\begin{aligned} & \forall v, i, f \\ & \text{Varco}(v) \wedge \text{in}(i, v) \wedge \text{out}(f, v) \\ & \rightarrow i < f \end{aligned}$$

<p>7 Tipo: <input checked="" type="checkbox"/> Entità Relationship (cerchiare)</p> <p>Nome: <u>Abbonamento</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td>istante</td> <td>dataora</td> <td></td> </tr> </tbody> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> <p><u>[V. Abbonam.date]</u></p> <p>$\forall a, ist, t, p, x \text{ Abbonamento}(a) \wedge \text{tipAb}(t, a)$</p> <p>$\wedge \text{istante}(ist, a) \wedge \text{durata}(p, t)$</p> <p>$\wedge da(p, x) \rightarrow ist \leq da$</p>	attributo	dominio	moltep. (*)	istante	dataora		<p>9 Tipo: <input checked="" type="checkbox"/> Entità Relationship (cerchiare)</p> <p>Nome: <u>Svolgimento</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td>iniz</td> <td>dataora</td> <td></td> </tr> <tr> <td>fine</td> <td>dataora</td> <td></td> </tr> </tbody> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	iniz	dataora		fine	dataora	
attributo	dominio	moltep. (*)														
istante	dataora															
attributo	dominio	moltep. (*)														
iniz	dataora															
fine	dataora															

<p>8 Tipo: <input checked="" type="checkbox"/> Entità Relationship (cerchiare)</p> <p>Nome: <u>Area</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td>capienza</td> <td>$int > 0$</td> <td></td> </tr> <tr> <td>valore</td> <td>str</td> <td></td> </tr> </tbody> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	capienza	$int > 0$		valore	str		<p>10 Tipo: <input checked="" type="checkbox"/> Entità Relationship (cerchiare)</p> <p>Nome: <u>Attività</u></p> <table border="1"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltep. (*)</th> </tr> </thead> <tbody> <tr> <td>nome</td> <td>str</td> <td></td> </tr> </tbody> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltep. (*)	nome	str	
attributo	dominio	moltep. (*)														
capienza	$int > 0$															
valore	str															
attributo	dominio	moltep. (*)														
nome	str															

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

Dominio Periodo

da: dataora

a: dataora

$$\forall p, x, y \text{ Periodo}(p) \wedge \text{da}(x, p) \wedge \text{a}(y, p) \rightarrow x < y$$

Dominio Denaro

valuta: char(3)

importo: reale > 0

[V. Istruttore.contr]

$$\forall p, c \text{ Istruttore}(p) \wedge \text{istrContr}(p, c) \rightarrow \text{Determinato}(c)$$

[V. Contratto.determ]

$$\forall i, f, c \text{ Determinato}(c) \wedge \text{inizio}(i, c) \wedge \text{fine}(f, c) \rightarrow i < f$$

[V. Contratto.indeterm]

$$\forall i, f, c \text{ Indeterminato}(c) \wedge \text{inizio}(i, c) \wedge \text{finePre}(f, c) \rightarrow i < f$$

[V. Svolgimento. date]

$$\forall s, i, f \text{ Svolgimento}(s) \wedge \text{inizio}(i, s) \wedge \text{fine}(f, s) \rightarrow i < f$$

Risposta alla Domanda 2 (segue)

[N. Svolgimento. sorapp]

$\forall a, s, s', i, f, i', f$

$\text{Area}(a) \wedge \text{Svolgimento}(s) \wedge \text{Svolgimento}(s') \wedge \text{arSvol}(a, s) \wedge$
 $\text{arSvol}(a, s') \wedge s \neq s' \wedge \text{inizio}(s, i) \wedge \text{fine}(s, f) \wedge$
 $\text{inizio}(s', i') \wedge \text{fine}(s', f') \rightarrow \exists t \text{ dataora}(t) \wedge$
 $(i \leq t \wedge t \leq f) \wedge (i' \leq t \wedge t \leq f')$

[V. Abbonamento. sottoscrizioneValida]

$\forall a, ist, t, x, y \quad \text{Abbonamento}(a) \wedge \text{istanteSip}(ist, a) \wedge \text{tipAbbo}(t, a)$
 $\wedge \text{perVendita}(p, t) \wedge \text{da}(p, x) \wedge \text{a}(p, y) \rightarrow x \leq ist \wedge ist \leq y$

