



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 – Modulo risposte prova scritta (diagramma delle classi UML)**

**Dati dello studente e dell'esame**

Cognome e nome: ..... Matricola: .....

Data: .....

Corso di laurea e canale di appartenenza:

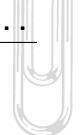
- Laurea in Informatica, canale 1 (Prof. G. Perelli)
- Laurea in Informatica, canale 2 (Prof.ssa M. De Marsico)

Firma di un membro della Commissione per  
avvenuta identificazione:  
.....

**Rinuncia alla prova**

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





# Istruzioni e regole d'esame

## Prima dell'esame

- Stampare questo modulo, preferibilmente fronte-retro, e rilegarlo con un fermaglio rimovibile, come quello disegnato in alto
- Compilare il frontespizio con i propri dati, come richiesto
- Scrivere la propria matricola nello spazio apposito nella parte alta di tutte le pagine

## Durante l'esame

- La prova è dimensionata per essere svolta in circa 3 ore. Tuttavia, data la sua natura fortemente progettuale, la Commissione offre agli studenti la più ampia disponibilità di tempo, al fine ovviare ad eventuali (e limitati) errori di analisi/progettazione rilevati più a valle del ciclo di vita.  
Il tempo massimo per la consegna è quindi rilassato a 5 ore (il massimo tempo compatibile con le disponibilità di aule).
- Scrivere le risposte negli spazi predisposti sotto le relative domande. Le ultime pagine sono vuote e possono essere usate come minute oppure, se puntate opportunamente, per contenere risposte in caso gli spazi appositi dovessero risultare insufficienti.
- Non è possibile usare alcun tipo di materiale didattico.
- In caso di necessità di ulteriori fogli (in proprio possesso), chiedere preventivamente alla Commissione una nuova procedura di controllo.
- La Commissione può rispondere solo a brevi domande inerenti al testo dei quesiti.
- Tra la seconda e la quarta ora d'esame, gli studenti possono effettuare **brevi pause** (uno studente alla volta) seguendo la seguente procedura:
  1. Alla lavagna è riportata una coda denominata 'Coda prenotazioni pause'. Sia  $n$  (un intero) l'elemento in fondo alla coda (si assuma  $n = 0$  in caso di coda vuota).
  2. Recarsi alla lavagna ed aggiungere l'intero  $n + 1$  come proprio contrassegno in fondo alla coda, seguito da una stringa a propria scelta (ad es., le proprie iniziali).
  3. Se il proprio contrassegno non è l'elemento affiorante della coda, tornare al lavoro in attesa che lo diventi.
  4. Consegnare tutti i fogli di lavoro e il testo d'esame alla Commissione ed uscire.
  5. Al rientro, cancellare il proprio contrassegno dalla coda di modo da permettere al successivo studente prenotato di uscire, e riprendere i fogli prima consegnati.

## Al momento della consegna

- Ordinare tutti i fogli che si vuole far valutare e rilegarli con un fermaglio rimovibile. Non includere fogli che la Commissione non deve valutare (ad es., requisiti, minute), ma includere ovviamente il frontespizio.
- Consegnare i fogli ordinati **nelle mani** di un membro della Commissione. **Non lasciare l'aula senza la conferma, da parte della Commissione, del buon esito delle operazioni di consegna.**

## In caso di rinuncia

- È possibile rinunciare alla consegna a partire dalla seconda ora d'esame. In caso di rinuncia, consegnare nelle mani della Commissione solo il frontespizio, dopo aver compilato e firmato la sezione dedicata.

## Sommario delle domande

Si richiede di progettare l'applicazione descritta dalla specifica dei requisiti effettuando le fasi di Analisi concettuale dei requisiti e di Progettazione logica della base dati e delle funzionalità, utilizzando la metodologia vista nel corso.

In particolare (vengono indicati i tempi suggeriti per i diversi passi chiave):

**Parte 1: Analisi concettuale dei requisiti** Effettuare la fase di Analisi concettuale dei requisiti producendo lo schema concettuale per l'applicazione, che includa:

- Analisi dei dati (45 minuti; 75 minuti al massimo):
  - un diagramma UML concettuale delle classi (\*)
  - (parte del)le specifiche formali delle classi e delle associazioni
  - le specifiche dei tipi di dato
  - la specifica formale dei vincoli esterni (\*)
- Analisi delle funzionalità:
  - un diagramma UML degli use-case (5 minuti; 10 minuti al massimo)
  - la segnatura di tutte le operazioni di use-case (10 minuti)
  - (parti del)le specifiche formali degli use-case. (30 minuti; 60 minuti al massimo)

Si richiede *esplicitamente* di modellare le specifiche formali delle operazioni di classe e/o use-case necessarie a modellare i requisiti contrassegnati dalla barra laterale (come quella qui a sinistra), *inclusa* tutte le eventuali operazioni ausiliarie, usando l'estensione della logica del primo ordine studiata nel corso. (\*)

**Parte 2: Progettazione della base dati e delle funzionalità** Effettuare la progettazione della base dati e delle funzionalità a partire dallo schema concettuale prodotto nella Parte 1, ed in particolare eseguire i seguenti passi:

- Progettazione della base dati relazionale con vincoli:
  - Ristrutturazione del diagramma UML concettuale delle classi e delle specifiche (20 minuti; 30 minuti al massimo):
    - \* scelta del DBMS da utilizzare
    - \* progettazione della corrispondenza tra i tipi di dato concettuali ed opportuni domini SQL (domini base o utente, oppure realizzati mediante relazioni aggiuntive) supportati dal DBMS scelto
    - \* ristrutturazione del diagramma UML concettuale delle classi e delle specifiche dei vincoli esterni.
  - Produzione dello schema relazionale della base dati e dei relativi vincoli (\*) (30 minuti; 60 minuti al massimo)
- Progettazione delle funzionalità (30 minuti; 45 minuti al massimo):
  - definizione della specifica realizzativa delle operazioni necessarie a modellare i requisiti contrassegnati dalla barra laterale, in modo conforme alla loro specifica concettuale prodotta nella fase di Analisi, in termini di algoritmi in pseudo-codice e comandi SQL immersi. (\*)

Le pagine seguenti contengono le domande specifiche a cui è richiesto rispondere, ulteriori delucidazioni per ogni singolo punto, e spazi per le risposte.

Le pagine da 31 in poi possono essere utilizzate per scrivere minute che non verranno valutate.

