



# Basi di Dati, Modulo 2

Sapienza Università di Roma

Facoltà di Ing. dell'Informazione, Informatica e Statistica

Laurea in Informatica

Prof. Toni Mancini

<http://tmancini.di.uniroma1.it>

Progetto 20220222 (P.20220222)

# Amazzon

Versione 2022-02-22

## 2

---

## Specifiche dei Requisiti

La piattaforma *AmaZZon* è utilizzata sia da negozi (che pubblicano offerte di loro prodotti) che dai clienti (che acquistano tali prodotti).

Di ogni **negozi** registrato, il sistema deve mantenere informazioni circa il **nome, l'indirizzo** della sede legale, **almeno un indirizzo email** e **opzionalmente un numero di telefono** per la richiesta di informazioni e assistenza post-vendita.

I **negozi registrati usano la piattaforma per pubblicare offerte di** (ovvero per mettere in vendita) **prodotti di vario genere.**

Gli **articoli disponibili su AmaZZon possono essere offerti da più negozi.** Le diverse offerte per uno stesso articolo possono differenziarsi nelle condizioni applicate. In particolare, all'atto dell'inserimento di un articolo in vendita, il negozio deve dichiarare **nome e descrizione dell'articolo, marca, numero del modello, categoria, un insieme di tag (stringhe), prezzo e istante di inizio e, eventualmente, di fine dell'offerta di vendita.** Si noti *AmaZZon* non è a conoscenza delle informazioni di quanti pezzi di un certo articolo possono essere vendute da un certo negozio. In caso un negozio dovesse terminare la disponibilità di magazzino di un articolo che ha posto in vendita, sarà sua cura eliminare la relativa offerta su *AmaZZon*.

Il seguente esempio potrebbe aiutare a chiarire alcuni dei requisiti precedenti. Il negozio 'New Tech s.r.l.' con sede legale in 'via A. Turing 1, Milano' mette in vendita il nuovo modello di televisore al plasma 'Pony PN123 plus' al prezzo di 1199.00 Eur. L'offerta è valida dal 23 ottobre al 15 novembre 2022 (salvo disponibilità). *AmaZZon* infatti non ha informazioni sulle disponibilità di magazzino del negozio: sarà cura di quest'ultimo eliminare l'offerta dal marketplace in caso di tutto esaurito.

*AmaZZon* permette ai **negozi di definire le spese di spedizione di un articolo in base al paese di destinazione e al numero di pezzi acquistati.** In particolare, il negozio deve associare ad ogni articolo le spese di spedizione per l'acquisto di un singolo pezzo in base al paese di destinazione.

Il negozio può inoltre **definire spese di spedizione ridotte in caso di acquisto di**

**più articoli identici.** Ad esempio, il negozio di cui sopra definisce le seguenti spese di spedizione (per un certo paese) per il 'Pony PN123 plus': acquisto di 2-4/5-9/10-99/100+ articoli: Eur 4.99/3.99/2.99/gratis. Le spese di spedizione per 7 pezzi equivarranno quindi a  $7 \times$  Eur 3.99.

Dato che più negozi possono mettere in vendita lo stesso articolo (a condizioni differenti), *AmaZZon* mantiene una sua vetrina di tutti gli oggetti disponibili, permettendo quindi all'utente di: (i) **navigare nella vetrina di tutti gli oggetti in vendita;** (ii) **consultare, per ognuno di essi, la scheda tecnica con codice identificativo univoco dell'articolo** (una stringa secondo un certo standard globale), nome, marca, numero di modello, categoria, tag; (iii) **accedere a tutte le offerte riguardanti l'articolo da parte dei diversi negozi** (accedendo quindi al prezzo ed ai costi di spedizione).

Come detto, tutti gli articoli in vendita su *AmaZZon* sono presenti in una vetrina unica. Dunque, all'atto della pubblicazione, da parte di un negozio, di un'offerta per un certo articolo, il sistema deve permettere di creare una scheda tecnica per quest'ultimo solo se non esiste già. Altrimenti, la nuova offerta dell'articolo consiste (oltre che al riferimento all'articolo nella vetrina comune) solo nel prezzo e nei costi di spedizione.

Per evitare l'inserimento di un articolo nella vetrina comune con dati errati (ad es., errata categoria), un negozio che, all'atto dell'inserimento di una offerta per un articolo già esistente in vetrina, dovesse notare un errore nei dati della scheda tecnica dell'oggetto, può **segnalare l'errore ed eventuali correzioni allo staff di *AmaZZon*.**

*AmaZZon* permette agli utenti di acquistare dei **buoni regalo**, i quali possono essere spesi sulla piattaforma per l'acquisto di altri prodotti. Di ogni buono regalo interessa conoscere **l'utente che l'ha acquistato, l'importo e il periodo di validità.** Al momento del pagamento, **l'utente può scegliere se utilizzare o meno ognuno dei suoi buoni regalo attivi.** In caso i buoni scelti non dovessero essere sufficienti, il cliente potrà effettuare il pagamento residuo con una delle **carte di credito** registrate (di cui il sistema deve memorizzare: **nome del titolare, numero –una stringa di 16 cifre–, e data di scadenza**), se non scaduta.

**Gli utenti privati possono registrarsi al sistema.** Gli utenti registrati (dei quali interessa il **nome utente** e l'**istante di registrazione**) possono visualizzare gli oggetti in vendita e le relative offerte (cosa per la quale non è necessaria la registrazione).

Gli utenti privati registrati possono effettuare **acquisti** sulla piattaforma, **includendo un qualsiasi numero di articoli e di pezzi per articolo.** Il sistema deve permettere all'utente di **ottenere l'importo totale dell'acquisto,** considerando il costo di ogni articolo e le spese di spedizione verso il suo paese (scegliendo ovviamente la tariffa di spedizione più conveniente tra quelle offerte dal venditore applicabili al numero di articoli acquistati).

Inoltre, gli utenti privati registrati possono salvare delle liste dei desideri (**wish list**), **contenenti oggetti che desiderano acquistare.** Di ogni wish list il sistema mantiene (oltre che l'utente registrato proprietario), il **nome** e l'insieme di oggetti che contiene.

**Periodicamente *AmaZZon* effettua una verifica del prezzo più basso disponibile per ogni oggetto presente nelle wish list di ogni utente e, se rileva un abbassamento di prezzo, notifica la cosa all'utente.**



Le wish list possono essere **pubbliche** o **private**. Mentre le wish list private sono visibili solo all'utente che le ha create, le wish list pubbliche sono consultabili anche agli utenti **amici** del proprietario, che quindi possono avere ispirazione circa regali particolarmente graditi.

L'amicizia tra utenti di *AmaZZon* viene instaurata con una **richiesta** dell'uno e l'accettazione dell'altro. L'amicizia è un concetto simmetrico, quindi in ogni coppia di utenti amici, ogni utente può consultare le wish list pubbliche dell'altro.