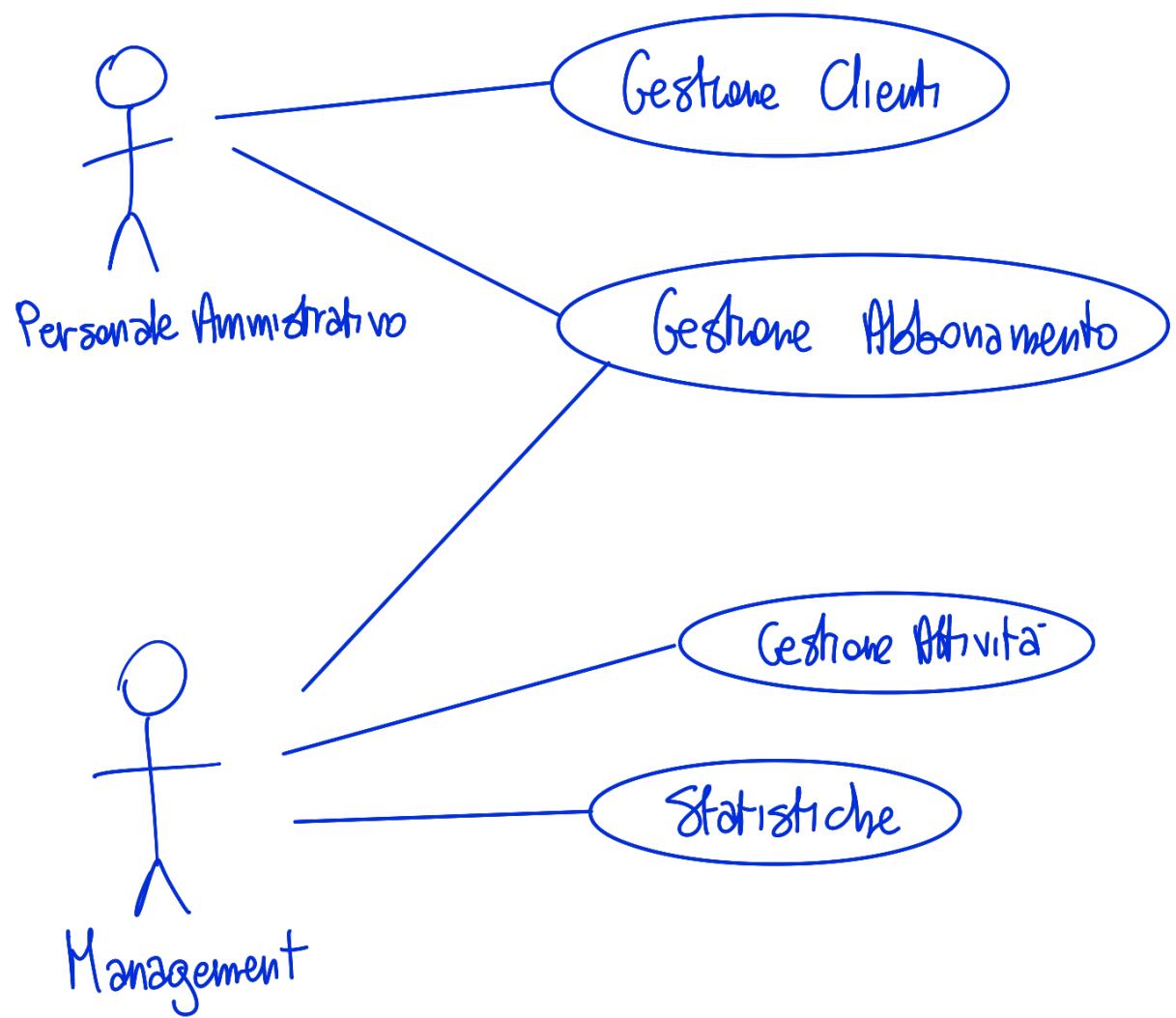
[V. Accessi. disgrunti]

