

Esame Es.20230130 – Prova scritta del 30 gennaio 2023

Si vuole progettare Slimmy, una nuova applicazione che permetta agli utenti di monitorare la loro alimentazione, tracciando i cibi che ingeriscono giornalmente (con relative quantità e nutrienti) per tenere sotto controllo apporto calorico ed equilibrio della loro dieta.

A seguito di interviste con il committente, è stata stilata la seguente specifica dei requisiti.

Slimmy deve consentire agli **utenti** di registrarsi al servizio fornendo **nome, cognome, nazionalità, email e password**.

Slimmy deve fornire un archivio di **schede alimenti** (**cibi** e **bevande**). Tale **archivio viene continuamente esteso, oltre che dalla redazione di Slimmy, anche dagli stessi utenti**. Per ogni **alimento** presente nell'archivio, la relativa scheda deve permettere di rappresentare il **nome**, la **marca** (se disponibile), la **grandezza della confezione** (se disponibile), il **codice identificativo** (una stringa) codificato nel codice a barre della confezione (se disponibile), il relativo **apporto calorico** (in kCal) per unità di alimento, il valore di **tale unità** (ad es., per i cibi liquidi: litri; per i solidi, grammi o etogrammi o chilogrammi; per alimenti come le uova, unità), la **quantità** (in unità di alimento) di una porzione standard, oltre **che la quantità** (questa volta sempre in grammi, per garantire uniformità) **di ogni nutriente per unità di alimento**.

La lista dei nutrienti è tipicamente stabile ed uguale per tutti gli alimenti (ad es., carboidrati totali, zuccheri, proteine, grassi totali, grassi saturi, grassi insaturi, sodio, potassio, fibre, alcol, etc.), ma Slimmy deve fare in modo che tale lista sia estendibile dall'autore dell'inserimento di una scheda alimento. Viceversa, Slimmy deve permettere l'inserimento di schede alimenti che non hanno valori per i **nutrienti definiti 'opzionali'**. La lista dei nutrienti opzionali è definita dalla redazione di Slimmy. Inoltre, tutti i nutrienti aggiunti dagli utenti alla lista dei nutrienti sono, per definizione, opzionali. Si mostrano di seguito alcuni esempi di schede alimento:

Fusilli Di Chicco (formato n. 34)

- Codice identificativo: "8033829843"
- Unità di alimento: 100 grammi
- kCal per unità di alimento (ovvero per 100 grammi): 351
- Porzione standard: 0.7 unità di alimento (ovvero 70 grammi)
- Nutrienti (in grammi per 100 grammi di prodotto; *: nutrienti obbligatori):
 - Grassi totali*: 1.5 g
 - Zuccheri*: 3.4 g
 - Sale: 0 g
 - Grassi saturi*: 0.3 g
 - Fibre*: 2.9 g
 - Proteine*: 14 g
 - Carboidrati*: 69 g

Buzz Cola

- Codice identificativo: "11928322032"
- Unità di alimento: 100 ml
- kCal per unità di alimento (ovvero per 100 ml): 42
- Porzione standard: 3.3 unità di alimento (ovvero 330 ml)
- Nutrienti (in grammi per 100 ml di prodotto; *: nutrienti obbligatori):

- Grassi totali*: 0 g
- Grassi saturi*: 0 g
- Carboidrati*: 11 g
- Zuccheri*: 11 g

- Fibre*: 0 g
- Proteine*: 0 g
- Caffeina: 0.008 g
- Calcio: 0.002 g

- Fosforo: 0.01 g
- Potassio: 0.002 g
- Ferro: 0.00011 g
- Sale: 0.001 g

Come detto, schede relative a nuovi alimenti possono essere aggiunte in ogni momento dalla redazione di Slimmy, ma anche da parte degli utenti, che quindi possono contribuire in prima persona all'ampliamento dell'archivio degli alimenti di Slimmy. Mentre le schede di alimenti inserite dalla redazione diventano subito pubbliche e visibili a tutti gli utenti, le schede aggiunte da utenti devono attraversare un processo di validazione. Una volta validate diventano pubbliche esattamente come quelle inserite dalla redazione.

Una scheda alimento inserita da un utente viene resa pubblica quando almeno 10 utenti l'hanno dichiarata valida. Fintantoché una scheda non supera il processo di validazione, può ancora essere ottenuta come risultato di una ricerca, ma può essere utilizzata (ad es., inserita in un diario alimentare) solo da parte dell'utente che l'ha compilata e di quelli che l'hanno dichiarata valida fino a quel momento (in questo modo si vogliono incentivare gli utenti interessati ad inserire quell'alimento a contribuire alla validazione della relativa scheda).

Nel tempo, le schede alimenti (sia quelle inserite dalla redazione che quelle inserite dagli utenti e diventate pubbliche a seguito di validazione) possono essere invalidate (ad es., perché obsolete, si pensi al caso in cui un alimento confezionato cambi profilo nutrizionale perché prodotto con una nuova ricetta). Questo avviene quando 100 utenti le hanno dichiarate non più valide. Le schede invalidate diventano inaccessibili a tutti gli utenti (tranne che, per le schede inizialmente inserite da un utente, all'utente che le ha inserite).

Gli utenti, durante la loro giornata, inseriscono in Slimmy la composizione dei loro pasti (diario alimentare), in termini di alimenti ingeriti e relative quantità. L'insieme dei pasti è attualmente definito dalla redazione di Slimmy come: colazione, pranzo, snack e cena, ma potrebbe essere oggetto di estensioni e perfino di personalizzazioni in futuro.