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

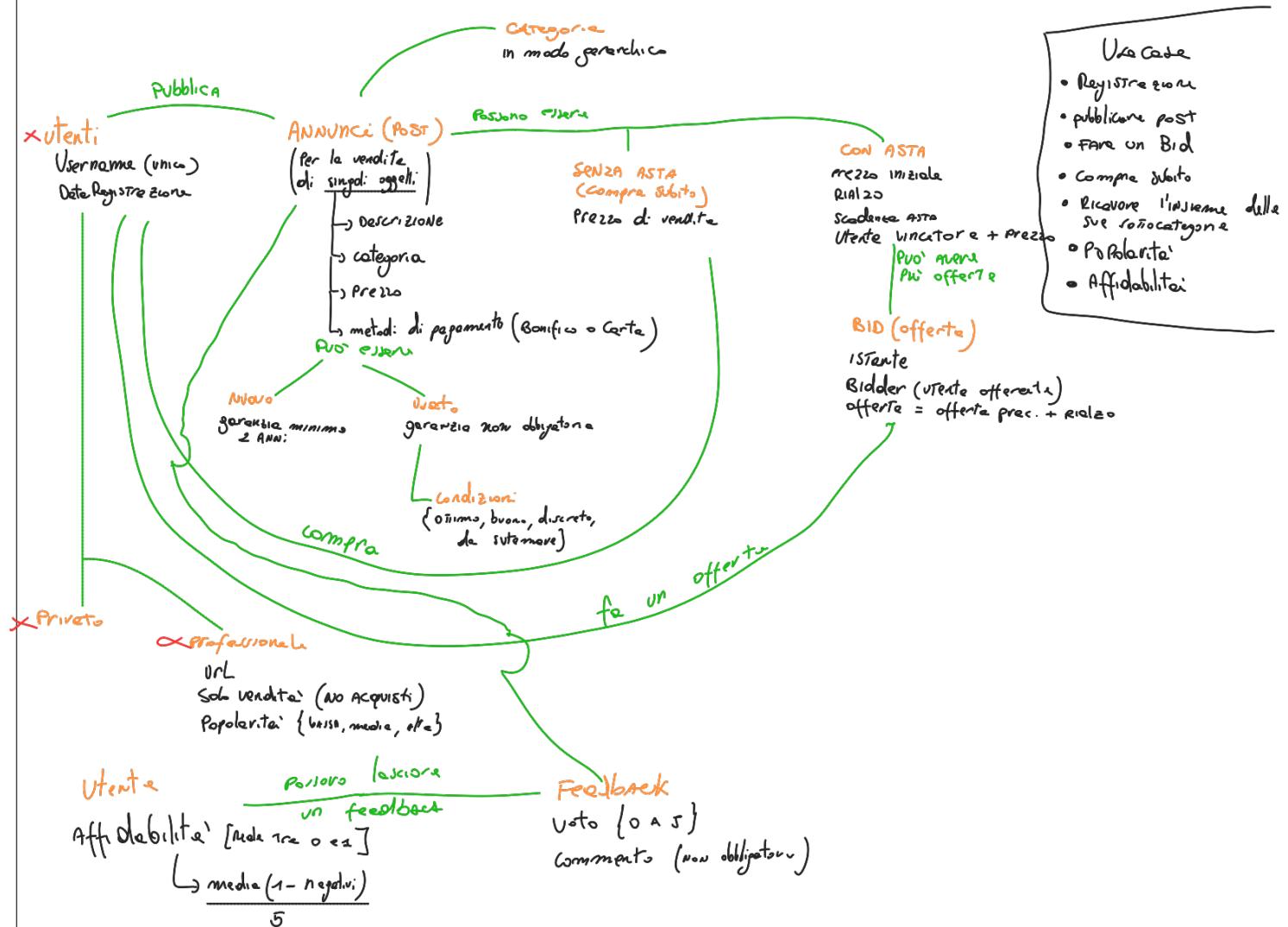


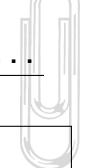
Questa pagina è stata intenzionalmente lasciata vuota

