Sapienza Università di Roma
Facoltà di Ing. dell'Informazione, Informatica e Statistica, Laurea in Informatica
Insegnamento di **Basi di Dati, Modulo 2**
Prof. Toni Mancini
Dipartimento di Informatica
http://tmancini.di.uniroma1.it

Esame **BD2.Esame.Risposte.ER – Modulo risposte prova scritta**

Dati dello studente e dell'esame

Cognome e nome: Caso Marco Matricola: .....

Data: 12/06/2024

Corso di laurea e canale di appartenenza:

- ☐ Laurea in Informatica, canale 1 (A-L, Prof. G. Perelli)
- ☐ Laurea in Informatica, canale 2 (M-Z, Prof.ssa M. De Marsico)
- ☐ Laurea in Informatica in Modalità Teledidattica Unitelma Sapienza

Firma di un membro della Commissione per avvenuta identificazione:

.....

Rinuncia alla prova

☐ Desidero rinunciare a questa prova d'esame. Firma: .....

TimeBank

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

moltiplicatore (d: calcolato in un dato istante

1. Utente — i — 2. Abilità
   2.1 utente che la fornisce
1.2 nome
1.3 saldo() → Scambi offerti - richiesti (
1.4 cognome
1.1 email
1.5 cell
abilità erogate
scambi
saldo bonus: x

3. Scambio
   3.1 utente richiedente
   3.2 abilità
   3.3 utente fornitore
   3.6 DataOra
   3.5 Durata } il valore dipende
   3.6 Valore } dalla durata
   3.7 Feedback [0..1]

saldo(d:Ist) = saldo dell'utente nell'ist. d

Offre
molt:
Utente — i — Abilità

V. utente richiede scambio
se non è in rosso

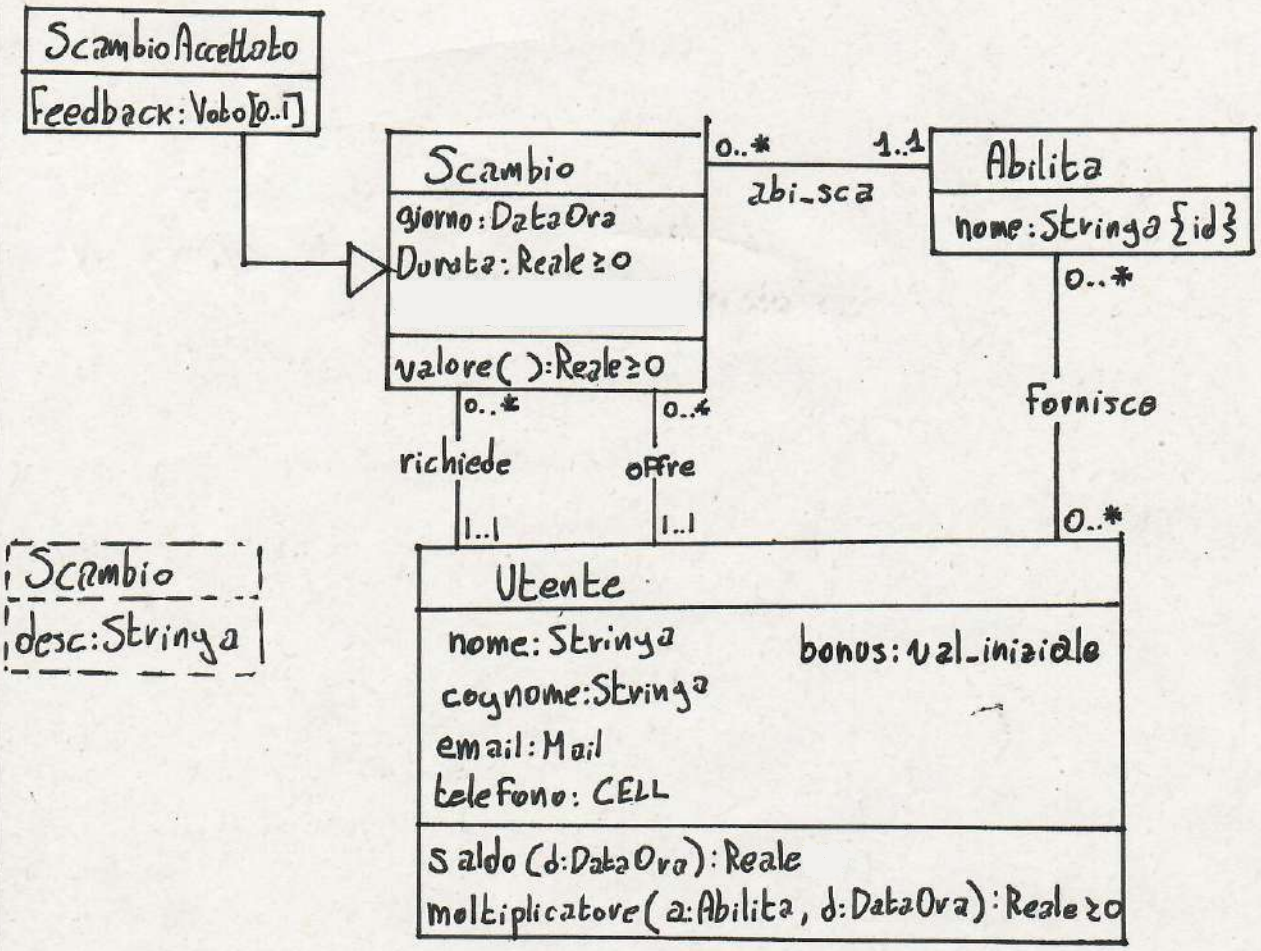
[continua alla pagina seguente]