Il sistema deve permettere di calcolare una serie di statistiche riguardo il diario alimentare di un utente in un certo giorno. In particolare: (i) Numero totale di kCal ingerite; (ii) grammi totali ingeriti di ogni nutriente.

Gli utenti di Slimmy possono acquistare abbonamenti Premium (la cui durata dipende dalla tipologia di abbonamento scelta tra quelle previste), durante i quali avranno accesso a servizi non offerti ai clienti Free. Il servizio principale offerto da Slimmy ai clienti Premium è relativo alla assistenza di coloro che desiderano seguire una dieta.

Le diete sono definite dai singoli utenti e sono private. Una dieta definisce delle regole, in termini di soglie minime e massime per alcune delle statistiche giornaliere di cui sopra. Ad esempio, un utente Premium che vuole perdere peso può definire una dieta, che decide di chiamare "ipocalorica ed iperproteica", con le seguenti regole circa le kCal totali e le quantità di nutrienti ingeriti al giorno:

Dieta: "ipocalorica ed iperproteica"

1. kCal totali: tra 1300 e 1500
2. Carboidrati totali: tra 150 e 180 g
3. Zuccheri: ≤ 30 g
4. Proteine totali: tra 80 e 110 g
5. Grassi saturi: ≤ 10 g
6. Sale: ≤ 3 g
7. Fibre: tra 35 e 40 g

Dato un pasto p , Slimmy deve permettere agli utenti di ottenere gli alimenti più comunemente aggiunti al diario per il pasto p da tutti gli altri utenti. Per ognuno, si vuole anche conoscere la quantità media con la quale viene aggiunta al diario.

Infine, Slimmy deve restituire, all'utente Premium che sta seguendo una dieta, le quantità di ognuno dei nutrienti che può ancora assumere (in base alle regole della dieta) in un certo giorno.

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

Utente:

- nome; Stringa
- cognome; Stringa
- email; Email
- password; Password
- associato alla Posizione

Nutriente:

- nome; Stringa
- obbligatorio; Booleano

Alimento:

- nome; Stringa
- marca; Stringa
- codice; CadBarre
- Kcal; Intero > 0
- porzione; Reale > 0
- Unità; Intero > 0
- associato all'Unità Di Misura
- associato ai Nutritivi
(in quantità); Intero > 0
- Specializzazione;
- Da Utente
- associato a 10 Utenti che lo validano
- Invalidato
- associato a 100 Utenti che lo invalidano

Posiz:

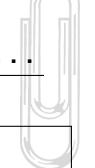
- associato alla Tipologia
- associato agli alimenti
con nPart; Reale > 0

Premium:

- duratagg; Intero > 0
- associato all'Utente

Diet:

- Kcal max; Intero > 0
- Kcal min; Intero > 0
- associato ai Nutritivi
(con maxgr; Intero > 0 e
mingr; Intero > 0)
- associato all'Utente



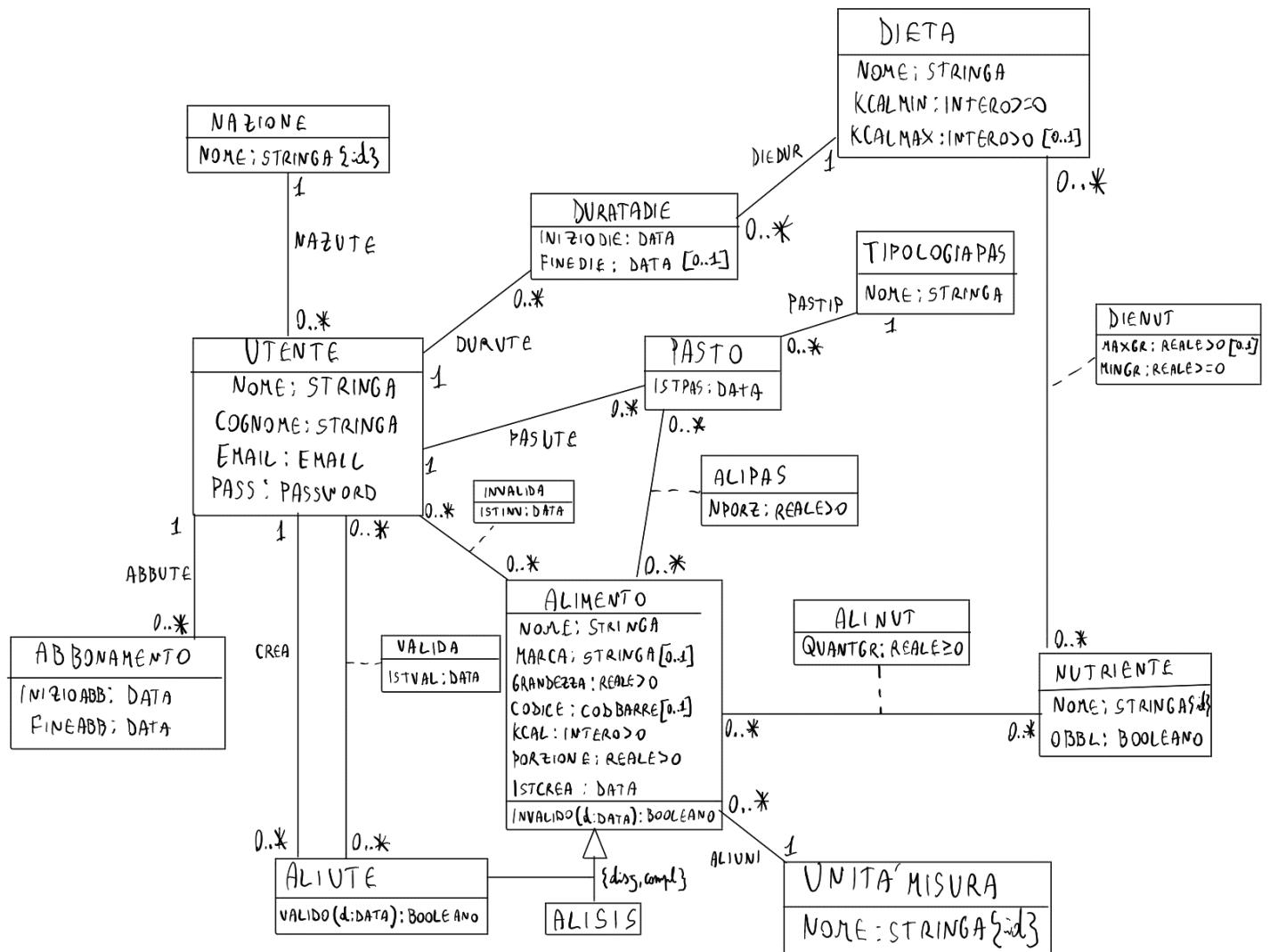
Risposta alla Domanda 1 (segue)