# 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





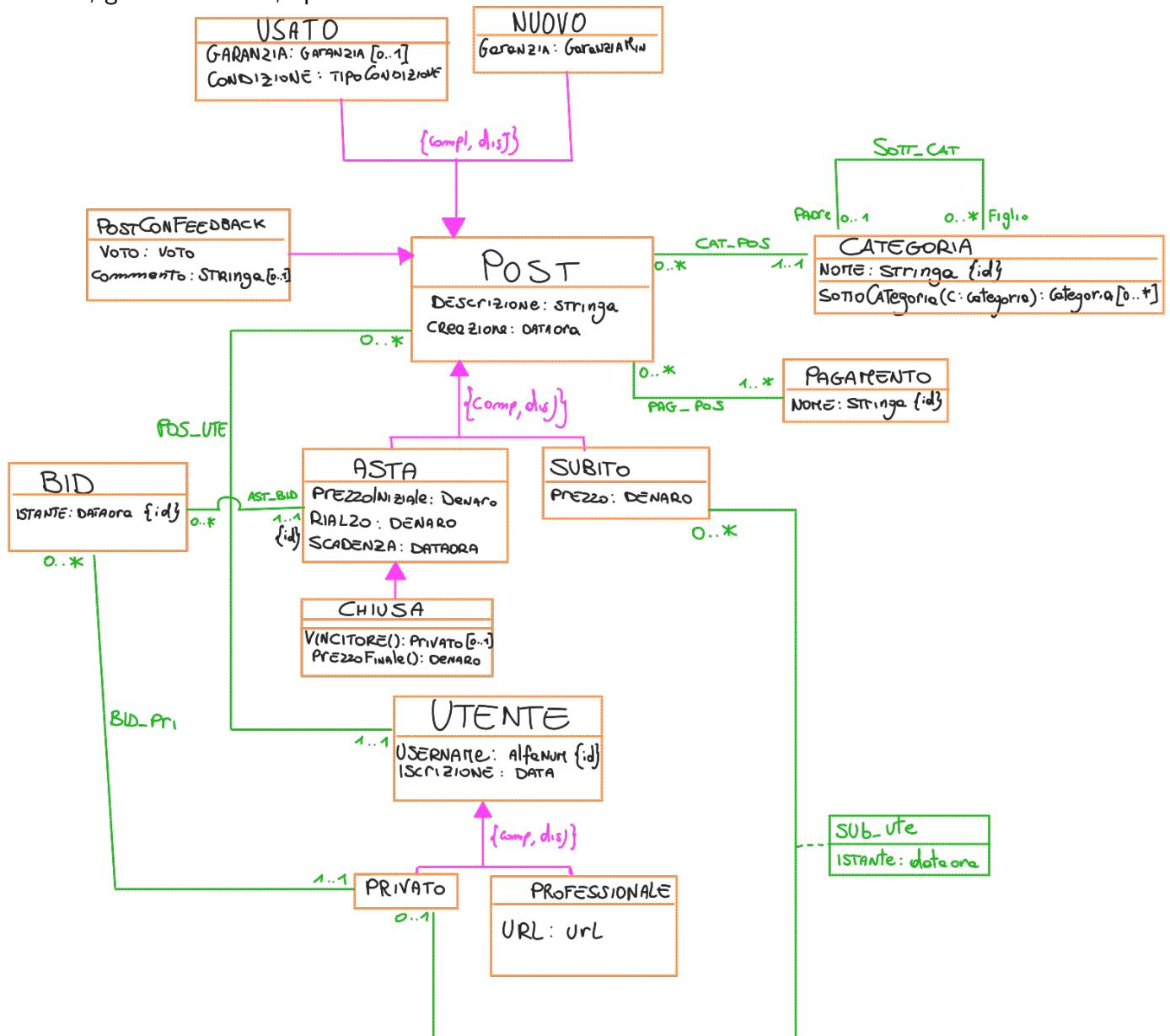
**Risposta alla Domanda 1 (segue)**

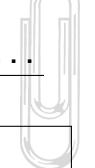
**Domanda 2 (45 minuti; 75 minuti al massimo)** Proseguire la fase di Analisi Concettuale dei requisiti, producendo un diagramma UML concettuale delle classi per l'applicazione, le specifiche di classi, associazioni, tipi di dato e vincoli esterni.

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

### Diagramma UML concettuale delle classi

Produrre un diagramma UML concettuale delle classi per l'applicazione in termini di classi, associazioni, attributi, generalizzazioni, operazioni di classe.





**Risposta alla Domanda 2 (segue)**

**Specifiche delle classi o associazioni** Per ogni classe o associazione del diagramma **con** operazioni o vincoli:

- Definire la specifica formale di eventuali operazioni necessarie a modellare i requisiti contrassegnati dalla barra laterale, ed eventuali vincoli esterni. Usare la logica del primo ordine estesa con teoria degli insiemi e semantica di mondo reale vista nel corso, usando il seguente alfabeto:
  - Un simbolo di predicato  $C/1$  per ogni classe  $C$ .  
Semantica di  $C(x)$ :  $x$  è una istanza di  $C$ .
  - Un simbolo di predicato  $T/1$  per ogni tipo di dato  $T$ .  
Semantica di  $T(x)$ :  $x$  è un valore di  $T$ .
  - Un simbolo di predicato  $\text{assoc}/2$  per ogni associazione binaria assoc.  
Semantica di  $\text{assoc}(c_1, c_2)$ :  $(c_1, c_2)$  è una istanza di assoc.
  - Un simbolo di predicato  $\text{attr}/2$  per ogni attributo attr di entità  
Semantica di  $\text{attr}(c, v)$ : uno dei valori dell'attributo attr dell'istanza  $c$  è  $v$ .
  - Un simbolo di predicato  $\text{attr}/3$  per ogni attributo attr di associazione binaria.  
Semantica di  $\text{attr}(c_1, c_2, v)$ : uno dei valori dell'attr. attr del link  $(c_1, c_2)$  è  $v$ .
  - Un simbolo di predicato  $\text{op}/(n+2)$  per ogni operazione di classe ad  $n$  argomenti.  
Semantica di  $\text{op}(c, \text{arg}_1, \dots, \text{arg}_n, v)$ : uno dei valori di ritorno di op, quando invocata sull'istanza  $c$  e con argomenti  $\text{arg}_1, \dots, \text{arg}_n$  è  $v$ .
  - Il simbolo di  $=/2$  (la cui interpretazione è la relazione che lega ogni elemento del dominio di interpretazione solo con se stesso) e opportuni simboli di predicato e di funzione, soggetti a semantica di modo reale, per relazioni e funzioni standard tra elementi dei tipi di dato, tra cui  $\text{adesso}/0$ , interpretato come il valore del dominio DataOra che rappresenta l'istante corrente.

## Risposta

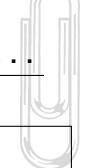
|   |  |
|---|--|
| <p>1 Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: <u>POST</u>.....</p> <p>Operazioni, vincoli:<br/> <math>\text{Vincolo: UN POST con FEEDBACK deve essere ACQUISTATO o VINTO all'ASTA}</math><br/> <math>\forall p \text{ POST} \text{ con FEEDBACK}(p) \rightarrow (\exists m \text{ SUB-UTE}(p, m)) \vee (\exists a, b \text{ Utente}(a) \wedge bid(b) \wedge BID\_PRI(b, p) \wedge AST\_BID(p, b) \wedge CHIUSA(p) \wedge VINCITORE(p) = m)</math></p> <p><u>Scadente &lt; di creazione</u><br/> <math>\forall p, s, c</math><br/> <math>(post(p) \wedge creazione(p, c) \wedge scadenza(p, s) \rightarrow c &lt; s)</math></p> | <p>2 Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: <u>CATEGORIA</u>.....</p> <p>Operazioni, vincoli:<br/> <math>\text{SottoCategorie}(c : \text{categoria}) : \text{Categorie}[0..*]</math><br/> <math>\text{Pre:}</math><br/> <math>\text{Post:}</math><br/> <math>F = \{c'   \text{categoria}(c') \wedge \text{SOTTOCAT}(c, c')\}</math><br/> <math>F' = \{c''   \exists c' \ c' \in F \wedge c'' \in \text{SottoCategorie}(c')\}</math><br/> <math>\text{Result} = F \cup F'</math></p> |
|---|--|