Infine, il sistema deve permettere:

- al management di calcolare, dato un periodo di tempo, il **paese dal quale sono provenuti più acquisti nel periodo dato**
- **visualizzare tutti gli oggetti in vendita di una data categoria e contenente un dato insieme di tag (mostrando, per ognuno, il prezzo più basso, comprensivo del costo di spedizione)**
- **visualizzare gli articoli attualmente più moda, che sono gli articoli il cui volume di vendite è salito percentualmente di più nell'ultimo mese, rispetto alla media mensile dei due mesi precedenti.**

# 1 Analisi concettuale

**Domanda 1 (10 minuti)** Raffinare la specifica dei requisiti eliminando inconsistenze, omissioni e ridondanze e producendo un elenco numerato di requisiti il meno ambiguo possibile. (La risposta a questa domanda non sarà valutata, ma si consiglia di svolgere accuratamente questo passo, in quanto può facilitare di molto le attività di progetto.)

## Risposta

### Negozio:

- nome: String
- > indirizzo: Indirizzo
- Email: Email [1..\*]
- Telefono: Tel [0..1]
- associato agli Oggetti  
con prezzo: Valuta,  
inizio: DataOra e  
fine: DataOra [0..1]

### Wishlist:

- nome: String
- > associato agli Oggetti  
che contiene
- > Specializzazioni:
  - Pubbliche
  - Private

### Buono Regalo:

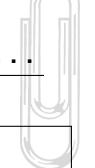
- importo: Valuta
- inizio: DataOra
- fine: DataOra
- > associato all' Utente  
che lo compra

### Oggetto:

- nome: String
- descrizione: Stringa
- numero: Intero
- subito: Categorie
- associato a Utenti
- > associato a Categorie
- > associato ai Tag

### Utente:

- nome: String
- > ist: Patchs
- > associato alle Wishlist
- > associato agli Utenti  
amici
- associato agli  
Oggetti



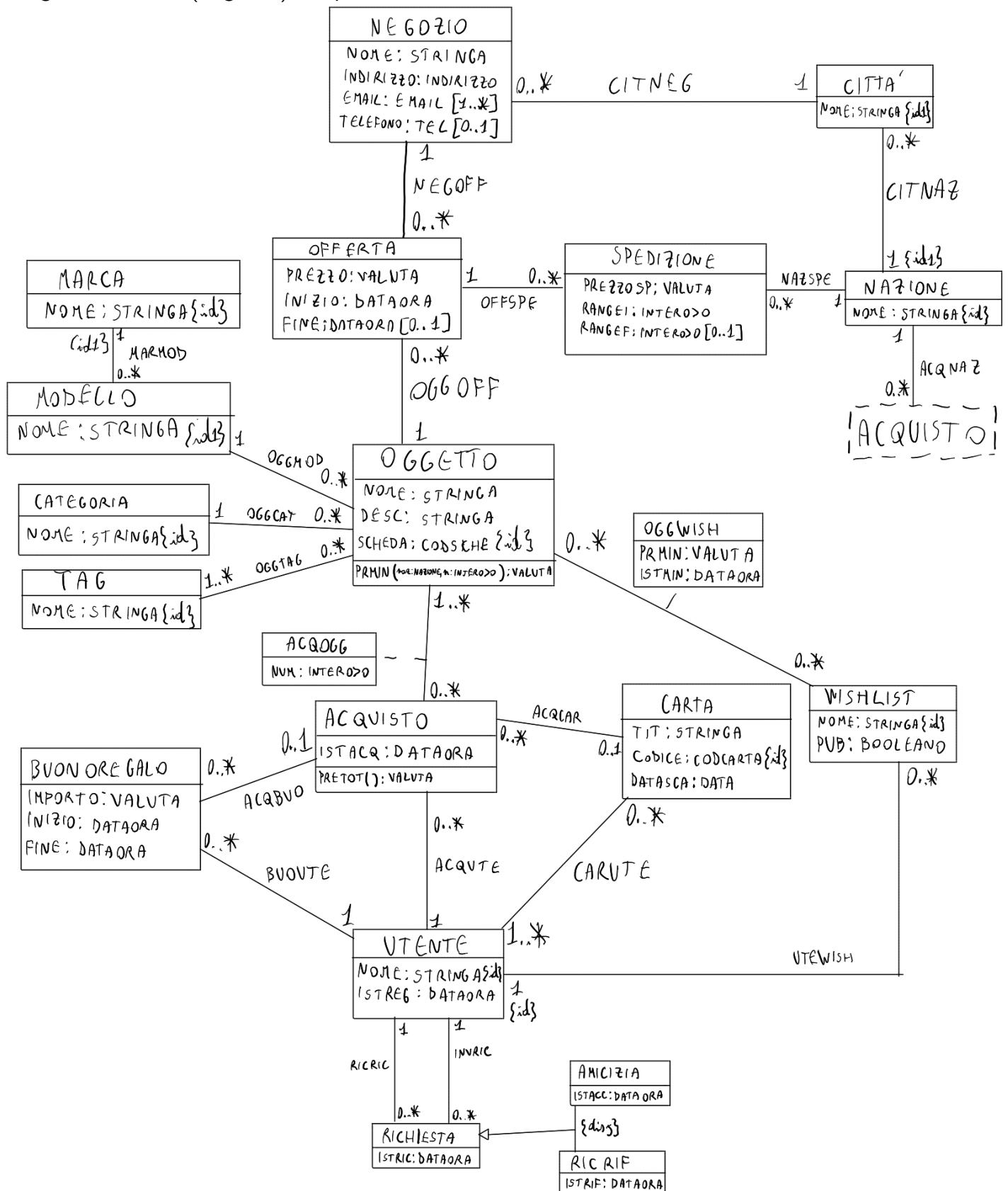
**Risposta alla Domanda 1 (segue)**

**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.

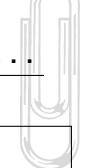
Una risposta soddisfacente a questa domanda è condizione *necessaria* (ma non sufficiente) per superare la prova.

### Diagramma ER

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



[continua alla pagina seguente]



**Risposta alla Domanda 2 (segue)**

**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) Nome: OFFERTA</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;">moltep. (*)</th></tr> </thead> <tbody> <tr> <td></td><td></td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> $\begin{aligned} &\forall .OFFERTA . DISGIUNTE \\ &\forall n, off, agg, i \ NegOff(n, off) \wedge OffOff(off, agg) \\ &\wedge \exists i \ NegOff(n, off) \rightarrow (\exists f \ Fine(f) \wedge \\ &\quad \neg \exists off' \exists i' \ NegOff(n, off') \wedge \exists i'' \exists d \ (off' \geq i' \wedge off' \leq i'' \wedge \\ &\quad \exists d \geq i' \wedge d \leq i'' \wedge \exists f' \ Fine(f') \wedge \\ &\quad d \leq f' \wedge \exists f \ Fine(f, off) \rightarrow \exists off' \exists i' \\ &\quad NegOff(n, off) \wedge OffOff(off, agg) \wedge i' \geq i) \end{aligned}$	attributo	dominio	moltep. (*)				<p>[2] Tipo: <b>Entità</b>   Relationship (cerchiare) Nome: OFFERTA</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;">moltep. (*)</th></tr> </thead> <tbody> <tr> <td></td><td></td><td></td></tr> </tbody> </table> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> $\begin{aligned} &\forall .OFFERTE . FINEDOPONIZIO \\ &\forall \alpha, i \ OFFERTA(\alpha) \wedge \exists i \ NegOff(i, off) \rightarrow (\exists f \ Fine(f) \wedge \\ &\quad i \leq f) \end{aligned}$	attributo	dominio	moltep. (*)			
attributo	dominio	moltep. (*)											
attributo	dominio	moltep. (*)											

