

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: Cazzu Marco Matricola: 204 62 12

Data: 09/06/2024

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:



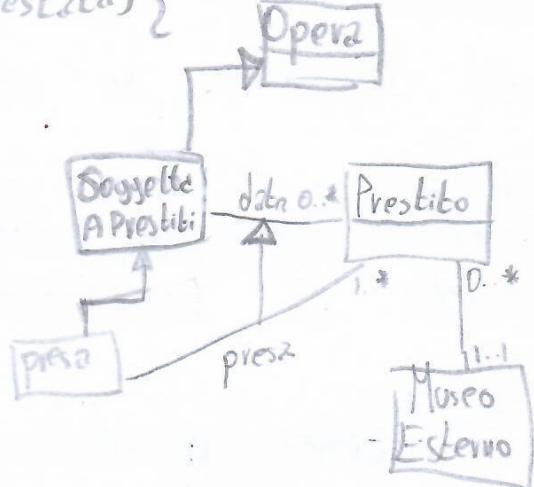
Questo modulo è ottimizzato per la stampa fronte-retro

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

01. Opera
- 01.1 nome
 - 01.2 categoria
 - 01.3 autore (Req. 2)
 - 01.4 anno realizzazione
 - 01.5 tecnica
 - 01.6 convenzioni artistiche
 - 01.7 Proprietaria (può essere prestata) } prestabile? Sì o No
 - 01.8 Presa In Prestito
 - 01.9 Museo origine



5. Restauro

- 5.1 opera
- 5.2 inizio
- 5.3 fine (> inizio) [0..1]

Prestito

- 3. Restauro
- 3.1 opera
- 3.2 inizio
- 3.3 fine

6. Biglietto

full access permanenti {disj, compl}
perm+temp

Data

4. Esposizione

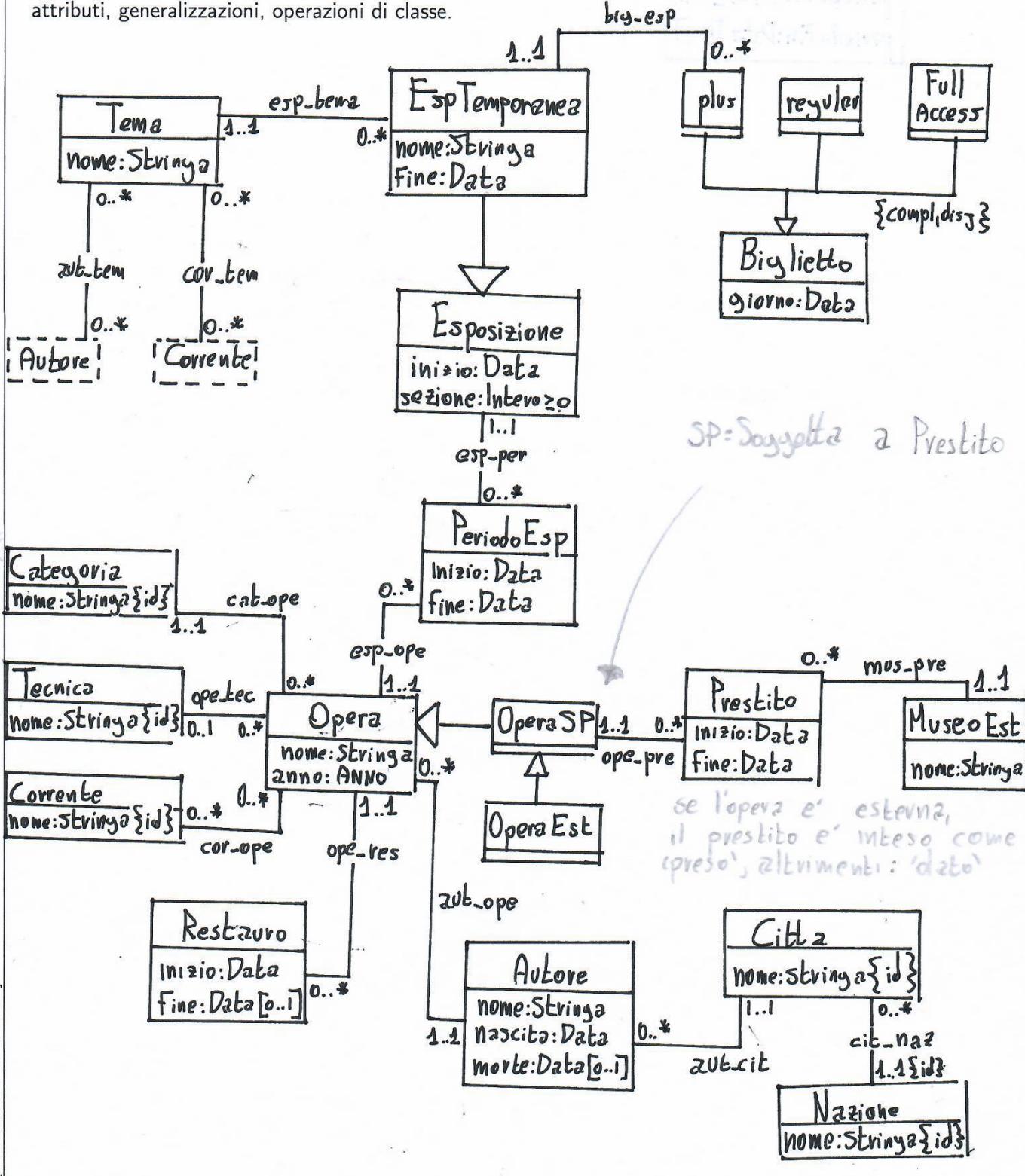
- 4.1 opere } periodo storico?
- 4.2 raggruppamento }
- 4.3 inizio esposizione
- Tempo? ne?
- 4.4 fine (maggiorre di inizio)
- 4.5 Tema
- 4.6 nome

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)