|   |  |
|---|--|
| <p><b>3</b> Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: .... ASTA.....</p> <p>Operazioni, vincoli:</p> <p><b>OPERAZIONE</b></p> <p><b>PREZZO_FINAL( )</b>: DENARO</p> <p>Pre: dove esistere almeno un bid per l'asta e deve essere finita<br/> <math>(\exists b \text{ bid}(b) \wedge \text{AST-BID}(\text{this}, b)) \wedge (\exists s \text{ Scadenza}(\text{this}, s) \wedge \text{adesso} &gt; s)</math></p> <p>Post:<br/> <math>B =  \{b \mid \text{bid}(b) \wedge \text{AST-BID}(\text{this}, b)\} </math></p> <p>Sia <math>p</math> t.c. <math>\text{PrezzoFinala}(\text{this}, p)</math><br/> Sia <math>r</math> t.c. <math>\text{Rialzo}(\text{this}, r)</math></p> <p>Result = <math>p + ((B-1) \times r)</math></p>  | <p><b>6</b> Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: .... UTENTE.....</p> <p>Operazioni, vincoli:</p> <p>NON si puo comprare un oggetto dove tu sei il venditore<br/> <math>\forall u, p</math><br/> <math>(\text{utente}(u) \wedge \text{POST}(p) \wedge \text{pos-ute}(p, u)) \rightarrow [(\neg \text{Sub-UTE}(p, u)) \wedge (\neg \exists b \text{ bid}(b) \wedge \text{BID-Pri}(b, u) \wedge \text{AST-BID}(p, b))] \wedge</math></p> <p>NON deve esistere un utente con un bid prima della sua iscrizione<br/> <math>\forall u, b, i, i'</math><br/> <math>(\text{utente}(u) \wedge \text{BID}(b) \wedge \text{BID-Pri}(b, p) \wedge \text{istante}(b, i) \wedge \text{iscrizione}(u, i') \rightarrow i' \leq i)</math></p> <p>NO UTENSI CON POST PRIMA DELLA LORO ISCRIZIONE<br/> <math>\forall u, p, i, c</math><br/> <math>(\text{utente}(u) \wedge \text{POST}(p) \wedge \text{iscrizione}(u, i) \wedge \text{creazione}(p, c) \wedge \text{pos-ute}(p, u) \rightarrow i \leq c)</math></p> |
| <p><b>4</b> Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: .... BID.....</p> <p>Operazioni, vincoli:</p> <p>IL BID deve essere durante l'asta</p> <p><math>\forall b, i, f, a, c</math></p> <p><math>(\text{BID}(b) \wedge \text{ISTANTE}(b, i) \wedge \text{esta}(a) \wedge \text{scadenza}(a, f) \wedge \text{AST-BID}(a, b) \wedge \text{creazione}(a, c) \rightarrow c \leq i \wedge i \leq f)</math></p>  | <p><b>7</b> Tipo: <b>Classe</b>   <b>Associazione</b> (cerchiare)</p> <p>Nome: .... SUB-UTE.....</p> <p>Operazioni, vincoli:</p> <p>La DATA di ACQUISTO di un oggetto deve essere maggiore della data di creazione del POST</p> <p><math>\forall u, p, c, i</math></p> <p><math>(\text{POST}(p) \wedge \text{PRIVATO}(u) \wedge \text{SUB-UTE}(p, u) \wedge \text{ISTANTE}(p, u, i) \wedge \text{CREAZIONE}(p, c) \rightarrow i \geq c)</math></p> <p>La DATA di ACQUISTO di un oggetto deve essere maggiore della data di iscrizione</p> <p><math>\forall u, p, t, i</math></p> <p><math>(\text{POST}(p) \wedge \text{PRIVATO}(u) \wedge \text{SUB-UTE}(p, u) \wedge \text{ISTANTE}(p, u, i) \wedge \text{iscrizione}(u, t) \rightarrow t \leq i)</math></p>  |
| <p><b>5</b> Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: .... ASTA.....</p> <p>Operazioni, vincoli:</p> <p><b>OPERAZIONE</b></p> <p><b>VINCITORE()</b>: utente</p> <p>Pre: dove esistere almeno un bid per l'asta e deve essere finita</p> <p><math>(\exists b \text{ bid}(b) \wedge \text{AST-BID}(\text{this}, b)) \wedge (\exists s \text{ Scadenza}(\text{this}, s) \wedge \text{adesso} &gt; s)</math></p> <p>Post:</p> <p>Result = <math>u \mid \text{utente}(u) \wedge \exists b \text{ bid}(b) \wedge \text{AST-BID}(\text{this}, b) \wedge \text{BID-Pri}(b, u) \wedge</math></p> <p><math>(\exists b', t, t' \text{ bid}(b') \wedge \text{AST-BID}(\text{this}, b') \wedge \text{dataore}(b, t) \wedge \text{dataore}(b', t') \wedge t' &gt; t)</math></p> <p>ALTRA idea per realizzarlo e' usare</p> <p>Argmax <math>U = \{(u, t) \mid \dots\}</math></p> <p>ESEMPIO <math>(u_{\text{max}}, t_{\text{max}}) \in \text{Argmax}_{(u, t) \in U}</math></p> | <p><b>8</b> Tipo: <b>Classe</b>   Associazione (cerchiare)</p> <p>Nome: .... CATEGORIA.....</p> <p>Operazioni, vincoli:</p> <p>Ogni categoria non deve avere come figlio uno dei suoi antenati</p> <p><math>\forall x, y (\text{Categoria}(x) \wedge \text{Categoria}(y) \wedge \text{SOT-CAT}(x, y) \rightarrow \text{Antenato}(x, y))</math></p> <p><math>\forall x, y, z (\text{Categoria}(x) \wedge \text{Categoria}(y) \wedge \text{Categoria}(z) \wedge \text{SOT-CAT}(x, y) \wedge \text{Antenato}(y, z) \rightarrow \text{Antenato}(x, z))</math></p> <p><math>\forall x, y (\text{Categoria}(x) \wedge \text{Categoria}(y) \wedge \text{SOT-CAT}(x, y) \rightarrow \neg \text{Antenato}(y, x))</math></p>   |