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.



Un Utente può':

- richiedere ed offrire contemporaneamente
- richiedere 2 scambi contemporaneamente

Non può:

- offrire 2 scambi contemporaneamente

[continua alla pagina seguente]

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 $r(x_1, \ldots, x_n)$: $x_1, \ldots, x_n$ 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 $a(x_1, \ldots, x_n, v)$: uno dei valori dell'attr. a dell'istanza $(x_1, \ldots, x_n)$ 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

1 Tipo: (Entità) | **Relationship** (cerchiare)
Nome: Utente

attributo	dominio	moltepl. (*)

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

Vincoli:

[V.saldo_positivo]
$\forall u, d, v\ [Utente(u) \wedge DataOra(d) \wedge saldo(u, d, v)] \rightarrow v \geq 0$

[V.offre_scambio]
$\forall u, s, a\ [Utente(u) \wedge offre(s, u) \wedge abi\_sca(a, s)] \rightarrow fornisce(a, u)$

2 Tipo: (Entità) | **Relationship** (cerchiare)
Nome: Utente

attributo	dominio	moltepl. (*)

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

Vincoli:

[V.no_auto_scambi]
$\forall u, s\ [Utente(u) \wedge Scambio(s) \wedge offre(s, u)] \rightarrow \neg richiede(s, u)$

3 Tipo: (Entità) | Relationship (cerchiare)

Nome: Utente

attributo	dominio	moltepl. (*)

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

Vincoli:

[V.no-2-scambi-contemporaneamente]

$\forall u, s1, s2, g1, g2, d1, d2$

$[Utente(u) \wedge s1 \neq s2 \wedge$

$\wedge offre(s1, u) \wedge offre(s2, u) \wedge giorno(s1, g1)$

$\wedge giorno(s2, g2) \wedge durata(s1, d1) \wedge$

$durata(s2, d2)] \rightarrow$

$[g1 + d1 < g2 \vee g2 + d2 < g1]$

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

Nome: .....

attributo	dominio	moltepl. (*)

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

Vincoli:

$(ScambioAccettato(s1) \wedge ScambioAccettato(s2))$

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

Nome: .....

attributo	dominio	moltepl. (*)

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

Vincoli:

6 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

moltiplicatore(a: Abilita, d: DataOra): Reale ≥ 0

- pre-cond: Fornisce(a, this)
- post-cond: $S = \{(s,f) \mid offre(this,s) \wedge \exists g\ giorno(s,g) \wedge g < d \wedge feedback(s,f) \wedge abi\_sca(a,s)\}$

Result è tale che:

$|S| = 0 \rightarrow Result = 1 \wedge$

$|S| \neq 0 \rightarrow Result = 1 + \arctan\left(0.05 * \sum_{(s,f)\in S} f \cdot \frac{1}{|S|}\right)$

valore( ): Reale ≥ 0

- pre-cond: nessuna
- post-cond:

Sia d tale che: durata(this, d)
Sia g tale che: giorno(this, g)
Sia a tale che: abi_sca(a, this)
Sia u tale che: offre(this, u)
Sia m tale che: moltiplicatore(u, a, g, m)

Result = d · m

saldo(d: DataOra): Reale

- pre-cond: nessuna
- post-cond: $R = \{(s,v) \mid richiede(s,this) \wedge \exists g\ giorno(s,g) \wedge g \leq d \wedge valore(s,v) \wedge ScambioAccettato(s)\}$

$O = \{(s,v) \mid offre(s,this) \wedge \exists g\ giorno(s,g) \wedge g \leq d \wedge valore(s,v) \wedge ScambioAccettato(s)\}$

$Result = \left(\sum_{(s,v)\in O} v\right) - \left(\sum_{(s,v)\in R} v\right) + x$

↑ saldo al momento dell'iscrizione

x := bonus(this, x)

[continua alla pagina seguente]

Risposta alla Domanda 2 (segue)

Tipi di Dato

val_iniziale:= ~~Intero~~ Reale = x

Voto = 1..10

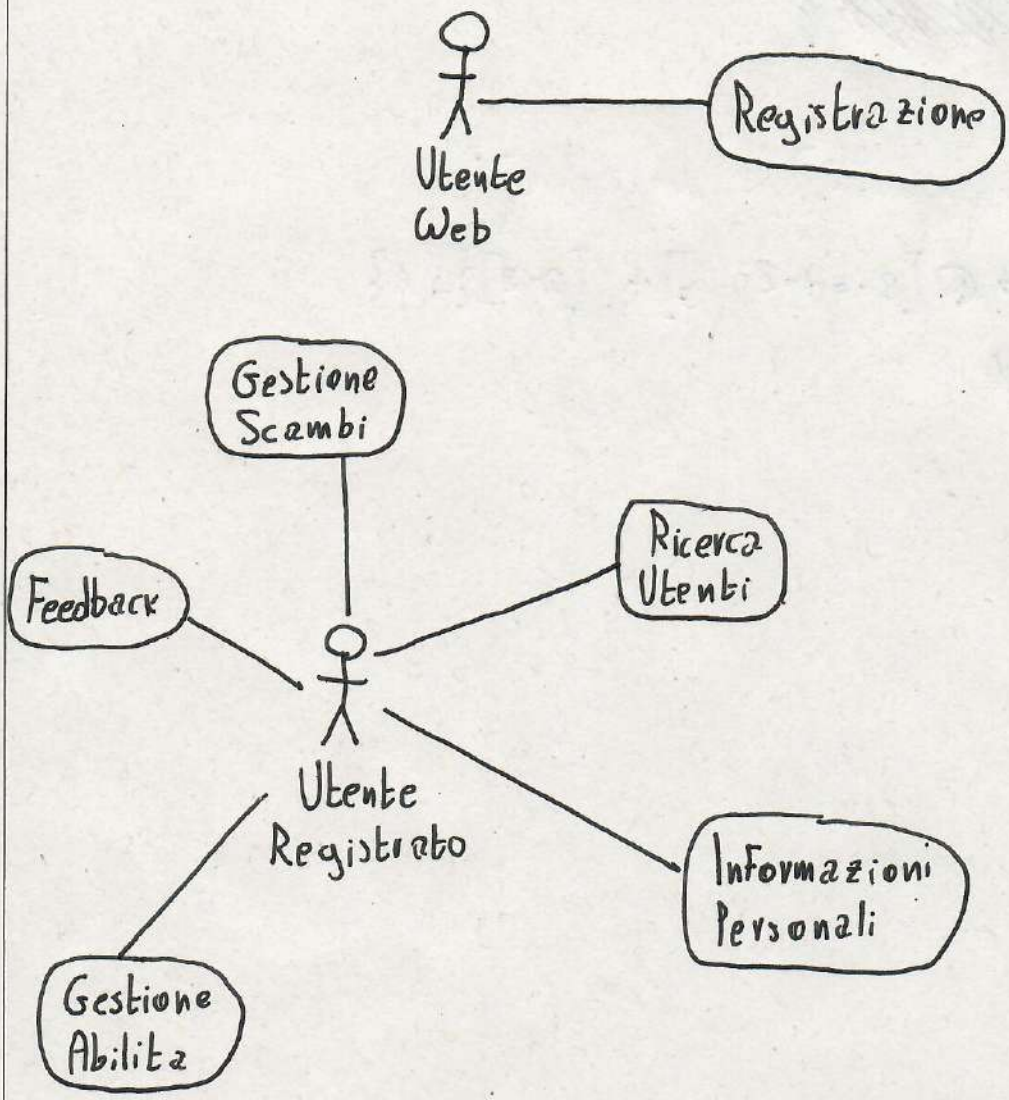
CELL = [0-9]{12} // Prefisso{2} e numero{10}