Esposizione ha anche
questi 2 attributi

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'attributo 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 $\text{op}(c, \arg_1, \dots, \arg_n, v)$: uno dei valori di ritorno di op , quando invocata sull'istanza c e con argomenti \arg_1, \dots, \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: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli: <u>[V.realizzata_durante_vita]</u></p> $\forall \sigma, p, a, dn$ $[\text{Opera}(\sigma) \wedge \text{Autore}(p) \wedge \text{anno}(\sigma, a) \wedge \text{aut-ope}(p, \sigma) \wedge \text{nascita}(p, dn)] \rightarrow$ $[\exists an \text{ Anno}(dn, an) \wedge an \leq a \wedge [\forall dm, am \text{ morte}(p, dm) \wedge \text{Anno}(dm, am)] \rightarrow am \geq a]$	<p>2 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli: <u>[V.prestito_dopo_realizzazione]</u></p> $\forall \sigma, p, ip, a, ap$ $[\text{Opera}(\sigma) \wedge \text{Prestito}(p) \wedge \text{inizio}(p, ip) \wedge \text{anno}(\sigma, a) \wedge \text{ope-pre}(\sigma, p) \wedge \text{Anno}(ip, ap)] \rightarrow ap \geq an$ $[V.restavro_dopo_realizzazione]$ $\forall \sigma, r, ir, a, ar$ $[\text{Opera}(\sigma) \wedge \text{Restavro}(r) \wedge \text{inizio}(r, ir) \wedge \text{anno}(\sigma, a) \wedge \text{ope-res}(\sigma, r) \wedge \text{Anno}(ir, ar)] \rightarrow ar \geq a$
--	--

②: significa che il simbolo è stato modificato durante la ristrutturazione.

<p>3 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli:</p> <p>[V.esposta_dopo_realizzata]</p> $\forall \sigma, p, i_p, a, ap \ [Opera(\sigma) \wedge \text{anno}(\sigma, a) \wedge \text{PeriodoEsp}(p) \wedge \text{inizio}(p, i_p) \wedge \text{Anno}(i_p, ap) \wedge \text{esp_ope}(\sigma, p)] \rightarrow ap \geq a$ <p>[V.no_esposizioni_contemporanee]</p> $\forall \sigma, e1, e2, i1, i2, f1, f2 \ [Opera(\sigma) \wedge e1 \neq e2 \wedge \text{PeriodoEsp}(e1) \wedge \text{PeriodoEsp}(e2) \wedge \text{esp_ope}(e1, \sigma) \wedge \text{esp_ope}(e2, \sigma) \wedge \text{inizio}(e1, i1) \wedge \text{fine}(e1, f1) \wedge \text{inizio}(e2, i2) \wedge \text{fine}(e2, f2)] \rightarrow [f2 < i1 \vee f1 < i2]$	<p>6 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli:</p> <p>[V.no_prestiti_contemporanei]</p> $\forall \sigma, p1, p2, i1, i2, f1, f2 \ [Opera(\sigma) \wedge p1 \neq p2 \wedge \text{Prestito}(p1) \wedge \text{Prestito}(p2) \wedge \text{ope_pre}(\sigma, p1) \wedge \text{ope_pre}(\sigma, p2) \wedge \text{inizio}(p1, i1) \wedge \text{fine}(p1, f1) \wedge \text{inizio}(p2, i2) \wedge \text{fine}(p2, f2)] \rightarrow [f1 < i2 \vee f2 < i1]$
<p>4 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli:</p> <p>[V.no_restauri_contemporanei]</p> $\forall \sigma, r1, r2, i1, i2 \ [Opera(\sigma) \wedge \text{Restauro}(r1) \wedge \text{Restauro}(r2) \wedge \text{ope_res}(\sigma, r1) \wedge \text{ope_res}(\sigma, r2) \wedge r1 \neq r2 \wedge \text{inizio}(r1, i1) \wedge \text{inizio}(r2, i2)] \rightarrow \neg \exists t \ \text{Data}(t) \wedge t \geq i1 \wedge [\forall f \ \text{fine}(r1, f) \rightarrow t \leq f] \wedge t \geq i2 \wedge [\forall f \ \text{fine}(r2, f) \rightarrow t \leq f]$	<p>7 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Autore</u></p> <p>Operazioni, vincoli:</p> <p>[V.nasce_poi_muore]</p> $\forall z, n, m \ [Autore(z) \wedge \text{nasce}(z, n) \wedge \text{morte}(z, m)] \rightarrow n \leq m$
<p>5 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli: esposizioni</p> <p>[V.restauro_e_prestito_disgiunti]</p> $\forall \sigma, r, e, ir, ie, fe \ [Opera(\sigma) \wedge \text{Restauro}(r) \wedge \text{PeriodoEsp}(e) \wedge \text{inizio}(e, ie) \wedge \text{inizio}(r, ir) \wedge \text{fine}(e, fe) \wedge \text{esp_ope}(\sigma, e) \wedge \text{ope_res}(\sigma, r)] \rightarrow \neg \exists t \ \text{Data}(t) \wedge t \geq ie \wedge t \geq ir \wedge t \leq fe \wedge [\forall fr \ \text{fine}(r, fr) \rightarrow t \leq fr]$	<p>8 Tipo: Classe Associazione (cerchiare)</p> <p>Nome: <u>Opera</u></p> <p>Operazioni, vincoli:</p> <p>[V.prestata_se_non_restaureta] R</p> $\forall \sigma, p, r, ip, fp, ir \ [Opera(\sigma) \wedge \neg \text{OperaEst}(\sigma) \wedge \text{Prestito}(p) \wedge \text{Restauro}(r) \wedge \text{inizio}(r, ir) \wedge \text{fine}(p, fp) \wedge \text{inizio}(p, ip) \wedge \text{ope_res}(\sigma, r) \wedge \text{ope_pre}(\sigma, p)] \rightarrow \neg \exists t \ \text{Data}(t) \wedge t \geq ip \wedge t \geq ir \wedge t \leq fp \wedge [\forall fr \ \text{fine}(r, fr) \rightarrow t \leq fr]$

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

$[V.\text{inizio_poi_fino}] // Sfrutta gli attributi di nome identico$

$\forall K, i, f [[Restauro(K) \vee Prestito(K) \vee PeriodoEsp(K) \vee Esposizione(K)] \wedge \text{inizio}(K, i) \wedge \text{fine}(K, f)] \rightarrow i \leq f$

Classe Opera

$[V.\text{prestata_se_non_esposta}]$

(R)

$\forall \sigma, p, e, i_p, f_p, ie, fe [Opera(\sigma) \wedge \neg \text{OperaEst}(\sigma) \wedge \text{Prestito}(p) \wedge \text{ope-pre}(\sigma, p) \wedge \text{inizio}(p, i_p) \wedge \text{fine}(p, f_p) \wedge \text{PeriodoEsp}(p) \wedge \text{inizio}(e, ie) \wedge \text{fine}(e, fe) \wedge \text{esp-ope}(e, \sigma)] \rightarrow [f_p < ie \vee fe < i_p]$

$[V.\text{restaurata_se_in_prestito}]$

(R)