## Specifiche dei tipi di dato, specifiche di ulteriori vincoli esterni ed altre specifiche

### Specifiche dei Tipi

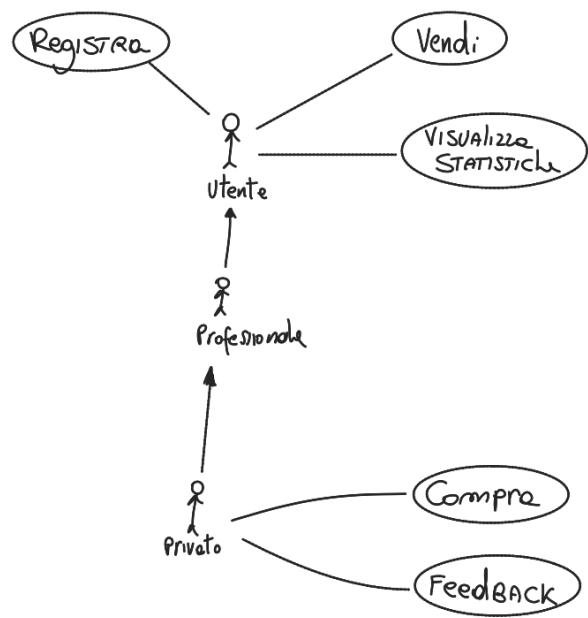
- VOTO = intero tra 0 e 5 (inclusi)
- Stringa = Stringa secondo STANDARD
- DATA = valore composto da interi (anni, mesi, giorni)
- DENARO = Reale  $\geq 0$  secondo STANDARD
- ALFANUM = Stringa che accetta solo lettere e numeri
- URL = Stringa secondo STANDARD
- garanzia = intero  $> 0$  che rappresentano i mesi
- garanziaMin = intero  $\geq 24$  (minimo 2 anni)
- DATAORA = Valore composto da (anni, mesi, giorni, ore, min, second.)
- tipoCondizione = {ottimo, buono, discreto, da sistemare}
- float\_0-1 = Reale compreso tra 0 e 1
- tipoPop = {bassa, media, alta}



**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





Questa pagina è stata intenzionalmente lasciata vuota

**Domanda 4 (10 minuti)** Proseguire la fase di Analisi Concettuale dei requisiti definendo la **segnatura** delle operazioni in ogni use-case.

### Risposta

- **Registra**

ISCRIVITI (n: alfanum) : PRIVATO

ISCRIVITI PRO (n: alfanum, u: url) : professionale

- **Vendi**

AS VENDITO (desc: stringa, pi: denaro, r: denaro, s: data/ora, g: garanzia, c: condizione) : ASTA

AS NUOVO (desc: stringa, pi: denaro, r: denaro, s: data/ora, g: garanziaMin) : ASTA

CS NUOVO (d: stringa, p: denaro, g: garanziaMin) : Subito

CS VENDUTO (d: stringa, p: denaro, g: garanzia, c: condizione) : subito

- **Compra**

ComproSubito (s: Subito)

offribid (a: ASTA)

- **Feed BACK**

Lasciafeedback (p: POST, v: voto, c: stringa [0..+])

- **VisualizzaStatiStidi**

def Popolarite (u: professionale) : tipo\_PoP

def AFFIDABilita (u: utente) : float\_0\_1



Questa pagina è stata intenzionalmente lasciata vuota

**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), ed includendo eventuali operazioni ausiliarie. 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

**Popolarità (v: professionale): tip.pop**

PRE:

nessuna

Post:

$$B = \left\{ b \mid \text{bio}(b) \wedge \exists a \text{ ASTA}(a) \wedge \text{POS-UTE}(a, u) \wedge \text{AST-BID}(a, b) \wedge \left( \exists i, \text{diff}, \text{annidiff} \text{ ISTANTE}(b, i) \wedge \text{diff} = \text{adesso} - i \wedge \text{ANNO}(\text{diff}, \text{annidiff}) \wedge \text{annidiff} \leq 1 \right) \right\}$$

$$A = \left\{ s \mid \text{subito}(s) \wedge \text{pos-ute}(s, u) \wedge \left( \exists u', i, \text{diff}, \text{annidiff} \text{ privato}(u') \wedge \text{sub-ute}(s, u') \wedge \text{ISTANTE}(s, u', i) \wedge \text{diff} = \text{adesso} - i \wedge \text{ANNO}(\text{diff}, \text{annidiff}) \wedge \text{annidiff} \leq 1 \right) \right\}$$

$$\text{Result} = \begin{cases} \text{'bassa'} & \text{Se } (|B| + |A|) < 50 \\ \text{'media'} & \text{Se } 50 \leq (|B| + |A|) \leq 300 \\ \text{'alta'} & \text{Se } (|B| + |A|) > 300 \end{cases}$$

**AFFIDABILITÀ (u: utente): Reale\_0..1**

PRE: ci deve essere almeno un feedback

$\exists p \text{ POSTConFeedback}(p) \wedge \text{POS-UTE}(p, u)$

POST:

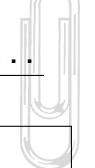
$$F = \left\{ (p, u, v) \mid \begin{array}{l} \text{POSTConFeedback}(p) \wedge \text{VOTO}(p, v) \\ \wedge \text{POS-UTE}(p, u) \end{array} \right\}$$