<b>3</b> Tipo: <b>Entità</b>   Relationship (cerchiare)	<b>5</b> Tipo: <b>Entità</b>   Relationship (cerchiare)
Nome: SPEDIZIONE.....	Nome: UTENTE.....
attributo      dominio      moltep. (*)	attributo      dominio      moltep. (*)

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

Vincoli:

$\forall \cdot, n, off, i \quad NAZSPE(n, i) \wedge OFFSPE(off, i) \wedge$   
 $RANGE1(i, s) \rightarrow (\exists f \text{ RANGE}(s, f) \rightarrow (\exists j \text{ } s, i, v \text{ }$   
 $NAZSPE(n, s) \wedge OFFSPE(s, off) \wedge RANGE(s, i) \wedge \text{INTERO}(v) \wedge v \geq i \wedge v \leq f \wedge v \geq i' \wedge (\exists f' \text{ RANGE}(f, s) \rightarrow v \leq f')) \wedge$   
 $(\exists f \text{ RANGE}(s, f) \rightarrow (\exists j \text{ } s, i' \text{ } NAZSPE(n, s) \wedge$   
 $OFFSPE(s, off) \wedge i' \geq i))$   
 $\forall \cdot, s \text{ SPEDIZIONE}(s) \wedge RANGE(i, s) \rightarrow (\exists f \text{ RANGE}(f) \rightarrow i \leq f)$

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

Vincoli:

$\forall \cdot, u, a, b, i \quad ACQUTE(a, u) \wedge ACQBUO(a, b) \wedge ISTRAC(a, i)$   
 $\rightarrow BUONTE(u, b) \wedge \exists d, df \text{ INIZIO}(d, df) \wedge FINTE(df, i) \wedge d \leq df$   
 $\forall \cdot, u, a, c, i \quad ACQUTE(a, u) \wedge ACQCAR(a, c) \wedge ISTRAC(i, a)$   
 $\rightarrow CARUTE(u, c) \wedge \exists d \text{ DATASC}(d, c) \wedge d \geq i$   
 $\forall \cdot, u, a, i, m \quad ACQUTE(u, a) \wedge ISTRREG(r, u) \wedge ISTRACA(i, a) \rightarrow r \leq i$

<b>4</b> Tipo: <b>Entità</b>   Relationship (cerchiare)	<b>6</b> Tipo: <b>Entità</b>   Relationship (cerchiare)
Nome: BUONO REGALO.....	Nome: RICHIESTA.....
attributo      dominio      moltep. (*)	attributo      dominio      moltep. (*)

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

Vincoli:

$\forall \cdot, u, a, f \quad BUONO(f) \wedge INIZIO(i, u) \wedge FINE(f, u) \rightarrow i \leq f$   
 $\forall \cdot, u, u', i, i' \quad INVRIC(m, u) \wedge RICRIC(m, u') \rightarrow u \neq u'$

<b>4</b> Tipo: <b>Entità</b>   Relationship (cerchiare)	<b>6</b> Tipo: <b>Entità</b>   Relationship (cerchiare)
Nome: BUONO REGALO.....	Nome: RICHIESTA.....
attributo      dominio      moltep. (*)	attributo      dominio      moltep. (*)

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

Vincoli:

$\forall \cdot, r, u, u' \quad INVRIC(m, u) \wedge RICRIC(m, u') \rightarrow u \neq u'$   
 $\forall \cdot, u, u', i, i' \quad INVRIC(m, u) \wedge RICRIC(m, u') \wedge ISTRREG(u, i) \wedge$   
 $ISTRREG(u', i') \wedge ISTRIC(i, m) \rightarrow i \leq m \wedge i' \leq m$   
 $\forall \cdot, u, u' \quad INVRIC(u, u') \wedge RICRIC(u, u') \rightarrow \neg (INVRIC(u, u') \wedge RICRIC(u, u'))$

7 Tipo: <b>Entità</b>   Relationship (cerchiare) Nome: <u>AMICIZIA</u>	9 Tipo: <b>Entità</b>   Relationship (cerchiare) Nome: <u>RICRIF</u>
attributo      dominio      moltep. (*)	attributo      dominio      moltep. (*)
<hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> <p><math>\forall_{AMICIZIA, DOPORIC}</math></p> <p><math>\forall_{a, im, ia} AMICIZIA(a) \wedge ISTRIC(a, im) \wedge</math></p> <p><math>ISTRACC(a, ia) \rightarrow ia = im</math></p>	<hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> <p><math>\forall_{RICRIF, DOPORIC}</math></p> <p><math>\forall_{M, m, m_f} RICRIF(m) \wedge ISTRIF(m_f, m) \wedge</math></p> <p><math>ISTRRIC(m, m_f) \rightarrow m_f \geq m</math></p>

8 Tipo: <b>Entità</b>   Relationship (cerchiare)	10 Tipo: <b>Entità</b>   Relationship (cerchiare)
Nome: <u>ACQUISTO</u>	Nome: <u>OBJETTO</u>
attributo	attributo
dominio	dominio
moltep. (*)	moltep. (*)
(*) solo se diversa da (1,1)	(*) solo se diversa da (1,1)
Vincoli:	Vincoli:
$\forall_{a,p} \text{ACQUISTO}(a) \wedge \text{PRTOR}(a,p) \wedge$	$\forall_{o,\sigma} \text{OBJETTO}, \text{ESISTEOFFERTO}$
$\forall_{a,p} \text{ACQUISTO}(a) \wedge \text{PRTOR}(a,p) \wedge$	$\forall_{a,o,i} \text{ACQOGG}(a,o) \wedge \text{ISTACQ}(i,a) \rightarrow$
$B = \left\{ (b,i) \mid \text{BONOREGALO}(b) \wedge \right.$	$\exists_{off,i} \text{OFFOGG}(off,o) \wedge \text{INIZIO}(i,off)$
$\left. \text{ACQBONO}(a,b) \wedge \text{IMPORTO}(b,i) \right\} \wedge$	
$\sum_{(b,i) \in B} i \leq p \leftrightarrow \exists c \text{ACQUAR}(a,c)$	
$\forall_{a,m,s,n,lm} \text{ACQNAZ}(a,s) \wedge \text{ACQOGG}(a,o) \wedge \text{NUM}(a,o,mlm)$	
$\rightarrow \exists_{s,m,i,M_f} \text{NATSPF}(m,s) \wedge \text{RANGEI}(m,i) \wedge \text{RANGEF}(M_f) \wedge$	
$mlm \geq mi \wedge num \leq M_f$	
$(\exists f_o \text{FINE}(off,f_o) \rightarrow f_o > i)$	