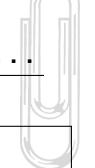
Domanda 2 (45 minuti; 75 minuti al massimo) Proseguire la fase di Analisi Concettuale dei requisiti, producendo un diagramma UML concettuale delle classi per l'applicazione, le specifiche di classi, associazioni, tipi di dato e vincoli esterni.

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'attrib. 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><input type="checkbox"/> Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ...<u>UTENTE</u>.....</p> <p>Operazioni, vincoli: V. DIETE DISGUNTE</p> $\forall u \text{ UTENTE}(u) \rightarrow \exists d, d', d'', t \text{ DURUTE}(u, d) \wedge \text{DURUTE}(u, d') \wedge \text{INIZIODIE}(d, d) \wedge \text{INIZIODIE}(d', d') \wedge \text{DATA}(t) \wedge d \neq d' \wedge t \geq d \wedge (\forall df \text{ FINE DIE}(df) \rightarrow t \leq df) \wedge t \geq d'' \wedge (\forall df' \text{ FINE DIE}(df') \rightarrow t \leq df')$	<p><input type="checkbox"/> Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ...<u>ALIVITE</u>.....</p> <p>Operazioni, vincoli: V. VALIDA DOPO CREA</p> $\forall a, u, dv, dc \text{ ALIVITE}(a) \wedge \text{ISTCREA}(a, dc) \wedge \text{ISTRVAL}(a, u, dv) \rightarrow dc \leq dv$ <p>V. CREA NON VALIDA</p> $\forall a, u \text{ ALICREA}(a, u) \rightarrow \neg \text{VALIDA}(a, u)$ <p>V. INVALIDA SE VALIDO</p> $\forall a, u, dv \text{ INVALIDA}(a, u) \wedge \text{ISTRINV}(a, u, dv) \rightarrow \text{VALIDO}(a, dv, \text{TRUE})$ <p>VALIDO(d: DATA): BOOLEANO</p> <ul style="list-style-type: none"> - pre: $\text{INVALIDO}(t_{\text{min}}, t_{\text{max}}, \text{FALSE})$ - post: $V = \left\{ u \mid \text{VALIDA}(u, t_{\text{min}}) \wedge \exists dv \text{ ISTRVAL}(u, t_{\text{min}}, dv) \right\}$ <p>RESULT = ($V \geq 10$)</p>
--	--

V. VALIDA SE NON INVALIDO

$$\forall a, u, iv \text{ VALIDA}(a, u) \wedge \text{ISTRVAL}(a, u, iv) \rightarrow \neg \text{INVALIDO}(a, iv, \text{FALSE})$$