Sia  $M = \frac{1}{|F|} \sum_{(p, u, v) \in F} v$

Sia  $N = \left\{ (p, u, v) \mid (p, u, v) \in F \wedge v \leq 2 \right\}$

Sia  $Z = \frac{|N|}{|F|}$

Result =  $\frac{M(1-Z)}{5}$



**Risposta alla Domanda 5 (segue)**

## 2 Progettazione della base dati e delle funzionalità

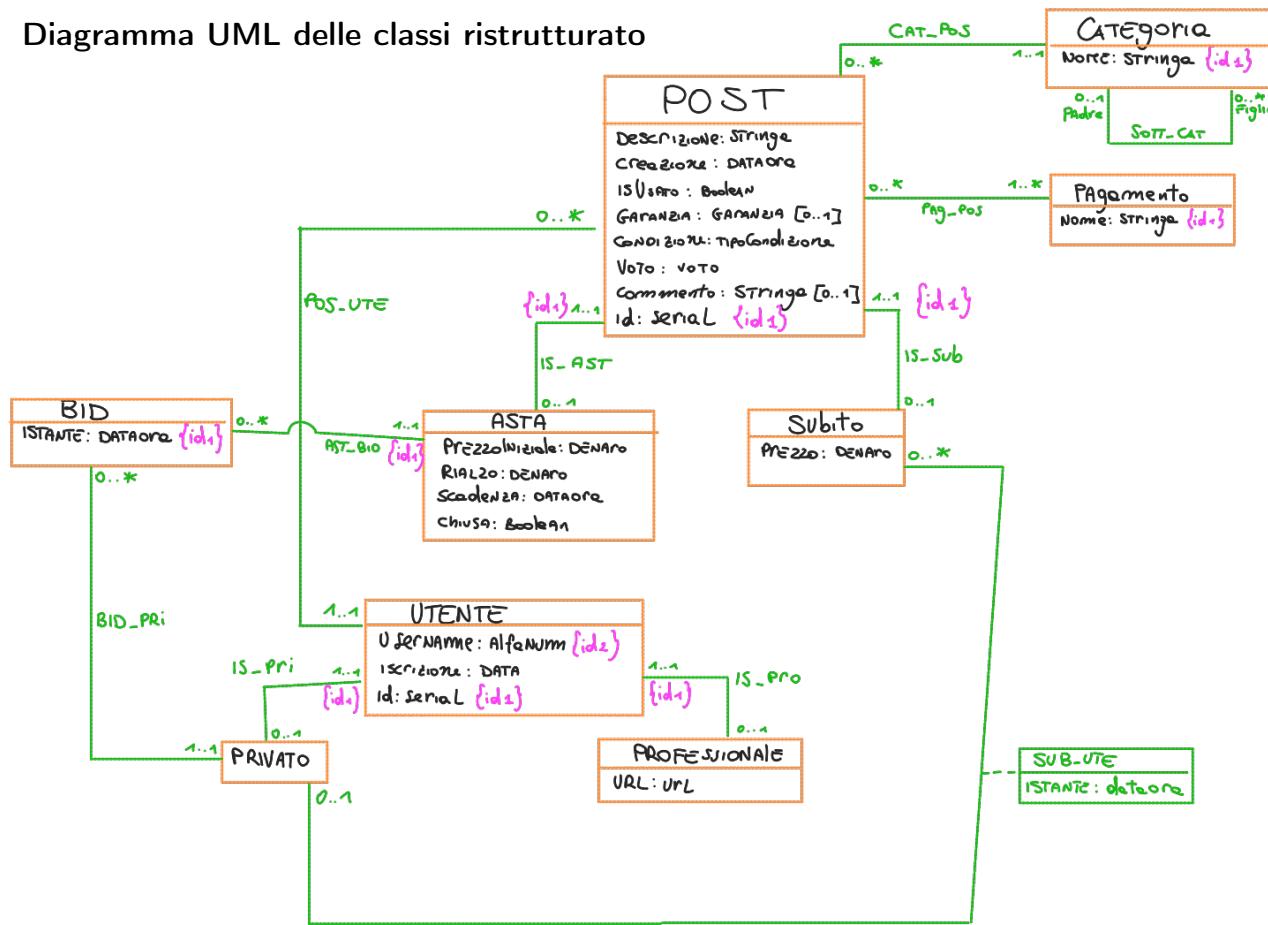
**Domanda 6 (20 minuti; 30 minuti al massimo)** Iniziare la fase di progettazione logica della base di dati decidendo il DBMS da utilizzare e ristrutturando lo schema UML delle classi concettuale, il dizionario dei dati e i vincoli esterni. In particolare:

- progettare una corrispondenza tra i tipi di dato 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 classe
- ristrutturare i vincoli esterni per renderli consistenti con la struttura del nuovo diagramma.

Descrivere brevemente le principali scelte effettuate.

|  |                   |
|--|-------------------|
| <b>DBMS da utilizzare . . . . .</b>  | <b>POSTGRESQL</b> |
| <b>Corrispondenza tra tipi di dato concettuali e domini supportati dal DBMS</b>  |                   |
| <pre> CREATE DOMAIN VOTO AS INT CHECK(value &gt;= 0 AND value &lt;= 5) CREATE DOMAIN STRINGA AS TEXT CHECK(value ~ '.*') CREATE DOMAIN DATA AS DATE CREATE DOMAIN DATAORA AS TIMESTAMP CREATE DOMAIN DENARO AS REAL CHECK(value &gt;= 0) CREATE DOMAIN ALFANUM AS TEXT CHECK(value ~ '.*') CREATE DOMAIN URL AS TEXT CHECK(value ~ '.*') CREATE DOMAIN GARANZIE AS INT CHECK(value &gt; 0) CREATE DOMAIN GARANZIALIN AS INT CHECK(value &gt;= 24)  CREATE TYPE TIPOCONDIZIONE AS ENUM ('ottimo', 'buono', 'discreto', 'da sistemare') CREATE DOMAIN FLOAT_0_1 AS REAL CHECK(value &gt;= 0 AND value &lt;= 1) CREATE TYPE TIPOPOP AS ENUM ('bassa', 'media', 'alta') </pre> |                   |

## Diagramma UML delle classi ristrutturato



### Breve descrizione delle scelte effettuate durante la ristrutturazione

IS-A Rimossa USATO - Nuovo per FUSIONE  
 IS-A feedback RIMOSSA per FUSIONE  
 IS-A chiusa RIMOSA per FUSIONE  
 IS-A privato/professionale RIMOSA per sostituzione  
 IS-A ATSA/subito RIMOSA per sostituzione

Aggiunti identificatori:

### 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)

Vincoli aggiornati/nuovi:

- C  $\nexists p \text{ POST}(p) \rightarrow [\text{isUser}(p, \text{TRUE}) \leftrightarrow (\exists c \text{ Condizione}(c))]$
- C  $\nexists p \text{ POST}(p) \wedge \text{isUser}(p, \text{FALSE}) \rightarrow (\exists g \text{ garanzia}(p, g) \wedge g \geq 24)$
- Vincoli aggiornati/nuovi:
- C  $\nexists p \text{ POST}(p) \rightarrow [\exists c \text{ Commento}(p, c) \rightarrow \exists v \text{ Voto}(p, v)]$
- $\nexists p \text{ POST}(p) \rightarrow [(\exists s, \text{subito}(s) \wedge \text{IS\_SUB}(p, s) \wedge \text{PRIVATO}(u) \wedge \text{SUB\_UTE}(s, u)) \vee (\exists a, \text{ASTA}(a) \wedge \text{IS\_AST}(a, p) \wedge \text{BID}(b) \wedge \text{AST\_BID}(a, b) \wedge \text{PRIVATO}(u) \wedge \text{BID\_PRI}(u) \wedge p.\text{VINCENTE}(u) = u)]$

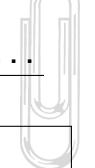
IS-A ASTA/subito

- $\nexists p \text{ POST}(p) \rightarrow [(\exists a \text{ IS\_AST}(a, p)) \vee (\exists s \text{ IS\_SUB}(p, s))]$
- $\nexists p \text{ POST}(p) \rightarrow [\exists a, s \text{ IS\_AST}(a, p) \wedge \text{IS\_SUB}(p, s)]$