Mail = [a-zA-Z0-9]+ @ [a-zA-Z0-9]+ . [a-z]{2,6}

Real_TEN = 1 ≤ Real ≤ 10

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

Operazioni dello use-case:

registrati(n:Stringa, c:Stringa, e:Mail, t:CELL):Utente

2 Specifica use-case: Gestione Abilità ..... (nome use-case)

Operazioni dello use-case:

aggiungi_abilita(a:Abilita)

rimuovi_abilita(a:Abilita)

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

Operazioni dello use-case:

lascia_Feedback(s:ScambioAccettato, v:Voto):ScambioAccettato

4 Specifica use-case: Gestione Scambi (nome use-case)

Operazioni dello use-case:

accetta_scambio(s:Scambio):ScambioAccettato

richiedi_scambio(u:Utente, a:Abilita, g:DataOra, d:Reale≥0):Scambio
desc:Stringa

5 Specifica use-case: Ricerca Utenti (nome use-case)

Operazioni dello use-case:

ottieni_abilita(u:Utente):Abilita[0..*]

utenti_qualificati(a:Abilita, m:Real_TEN, d:DataOra):Utente[0..*]

6 Specifica use-case: Informazioni Personali (nome use-case)

Operazioni dello use-case:

calcola_saldo(u:Utente, d:DataOra):Real_GEZ

calcola_molt(u:Utente, d:DataOra, a:Abilita):Real≥0

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

Operazioni dello use-case:

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

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

calcola_molt(u:Utente, d:DataOra, z:Abilita): Real≥0

- pre-cond: fornisce(z,u)
- post-cond: Result è tale che := moltiplicatore(u,z,d,Result)

---

utenti_qualificati(a:Abilita, m:Real_TEN, d:DataOra):Utente[0..*]

- pre-cond: nessuna
- post-cond: $U=\{u \mid fornisce(z,u) \wedge \exists k \ \ media\_utente(u,z,d,k) \wedge k \geq m\}$

  Result = U

[continua alla pagina seguente]

Risposta alla Domanda 5 (segue)

media_utente(u:Utente, a:Abilita, d:DataOra):Real_TEN

- pre-cond: fornisce(a, u)
- post-cond:

$$S=\left\{(s,f)\;\middle|\;\begin{array}{l}offre(u,s)\wedge\exists g\ giorno(s,g)\wedge g\leq d\\ \wedge\ abi\_sca(a,s)\wedge Feedback(s,f)\end{array}\right\}$$

Result è tale che:

$|S|=0 \rightarrow Result=1\ \wedge$

$|S|\neq 0 \rightarrow Result=\sum_{(s,f)\in S} F\cdot\frac{1}{|S|}$

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 multivalore 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.