<p>3 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... ALISIS</p> <p>Operazioni, vincoli:</p> <p>$\forall u, d \text{ ALISIS}(u) \wedge \text{ISTCREA}(u, d) \wedge \text{ISTINV}(u, d) \rightarrow d \leq u$</p>	<p>6 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... PASTO</p> <p>Operazioni, vincoli:</p> <p>$\forall u, d \text{ PASTO}(u) \wedge \text{ALIMENTOVALIDO}(u, d) \wedge \text{PASUTE}(u, d) \wedge \text{ALIPAS}(u, d) \wedge \text{ISTPAS}(d, u) \rightarrow [\text{VALIDO}(u, d, \text{true}) \wedge \text{ALIUTE}(u)] \vee [\text{VALIDA}(u, d) \wedge \exists dv \text{ ISTVAL}(u, d, dv) \wedge dv \leq d] \wedge \text{INVALIDO}(u, d, \text{false}) \vee [\text{INVALIDO}(u, d, \text{false}) \wedge \text{ALISIS}(u)] \vee [\text{ALICREA}(u, d) \wedge \exists dc \text{ ISTCREA}(u, d, dc) \wedge dc \leq d]$</p>
<p>4 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... NUTRIENTE</p> <p>Operazioni, vincoli:</p> <p>$\forall n \text{ NUTRIENTE OBBLIGATORIO}$</p> <p>$\forall a, m \text{ ALIMENTO}(a) \wedge \text{NUTRIENTE}(n) \wedge \text{OBBL}(n, \text{true}) \rightarrow \text{ALINUT}(a, m)$</p>	<p>7 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... ABBONAMENTO</p> <p>Operazioni, vincoli:</p> <p>$\forall f \text{ FINE DOPOLNIZIO}$</p> <p>$\forall a, f, i \text{ ABBONAMENTO}(a) \wedge \text{INIZIOABB}(a, i) \wedge \text{FINEABB}(a, f) \rightarrow f \geq i$</p> <p>$\forall t \text{ DISGIUNTI}$</p> <p>$\forall u \text{ UTENTE}(u) \rightarrow \exists a, a', i, i', f, f', t \text{ at} a \wedge \text{ABBUTE}(a, u) \wedge \text{ABBUTE}(a', u) \wedge \text{INIZIOABB}(a, i) \wedge \text{INIZIOABB}(a', i') \wedge \text{FINEABB}(a, f) \wedge \text{FINEABB}(a', f') \wedge \text{DATA}(t) \wedge t >= i \wedge t <= f \wedge t >= i' \wedge t <= f'$</p>
<p>5 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... DIETA</p> <p>Operazioni, vincoli:</p> <p>$\forall d \text{ KCALCONTINUE}$</p> <p>$\forall d, k_{\min}, k_{\max} \text{ DIETA}(d) \wedge \text{KCALMIN}(k_{\min}, d) \wedge (\exists k_{\max} \text{ KCALMAX}(k_{\max}, d)) \rightarrow k_{\min} < k_{\max}$</p> <p>$\forall d \text{ DIENUTCONTINUE}$</p> <p>$\forall d, n, m \text{ DIENUT}(d, n) \wedge \text{MINGR}(d, n, m) \wedge (\exists max \text{ MAXGR}(d, n, max)) \rightarrow max > m$</p>	<p>8 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: ... DURATA DIE</p> <p>Operazioni, vincoli:</p> <p>$\forall f \text{ FINE DOPOLNIZIO}$</p> <p>$\forall d, i, f \text{ DURATADIE}(d) \wedge \text{INIZIODIE}(i) \wedge \text{FINEDIIE}(d, f) \rightarrow i < f$</p>

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

EMAIL = (STRINGA SECONDO STANDARD)

PASSWORD = (STRINGA SECONDO STANDARD)

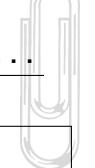
CODBARRE = (STRINGA SECONDO STANDARD)

ALIMENTO.INVALIDO (d; DATA) : BOOLEANO

- pre:

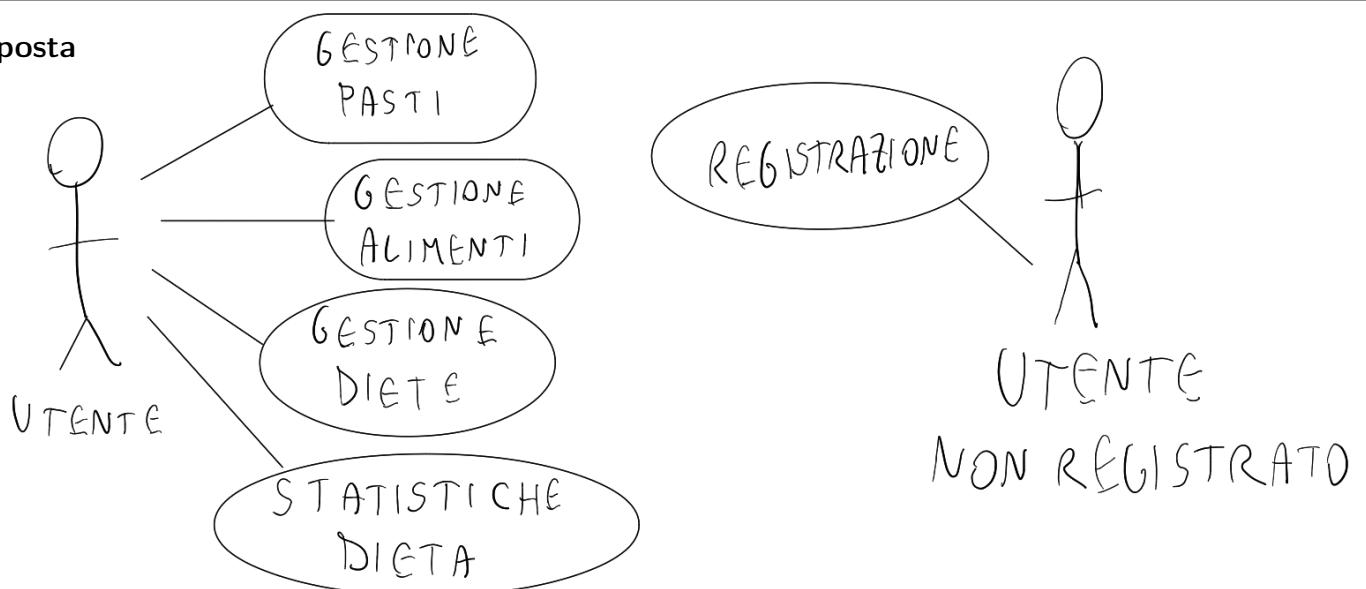
- post: $N = \{ u \mid \text{INVALIDA}(u, \text{ths}) \wedge \exists d_i \in \text{DATA} \text{ such that } \text{ISTINV}(u, \text{ths}, d_i) \wedge d_i = d \}$

RESULT = (|IV| >= 100)



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 la **segnatura** delle operazioni in ogni use-case.