$\forall a, ac, ac', i, o, i', o' \quad \text{Abbonamento}(a) \wedge \text{abbAcc}(a, ac) \wedge \text{abbAcc}(a, ac')$
 $\wedge ac \neq ac' \wedge \text{in}(i, ac) \wedge \text{out}(o, ac) \wedge \text{in}(i', ac') \wedge \text{out}(o', ac')$
 $\rightarrow \exists t \text{ dataora}(t) \wedge (i \leq t \wedge t \leq f) \wedge (i' \leq t \wedge t \leq f')$

[V. Accesso. Varco]

$\forall a, ac, t, in, d, x \quad \text{Abbonamento}(a) \wedge \text{abbAcc}(a, ac) \wedge \text{in}(acc, in) \wedge$
 $\text{tipAbbo}(t, a) \rightarrow \exists v, ar, s, att, i, f \quad \text{accVar}(ac, v) \wedge \text{entra}(v, ar)$
 $\wedge \text{arSvol}(ar, s) \wedge \text{attende}(att, s) \wedge \text{tipAtt}(t, att)$
 $\wedge \text{inizio}(i, s) \wedge \text{fine}(f, s) \wedge in \geq i \wedge in < f$

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: *Gestione Clienti* (nome use-case)

Operazioni dello use-case:

registra(vn: str, co: str, na: data): Cliente

cerca(vn: str(0,1), co: str(0,1), na: data(0,1)): Cliente(0,N)

2 Specifica use-case: *Gestione Abbonamento* (nome use-case)

Operazioni dello use-case:

clienteAbbon(c: Cliente, t: TipAb, ist: dataora): Abbonamento

creaAbbon(no: str, d: Periodo, p: Denaro, pV: Periodo): TipAb

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

Operazioni dello use-case:

calcolaAccessoArea(da: dataora, a: dataora): (ar: Area, min:int ≥ 0, max:int ≥ 0, med:int ≥ 0)(0,N)

clientiArea(ar: Area, da: dataora, a: dataora): Clienti(0,N)

calcolaAccessoCliente(c: Cliente): Area(0,N)

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: *Gestione Attività* (nome use-case)

Operazioni dello use-case:

*creaAtt(*no: str*): Attività*

*cerca(*no: str(0,1)*): Attività(0,N)*

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

Operazioni dello use-case:

3 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

calcolaAccessoArea(da:ora, a:ora) : (ar:Area, min:int ≥ 0,
max:int ≥ 0, med:int ≥ 0)(O,N)

pre: da < a

post: No side-effect

Val. ritorno:

$$\text{Acc} = \left\{ (ar, g, n) \mid \text{Area}(ar) \wedge \text{data}(g) \wedge \right. \\ \text{adesso - } g \leq '30 gg' \wedge \\ \exists i, f, i_0, f_0 \text{ data}(i, g) \wedge \\ \text{data}(f, g) \wedge \text{ora}(i, da) \wedge \text{ora}(f, a) \\ \left. \wedge n = \text{dilInArea}(ar, i, f) \right\}$$

$$\text{result} = \left\{ ar, min, max, med \mid \text{Area}(ar) \wedge \right. \\ \forall g, h \mid (ar, g, n) \in \text{Acc} \rightarrow \begin{array}{l} \min = 0 \\ \max = 0 \\ \text{med} = 0 \end{array} \\ \wedge \exists g, h \mid (ar, g, n) \in \text{Acc} \rightarrow \begin{array}{l} \text{med} = \text{avg}(h) \\ \min = \text{argMin}(h) \\ ((a, g, h) \mid (a, g, h) \in \text{Acc} \wedge a = ar) \end{array} \\ \left. \wedge \max = \text{argMax}(h) \\ ((a, g, h) \mid (a, g, h) \in \text{Acc} \wedge a = ar) \right\}$$

dilInArea(ar:Area, i:dataora, f:dataora) : Persona(O,N)

pre: i < f

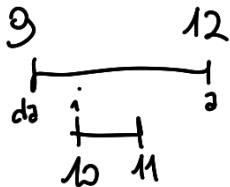
post: result = {c | Cliente(c) \ \exists acc AccessoDa(c, ar, i, f)}

Risposta alla Domanda 5 (segue)

AccessoDa(p : Cliente, ar : Area, da : dataora, a : dataora) : Accesso(O, N)

pre : $da < a$

$da - a$



post : No side-effect.