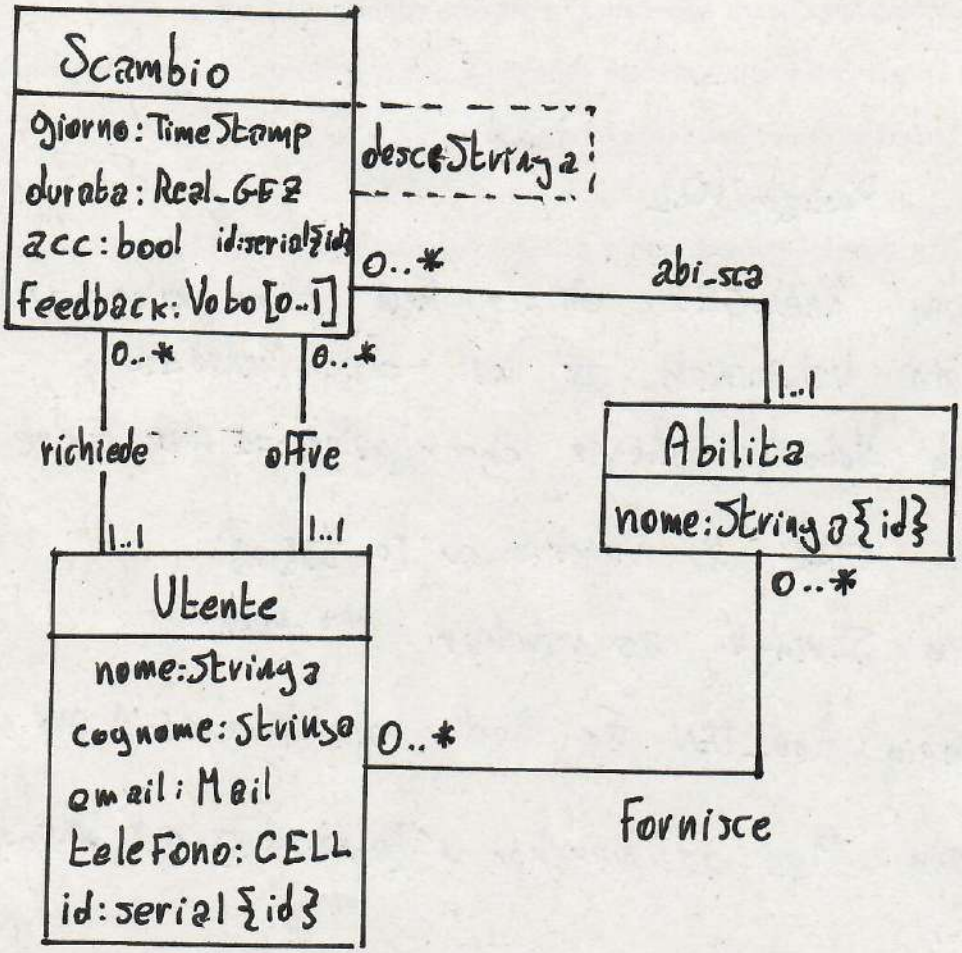
Descrivere brevemente le principali scelte effettuate.

DBMS da utilizzare PostgreSQL

Corrispondenza tra domini concettuali e domini supportati dal DBMS

```
create domain Real_GEZ as Real check(value >= 0);
create domain val_iniziale as Real check (value =x);
create domain Voto as Integer check(value >=1 AND value <=10);
create domain CELL as varchar ~'[0-9]{12}'
create domain Stringa as varchar NOT NULL;
create domain Real_TEN as Real check(value >=1 AND value <=10);
create domain Mail as varchar ~'[a-zA-Z0-9]+@[a-zA-Z0-9]+.[a-z]{2,6}'
```

Diagramma ER ristrutturato



Breve descrizione delle scelte effettuate durante la ristrutturazione

Fusione su Scambio

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. Feedback_se_accettato]

$\forall s \; [Scambio(s) \wedge \exists k \; Feedback(s,k)] \rightarrow acc(s, True)$

[V. no_2_scambi_contemporaneamente]

$\forall u, s1, s2, g1, g2, d1, d2 \; [Utente(u) \wedge s1 \neq s2 \wedge Scambio(s1) \wedge Scambio(s2) \wedge acc(s1, True) \wedge acc(s2, True) \wedge offre(u, s1) \wedge offre(u, s2) \wedge giorno(s1, g1) \wedge giorno(s2, g2) \wedge durata(s1, d1) \wedge durata(s2, d2)] \rightarrow$

$[g1 + d1 < g2 \vee g2 + d2 < g1]$

[continua alla pagina seguente]

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 Utente ..... (nome) Derivante da: (entità) | **relationship** (cerchiare)

Attributi	nome	cognome	email	telefono	id	bonus		
Domini	Stringa	Stringa	Mail	CELL	serial	val_iniziale		

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 accorpa le relazioni che implementano le seguenti relationship: .....