Risposta

GESTIONE PASTI:

NEW PASTO (t: TIPOLOGIA PAS, A: ALIMENTO [0..*], u: UTENTE): PASTO

GESTIONE ALIMENTI:

NEW ALIMENTO (n: STRINGA, m: STRINGA [0..1], g: REALE >= 0, ut: UTENTE,
c: CODBARRE [0..1], kcal: INTERO >= 0, p: REALE >= 0, u: UNITA',
nut: NUTRIENTE [0..*]): ALIMENTO

VALIDA (a: ALIMENTO)

INVALIDA (a: ALIMENTO)

GESTIONE DIETE:

NEW DIETA (): DIETA

NEW PERIODO (): DURATA DIE

STATISTICHE DIETA:

ALIMENTOCOMUNE (t: TIPOLOGIA PAS): (ALIMENTO, REALE >= 0) [0..*]

NUTRIENTIMANCANTI (d: DATA): (NUTRIENTE, REALE) [0..*]

CREA ALIMENTI:

NEW ALIMENTO (): ALIMENTO

GESTIONE NUTRIENTI:

NEW NUTR (n: STRINGA, s: BOOLEANO): NUTRIENTE

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

ALIMENTO COMUNE (t ; TIPOLOGIA PAS): $(ALIMENTO, REALE \geq 0)[0..*]$

- pre: $\exists p \text{ PASTIPO}(p, t)$

- post: $A = \left\{ (a, n) \mid \begin{array}{l} \exists p \text{ PASTIPO}(p, t) \wedge ALIPAS(a, p) \wedge \\ n = |\{\nu \mid PASTIPO(\nu, t) \wedge ALIPAS(a, \nu)\}| \end{array} \right\}$

$$A_{\max} = \operatorname{ARGMAX}_{(a, n)} (n)$$

$$\text{RESULT} = \left\{ (a, m) \mid \begin{array}{l} \exists n \in A_{\max} \wedge N = \left\{ (\nu, q) \mid \begin{array}{l} PASTIPO(\nu, t) \wedge ALIPAS(\nu, a) \wedge \\ NPORZ(\nu, a, q) \end{array} \right\} \\ m = \left(\sum_{(\nu, q) \in N} q \right) / |N| \end{array} \right\}$$

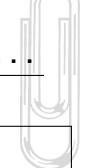
NUTRIENTI MANCATI (u : UTENTE, d : DATA): $(n: NUTRIENTE, q: REALE)[0..*]$

- pre: $\exists die, dur, di \text{ DIE_DUR}(die, dur) \wedge DURUTE(dur, u) \wedge INIZIODIE(di, dur) \wedge (\exists df \text{ FINEDI}(df, die) \rightarrow df \geq d) \wedge di \leq d \wedge$

$\exists a, ai, af \text{ ABBUTE}(a, u) \wedge INIZIOABB(a, ai) \wedge FINEABB(a, af) \wedge ai \leq d \wedge af \geq d$

- post:

$$\text{RESULT} = \left\{ (n, q_{\max} - q_{\text{tot}}) \mid \begin{array}{l} \text{NUTRIENTE}(n) \wedge \exists die, dur, di, df \text{ DIE_NUT}(die, n) \wedge \\ \text{DIE_DUR}(die, dur) \wedge DURUTE(dur, u) \wedge MAXGR(die, n, q_{\max}) \wedge \\ INIZIODIE(dur, di) \wedge FINEDI(df, die) \wedge di \leq d \wedge df \geq d \wedge \\ Q = \left\{ (\nu, a, q) \mid \begin{array}{l} ALIMENTO(a) \wedge \exists mp, q_{gr} \text{ ALIPAS}(a, mp) \wedge PASUTE(\nu, a) \wedge \\ NPORZ(\nu, a, mp) \wedge ISTPAS(\nu, a) \wedge ALINUT(a, n) \wedge \\ QUANTITAGR(a, n, q_{gr}) \wedge q = q_{gr} \cdot mp \end{array} \right\} \wedge \\ q_{\text{tot}} = \sum_{(a, q) \in Q} q \end{array} \right\}$$



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.