<p><b>11</b> Tipo: <b>Entità   Relationship</b> (cerchiare)</p> <p>Nome: <u>WISHLIST</u></p> <table border="1" data-bbox="92 242 790 303"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltepl. (*)</th> </tr> </thead> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p> $\begin{aligned} & \forall u, w, i_m, i_{mw}, o \quad \text{OGGWISH}(o, w) \wedge \\ & \text{UTEWISH}(u, w) \wedge \text{ISTRREG}(u, i_m) \wedge \\ & \text{ISTMIN}(w, o, i_{mw}) \rightarrow i_{mw} > i_m \end{aligned}$	attributo	dominio	moltepl. (*)	<p><b>13</b> Tipo: <b>Entità   Relationship</b> (cerchiare)</p> <p>Nome: .....</p> <table border="1" data-bbox="806 242 1511 303"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltepl. (*)</th> </tr> </thead></table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)
attributo	dominio	moltepl. (*)					
attributo	dominio	moltepl. (*)					

<p><b>12</b> Tipo: <b>Entità   Relationship</b> (cerchiare)</p> <p>Nome: .....</p> <table border="1" data-bbox="84 1293 790 1354"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltepl. (*)</th> </tr> </thead> </table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)	<p><b>14</b> Tipo: <b>Entità   Relationship</b> (cerchiare)</p> <p>Nome: .....</p> <table border="1" data-bbox="806 1293 1511 1354"> <thead> <tr> <th>attributo</th> <th>dominio</th> <th>moltepl. (*)</th> </tr> </thead></table> <hr/> <p>(*) solo se diversa da (1,1)</p> <p>Vincoli:</p>	attributo	dominio	moltepl. (*)
attributo	dominio	moltepl. (*)					
attributo	dominio	moltepl. (*)					

15 Tipo: **Entità | Relationship** (cerchiare)

Nome: .....

attributo	dominio	moltepl. (*)
-----------	---------	--------------

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

Vincoli:

17 Tipo: **Entità | Relationship** (cerchiare)

Nome: .....

attributo	dominio	moltepl. (*)
-----------	---------	--------------

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

Vincoli:

16 Tipo: **Entità | Relationship** (cerchiare)

Nome: .....

attributo	dominio	moltepl. (*)
-----------	---------	--------------

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

Vincoli:

18 Tipo: **Entità | Relationship** (cerchiare)

Nome: .....

attributo	dominio	moltepl. (*)
-----------	---------	--------------

(\*) 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

INDIRIZZO:  $(\text{VIA} = \text{STRINGA}, \text{CIVICO} = \text{STRINGA}[0..l], (\text{AP} = [0..9]\{5\}))$

EMAIL: STRINGA SECONDO STANDARD

TEL:  $[0..9]\{10\}$

VALUTA: REALE  $\geq 0$

CODSCHÉ: STRINGA SECONDO STANDARD

CODCARTA:  $[0..9]\{16\}$

OGGETTO. PRMIN( $n_{\sigma}; n_{\tau}; n; \text{INTERO} > 0$ ): VALUTA

-pre:

-post:  
 $O = \{(o, p_i, p_o) \mid \text{OFFOGG}(o, this) \wedge (\exists f \text{ FINE}(o, f) \rightarrow f \geq \text{ADESSO}) \wedge \text{PREZZO}(p_i, o) \wedge \exists p_f, m_i, i \text{ INZIO}(o, i) \wedge \text{OFFSPE}(o, m_i) \wedge p_i = p_f \wedge \text{RANGE1}(m_i, p_f) \wedge (\exists m_f \text{ RANGEF}(m_f, p_f) \rightarrow m_f \geq n) \wedge \text{NAZSPE}(p_f, m_f) \wedge \text{PREZZOSP}(p_f, p_o) \wedge i \leq \text{ADESSO} \wedge m \geq m_i\}$

$$\text{RESULT} = \min_{(o, p_i, p_o) \in O} (p_i \cdot n + p_o)$$

ACQUISTO. PRTOT(): VALUTA

-pre:

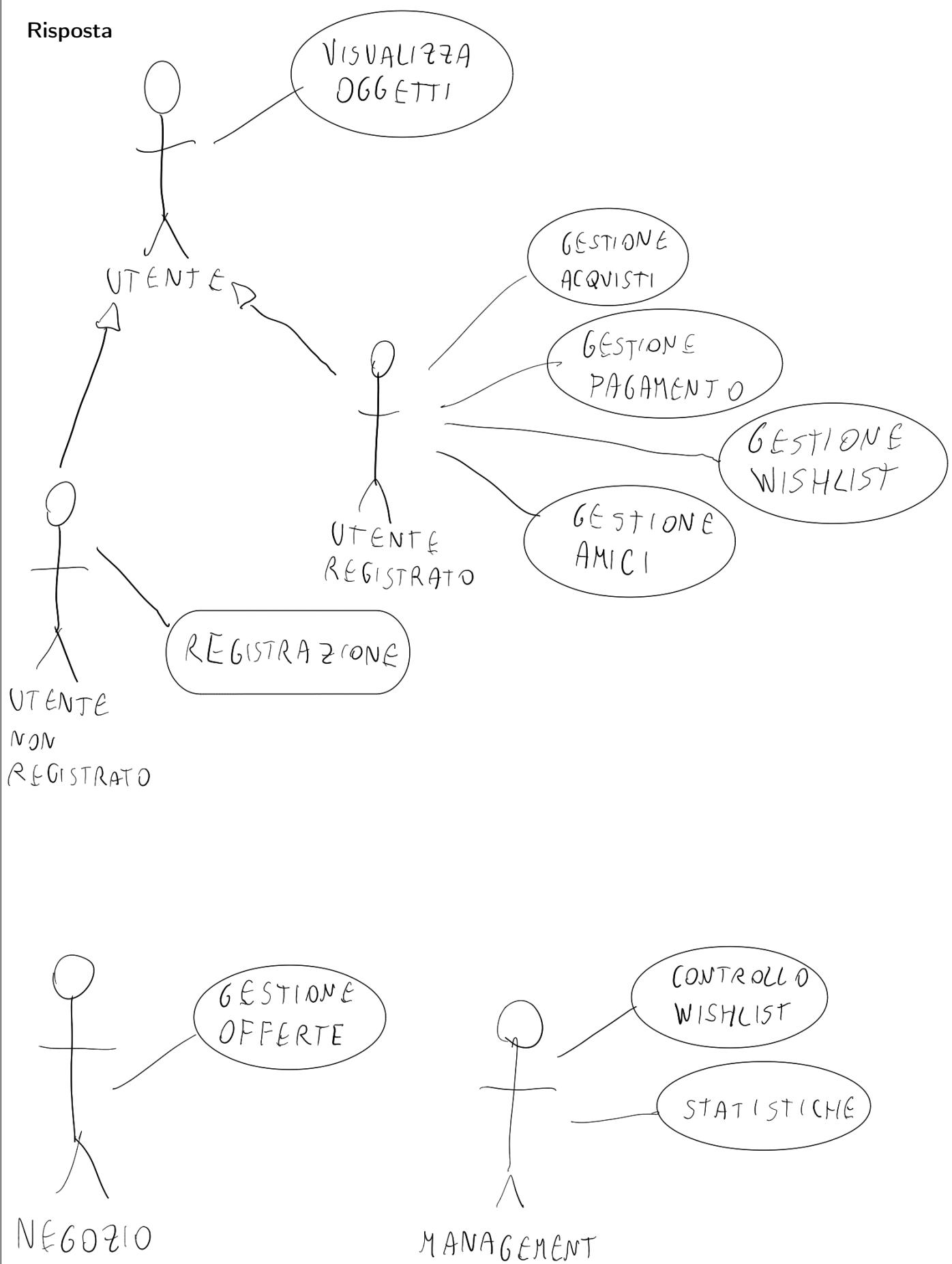
-post: SIA  $m_{\sigma}$  TALE DA SODDISFARE: ACQNAZ( $this, m_{\sigma}$ )