IS-A privato/professionale

- $\forall u \text{ utente}(u) \rightarrow [(\exists pri \text{ IS\_PRI}(u, pri)) \vee (\exists pro \text{ IS\_PRO}(u, pro))]$
- $\forall u \text{ utente}(u) \rightarrow [\exists pri, pro \text{ IS\_PRI}(u, pri) \wedge \text{IS\_PRO}(u, pro)]$

da finire



**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 UML delle classi ristrutturato.

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

| 1 Relazione .POST..... (nome)   | Derivante da: <u>classe</u>   associazione (cerchiare) |  |  |  |  |  |  |  |
|---|--|--|--|--|--|--|--|--|
| Attributi   <u>ID</u>   Descrizione   CREAZIONE   ISUSATO   GARANZIA *   Condizione   VOTO *   Commento * |  |  |  |  |  |  |  |  |
| Domini   SERIAL   Stringa   DATAORA   Boolean   GARANZIE   TIPOGRADIZIONE   VOTO   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 ((Condizione IS NULL AND ISUSATO = FALSE) OR (Condizione IS NOT NULL AND ISUSATO = TRUE))

Check ( ISUSATO = TRUE OR GARANZIA > 24 ) Check (Commento IS NULL OR VOTO IS NOT NULL)

$ID \subseteq \text{PAG\_POS}(\text{POST})$

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

| 2 Relazione .POST (Cont.) .. (nome)   | Derivante da: <u>classe</u>   associazione (cerchiare) |  |  |  |  |  |  |  |
|---------------------------------------|--|--|--|--|--|--|--|--|
| Attributi   <u>CATEGORIA</u>   Utente |  |  |  |  |  |  |  |  |
| Domini   Stringa   INT                |  |  |  |  |  |  |  |  |

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)

FK Utente REF Utente (id)

La relazione accorda le relazioni che implementano le seguenti associazioni: .CAT\\_POS.. POS\\_UTE .....

| 3 Relazione .CATEGORIA.... (nome) | Derivante da: <u>classe</u>   associazione (cerchiare) |  |  |  |  |  |  |  |
|-----------------------------------|--|--|--|--|--|--|--|--|
| Attributi   <u>NOME</u>   PADRE * |  |  |  |  |  |  |  |  |
| 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):

FK PADRE REF CATEGORIA (Nome)

La relazione accorda le relazioni che implementano le seguenti associazioni: .SOTT\\_CAT .....

| 4 Relazione .PAGAMENTO..... (nome) | Derivante da: <u>classe</u>   associazione (cerchiare) |  |  |  |  |  |  |  |
|------------------------------------|--|--|--|--|--|--|--|--|
| Attributi   <u>Nome</u>            |  |  |  |  |  |  |  |  |
| 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 accorda le relazioni che implementano le seguenti associazioni: .....

| 5 Relazione .PAG\_POS..... (nome)   | Derivante da: <u>classe</u>   <u>associazione</u> (cerchiare) |  |  |  |  |  |  |  |
|-------------------------------------|---|--|--|--|--|--|--|--|
| Attributi   <u>POST</u>   PAGAMENTO |   |  |  |  |  |  |  |  |
| Domini   INT   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 POST REF POST(id)

FK PAGAMENTO REF PAGAMENTO (Nome)

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

|  |  |
|--|--|
| <b>6 Relazione SUBITO..... (nome)</b>                    | Derivante da: <b>classe</b> ) associazione (cerchiare) |
| Attributi   <u>POST</u>   PREzzo   PRIVATO *   IstanTe * |  |
| Domini   INT   DENARO   INT   dataOra                    |  |

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 POST REF POST(ID)      check ((PRIVATO IS NOT NULL AND IstanTe IS NOT NULL) OR (PRIVATO IS NULL AND IstanTe IS NULL))  
 FK PRIVATO REF PRIVATO(utente)

La relazione accorda le relazioni che implementano le seguenti associazioni: .IS - Sub, sub-ute .....

|   |  |
|---|--|
| <b>7 Relazione ASTA..... (nome)</b>                                 | Derivante da: <b>classe</b> ) associazione (cerchiare) |
| Attributi   <u>POST</u>   PrezzoMinimo   Scadenza   Rialzo   Chiusa |  |
| Domini   INT   DENARO   dataOra   DENARO   Boolean                  |  |

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 POST REF POST(ID)      check ((scadenza < now AND chiusa IS TRUE) OR (scadenza >= now AND chiusa IS FALSE))

La relazione accorda le relazioni che implementano le seguenti associazioni: .IS - AST .....

|  |  |
|--|--|
| <b>8 Relazione UTENTE..... (nome)</b>                | Derivante da: <b>classe</b> ) associazione (cerchiare) |
| Attributi   <u>username</u>   Iscrizione   <u>ID</u> |  |
| Domini   Alphanumeric   dataOra   Serial             |  |

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):

UNIQUE(username)

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

|  |  |
|--|--|
| <b>9 Relazione PRIVATO..... (nome)</b> | Derivante da: <b>classe</b> ) associazione (cerchiare) |
| Attributi   <u>utente</u>              |  |
| Domini   int                           |  |

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(id)

La relazione accorda le relazioni che implementano le seguenti associazioni: .IS - Pri .....

|   |  |
|---|--|
| <b>10 Relazione PROFESSIONALE. (nome)</b> | Derivante da: <b>classe</b> ) associazione (cerchiare) |
| Attributi   <u>utente</u>   URL           |  |
| Domini   int   URL                        |  |

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(id)

La relazione accorda le relazioni che implementano le seguenti associazioni: .IS - Pro .....

11 Relazione ...BID..... (nome) Derivante da: **classe** | associazione (cerchiare)

|           |                |             |                |  |  |  |  |  |
|-----------|----------------|-------------|----------------|--|--|--|--|--|
| Attributi | <u>ISTANTE</u> | <u>ASTA</u> | <u>PRIVATO</u> |  |  |  |  |  |
| Domini    | DATA/ORA       | INT         | ID             |  |  |  |  |  |

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 ASTA REF ASTA(POST)  
 FK PRIVATO REF PRIVATO(UTENTE)

La relazione accorda le relazioni che implementano le seguenti associazioni: ...AST-BID, BID-PR; ....