$i - f$

Valore di ritorno.

$da \leq i \wedge i \leq a \quad \text{or}$
 $da \leq f \wedge f \leq a$

result = { acc | Accesso(acc) $\wedge \exists ab, v, i, o$

$pAbb(ab, p) \wedge abbAcc(ab, acc) \wedge accVar(acc, n)$

$\wedge [entra(v, ar) \wedge esce(v, ar) \wedge in(i, acc)$

$\wedge out(o, acc) \wedge (da \leq i \wedge i \leq a$

$\vee da \leq f \wedge f \leq a)$

calcolaAccessoCliente(c : Cliente) : (a : Area, n : int ≥ 0) (O, N)

pre : $\exists acc, i, ab$ Accesso(acc \wedge in(i, acc) \wedge adesso - i \leq '30 gg'

\wedge abbAcc(ab, acc) \wedge pAbb(c, ab)

post : No side-effect.

Valore di ritorno :

$$T = \{ ar, n \mid Area(ar) \wedge$$

$n = |AccessoDa(c, ar, adesso - '30 gg', adesso)|\}$

result = sort(T, confronto)

Confronto ((ar_1, n_1) : (Area, int ≥ 0), (ar_2, n_2) : (Area, int ≥ 0)) : boolean

pre : nessuna

post : result = true $\Leftrightarrow n_2 \geq n_1$

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);
"      "      StringM as varchar(200);
"      "      StringL as varchar(500);
"      "      IntegerGZ as integer check (value > 0);
"      "      IntegerGEZ as integer check (value ≥ 0);

```

```

create type Denaro as(
    valuta : char(3)
    importo : real );

```

```

create type Periodo as(
    da : timestamp
    a : timestamp );

```

```

create type TipDC as enum ('Determin'; 'Indetermin')

```

```

create type TipOffr as enum ('Comune', 'Non Comune')

```

```

create type TipOffi as enum ('Ammin', 'Istrutt')

```

```

create type TipOp as enum ('staff', 'cliente')