$$O = \{(o, n) \mid \text{ACQOGG}(this, o) \wedge \text{NUM}(this, o, n)\}$$

$$\text{RESULT} = \sum_{(o, n) \in O} \text{PRMIN}(o, m_{\sigma}, n)$$

**Risposta alla Domanda 2 (segue)**

**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.

CONTROLCLOWISHLIST :

CALCOLAPREZZIMIN (w: WISHLIST)

**1 Specifica use-case:** VISUALIZZA...OGGETTI..... (nome use-case)

Operazioni dello use-case:

CERCAOFFERTE (o: OGGETTO) : OFFERTA [0..\*]

CONTROLLASCHEDA (o: OGGETTO)

CERCAOGGETTI (c: CATEGORIA, t: TAG [0..\*]) : (OGGETTO, VALUTA) [0..\*]

REGISTRAZIONE :

NEWUTENTE (n: STRINGA) : UTENTE

**2 Specifica use-case:** GESTIONEWISHLIST..... (nome use-case)

Operazioni dello use-case:

ADDOGGETTO (w: WISHLIST, o: OGGETTO)

NEWWISHLIST (n: STRINGA) : WISHLIST

VEDIWISHLIST (u: UTENTE) : WISHLIST [0..\*]

**3 Specifica use-case:** GESTIONE...ACQUISTI..... (nome use-case)

Operazioni dello use-case:

NEWACQUISTO (o: OGGETTO, b: BUONOREGALO [0..\*], c: CARTA [0..1], num: INTERO>0  
(u: UTENTE, n: NAZIONE) : ACQUISTO

ADDOGGETTO (a: ACQUISTO, o: OGGETTO, n: INTERO>0)

**4 Specifica use-case:** GESTIONE PAGAMENTO ..... (nome use-case)

Operazioni dello use-case:

NEW BUONO ( $i: VALUTA, m: DATAORA, f: DATAORA$ ): BUONO REGALO

NEW CARTA ( $t: STRINGA, c: CODCARTA, d: DATA$ ): CARTA

**5 Specifica use-case:** GESTIONE AMICI ..... (nome use-case)

Operazioni dello use-case:

INVIA RICHIESTA ( $u1: UTENTE, u2: UTENTE$ ): RICHIESTA

ACCETTA RICHIESTA ( $r: RICHIESTA$ ): AMICIZIA

R-INTA RICHIESTA ( $r: RICHIESTA$ ): RICRIF

**6 Specifica use-case:** GESTIONE OFFERTE ..... (nome use-case)

Operazioni dello use-case:

NEW OFFERTA ( $o: OGGETTO, n: NEGOTIO, p: VALUTA, f: DATAORA[0..1]$ ): OFFERTA

NEW SPED ( $o: OFFERTA, p: VALUTA, n: NAZIONE, i: INTERSO, f: INTERSO[0..1]$ ): SPEDIZIONE

SEGNALA ERRORE ( $o: OGGETTO, o2: OGGETTO$ ): OGGETTO

**7 Specifica use-case:** STATISTICHE ..... (nome use-case)

Operazioni dello use-case:

PAESE MAX ACQUISTI ( $i: DATAORA, f: DATAORA$ ): NAZIONE [0..\*]

OGGETTI MODA (): OGGETTO [0..\*]

**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

PAESEMA X ACQUISTI ( $i: DATAORA, f: DATAORA$ ) : (NAZIONE, INTEROZO) [0..\*]

- pre:  $i \leq f$

- post:

$$P = \left\{ (p, n) \mid NAZIONE(p) \wedge n = \left\{ a \mid \exists m, d \quad ACQAZ(m, a) \wedge ISTACQ(d, a) \wedge i \leq d \leq f \right\} \right\}$$

$$\text{RESULT} = \underset{(p, n) \in P}{\text{ARGMAX}}(n)$$

CERCAOGGETTI (C: CATEGORIA, T: TAG [0..\*], n: NAZIONE, num: INTEROZO) : (OGGETTO, VALUTA) [0..\*]

- pre:

- post:

$$\text{RESULT} = \left\{ (o, p) \mid \text{CATOGG}(o, c) \wedge (\forall t \in T \quad \text{TACOGG}(t, o)) \wedge \text{PRMIN}(o, n, num, p) \right\}$$

OGGETTI MODA () : OGGETTO [0..\*]

- pre:

- post:

$$O = \left\{ (o, p) \mid \begin{array}{l} \text{OGGETTO}(o) \wedge m_1 = \text{SOMMAMESE}(o, \text{ADESSO}-1\text{ MESE}, \text{ADESSO}) \wedge m_2 = \text{SOMMAMES}(o, \text{ADESSO}-2\text{ MESI}, \text{ADESSO}-1\text{ MESE}) \\ \wedge m_3 = \text{SOMMAMESE}(o, \text{ADESSO}-3\text{ MESI}, \text{ADESSO}-2\text{ MESI}) \wedge p = \frac{m_1}{\frac{m_2+m_3}{2}} \end{array} \right\}$$

$$\text{RESULT} = \left\{ o' \mid \exists (o, p) \in \text{ARGMAX}(p) \right. \\ \left. (o, p) \in O \right\}$$

**Risposta alla Domanda 5 (segue)**

$\text{SOMMAMESSE}(o; \text{OGGETTO}, i; \text{DATA}, f; \text{DATA}): \text{INTERO} \geq 0$