$\forall \sigma, r, i_r [Opera(\sigma) \wedge \text{OperaEst}(\sigma) \wedge \text{Restauro}(r) \wedge \text{inizio}(r, i_r) \wedge \text{ope-res}(\sigma, r)] \rightarrow \exists p, i_p, f_p [\text{Prestito}(p) \wedge \text{inizio}(p, i_p) \wedge \text{fine}(p, f_p) \wedge \text{ope-pre}(\sigma, p) \wedge i_r \geq i_p \wedge [\forall fr \text{ fine}(r, fr) \rightarrow fr \leq f_p]]$

$[V.\text{esposta_se_in_prestito}]$

(R)

$\forall \sigma, e, ie, fe [\text{OperaEst}(\sigma) \wedge \text{PeriodoEsp}(e) \wedge \text{inizio}(e, ie) \wedge \text{fine}(e, fe) \wedge \text{esp-ope}(e, \sigma)] \rightarrow [\exists p, i_p, f_p [\text{Prestito}(p) \wedge \text{inizio}(p, i_p) \wedge \text{fine}(p, f_p) \wedge \text{ope-pre}(\sigma, p) \wedge i_p \leq ie \wedge f_p \leq fe]]$

Classe Esposizione

$[V.\text{presenza_periodo_storico}]$

$\forall e \text{ Esposizione}(e) \rightarrow [\exists K \text{ PeriodoIn}(e, K) \leftrightarrow \exists K' \text{ PeriodoFin}(e, K')]$

$[V.\text{esposta_durante_esposizione}]$ espresso nel periodo di esposizione (R)

$\forall e, pe, ie, ipe, Epe [\text{Esposizione}(e) \wedge \text{PeriodoEsp}(pe) \wedge \text{esp-per}(e, pe) \wedge \text{inizio}(e, ie) \wedge \text{inizio}(pe, ipe) \wedge \text{fine}(pe, Epe)] \rightarrow ie \leq ipe \wedge [\forall fe \text{ fine}(e, fe) \rightarrow fe \leq Epe]$

Classe Biglietto

$[V.\text{data_biglietto_esposizione}]$

(R)

$\forall b, e, db, ie, fe [\text{plus}(b) \wedge \text{Esptemporanea}(e) \wedge \text{big-esp}(b, e) \wedge \text{giorno}(b, db) \wedge \text{inizio}(e, ie) \wedge \text{fine}(e, fe)] \rightarrow ie \leq db \leq fe$

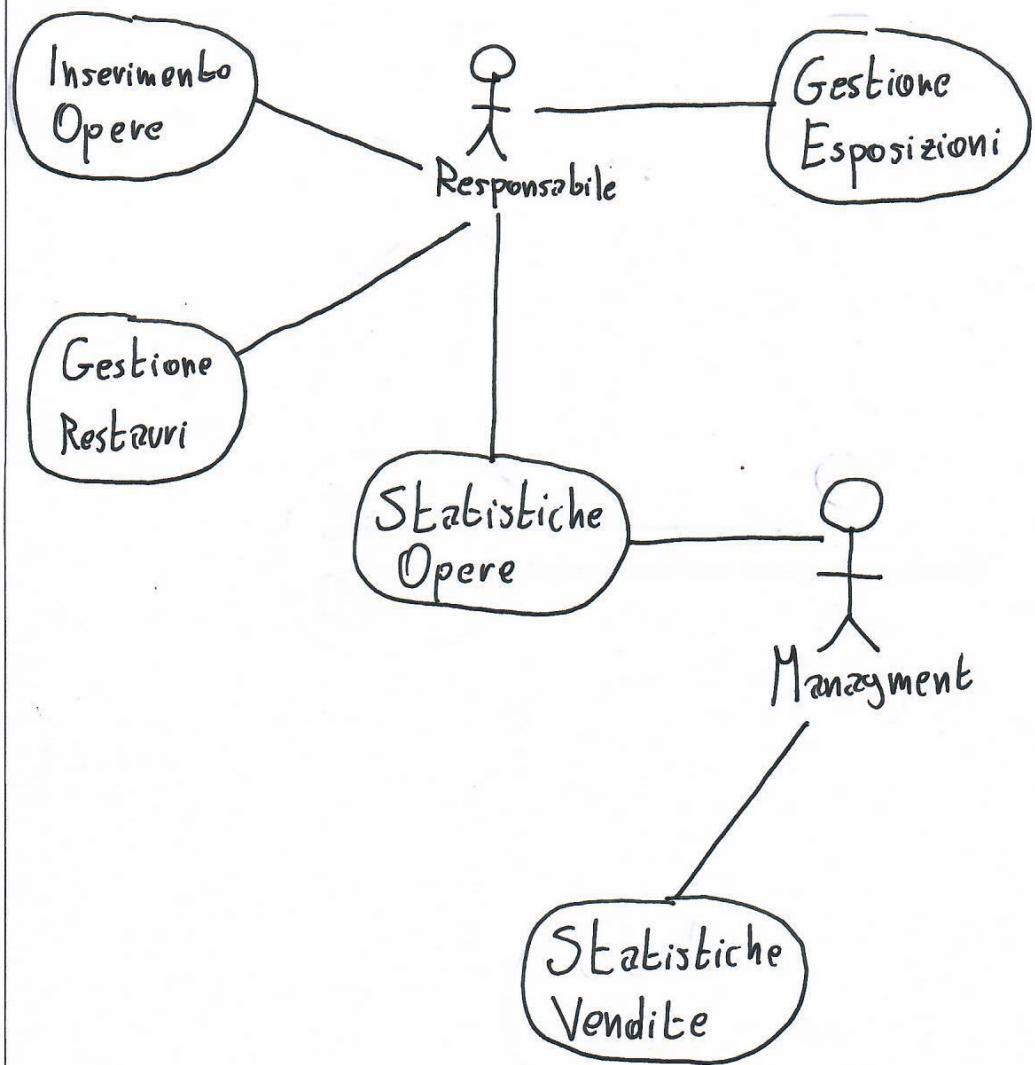
Risposta alla Domanda 2 (segue)

Tipi di Dato

ANNO: Intero che rappresenta un anno (prima o dopo Cristo)
Integer ≥ -8000

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 INSEGNAMENTO OPERE

inserisci_opera(n:Stringa, anno:ANNO, c:Categoria[0..1], t:Technica[0..*], cr:Corrente[0..*],
a:Autore):Opera

rimuovi_opera(o:Opera)