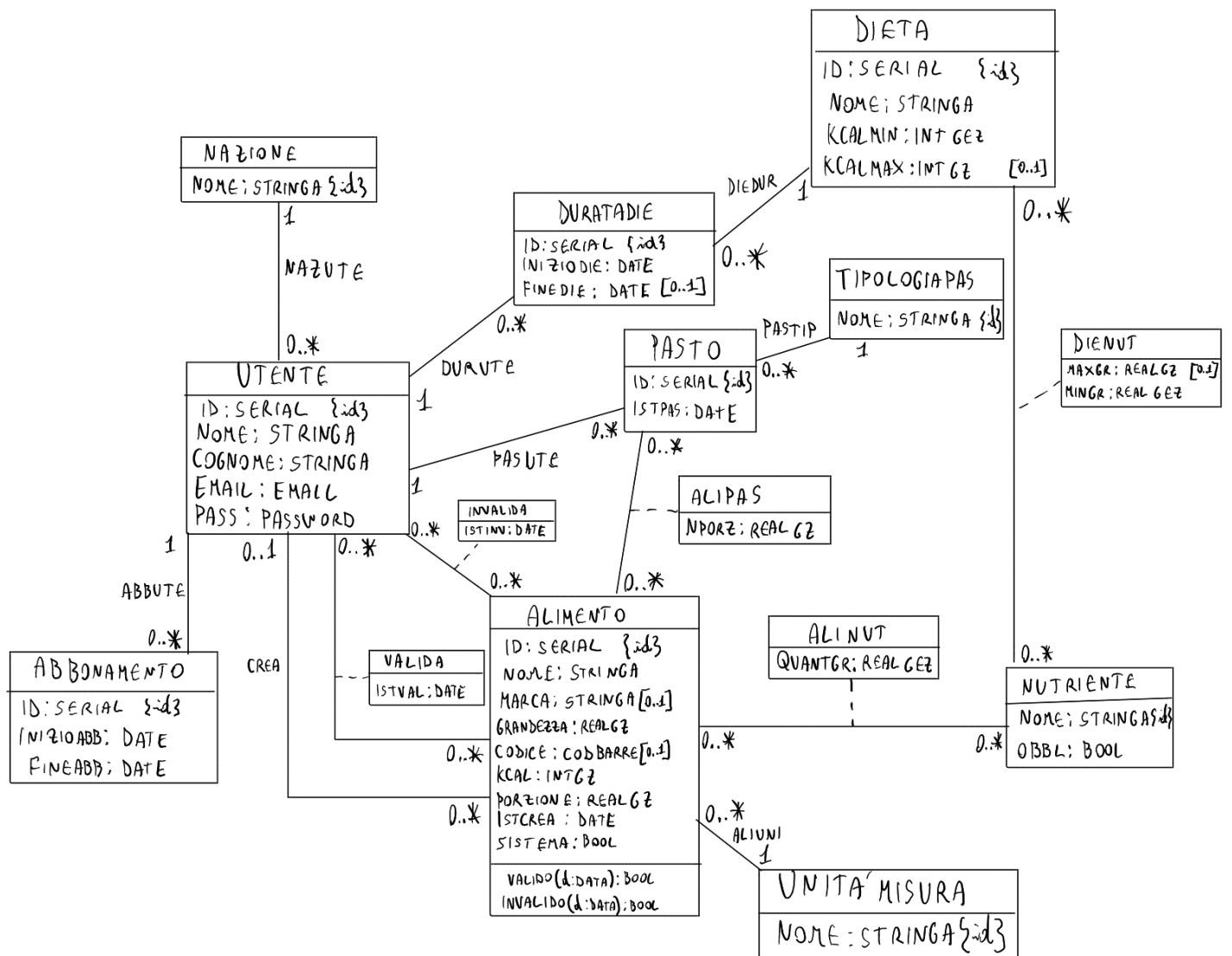
Descrivere brevemente le principali scelte effettuate.

DBMS da utilizzare ... PostgreSQL

Corrispondenza tra tipi di dato concettuali e domini supportati dal DBMS

```
CREATE DOMAIN STRINGA AS VARCHAR NOT NULL;  
CREATE DOMAIN EMAIL AS VARCHAR;  
CREATE DOMAIN PASSWORD AS VARCHAR;  
CREATE DOMAIN INTGEZ AS INTEGER CHECK(VALUE > 0);  
CREATE DOMAIN INTGET AS INTEGER CHECK(VALUE >= 0);  
CREATE DOMAIN REALGEZ AS REAL CHECK(VALUE > 0);  
CREATE DOMAIN REALGET AS REAL CHECK(VALUE >= 0);  
CREATE DOMAIN CODBARRE AS VARCHAR;
```

Diagramma UML delle classi ristrutturato



Breve descrizione delle scelte effettuate durante la ristrutturazione

FUSIONE SU ALIMENTO

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)

$\forall \text{ALIMENTO}, \text{CREA} \rightarrow \text{SISTEMA}$

$\forall a, u \text{ ALIMENTO}(a) \wedge \text{SISTEMA}(a, \text{FALSE}) \leftrightarrow \exists u \text{ ALICREA}(a, u)$

$\forall \text{ALIMENTO}, \text{VALIDA} \rightarrow \text{SISTEMA}$

$\forall a, u \text{ VALIDA}(a, u) \rightarrow \text{SISTEMA}(a, \text{TRUE})$

$\forall \text{ALIMENTO}, \text{CREA} \rightarrow \text{NOT VALIDA}$

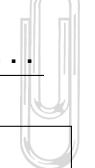
$\forall a, u \text{ ALICREA}(a, u) \rightarrow \neg \text{VALIDA}(a, u)$

$\forall \text{ALIMENTO}, \text{VALIDA} \rightarrow \text{NOT CREA}$

$\forall a, u, ic, iv \text{ ALIMENTO}(a) \wedge \text{ISTCREA}(a, ic) \wedge \text{ISTVALIDA}(a, u, iv) \rightarrow ic < iv$

$\forall \text{ALIMENTO}, \text{INVALIDA} \rightarrow \text{NOT VALIDA}$

$\forall a, u, iv \text{ ALIMENTO}(a) \wedge \text{INVALIDA}(a, u) \wedge \text{ISTINV}(a, u, iv) \rightarrow \text{VALIDO}(a, u, iv, \text{TRUE})$



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	<u>UTENTE</u> ... (nome)	Derivante da: classe associazione (cerchiare)
--------------------	--------------------------	---

Attributi	<u>ID</u>	NOME	COGNOME	E-MAIL	PASS	NAZIONE	
-----------	-----------	------	---------	--------	------	---------	--

Domini	<u>SERIAL</u>	ISTRINGA	ISTRINGA	EMAIL	PASSWORD	ISTRINGA	
--------	---------------	----------	----------	-------	----------	----------	--

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)

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

2 Relazione	<u>NAZIONE</u> ... (nome)	Derivante da: classe associazione (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 associazioni:

3 Relazione	<u>ABBONAMENTO</u> (nome)	Derivante da: classe associazione (cerchiare)
--------------------	---------------------------	---

Attributi	<u>ID</u>	INIZIOABB	FINEABB	UTENTE			
-----------	-----------	-----------	---------	--------	--	--	--

Domini	<u>SERIAL</u>	DATE	DATE	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(ID)

CHECK(INIZIOABB <= FINEABB)

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

4 Relazione	<u>ALIMENTO</u> ... (nome)	1	Derivante da: classe associazione (cerchiare)
--------------------	----------------------------	---	---

Attributi	<u>ID</u>	NOME	MARCA*	GRANDEZZA	CODICE*	KCAL	PORZIONE	
-----------	-----------	------	--------	-----------	---------	------	----------	--

Domini	<u>SERIAL</u>	ISTRINGA	ISTRINGA	REALGZ	CODBARRE	INTGZ	REALGZ	
--------	---------------	----------	----------	--------	----------	-------	--------	--

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	<u>ALIMENTO</u> (nome)	2	Derivante da: classe associazione (cerchiare)
--------------------	-----------------------------	---	---

Attributi	<u>ISTCREA</u>	SISTEMA	UNITA'	UTECREA*				
-----------	----------------	---------	--------	----------	--	--	--	--

Domini	DATE	BOOL	ISTRINGA	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(UNITA') REF UNITADMISURA(nome)

CHECK((SISTEMA)=(UTECREA IS NULL))

FK(UTECREA) REF UTENTE(ID)

La relazione accorda le relazioni che implementano le seguenti associazioni: ALIVUNI, ALICREA.....

6 Relazione ..VALIDA... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>UTENTE</u> ALIMENTO <u>LIST VAL</u>	
Domini INTEGER INTEGER 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):

FK(UTENTE) REF UTENTE(ID)

FK(ALIMENTO) REF ALIMENTO(ID)

La relazione accorda le relazioni che implementano le seguenti associazioni:

7 Relazione !VALIDA... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>UTENTE</u> ALIMENTO <u>LIST INV</u>	
Domini INTEGER INTEGER 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):

FK(UTENTE) REF UTENTE(ID)

FK(ALIMENTO) REF ALIMENTO(ID)

La relazione accorda le relazioni che implementano le seguenti associazioni:

8 Relazione ..PASTO..... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>ID</u> <u>LIST PAS</u> TIPO UTENTE	
Domini SERIAL DATE 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(TIPO) REF TIPOLOGIA_PAS(NOME)

FK(UTENTE) REF UTENTE(ID)

La relazione accorda le relazioni che implementano le seguenti associazioni: ..PASTIP.., ..PASUTE.....

9 Relazione ...ALIMENTO... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>ALIMENTO</u> PASTO <u>IMPORT</u>	
Domini INTEGER INTEGER REALGZ	

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(ALIMENTO) REF ALIMENTO(ID)

FK(PASTO) REF PASTO(ID)

La relazione accorda le relazioni che implementano le seguenti associazioni:

10 Relazione ..NUTRIENTE... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>NOME</u> OBBL	
Domini STRINGA BOOL	

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:

11 Relazione ...ALI.N.V.T.... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>ALIMENTO</u> <u>NUTRIENTE</u> <u>QUANTGR</u>	
Domini <u>INTEGER</u> <u>STRINGA</u> <u>REALGEZ</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 (ALIMENTO) REF ALIMENTO (ID)

FK (NUTRIENTE) REF NUTRIENTE (NAME)

La relazione accorda le relazioni che implementano le seguenti associazioni:

12 Relazione .DIETA..... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>ID</u> <u>NAME</u> <u>KCALMIN</u> <u>KCALMAX*</u>	
Domini <u>SERIAL</u> <u>STRINGA</u> <u>INTGEZ</u> <u>INTGEZ</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):

CHECK (KCALMAX > KCALMIN)

La relazione accorda le relazioni che implementano le seguenti associazioni:

13 Relazione .DURATA.DIE... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>ID</u> <u>INIZIODIE</u> <u>FINEDIE*</u> <u>UTENTE</u> <u>DIETA</u>	
Domini <u>SERIAL</u> <u>DATE</u> <u>DATE</u> <u>INTEGER</u> <u>INTEGER</u>	

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

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

FK (UTENTE) REF UTENTE (ID)

FK (DIETA) REF DIETA (ID)

CHECK (INIZIODIE < FINEDIE)

La relazione accorda le relazioni che implementano le seguenti associazioni: .DUR.UYE..., .DIE.DUR.....

14 Relazione ...DIENUT... (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>DIETA</u> <u>NUTRIENTE</u> <u>MINGR</u> <u>MAXGR*</u>	
Domini <u>INTEGER</u> <u>STRINGA</u> <u>REALGEZ</u> <u>REALGEZ</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 (DIETA) REF DIETA (ID)

CHECK (MAXGR > MINGR)

FK (NUTRIENTE) REF NUTRIENTE (NAME)

La relazione accorda le relazioni che implementano le seguenti associazioni:

15 Relazione TIPOLOGIAS. (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>NOME</u>	
Domini <u>STRINGA</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 associazioni:

16 Relazione	<u>UNITA' DI MISURA</u> (nome)	Derivante da:	classe	associazione (cerchiare)
Attributi	<u>NOME</u>			
Domini	<u>STRINGA</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 associazioni:

17 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:

18 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:

19 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:

20 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:

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 ennupple); (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_UTENTE_DIETE_DISGIUNTE