- pre; i ≤ f

- post:

$$A = \left\{ (a, n) \mid \begin{array}{l} \text{ACQOCC}(a, o) \wedge \text{NUM}(a, o, n) \wedge \exists ia, d \text{ISTACQ}(ia, a) \wedge \\ \text{DATA}(d, ia) \wedge d \geq i \wedge d \leq f \end{array} \right\}$$

$$\text{RESULT} = \sum_{(a, n) \in A} n$$

## 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** ....

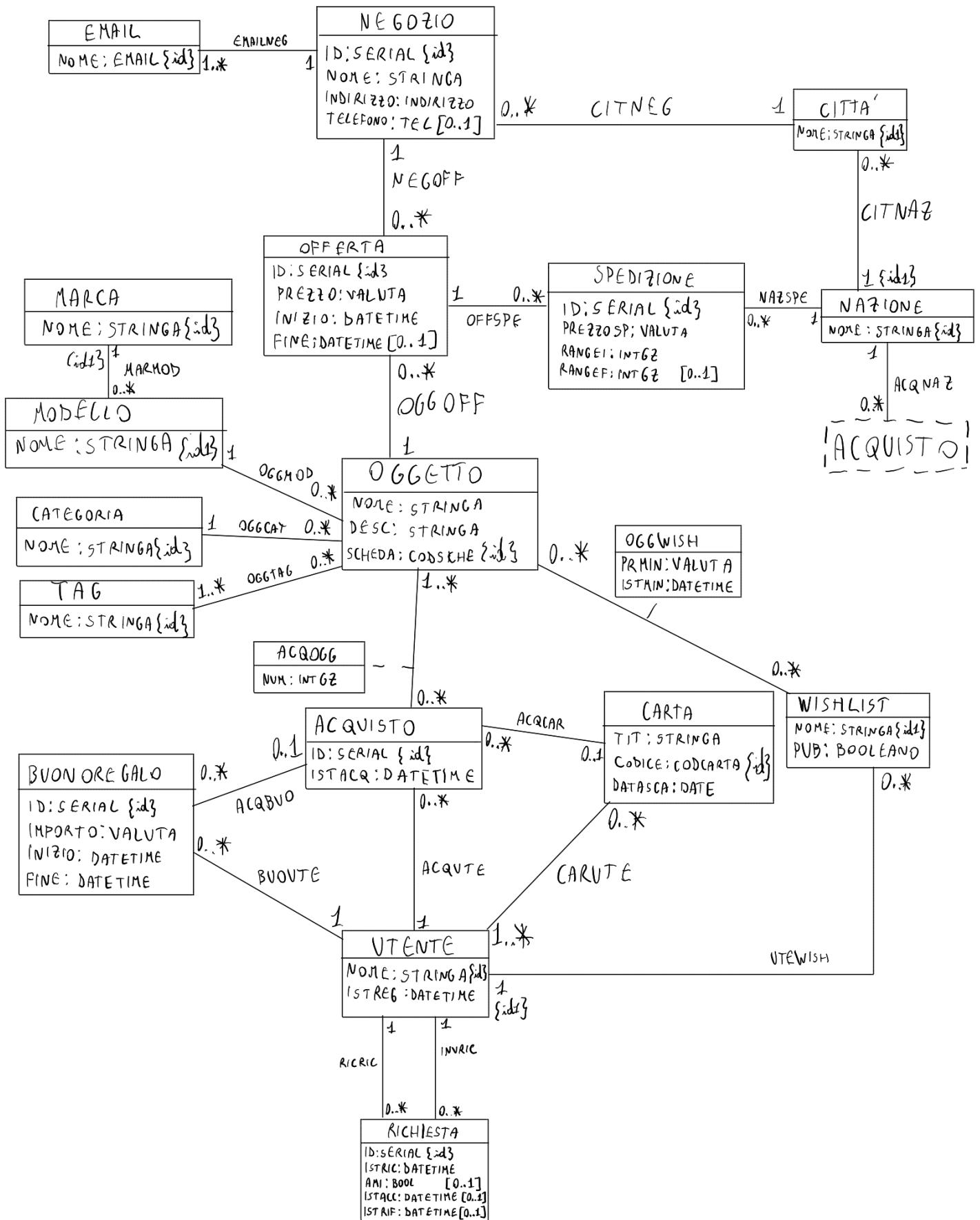
**Corrispondenza tra domini concettuali e domini supportati dal DBMS**

```

CREATE DOMAIN STRINGA AS VARCHAR NOT NULL;
CREATE DOMAIN INTGEZ AS INTEGER CHECK(VALUE >= 0);
CREATE DOMAIN EMAIL AS VARCHAR;
CREATE TYPE INDIRIZZO (
    VIA STRINGA
    CIVICO VARCHAR
    CAP CHAR(5));
CREATE DOMAIN CAP AS VARCHAR ILIKE '[0-9]{5}';
CREATE DOMAIN VALUTA AS REAL CHECK(VALUE >= 0);
CREATE DOMAIN CODCARTA AS VARCHAR ILIKE '[0-9]{16}';
CREATE DOMAIN CODSCHE AS VARCHAR;
CREATE DOMAIN TEL AS VARCHAR ILIKE '[0-9]{10}';
CREATE DOMAIN INTGEZ AS INTEGER CHECK(VALUE >= 0);

```

## Diagramma ER ristrutturato



**Breve descrizione delle scelte effettuate durante la ristrutturazione**

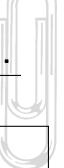
**Vincoli esterni introdotti o modificati durante la fase di ristrutturazione**  
(si omettano i vincoli esterni la cui formulazione è rimasta identica a seguito della ristrutturazione)

$$\forall m, i \text{ RICHIESTA}(m) \wedge \text{ISTRIC}(i, m) \wedge \text{AMI}(m, \text{true}) \leftrightarrow$$

$$[ \exists a \text{ ISTACC}(ia, m) \wedge ia > i ]$$

$$\forall m, i \text{ RICHIESTA}(m) \wedge \text{ISTRIC}(i, m) \wedge \text{AMI}(m, \text{false}) \leftrightarrow$$

$$[ \exists m \text{ ISTRIF}(im, m) \wedge im > i ]$$



**Risposta alla Domanda 6 (segue)**

**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.

<b>1 Relazione</b> <u>UTENTE</u> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi <u>NOME</u>  ISTRREG	
Domini  STRINGA DATETIME	

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>2 Relazione</b> <u>RICHIESTA</u> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   <u>ID</u>  ISTRIC AMI* ISTACC* ISTRIF* UTEINV UTERIC	
Domini  SERIAL DATETIME BOOL DATETIME DATETIME STRINGA STRINGA	

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): CHECK(UTEINV <> UTERIC)

FK(UTEINV) REF UTENTE(NOME) CHECK((AM>TRUE & ISTACC <> NULL) V (AM<>TRUE & ISTACC = NULL)) CHECK(ISTRIC < ISTACC)

FK(UTERIC) REF UTENTE(NOME) CHECK((AM>FALSE & ISTRIF <> NULL) V (AM<>FALSE & ISTRIF = NULL)) CHECK(ISTRIC < ISTRIF)

La relazione accorda le relazioni che implementano le seguenti relationship: ....INVRIC, RICRIC.....