GESTIONE RESTAURI

registra_rest(i:Data, o:Opera):Restauro

termina_rest(f:Data, r:Restauro):Restauro

GESTIONE ESPOSIZIONI

crea_esp_perv(p:PeriodoEsp[0..*], i:Data, s:Sezione, Intervzo, (pi:Data, po:Data)[0..1])
:Esposizione

crea_esp_temp(p:PeriodoEsp[0..*], i:Data, s:Intervzo, (pi:Data, po:Data)[0..1], f:Data,
n:Stringa, t:Tema):Esptemporanea

STATISTICHE VENDITE

biglietti_per_tip(d:Data):(Intervzo, Intervzo, Intervzo)

biglietti_in_periodo(i:Data, f:Data):Reale ≥ 0

esp_moda(i:Data, f:Data):Esptemporanea[0..*]

STATISTICHE OPERE

artisti_per_corrente(i:Data, f:Data):(Corrente, Intervzo)[0..*]

opere_artista_non_ves(i:Data, f:Data, a:Autore):Opera[0..*]

opere_hab_meno_esposte():Opera[0..*]

Domanda 5 (30 minuti; 60 minuti al massimo) Proseguire la fase di Analisi Concettuale dei requisiti producendo le specifiche concettuali per le operazioni di use-case, **limitandosi** a quelle necessarie a modellare i requisiti contrassegnati dalla barra laterale (come quella qui a sinistra), 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

biglietti_per_tip(d:Data): (Intervallo, Intervallo, Intervallo)

- pre-cond: nessuna

- post-cond: $P = \{b \mid plus(b) \wedge giorno(b, d)\}$ $R = \{r \mid regular(r) \wedge giorno(r, d)\}$
 $F = \{f \mid fullAccess(f) \wedge giorno(f, d)\}$

$$\text{Result} = (|P|, |R|, |F|)$$

biglietti_in_periodo(i:Data, f:Data): Reale

- pre-cond: $i \leq f$

- post-cond: $B = \{(g, n) \mid Data(g) \wedge i \leq g \leq f \wedge n = |\{b \mid Biglietto(b) \wedge giorno(b, g)\}| \}$

$$\text{Result} = \sum_{(g,n) \in B} n \cdot \frac{1}{|B|}$$

esp_moda(i:Data, f:Data): EspTemporanea [0..*]

- pre-cond: $i \leq f$

- post-cond: $E = \{(e, n) \mid \begin{array}{l} \text{EspTemporanea}(e) \wedge \\ n = |\{b \mid plus(b) \wedge big-esp(b, e) \wedge \exists g \text{ giorno}(b, g) \wedge i \leq g \leq f\}| \end{array}\}$

$$EM = \arg \max_{(e,n) \in E} (n)$$

$$\text{Result} = \{e \mid (e, n) \in EM\}$$

Risposta alla Domanda 5 (segue)

$\text{Artisti_per_corrente}(i:\text{Data}, f:\text{Data}): (\text{Corrente}, \text{Intervallo}) [0..*]$

• pre-cond: $i \leq f$

• post-cond:

$$C = \left\{ (cr, n) \mid \begin{array}{l} \text{Corrente}(cr) \wedge \\ n = \left| \left\{ a \mid \begin{array}{l} \text{Autore}(a) \wedge \exists \sigma, an, ik, fk \quad \text{Opera}(\sigma) \wedge \text{anno}(\sigma, an) \\ \wedge \text{cor-ope}(cr, \sigma) \wedge \text{aut-ope}(a, \sigma) \wedge \text{Anno}(i, ik) \wedge \\ \text{Anno}(f, fk) \wedge ik \leq an \leq fk \end{array} \right\} \right| \end{array} \right\}$$

Result = C

$\text{Opere_artista_non_res}(i:\text{Data}, f:\text{Data}, a:\text{Autore}): \text{Opera} [0..*]$

• pre-cond: $i \leq f$

• post-cond:

$$O = \left\{ \sigma \mid \begin{array}{l} \text{Opera}(\sigma) \wedge \text{aut-ope}(a, \sigma) \wedge \\ \left[\left[\forall r, ir \quad \text{Restauro}(r) \wedge \text{inizio}(r, ir) \wedge \text{ope-res}(\sigma, r) \right] \rightarrow \right. \\ \left. \neg \exists t \quad \text{Data}(t) \wedge t \geq ir \wedge t \geq i \wedge t \leq f \right. \\ \wedge \left[\forall fr \quad \text{fine}(r, fr) \rightarrow t \leq fr \right] \end{array} \right\}$$

Result = O

$\text{opere_rest_meno_esposte}(): \text{Opera} [0..*]$

• pre-cond: nessuna

• post-cond:

$$O = \left\{ (\sigma, fe) \mid \begin{array}{l} \text{Opera}(\sigma) \wedge \exists r, ir \quad \text{ope-res}(\sigma, r) \wedge \exists k \quad \text{Data}(k, addresso) \wedge \text{inizio}(r, ir) \\ \wedge \left[\forall fr \quad \text{fine}(r, fr) \rightarrow fr \geq k \right] \wedge ir \leq k \wedge \exists pe \quad \text{PeriodoEsp}(pe) \wedge \\ \text{Fine}(pe, fe) \wedge \text{esp-ope}(pe, \sigma) \wedge \neg \exists pe1, fe1 \quad \text{PeriodoEsp}(pe1) \wedge \\ \text{esp-ope}(pe1, \sigma) \wedge \text{fine}(pe1, fe1) \wedge fe1 > fe \end{array} \right\}$$

Result = $\underset{(\sigma, fe) \in O}{\operatorname{argmin}}(fe)$

* continuaz

2 minute 31 e 32

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 Int-GEZ as Integer check(value >= 0);