- INSERT OR UPDATE DURATA_DIE

VALID = NOT EXIST (SELECT * FROM DURATA_DIE

WHERE ID > NEW.ID AND (INIZIodie, FINEdie) OVERLAPS (NEW.INIZIodie, NEW.FINEdie))

IF VALID:

 COMMIT

ELSE:

 ERROR

T_ALIMENTO_VALIDA_CONTROLLI

- INSERT OR UPDATE VALIDA

- ERROR = EXIST (SELECT * FROM ALIMENTO WHERE ID = NEW.ALIMENTO

AND (UTCREA = NEW.UTENTE OR ISTCREA > NEW.ISTVAL OR SISTEMA = TRUE
OR INVALIDO(ID, NEW.ISTVAL) = TRUE))

IF ERROR:

 ERROR

ELSE:

 COMMIT

T_ALIMENTO_INVALIDA_SE_VALIDO

- INSERT OR UPDATE INVALIDA

- VALID = SELECT VALIDO(NEW.ALIMENTO, NEW.ISTINV)

IF VALID:

 COMMIT

ELSE:

 ERROR

T_NUTRIENTE_OBLIGATORIO

- INSERT OR UPDATE ALIMENTO

- ERROR = EXIST (SELECT NOME FROM NUTRIENTE
WHERE OBLG = TRUE
EXCEPT
SELECT NUTRIENTE FROM ALINUT
WHERE ALIMENTO = NEW.ID)

IF ERROR: ERROR

ELSE: COMMIT

[continua alla pagina seguente]

Risposta alla Domanda 7 (segue)

T. PASTO, ALIMENTOVALIDO

- INSERT OR UPDATE ALIPAS

~ VALID := EXIST (WITH P AS (SELECT * FROM PASTO WHERE ID = NEW, PASTO)
 SELECT * FROM ALIMENTO a, VALIDA v
 WHERE a.ID = NEW.ALIMENTO AND v.ALIMENTO = a.ID
 AND (a.UTECREA = p.UTENTE OR VALIDO(a.ID, p.ISTPAS))
 OR (v.UTENTE = p.UTENTE AND v.ISTVAL <= p.ISTPAS AND NOT INVALIDO(a.ID, p.ISTPAS)))

IF VALID;

COMMIT

ELSE:

ERROR

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

ALIMENTOCOMUNE ($t: \text{STRINGA}$) ; ($\text{INTEGER}, \text{REAL} G2$) [$0..*$]

VALID = EXIST (SELECT * FROM PASTO WHERE TIPO = t)

$R = \text{WITH } A \text{ AS} (\text{SELECT op.ALIMENTO, COUNT(p.ID) } n, \text{SUM(op.NPORZ)} \text{ taf} \text{ FROM PASTO p, ALIPAS op}$
 $\text{WHERE p.TIPO} = t \text{ AND op.PASTO} = p.ID$
 $\text{GROUP BY op.ALIMENTO})$

SELECT ALIMENTO, taf/n FROM A
WHERE $n = (\text{SELECT MAX}(n) \text{ FROM A})$

RESULT = R

NUTRIENTIMANCANTI ($u: \text{INTEGER}, d: \text{DATE}$) : ($\text{STRINGA}, \text{REAL}$) [$0..*$]

VALID = EXIST (SELECT * FROM ABBONAMENTO a, DURATADIE dd
WHERE a.UTENTE = dd.UTENTE AND a.INIZIOABB <= d AND a.FINEABB >= d
AND dd.INIZIODIE <= d AND dd.FINEDIE >= d)

$R = \text{WITH } Q \text{ AS} (\text{SELECT am.NUTRIENTE, SUM(op.NPORZ * am.QUANTGR)} \text{ taf}$
 $\text{FROM PASTO p, ALIPAS op, ALINUT am}$
 $\text{WHERE p.ID} = op.PASTO \text{ AND op.ALIMENTO} = am.ALIMENTO \text{ AND p.UTENTE} = u$
 $\text{AND p.ISTPAS} = d)$

$D \text{ AS} (\text{SELECT dm.NUTRIENTE, dm.MAXGR FROM DURATA DIE dd, DIENUT dm}$
 $\text{WHERE dd.DIETA} = dm.DIETA \text{ AND dd.UTENTE} = u$
 $\text{AND dd.INIZIODIE} <= d \text{ AND dd.FINEDIE} >= d \text{ AND}$
 $dm.MAXGR > NULL)$

SELECT Q.NUTRIENTE, D.MAXGR - Q.taf FROM Q, D
WHERE Q.NUTRIENTE = D.NUTRIENTE

RETURN R

Risposta alla Domanda 8 (segue)

CREATE FUNCTION INVALIDO(a: INTEGER, d: DATE): BOOL

$R = (\text{SELECT COUNT}(u.\text{id}) \text{ C FROM INVALIDA}$
 $\text{WHERE ISTINV} \leq d \text{ AND ALIMENTO} = a)$

RESULT = ($R.C \geq 100$)

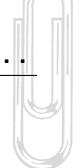
CREATE FUNCTION VALIDO(a: INTEGER, d: DATE): BOOL

ERROR = INVALIDO(a, d) OR EXIST (SELECT * FROM ALIMENTO WHERE ID = a AND
 $SISTEMA = \text{TRUE}$)

$R = (\text{SELECT COUNT}(u.\text{id}) \text{ C FROM VALIDA}$
 $\text{WHERE ISTVAL} \leq d \text{ AND ALIMENTO} = a)$

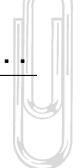
RESULT = ($R.C \geq 10$)

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.]