<b>3 Relazione</b> <u>CARTA</u> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   <u>TIT</u>  CODICE DATASCA	
Domini  STRINGA CODCARTA DATE	

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):

CODICE ⊂ CARUTE(CARTA)

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

<b>4 Relazione</b> <u>CARUTE</u> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   <u>UTENTE</u>  CARTA	
Domini  STRINGA CODCARTA	

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):

FK(UTENTE) REF UTENTE(NOME)

FK(CARTA) REF CARTA(CODICE)

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

<b>5 Relazione</b> <u>BUONOREGALO</u> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   <u>ID</u>  IMPORTO INIZIO FINE UTENTE ACQ*	
Domini  SERIAL VALUTA DATETIME DATETIME STRINGA 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):

FK(UTENTE) REF UTENTE(NOME) CHECK(INIZIO < FINE)

FK(ACQ) REF ACQVISTO(ID)

La relazione accorda le relazioni che implementano le seguenti relationship: ...BUONUTE,...ACQBUO...

<b>6 Relazione .ACQV.I.S.T.O..</b> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   <u>ID</u>   LISTACQ   CARTA*   NAZ   UTENTE	
Domini   SERIAL   DATETIME   CODCARTA   STRINGA   STRINGA	

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):

FK(CARTA) REF CARTA(CODICE)      FK(NAZ) REF NAZIONE(NOME)  
 FK(UTENTE) REF UTENTE(NOME)      ID ⊆ ACQOGG(ACQ)  
 La relazione accorda le relazioni che implementano le seguenti relationship: ...ACQCAR.ACQNAZ....

<b>7 Relazione .OGGETTO....</b> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   NOME   DESC   SCHEDA   MODELLO   MARCA   CATEGORIA	
Domini   STRINGA   STRINGA   CODSCHEDA   STRINGA   STRINGA   STRINGA	

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):

FK(CATEGORIA) REF CATEGORIA(NOME)      SCHEDA ⊆ OGGETTO(OGGETTO)  
 FK(MODELLO, MARCA) REF MARCA(NOME, MODELLO)  
 La relazione accorda le relazioni che implementano le seguenti relationship: ...CATOGG.CAT.MAR.....

<b>8 Relazione .ACQ.OG.G....</b> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   OGGETTO   ACQ   NUM	
Domini   CODSCHEDA   INTEGER   INT62	

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):

FK(OBJECTO) REF OGGETTO(SCHEDA)  
 FK(ACQ) REF ACQUISTO(ID)  
 La relazione accorda le relazioni che implementano le seguenti relationship: .....

<b>9 Relazione .WISHLIST...</b> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   NOME   PUB   UTENTE	
Domini   STRINGA   BOOL   STRINGA	

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):

FK(UTENTE) REF UTENTE(NOME)  
 La relazione accorda le relazioni che implementano le seguenti relationship: .....

<b>10 Relazione .OGGETTO.WISH...</b> (nome)	Derivante da: <b>entità</b>   relationship (cerchiare)
Attributi   OGGETTO   WISH   UTENTE   PRMIN   STMIN	
Domini   CODSCHEDA   STRINGA   STRINGA   VALUTA   DATETIME	

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):

FK(OBJECTO) REF OGGETTO(SCHEDA)  
 FK(WISH, UTENTE) REF WISHLIST(NOME, UTENTE)  
 La relazione accorda le relazioni che implementano le seguenti relationship: .....

**[11] Relazione ...TA.6.....(nome)** Derivante da: **entità** | **relationship** (cerchiare)

Attributi	<u>NOME</u>						
Domini	<u>ISTRINGA</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: .....

**[12] Relazione ...OGGETTO...(nome)** Derivante da: **entità** | **relationship** (cerchiare)

Attributi	<u>TAG</u>	<u>OGGETTO</u>					
Domini	<u>ISTRINGA</u>	<u>CODSCHE</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):

$FK(TAG) \text{ REF } TAG(NOME)$

$FK(OGGETTO) \text{ REF } OGGETTO(SCHEDA)$

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

**[13] Relazione CATEGORIA...(nome)** Derivante da: **entità** | **relationship** (cerchiare)

Attributi	<u>NOME</u>						
Domini	<u>ISTRINGA</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: .....

**[14] Relazione ...MODELLO...(nome)** Derivante da: **entità** | **relationship** (cerchiare)

Attributi	<u>NOME</u>	<u>MARCA</u>					
Domini	<u>ISTRINGA</u>	<u>ISTRINGA</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):

$FK(MARCA) \text{ REF } MARCA(NOME)$

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

**[15] Relazione ...MARCA.....(nome)** Derivante da: **entità** | **relationship** (cerchiare)

Attributi	<u>NOME</u>						
Domini	<u>ISTRINGA</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: .....

16 Relazione .OFFERTA.(nome)	Derivante da: entità   relationship (cerchiare)
------------------------------	---

Attributi   <u>ID</u>   PREZZO   INIZIO   FINE *   NEGOZIO   OGGETTO	
Domini   SERIAL   VALUTA   DATETIME   DATETIME   INTEGER   COSSCHE	

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):

FK(NEGOZIO) REF NEGOZIO(ID) CHECK(INIZIO<FINE)

FK(OBJECTO) REF OBJECTO(SCHEMA)

La relazione accorda le relazioni che implementano le seguenti relationship: .OFFERG,.OFFNEG.....

17 Relazione ...NEGOTIO.(nome)	Derivante da: entità   relationship (cerchiare)
--------------------------------	---

Attributi   <u>ID</u>   NOME   INDIRIZZO   TELEFONO *   CITTÀ   NAZIONE	
Domini   SERIAL   STRINGA   INDIRIZZO   TEL   STRINGA   STRINGA	

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):

FK(CITTÀ,NAZIONE) REF CITTÀ(NOME,NAZIONE)

ID ⊂ EMAIL(NEGOZIO)

La relazione accorda le relazioni che implementano le seguenti relationship: ...CITNE6.....

18 Relazione ...EMAIL.....(nome)	Derivante da: entità   relationship (cerchiare)
----------------------------------	---

Attributi   <u>NOME</u>   NEGOZIO	
Domini   EMAIL   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):

FK(NEGOZIO) REF NEGOZIO(ID)

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

19 Relazione SPEDIZIONE.(nome)	Derivante da: entità   relationship (cerchiare)
--------------------------------	---

Attributi   <u>ID</u>   PREZZO   RANGEI   RANGEF *   NAZIONE   OFFERTA	
Domini   SERIAL   VALUTA   INT 62   INT 62   STRINGA   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):

FK(NAZIONE) REF NAZIONE(NOME) CHECK(RANGEI<RANGEF)

FK(OFFERTA) REF OFFERTA(ID)

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

20 Relazione ...CITTÀ.....(nome)	Derivante da: entità   relationship (cerchiare)
----------------------------------	---

Attributi   <u>NOME</u>   NAZIONE	
Domini   STRINGA   STRINGA	

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: ...CITNAZ.....