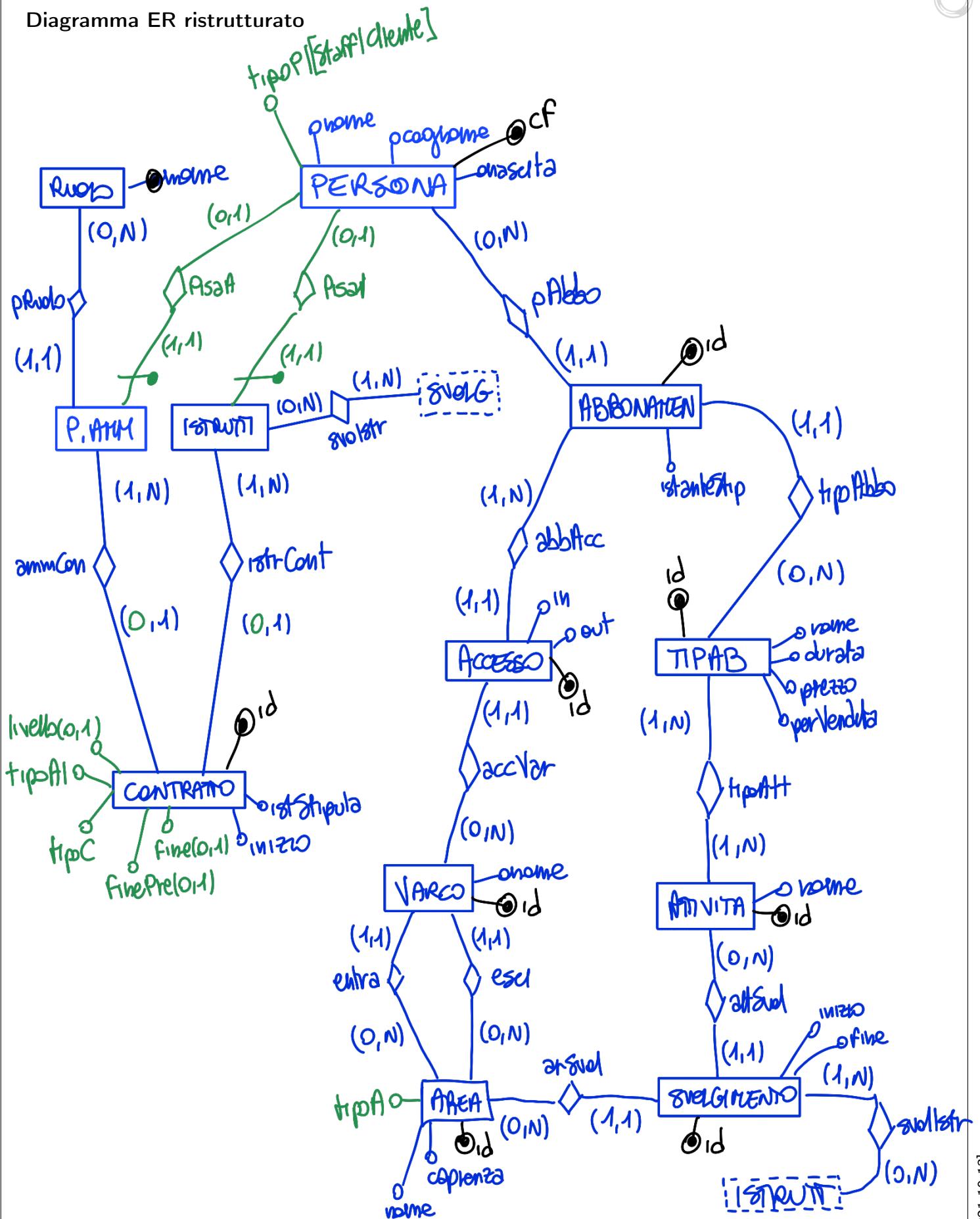
```

```

create domain CodFisc as char(16)
    check (isValidCF(value))

```

Diagramma ER ristrutturato



Breve descrizione delle scelte effettuate durante la ristrutturazione

unione is-a in Area

unione is-a dei tipiC in Contratto

unione is-a dei tipiLavoratori in Contratto

unione is-a dei tipiP in Persona

sostituz. di is-a con relationship

[V. Contratto. tipoC]

$$\forall c \text{ Contratto}(c) \rightarrow [tipo(c, 'Determin'))] \leftrightarrow [\exists f \ fine(f, c)]$$

$$\wedge [tipo(c, 'Indet')] \leftrightarrow [\exists f \ finePrev(f, c)]$$

[V. Area. tipoA]

$$\forall a \text{ Area}(a) \rightarrow [tipo(a, 'Com')] \vee [tipo(a, 'Non Com')]$$

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)

[V. Contratto. tipoFI]

$$\forall c \text{ Contratto}(c) \rightarrow [tipo(c, 'Istrutt')] \leftrightarrow [\exists i \ istrCon(i, c)]$$

$$\wedge [tipo(c, 'Ammin')] \leftrightarrow [\exists a \ ammCon(a, c)]$$

$$\wedge [tipo(c, 'Ammin')] \leftrightarrow [\exists l \ livello(l, c)]$$

[V. Persona. cliente]

$$\forall p \text{ Persona}(p) \rightarrow [tipo(p, 'Cliente')] \leftrightarrow [\exists a \ pAbbo(a, p)]$$

[V. Persona. staff] Trigger

$$\forall p \text{ Persona}(p) \rightarrow [\exists a \ PisaA(a, p)] \rightarrow [\exists i \ Pisal(i, p)]$$

$$[\exists a \ PisaA(a, p)] \vee [\exists i \ Pisal(i, p)]$$

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>PERSONA</u>	(nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>cf</u>	<u>name</u>	<u>cognome</u>	<u>nascita</u>	<u>tipop</u>	
Domini	<u>CodFis</u>	<u>StringS</u>	<u>StringS</u>	<u>data</u>	<u>TipoPer</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>ennupla: tipo = 'STAFF' V tipo = 'CLIENTE'</u> <u>tipo = 'STAFF' → tipo = PAMM V tipo = ISTRUT</u>						
La relazione accorda le relazioni che implementano le seguenti relationship:						

2	Relazione ...	<u>PAMM</u>	(nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>Persona</u>	<u>wodo</u>				
Domini	<u>CodFis</u>	<u>StringS</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>PK: persona refer Persona(cf)</u> <u>Inclusione: persona ⊆ ammCon(Persona)</u> <u>PK: wodo refer Wodo(name)</u>						
La relazione accorda le relazioni che implementano le seguenti relationship:						

3	Relazione ...	<u>RWODO</u>	(nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>wome</u>					
Domini	<u>StringS</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:						

4	Relazione ...	<u>ISTRUTTORE</u>	(nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>Persona</u>					
Domini	<u>CodFis</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>PK: persona refer Persona(cf)</u> <u>Inclusione: persona ⊆ istrCon(Persona)</u>						
La relazione accorda le relazioni che implementano le seguenti relationship:						

5	Relazione ...	<u>SUDGIMENTO</u>	(nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>id</u>	<u>inizio</u>	<u>fine</u>	<u>attività</u>	<u>area</u>	
Domini	<u>integer</u>	<u>Timestamp</u>	<u>Timestamp</u>	<u>integer</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>seriale: id</u> <u>PK: area refer Area(id)</u> <u>Inclusione: id ⊆ sudgr(Sudgimento)</u> <u>PK: attività refer Attività(id)</u> <u>ennupla: inizio < fine</u>						