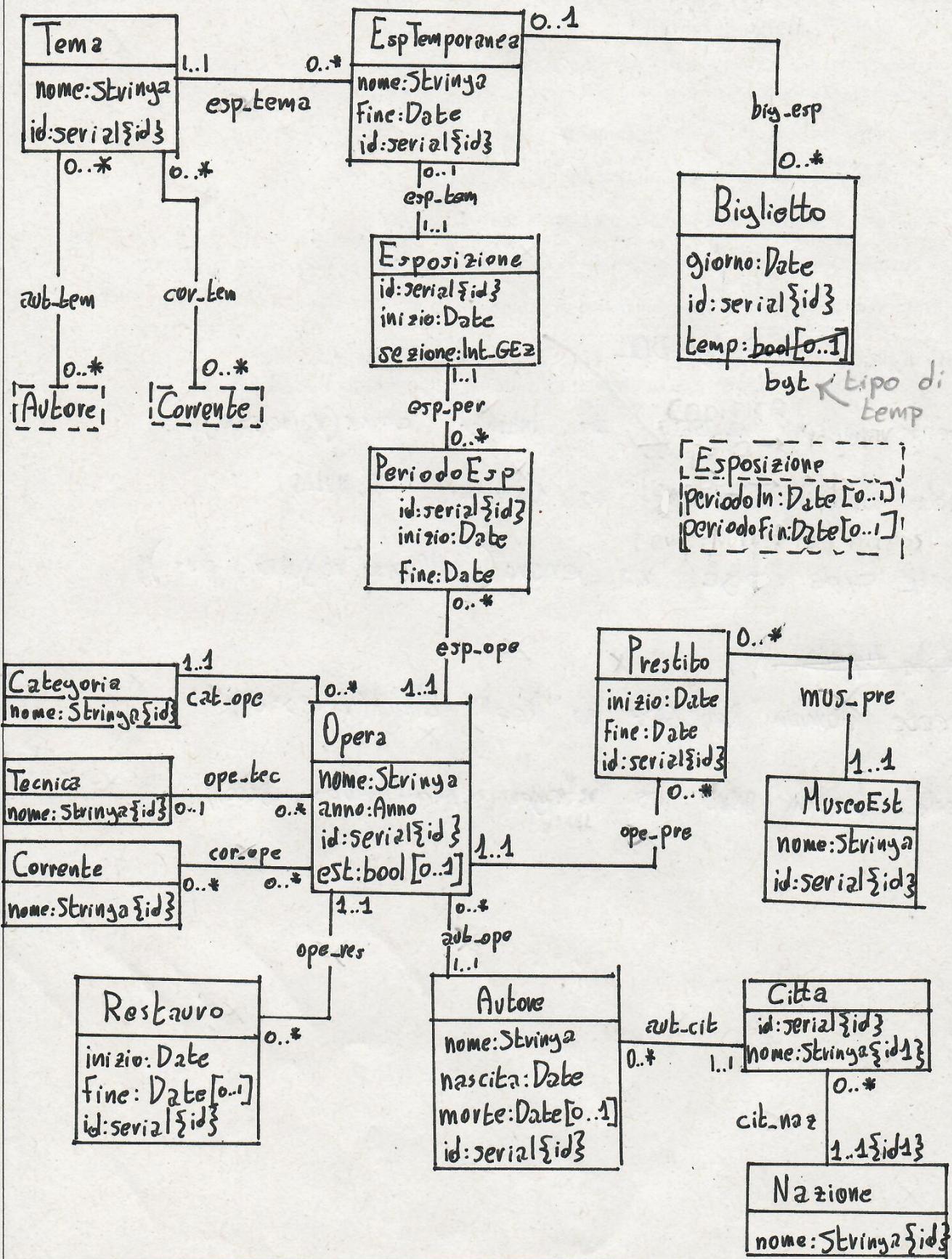
create domain Stringa as varchar NOT NULL;

create type bogt as enum('FullAccess','regular','plus');

create domain Reel-GEZ as Real check(value >= 0);

create domain ANNO as Integer check(value >= -8000);

Diagramma UML delle classi ristrutturato



Breve descrizione delle scelte effettuate durante la ristrutturazione

Fusione su Biglietto

Fusione su Opera

sost. di is-a con associazione su esposizione

utilizzo di bool [0..*] per rappresentare i diversi stati

Opera :

$est=0 \rightarrow$ Interna e prestabile
 $est=1 \rightarrow$ esterna
 $est=NULL \rightarrow$ non prestabile

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.biglietto-plus]
 $\forall b [Biglietto(b)] \rightarrow [temp(b, \text{plus}) \leftrightarrow \exists e \ big-exp(b,e)]$

[V.opera-prestabile]

$\forall \sigma [Opera(\sigma) \wedge \exists p \ Prestito(p) \wedge op-pre(\sigma, p)] \rightarrow \exists k \ est(\sigma, k)$

[V.prestata-se-non-restaurata]

$\forall \sigma, p, r, cp, fp, ir [Opera(\sigma) \wedge est(\sigma, False) \wedge Prestito(p) \wedge Restauro(r) \wedge inizio(r, ir) \wedge fine(p, fp) \wedge inizio(p, cp) \wedge op-res(\sigma, r) \wedge op-pre(\sigma, p)] \rightarrow \neg \exists t \ Data(t) \wedge t \geq cp \wedge t \geq ir \wedge t \leq fp \wedge [\forall fr \ fine(r, fr) \rightarrow t \leq fr]$

[V.prestata-se-non-exposta]

$\forall \sigma, p, e, cp, fp, ie, fe [Opera(\sigma) \wedge est(\sigma, False) \wedge Prestito(p) \wedge op-pre(\sigma, p) \wedge inizio(p, cp) \wedge fine(p, fp) \wedge PeriodoExp(e) \wedge inizio(e, ie) \wedge fine(e, fe) \wedge esp-op(e, \sigma)] \rightarrow [fp < ie \vee fe < cp]$

Risposta alla Domanda 6 (segue)

[V. restaurata_se_in_prestito]

$\forall \sigma, r, ir [Opera(\sigma) \wedge est(\sigma, True) \wedge Restauro(r) \wedge inizio(r, ir) \wedge ope-res(\sigma, r)]$

$\rightarrow \exists p, cp, fp [Prestito(p) \wedge inizio(p, cp) \wedge fine(p, fp) \wedge ope-pre(\sigma, p) \wedge ir \geq cp \wedge \forall fr [fine(r, fr) \rightarrow fr \leq fp]]$

[V. esposta_se_in_prestito]

$\forall \sigma, e, ie, fe [Opera(\sigma) \wedge est(\sigma, True) \wedge inizio(e, ie) \wedge PeriodoEsp(e) \wedge fine(e, fe) \wedge esp-ope(e, \sigma)] \rightarrow [\exists p, cp, fp [Prestito(p) \wedge inizio(p, cp) \wedge fine(p, fp) \wedge ope-pre(\sigma, p) \wedge cp \leq ie \wedge fp \geq fe]]$

[V. dato_biglietto_esposizione]

$\forall b, e, db, ie, fe [Biglietto(b) \wedge temp(b, \cancel{free}) \wedge EspTemporanea(e) \wedge big-esp(b, e) \wedge giorno(b, db) \wedge inizio(e, ie) \wedge fine(e, fe)] \rightarrow ie \leq db \leq fe$

[V. esposta_durante_esposizione]