2 Relazione Abilita ..... (nome) Derivante da: (entità) | **relationship** (cerchiare)

Attributi	nome							
Domini	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 accorpa le relazioni che implementano le seguenti relationship: .....

3 Relazione Fornisce ..... (nome) Derivante da: **entità** | (relationship) (cerchiare)

Attributi	Ubente	Abilita						
Domini	Integer	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):

Foreign key abilita references Abilita(nome);

Foreign key ubente references Utente(id);

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

4 Relazione Scambio ..... (nome) Derivante da: (entità) | **relationship** (cerchiare)

Attributi	giorno	durata	id	acc	Feedback*	Abilita	richiede	offre
Domini	TimeStmp	Real_GEZ	serial	bool	Voto	Stringa	Integer	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): check(richiede <> offre);

Foreign key abilita ref Abilita(nome); fk richiede ref Utente(id);

fk offre ref Utente(id);

La relazione accorpa le relazioni che implementano le seguenti relationship: richiede, offre, abi_sco

5 Relazione Scambio ..... (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):

check(Feedback is NULL or (Feedback is NOT NULL AND acc=True));

La relazione accorpa 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.

T.no_2_scambi_contemporaneamente
op: Insert o Update Scambio

```
Error = EXISTS( SELECT
        FROM Scambio s WHERE new.id <> s.id AND s.utente
        = new.utente AND
(s.giorno, s.giorno + cast(s.durata as 'hour')::Timestamp) OVERLAPS
(new.giorno, new.giorno + cast(new.durata as 'hour')::Timestamp));
```

iF Error = True : errore e rollback
else : permetti oper.

T.saldo_positivo
op: Insert o Update Scambio

```
S = SELECT saldo(new. richiede, new.giorno) as sal;
```

iF(S.sal < 0): Errore e vollback
else: permetti oper.

T.offre_scambio
op: Insert o Update Scambio

```
Ok = EXISTS (SELECT * FROM fornisce F WHERE
        F.utente = new. offre AND f.abilita = new. abilita);
```

iF OK = True : permetti op.
else: errore e vollback

[continua alla pagina seguente]

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

```
Create function media(u:Integer, a:Stringa, d:TimeStamp): Real_TEN
E=EXISTS(SELECT * FROM fornisce WHERE abilita=a AND
                                     utente=u);
if(E==False): termina operazione
Q=SELECT AVG(feedback) as m, count(*) as n
   FROM Scambio WHERE offre=u AND giorno<d AND acc=True
   AND abilita=a AND feedback IS NOT NULL;
if(Q.n==0): Result=1
else: Result=Q.m
```

```
create function valore(s:Integer): Real_GEZ
Q=SELECT durata*moltiplicatore(offre, abilita, giorno) as val
   FROM Scambio SC WHERE id=S;
Result=Q.val
```

```
create function moltiplicatore(u:Integer, a:Stringa, d:Timestamp): Real_GEZ
Q=SELECT 1+ARCTAN(0.05*media(u,a,d)) as mol;
Result=Q.mol
```

[continua alla pagina seguente]

Risposta alla Domanda 8 (segue) TimeStamp

saldo(u:Integer, d:~~Data~~):Real

```
Q=WITH  R as (SELECT sum(valore(id)) as ric
              FROM Scambio WHERE richiede=u AND
              giorno <= d AND acc=True)

        O as (SELECT sum(valore(id)) as off
              FROM Scambio WHERE offre=u AND
              giorno<=d AND acc=True)

        SELECT bonus+Q.off-R.ric as saldo
        FROM Utente, R, Q
        WHERE id=u
Result = Q.saldo
```

utenti_qualificati(a:Stringa, m:Real_TEN, d:Timestamp):Insieme(<Integer>)

```
Q = SELECT u.id
    FROM Utente u JOIN fornisce F on u.id=F.utente
    WHERE f.abilita = a AND
    media(u.id, a, d) >= m;
Result = Q
```