La relazione accorda le relazioni che implementano le seguenti relationship:						

6 Relazione	<u>svolgimento</u> (nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>istruttore</u> <u>svolgimento</u>			
Domini	<u>CodFisc</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):

PK : istruttore refer Personale(cf)

PK : svolgimento refer Svolgimento(id)

La relazione accorda le relazioni che implementano le seguenti relationship:

7 Relazione	<u>contratto</u> ... (nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>id</u> <u>inizio</u> <u>istStep</u> <u>fine</u> *		<u>finePre</u> * <u>livello</u> *	
Domini	<u>integer</u> <u>Timestamp</u> <u>Timestamp</u> <u>Timestamp</u> <u>Timestamp</u> <u>IntegerG2</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): seriale: id

ennupla: tipoA1 = pAnn AND livello ≠ NULL

ennupla: istStep ≤ inizio

La relazione accorda le relazioni che implementano le seguenti relationship:

8 Relazione	<u>contratto</u> ... (nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>tipoti</u> <u>tipoc</u> <u>pAnn</u> * <u>istruir</u> *			
Domini	<u>TIPOTI</u> <u>TIPOC</u> <u>CodFis</u> <u>CodFis</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 : pAnn refer Personale(cf) ennupla: tipoc = 'DETERM' → fine ≠ NULL ∧ inizio < fine

PK : istruttore refer Personale(cf) ennupla: tipoc = 'Indet' → finePre ≠ NULL ∧ inizio < finePre

La relazione accorda le relazioni che implementano le seguenti relationship:

9 Relazione	<u>abbonamento</u> (nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>id</u> <u>istStep</u> <u>persona</u> <u>tipAb</u>			
Domini	<u>integer</u> <u>Timestamp</u> <u>CodFisc</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): seriale: id

PK : persona refer Personale(cf) inclusione: id ⊆ abbrAcc (Abbonamento)

PK : tipAb refer TipAbbo(id)

La relazione accorda le relazioni che implementano le seguenti relationship:

10 Relazione	<u>accesso</u> (nome)	Derivante da:	entità	relationship (cerchiare)
Attributi	<u>id</u> <u>in</u> <u>out</u> <u>abbonamento</u> <u>Varco</u>			
Domini	<u>integer</u> <u>Timestamp</u> <u>Timestamp</u> <u>integer</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): seriale: id

PK : abbonamento refer Abbonamento(id) ennupla: in < out

PK : varco refer Varco(id) PK : contratto refer Contratto(id)

La relazione accorda le relazioni che implementano le seguenti relationship:

11	Relazione <u>NARCO</u> (nome)	Derivante da: entità relationship (cerchiare)
Attributi	<u>id</u> nome <u>AreaEntra</u> <u>AreaEsc</u>	
Domini	<u>integer</u> <u>string</u> <u>integer</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):

PK : areaEntra refer Area(id)

PK : areaEsc refer Area(id)

La relazione accorda le relazioni che implementano le seguenti relationship: entra, esci

12	Relazione <u>AREA</u> (nome)	Derivante da: entità relationship (cerchiare)
Attributi	<u>id</u> tipoA capienza nome	
Domini	<u>integer</u> <u>TipoA</u> <u>integerGZ</u> <u>stringS</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): seriale: id

ennupla : tipoA = 'comune' V tipoA = 'Non com'

La relazione accorda le relazioni che implementano le seguenti relationship:

13	Relazione <u>ATTIVITA</u> (nome)	Derivante da: entità relationship (cerchiare)