$\forall e, pe, ie, ipe, fpe [Esposizione(e) \wedge PeriodoEsp(pe) \wedge esp-per(e, pe) \wedge inizio(e, ie) \wedge inizio(pe, ipe) \wedge fine(pe, fpe)] \rightarrow ie \leq ipe \wedge$

$[\forall et, fe [EspTemporanea(et) \wedge esp-tem(e, et) \wedge fine(e, fe)] \rightarrow fe \geq fpe]$

[V. inizio_esp_fine]

$\forall e, ie, et, fe$

$[Esposizione(e) \wedge inizio(e, ie) \wedge esp-ter(e, et) \wedge fine(et, fe)] \rightarrow fe \geq ie$

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>Categoria</u> (nome)	Derivante da: <u>classe</u> 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:

2 Relazione <u>Tecnica</u> (nome)	Derivante da: <u>classe</u> 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:

3 Relazione <u>Corrente</u> (nome)	Derivante da: <u>classe</u> 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:

4 Relazione <u>Nazione</u> (nome)	Derivante da: <u>classe</u> 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:

5 Relazione <u>Città</u> (nome)	Derivante da: <u>classe</u> associazione (cerchiare)						
Attributi <u>nome</u> <u>id</u> <u>nazione</u>							
Domini <u>Stringa</u> <u>Serial</u> <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):

foreign key nazione references Nazione(nome);

Unique(nome,nazione);

La relazione accorda le relazioni che implementano le seguenti associazioni:

foreign key \approx FK \sim references \approx ref abbreviazione

6 Relazione Autore (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>nome</u> <u>nascita</u> <u>morte*</u> <u>id</u> <u>città</u>	
Domini Stringa Date Date serial 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 città ref Città(id);
check(nascita <= morte);

La relazione accorda le relazioni che implementano le seguenti associazioni: ...aut-ope...

7 Relazione Opera (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>nome</u> <u>anno</u> <u>id</u> <u>est*</u> <u>autore</u> <u>cat</u> <u>tecnica*</u>	
Domini Stringa ANNO serial bool Integer 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 autore ref Autore(id);

fk cat ref Categoria(nome); fk tecnica ref Tecnica(nome);

La relazione accorda le relazioni che implementano le seguenti associazioni: ...aut-ope, cat-ope, ope-tec...

8 Relazione cor_ope (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>corrente</u> <u>ope</u>	
Domini nome 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 ope ref Opera(id);

fk corrente ref Corrente(nome);

La relazione accorda le relazioni che implementano le seguenti associazioni:

9 Relazione Restauro (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>id</u> <u>inizio</u> <u>Fine*</u> <u>ope</u>	
Domini serial 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):

check(inizio <= fine);

fk ope ref Opera(id);

La relazione accorda le relazioni che implementano le seguenti associazioni: ...ope-kes...

10 Relazione Museo_Est (nome)	Derivante da: classe associazione (cerchiare)
Attributi <u>id</u> <u>nome</u>	
Domini serial 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:

11 Relazione Presti (nome)	Derivante da: <u>classe</u> associazione (cerchiare)
Attributi <u>inizio</u> fine id opera Museo		
Domini Date Date serial 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 (inizio <= Fine); fk opera ref Opera(id);

La relazione accorda le relazioni che implementano le seguenti associazioni: *ope-pre, mus-pre*

12 Relazione Esposizione (nome)	Derivante da: <u>classe</u> associazione (cerchiare)
Attributi <u>id</u> inizio sezione periodoIN periodoFIN		
Domini serial Date Int-GEZ Date 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):

*check (periodoIN is NULL AND periodoFIN is NULL) OR
(periodoIN is NOT NULL AND periodoFIN is NOT NULL);*

La relazione accorda le relazioni che implementano le seguenti associazioni:

13 Relazione PeriodoEsp (nome)	Derivante da: <u>classe</u> associazione (cerchiare)
Attributi <u>id</u> inizio fine opera exposizione		
Domini serial Date Date 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(inizio <= fine); fk exposizione ref Esposizione(id);
fk opera ref Opera(id);*

La relazione accorda le relazioni che implementano le seguenti associazioni: *esp-ope, esp-per*

14 Relazione Tema (nome)	Derivante da: <u>classe</u> associazione (cerchiare)
Attributi nome <u>id</u>		
Domini Stringa 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):

La relazione accorda le relazioni che implementano le seguenti associazioni:

15 Relazione EspTempome (nome)	Derivante da: <u>classe</u> associazione (cerchiare)
Attributi <u>id</u> nome fine tema esposizione		
Domini serial Stringa Date 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):

FK esposizione ref Esposizione(id); Unique(esposizione);

Fk tema ref Tema(id);

La relazione accorda le relazioni che implementano le seguenti associazioni: *esp-tempo, esp-tem*

16 Relazione <u>Bisielto</u> (nome)	Derivante da: <u>classe</u> <u>associazione</u> (cerchiare)
Attributi <u>dorno</u> <u>id</u>	<u>esposizione</u> <u>temp</u>
Domini <u>Date</u> <u>serial</u>	<u>Integer</u> <u>byte</u>
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *	
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio): <u>plus</u> fk esposizione ref EspTemporanea(id); Check((temp > 0 AND esposizione IS NOT NULL) OR (temp < 0 AND esposizione IS NULL))	
La relazione accorda le relazioni che implementano le seguenti associazioni: <u>big-exp</u>	

17 Relazione <u>aut_temp</u> (nome)	Derivante da: <u>classe</u> <u>associazione</u> (cerchiare)
Attributi <u>temp</u>	<u>Autore</u>
Domini <u>Integer</u>	<u>Integer</u>
Gli attributi chiave primaria sono sottolineati, quelli i cui valori possono essere NULL sono contrassegnati con *	
Vincoli (foreign key, inclusione, altra chiave, di ennupla, di dominio):	
fk tema ref Tema(id); fk autore ref Autore(id);	
La relazione accorda le relazioni che implementano le seguenti associazioni:	

18 Relazione <u>cor_tema</u> (nome)	Derivante da: <u>classe</u> <u>associazione</u> (cerchiare)
Attributi <u>tema</u>	<u>corrente</u>
Domini <u>Integer</u>	<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):	
fk tema ref Tema(id); fk corrente ref Corrente(nome);	
La relazione accorda le relazioni che implementano le seguenti associazioni:	

19 Relazione (nome)	Derivante da: <u>classe</u> <u>associazione</u> (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: <u>classe</u> <u>associazione</u> (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.