## 12 Relazione ..... (nome) Derivante da: classe | associazione (cerchiare)

|           |  |  |  |  |  |  |  |  |
|-----------|--|--|--|--|--|--|--|--|
| Attributi |  |  |  |  |  |  |  |  |
| Domini    |  |  |  |  |  |  |  |  |

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

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

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

## 13 Relazione ..... (nome) Derivante da: classe | associazione (cerchiare)

|           |  |  |  |  |  |  |  |  |
|-----------|--|--|--|--|--|--|--|--|
| Attributi |  |  |  |  |  |  |  |  |
| Domini    |  |  |  |  |  |  |  |  |

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

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

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

## 14 Relazione ..... (nome) Derivante da: classe | associazione (cerchiare)

|           |  |  |  |  |  |  |  |  |
|-----------|--|--|--|--|--|--|--|--|
| Attributi |  |  |  |  |  |  |  |  |
| Domini    |  |  |  |  |  |  |  |  |

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

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

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

## 15 Relazione ..... (nome) Derivante da: classe | associazione (cerchiare)

|           |  |  |  |  |  |  |  |  |
|-----------|--|--|--|--|--|--|--|--|
| Attributi |  |  |  |  |  |  |  |  |
| Domini    |  |  |  |  |  |  |  |  |

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

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

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

|           |                               |  |
|-----------|-------------------------------|--|
| <b>16</b> | <b>Relazione</b> ..... (nome) | Derivante da: <b>classe   associazione</b> (cerchiare) |
| Attributi |                               |  |
| Domini    |                               |  |

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

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

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

|           |                               |  |
|-----------|-------------------------------|--|
| <b>17</b> | <b>Relazione</b> ..... (nome) | Derivante da: <b>classe   associazione</b> (cerchiare) |
| Attributi |                               |  |
| Domini    |                               |  |

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

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

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

|           |                               |  |
|-----------|-------------------------------|--|
| <b>18</b> | <b>Relazione</b> ..... (nome) | Derivante da: <b>classe   associazione</b> (cerchiare) |
| Attributi |                               |  |
| Domini    |                               |  |

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

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

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

|           |                               |  |
|-----------|-------------------------------|--|
| <b>19</b> | <b>Relazione</b> ..... (nome) | Derivante da: <b>classe   associazione</b> (cerchiare) |
| Attributi |                               |  |
| Domini    |                               |  |

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

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

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

|           |                               |  |
|-----------|-------------------------------|--|
| <b>20</b> | <b>Relazione</b> ..... (nome) | Derivante da: <b>classe   associazione</b> (cerchiare) |
| Attributi |                               |  |
| Domini    |                               |  |

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

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

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

## 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.

**Trigger: FEEDBACK Solo Se Vinto o Campionato**

BEFORE INSERT or update ON POST

```
OK = EXIST((SELECT 1 FROM ASTA a JOIN BID b ON b.ASTA = a.POST
    WHERE a.POST = NEW.ID)
    UNION (SELECT 1 FROM Subito s
    WHERE s.UTENTE IS NOT NULL AND s.POST = NEW.ID))
```

IF NOT OK AND NEW.VOTO IS NOT NULL then  
RAISE EXCEPTION

Permit

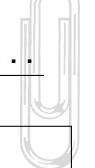
**Trigger: bidDentroIntervalloASTA**

BEFORE INSERT o UPDATE ON BID

```
IF NOT EXISTS (
    SELECT 1
    FROM POST p JOIN ASTA a ON p.id = a.POST
    WHERE a.POST = NEW.ASTA AND NEW.ISTANTE BETWEEN p.CREAZIONE AND a.SCADENZA
) Then
    RAISE EXCEPTION
```

```

|
|
| non li scrivo tutti
| trigger
```



**Risposta alla Domanda 7 (segue)**

**Domanda 8 (30 minuti; 45 minuti al massimo)** Proseguire la fase di progettazione dell'applicazione producendo le specifiche realizzative delle operazioni di classe e/o 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. Specificare, per ogni operazione, se debba essere implementata nel DBMS o nel *back-end*.

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

### Risposta

SottoCategoria(*n*: Stringa): Stringa [0..\*]

```
WITH RECURSION subcat AS (
    Select Nome FROM Categoria
    Where PADRE = n
    UNION ALL
    Select c.Nome FROM Categoria c
    INNER Join subcat s ON c.PADRE = s.Nome
)
Select * FROM subcat
```

VINCITORE(*A* : int): int  
<sub>id ASTA</sub>  
<sub>id utente</sub>

```
ERROR = EXIST(Select a FROM BID
               WHERE ASTA = A)
IF ERROR THEN
    RAISE EX_Caption
```

```
Select p.utente FROM BID b Join PRIVATO p ON b.PRIVATO = p.utente
   WHERE b.ASTA = A
ORDER BY b.ISTANTE DESC
Limit 1
```

Popolarità(*u*: int): TipoPop

```
Select COUNT(p.ID) INTO A
FROM POST p Join Subito s ON s.POST = p.ID
WHERE p.utente = u AND s.PRIVATO IS NOT NULL
```

```
Select COUNT(*) INTO B
FROM POST p Join ASTA a ON a.POST = p.id
   Join BID b ON b.ASTA = a.POST
WHERE p.utente = u
```

```
IF (A + B) < 50 THEN
    RETURN 'bassa'
IF (A + B) ≥ 50 AND (A + B) ≤ 300 THEN
    RETURN 'media'
ELSE
    RETURN 'alta'
```

[continua alla pagina seguente]

### Risposta alla Domanda 8 (segue)

AFFIDABILITA(U: int): Reale\_0..1

```
ERROR = EXISTS ( SELECT 1 FROM POST
                  WHERE UTENTE = u AND VOTO IS NOT NULL)
```

```
IF NOT ERROR THEN
    RAISE EXCEPTION
```

```
SELECT AVG(P.VOTO) INTO M
FROM POST P
WHERE P.UTENTE = u
```

```
SELECT COUNT(*) INTO N
FROM POST P
WHERE P.UTENTE = u AND P.VOTO <= 2
```

```
SELECT COUNT(*) INTO F
FROM POST P
WHERE P.UTENTE = u
```

$$\text{RESULT} = M * (1 - (N/F)) / 5$$

**SELECT**

AVG(P.VOTO) \*

(1 - (SUM(CASE WHEN P.VOTO <= 2 THEN 1 ELSE 0 END) / COUNT(\*)))

) / 5 AS RESULT

FROM POST P

WHERE P.UTENTE = u

PREZZOFinale(IDASTA : int): DENARO

```
ERROR = EXIST (Select 1 FROM BID b
                  WHERE LASTA = IDASTA)
```

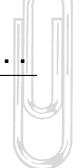
```
IF ERROR THEN
    RAISE EXCEPTION
```

```
Select COUNT(*) INTO B
FROM ASTA a JOIN BID b ON b.ASTA = IDASTA
```

```
Select PrezzoIniziale, Prezzo INTO P, R
FROM ASTA Where Post = IDASTA
```

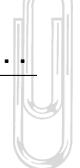
$$\text{Result} = P + ((B-1) * R)$$

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.]