21 RELAZIONE: NAZIONE

ENTITÀ

ATRIBUTI   <u>NOME</u>	
DOMINI   STRINGA	

[continua a pagina ..... delle minute]

**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.

**T. OFFERTA . DISGIUNTE**

- INSERT OR UPDATE OFFERTA

- ERROR = EXISTS (SELECT \* FROM OFFERTA  
 WHERE OGGETTO = NEW.OGGETTO AND NEGOZIO = NEW.NEGOZIO  
 AND TSRANGE(INIZIO, FINE) \* TSRANGE(NEW, INIZIO, NEW, FINE) <= EMPTY')

**T. SPEDIZIONE . DISGIUNTE**

- INSERT OR UPDATE SPEDIZIONE

- ERROR = EXIST (SELECT \* FROM SPEDIZIONE  
 WHERE OFFERTA = NEW.OFFERTA AND NAZIONE = NEW.NAZIONE  
 AND INT4RANGE(INIZIO, FINE) \* INT4RANGE(NEW, INIZIO, NEW, FINE) <= EMPTY')

REVOKE UPDATE ON `(ACQUISTO)`

**T. ACQUISTO, CARTA BUONO POSSESSUTI**

- INSERT ACQUISTO, BUONO REGALO, CARUTE

- ERROR = EXISTS (SELECT \* FROM ACQUISTO a, BUONO REGALO b, CARTA c  
 WHERE a.ID = b.ACQ AND c.CARTA = a.CARTA  
 AND (b.UTE < a.UTE OR c.UTE < a.UTE)) OR  
 EXIST (SELECT \* FROM ACQUISTO a, BUONO REGALO b, CARTA c  
 WHERE b.ACQ = a.ID AND a.CARTA = c.CODICE  
 AND (b.INIZIO > a.ISTACQ OR b.FINE < a.ISTACQ) OR c.DATASCAD < a.ISTACQ)

**T. ACQUISTO, DOPOREG**

- INSERT ACQUISTO

- ERROR = EXIST (SELECT \* FROM UTENTE  
 WHERE NOME = NEW.UTENTE AND ISTREG > NEW.ISTACQ)

**Risposta alla Domanda 7 (segue)**

T. RICHIESTA. DOPOREG

- INSERT OR UPDATE RICHIESTA
- OK = EXIST (SELECT \* FROM UTENTE u1, UTENTE u2  
WHERE u1.NOME = NEW.VTEINV AND u2.NOME = NEW.UTERIC  
AND u1.ISTRREG < NEW.ISTRIC AND u2.ISTRREG < NEW.ISTRIC)

T. RICHIESTA. NODOPPIA

- INSERT OR UPDATE RICHIESTA
- ERROR = EXIST (SELECT \* FROM RICHIESTA  
WHERE VTEINV = NEW.UTERIC AND VTERIC = NEW.VTEINV)

T. ACQUISTO. FATTIBILE

- INSERT ACQUISTO
- OK = (SELECT SUM(IMPORTO) FROM BUONOREGALO  
WHERE ACQ = NEW.ID) > PRTOT(NEW)  
OR (NEW.CARTA <> NULL)

T. WISHLIST. DOPOISCRIZIONE

- INSERT OR UPDATE OGGWISH
- ERROR = EXIST (SELECT \* FROM UTENTE  
WHERE NEW.UTENTE = NONE  
AND NEW.ISTMING < ISTRREG)

**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

PRTOT(acq : INTEGER): VALUTA

R = WITH ACQDATA AS (SELECT \* FROM ACQUISTO WHERE ID = acq)  
 SELECT SUM(PRMIN) FROM ACQDATA, (SELECT OGGETTO, PRMIN(OGGETTO, ACQDATA.NAZIONE, NUM) PRMIN FROM ACQOGG WHERE ACQ = acq)

RETURN R

PRMIN(o: INTEGER, maz: STRINGA, n: INTEGER): VALUTA

R = SELECT MIN(aff.PREZZO \* n + s.PREZZO SP) FROM OFFERTA aff, SPEDIZIONI s  
 WHERE aff.ID = s.OFFERTA AND aff.OGGETTO = o AND s.NAZIONE = maz  
 AND s.RANGE1 ≤ n AND s.RANGE2 ≥ n

RETURN R

PAESE MAX ACQUISTI(i: TIMESTAMP, f: TIMESTAMP): (STRINGA, INTEGER)[0..\*]

ERROR IF i > f

R = WITH Q AS (SELECT NAZIONE, COUNT(ID) NACQ FROM ACQUISTO  
 WHERE ISTACQ ≥ i AND ISTACQ ≤ f  
 GROUP BY NAZIONE)  
 SELECT \* FROM Q  
 WHERE NACQ = (SELECT MAX(NACQ) FROM Q)

RETURN R

**Risposta alla Domanda 8 (segue)**

CERCAOGGETTI(C:STRINGA,T:STRINGA[0..\*],n:STRINGA,mun:INTEGER):(**INTEGER**,VALUTA)[0,\*]  
 R:=SELECT DISTINCT ID,PRMIN(ID,n,mun) FROM OGGETTO  
 WHERE CATEGORIA=C AND NOT EXISTS(  
 SELECT ID,T FROM T  
 EXCEPT SELECT \* FROM OGGETAG)  
 RETURN R

OGGETTI MODA();**INTEGER**[0..\*]

R:=WITH M1 AS(SELECT ar.OGGETTO,SUM(NUM) SOMMA FROM ACQOGG ar,ACQUISTO a  
 WHERE ar.ACQ=a.ID  
 AND a.ISTACQ>NOW()-INTERVAL '1 MONTH'  
 GROUP BY ar.OGGETTO),

M2 AS (SELECT ar.OGGETTO,SUM(NUM) SOMMA FROM ACQOGG ar,ACQUISTO a  
 WHERE ar.ACQ=a.ID  
 AND a.ISTACQ>NOW()-INTERVAL '2 MONTH'  
 AND a.ISTACQ<NOW()-INTERVAL '1 MONTH'  
 GROUP BY ar.OGGETTO),

M3 AS (SELECT ar.OGGETTO,SUM(NUM) SOMMA FROM ACQOGG ar,ACQUISTO a  
 WHERE ar.ACQ=a.ID  
 AND a.ISTACQ>NOW()-INTERVAL '3 MONTH'  
 AND a.ISTACQ<NOW()-INTERVAL '2 MONTH'  
 GROUP BY ar.OGGETTO)

SELECT OGGETTO FROM M1,M2,M3

WHERE M1.OGGETTO=M2.OGGETTO AND M2.OGGETTO=M3.OGGETTO

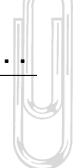
AND M1.SOMMA/((M2.SOMMA+M3.SOMMA)/2)=

SELECT MAX(M1.SOMMA/((M2.SOMMA+M3.SOMMA)/2)) FROM M1,M2,M3

WHERE M1.OGGETTO=M2.OGGETTO AND M2.OGGETTO=M3.OGGETTO)

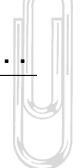
RETURN R

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.]