TUTTI I TRIGGER SONO CONTROLLATI POST INSERIMENTO

T. realizzata_durante_vita

op: Insert o Update Opera

Q = EXISTS (SELECT *

```
FROM Opera σ JOIN Autore a ON σ.autore = a.id
WHERE σ.id = new.id AND
(σ.anno < EXTRACT(year FROM a.nascita) OR
σ.anno > EXTRACT(year FROM a.morte)));
```

if Q = True: genera errore e rollback

else: permetti operazione

T. prestito_dopo_realizzazione

op: Insert o Update Prestito

Error = EXISTS (SELECT *

```
FROM Opera σ WHERE σ.id = new.opera AND
EXTRACT(year FROM new.inizio) < σ.anno);
```

if Error = True: genera errore e rollback

else: permetti operazione

T. restauro_dopo_realizzazione

op: Insert o Update Restauro

Error = EXISTS (SELECT * FROM Opera σ WHERE new.opera = σ.id AND
EXTRACT(year FROM new.inizio) < σ.anno);

if Error = True: genera errore e rollback

else: permetti operazione

T. asposta_dopo_realizzata

op: Insert o Update PeriodoEsp

Error = EXISTS (SELECT * FROM Opera σ WHERE new.opera = σ.id AND
EXTRACT(year FROM new.inizio) < σ.anno);

if Error = True: genera errore e rollback

else: permetti op.

Risposta alla Domanda 7 (segue)

Trigger. no-restauri- σ -esposti-zioni-contemporanee

op: Insert o Update Restauro o PeriodoEsp

Error = EXISTS (SELECT * FROM Restauro r, PeriodoEsp e WHERE
 $e.\text{opera} = \text{new}.\text{opera}$ AND $r.\text{opera} = \text{new}.\text{opera}$ AND
 $e.id <> \text{new}.id$ AND $r.id <> \text{new}.id$ AND
 $((e.\text{inizio}, e.\text{fine}) \text{ OVERLAPS } (\text{new}.\text{inizio}, \text{new}.\text{fine})) \text{ OR }$
 $((r.\text{inizio}, r.\text{fine}) \text{ OVERLAPS } (\text{new}.\text{inizio}, \text{new}.\text{fine}))$);

if Error=True: errore e rollback

Else: permetti op.

Trigger. no-presbiti-contemporanei

op: Insert o Update Prestito

Error = EXISTS (SELECT * FROM Prestito p WHERE $p.\text{opera} = \text{new}.\text{opera}$ AND
 $\text{new}.id <> p.id$ AND $(p.\text{inizio}, p.\text{fine}) \text{ OVERLAPS } (\text{new}.\text{inizio}, \text{new}.\text{fine})$);

if Error=True: errore e rollback

else: permetti op.

T.opera-prestabile

op: Insert o Update Prestito

Error = EXISTS (SELECT *
 FROM Opera σ
 WHERE ($\text{new}.\text{opera} = \sigma$ AND $\sigma.est$ IS NULL));

if Error=True: errore e rollback

else: permetti op

T.opera-presta-se-non-esposta- σ -restaurata

op: Insert o Update Prestito

Error = EXISTS (SELECT Opera σ , Restauro r, PeriodoEsp e WHERE $\text{new}.\text{opera} = \sigma.id$
 AND $\sigma.est = \text{False}$ AND $r.\text{opera} = \sigma.id$ AND $e.\text{opera} = \sigma.id$ AND
 $((\text{new}.id, \text{new}.fine) \text{ OVERLAPS } (e.\text{inizio}, e.\text{fine})) \text{ OR } ((\text{new}.\text{inizio}, \text{new}.\text{fine}) \text{ OVERLAPS } (r.\text{inizio}, r.\text{fine}))$);

if Error=True: errore e rollback, else: permetti op

* CONTINUA A MINUTE 32 e 33

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

biglietti_per_tip(d:Date): Insieme(<biglietto, Int-GEZ>)

* Q = SELECT temp, count(*)
FROM Biglietto
WHERE giorno = d
GROUP BY temp;

Result = Q

biglietti_in periodo(i:Date, f:Date): Real-GEZ

* Q = WITH BG as (SELECT giorno, count(*) as num
FROM Biglietto
WHERE giorno >= i AND giorno <= f
GROUP BY giorno)

SELECT AVG(num) FROM BG;

Result = Q

esp_moda(i:Date, f:Date): Insieme(EspTempo<Integer>)

* Q = WITH EN as (SELECT e.id, count(*) as num
FROM EspTemporanea e, Biglietto b
WHERE e.id = b.esposizione AND
b.giorno >= i AND b.giorno <= f
GROUP BY e.id)

SELECT e.id
FROM EspTemporanea e, EN
WHERE EN.e.id = e.id
AND EN.num = (SELECT max(num) FROM EN)

* if (i > f): termina operazione

Risposta alla Domanda 8 (segue)

$\text{opere_artista_non_res}(i:\text{Date}, f:\text{Date}, a:\text{Integer}): \text{Insieme}(\langle \text{Integer} \rangle)$

if ($i > f$): termina operazione

$Q = (\text{SELECT id FROM Opera WHERE autore} = a)$

EXCEPT

$(\text{SELECT } \sigma.\text{id}$

FROM Opera σ , Restauro r WHERE $\sigma.\text{autore} = a$

AND r.opera = $\sigma.\text{id}$ AND (r.inizio, r.fine) OVERLAPS (i, f)) ;

Result = Q

$\text{artisti_per_corrente}(i:\text{Date}, f:\text{Date}): (\langle \text{Stringa}, \text{Int_GEZ} \rangle) \text{Insieme}$

if ($i > f$): termina operazione

$Q = \text{SELECT co.corrente, COUNT(DISTINCT a.id)}$

FROM Autore a, Opera σ , cor-ope co

WHERE a.id = $\sigma.\text{autore}$ AND co.opera = $\sigma.\text{id}$

AND EXTRACT('year' from i) $\leq \sigma.\text{anno}$

AND EXTRACT('year' from f) $\geq \sigma.\text{anno}$

GROUP BY co.corrente;

Result = Q

$\text{opere_rest_meno_esposte}(): \text{Insieme}(\langle \text{Integer} \rangle)$

$Q = \text{WITH OF AS} (\text{SELECT } \sigma.\text{id}, \text{MAX}(e.\text{fine}) \text{ AS Fine}$

FROM Opera σ , Restauro r, PeriodoEsp e

WHERE $\sigma.\text{id} = r.\text{opera}$

AND NOWC $\geq r.\text{inizio}$ AND

(r.fine IS NULL OR NOWC $\leq r.\text{fine}$) AND

e.opera = $\sigma.\text{id}$

GROUP BY $\sigma.\text{id}$)

$\text{SELECT } \sigma.\text{id}$

FROM Opera σ , OF WHERE $\sigma.\text{id} = \text{OF}.\sigma.\text{id}$ AND

OF.fine = (SELECT MIN(fine) FROM OF);

Result = Q

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.]