Attributi	<u>id</u> nome	
Domini	<u>integer</u> <u>stringM</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): seriale: id

inclusione : id \leq tipoAtt(Attivita)

La relazione accorda le relazioni che implementano le seguenti relationship:

14	Relazione <u>TIPOABB</u> (nome)	Derivante da: entità relationship (cerchiare)
Attributi	<u>id</u> nome durata prezzo perVendita	
Domini	<u>integer</u> <u>stringS</u> <u>Periodo</u> <u>denaro</u> <u>Periodo</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): seriale: id

ennupla : periodo.da < periodo.a

La relazione accorda le relazioni che implementano le seguenti relationship:

15	Relazione <u>tipAbb</u> (nome)	Derivante da: entità relationship (cerchiare)
Attributi	<u>tipAbb</u> attivita	
Domini	<u>integer</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):

PK : tipAbb refer TipAbb(id)

PK : attivita refer Attivita(id)

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. Persona.staff.diss]

$\forall p \text{ Persona}(p) \rightarrow [\exists a \text{ PisaA}(a,p)] \rightarrow [\nexists i \text{ Pisal}(i,p)]$

Operazione: insert in PAHM e Istr

Istante: before

Funzione:

if: Persona ≠ 'staff' : genera errore

else:

1 isError = select(exists(select *
from PAHM pa, Istr i
where pa.cf = new.persona
and i.cf = new.persona))

)

2 if isError: then blocca

3 else: permetti l'operazione

Risposta alla Domanda 7 (segue)

$[\forall \text{Person}. \text{staff.compl}]$

$$\forall p \text{ Person}(p) \rightarrow [\exists a \text{ Pisaf}(a,p)] \vee [\exists i \text{ Pisal}(i,p)]$$

Operazione: insert in persona

Istante: before

Funzione:

if: Person != 'staff' : genera errore

else:

1 isError = select exists (select *

from PAHM pa, lstr i

where pa.cf = new.persona

or i.cf = new.persona)

)

2 if isError: then blocca

3 else: permetti l'operazione

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

`calcolaAccessoArea(da: time, a: time): (< ar: integer, min: IntegerGEZ
max: IntegerGEZ, med: IntegerGEZ >)`

pre: da < a

post:

$$Q = \text{select area, min(num_cli), max(num_cli), avg(num_cli)}$$

$$\text{From (select ar.id as area, ace.in as giorni,}$$

$$\text{Count(distinct acc.abbonamento) as num_cli}$$

$$\text{from Accesso acc, Varco v, Area ar,}$$

$$\text{Abbonamento ab,}$$

$$\text{where acc.varco = v.id and acc.abbon = ab.id}$$

and (case
????
when v.areaEntra = ar.id then AreaEntra
else AreaEsci)

and (acc.in \geq da and acc.in \leq a)
or acc.out \geq da and acc.out \leq a)
and (acc.in \geq CURRENT-TIME - 30 * '1 day'
or acc.out \geq CURRENT-TIME - 30 * '1 day')
group by ar.id, ace.in) Accessi

group by area

Risposta alla Domanda 8 (segue)

calcolaAccessoCliente(p: CF) : (< ar: integer, n: IntegerGEZ)

Q = select area, count(accessi) as n-volte

From (select p.cf as cliente, ar.id as area, acc.id as accessi

from Abbonamento ab, Accesso ace, Varco v, Area ar

where p.TipoP = 'cliente' and ab.persona = :p

and acc.abbonamento = ab.id and ace.varco = v.id

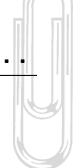
and acc.in < CURRENT_TIMESTAMP - 30 * '1 day'

and v.areaEntra = ar.id) AccessiCliente

group by area

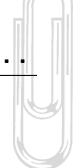
order by n-volte DESC

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.]