Use Case FOL

inserisci_opera(n:Stringa, anno:Anno, C:Categorie, t:Technica[0..1], cr:Corrente[0..*], a:Autore):Opera modifica il livello estensionale

- pre-cond: nessuna

- post-cond: Sia α il nuovo oggetto del dominio tale che:

$\text{Opera}(\alpha) \wedge \text{nome}(\alpha, n) \wedge \text{anno}(\alpha, \text{anno}) \wedge \text{aut-ope}(\alpha, a) \wedge \text{cat-ope}(\alpha, c) \wedge$

$[\text{t} \neq \text{NULL} \rightarrow \text{ope-tec}(\alpha, t)] \wedge [\forall \text{cor cor} \in \text{cr} \rightarrow \text{cor-ope}(\text{cor}, \alpha)]$

Result = α

$M_{\text{out}} = M_{\text{in}} \cup \{\alpha\}$

rimuovi_opera(o) modifica il livello estensionale

- pre-cond: nessuna

- post-cond:
 $R = \{k \mid \begin{array}{l} \text{ope-res}(o, k) \vee \text{ope-pre}(o, k) \\ \vee \text{esp-ope}(k, o) \end{array}\}$

l'oggetto σ ed ogni oggetto dell'insieme R non faranno più parte del dominio.

$M_{\text{out}} = M_{\text{in}} \setminus (R \cup \{\sigma\})$

registra_rest(i:Data, o:Opera):Restauro modifica il livello estensionale

- pre-cond: nessuna già controllato dai VINCOLI

- post-cond: Sia α il nuovo oggetto del dominio tale che

$\text{Restauro}(\alpha) \wedge \text{inizio}(\alpha, i) \wedge \text{ope-res}(\alpha, r)$ $M_{\text{out}} = M_{\text{in}} \cup \{\alpha\}$ result = α

termina_rest(f:Data, r:Restauro):Restauro modifica il livello estensionale

- pre-cond: $\exists i \text{ inizio}(r, i) \wedge i \leq f \wedge \exists k \text{ fine}(r, k)$

- post-cond: l'oggetto del dominio r ora soddisfa:

$\text{fine}(r, f)$ Result = r

crea_esp_perm(p:PeriodoEsp[0..*], i:Data, s:Intero ≥ 0 , (pi:Data, po:Data)[0..1]):Esposizione
 • pre-cond: nessuna

- post-cond: Sia α il nuovo oggetto del dominio tale che: modifica il livello estensionale

$\text{Esposizione}(\alpha) \wedge [\forall p \in p \rightarrow \text{esp-per}(\alpha, p)] \wedge \text{inizio}(\alpha, i) \wedge \text{sezione}(\alpha, s) \wedge$
 $[(p_i, p_o) \neq \text{NULL} \rightarrow \text{PeriodoIn}(\alpha, p_i) \wedge \text{periodoFin}(\alpha, p_o)]$

$M_{\text{out}} = M_{\text{in}} \cup \{\alpha\}$

Result = α

[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]

Crea-esp-temp($p:\text{PeriodoEsp}[0..*]$, $i:\text{Data}$, $s:\text{Intervallo}$, $(pi:\text{Data}, po:\text{Data})[0..1]$,

$f:\text{Data}$, $n:\text{Stringz}$, $t:\text{Tema}):\text{EspTemporanea}$ modifica il livello estensionale

opre-cond: $i \leq f$

opost-cond: Sia α il nuovo oggetto del dominio tale che:

$\text{EspTemporanea}(\alpha) \wedge [\forall p \in p \rightarrow \text{esp-por}(\alpha, p)] \wedge \text{inizio}(\alpha, i) \wedge \text{fine}(\alpha, f)$

$\wedge [(pi, po) \neq \text{NULL} \rightarrow \text{periodoIn}(po, pi) \wedge \text{periodoFin}(\alpha, po)] \wedge \text{fine}(\alpha, f) \wedge \text{nome}(\alpha, n)$

$\wedge \text{esp-tema}(\alpha, t)$

$$M_{\text{out}} = M_{\text{in}} \cup \{\alpha\}$$

Result: α

TRIGGER

T. restaura- σ -esposta-se-in-prestito

op: Insert o Update PeriodoEsp o Restauro

OK = EXISTS (SELECT * FROM Opera σ , Prestito p
 WHERE ($\sigma.\text{est} = \text{True}$ AND $p.id = \sigma.\text{id}$ AND $p.\text{opera} = \sigma.\text{id}$ AND
 $p.\text{inizio} \leq \text{new}.\text{inizio}$ AND ($\text{new}.\text{fine}$ IS NULL OR
 $\text{new}.\text{fine} \leq p.\text{fine}$)
 OR $\sigma.\text{est} \neq \text{True}$ AND $\text{new}.\text{opera} = \sigma.\text{id}$);

if OK=true: permetti op

else: errore e rollback

T. biglietto-data-esposizione

op: Insert o Update Biglietto

OK = EXISTS (SELECT * FROM EspTemporanea et , Esposizione e
 WHERE $\text{new}.\text{temp} \neq \text{'Plus'}$ OR ($et.\text{esposizione} = e.\text{id}$ AND
 $\text{new}.\text{esposizione} = et.\text{id}$ AND $b.\text{giorno} \geq e.\text{inizio}$ AND
 $b.\text{giorno} \leq et.\text{fine}$)

if OK=True: permetti op

else: Errore e rollback

T. inizio-exp-fine

op: Insert o Update EspTemporanea

Error: EXISTS (SELECT * FROM Esposizione e WHERE $\text{new}.\text{esposizione} = e.\text{id}$ AND $e.\text{inizio} > \text{new}.\text{fine}$);

if Error: genera errore e rollback

else: permetti op.

[Spazio per minute. Questa pagina non sarà valutata a meno che non sia puntata da pagine precedenti.]

T.esposta_durante_esposizione

op: Insert o Update PeriodoEsp

OK = EXISTS (SELECT *

FROM Esposizione e left outer JOIN EspTemporanea et
ON e.id=et.esposizione

WHERE e.id=new.esposizione

AND e.inizio <= new.inizio AND (et.fine IS NULL

OR et.fine >= new.fine));

IF OK=True: permetti op

else: errore e rollback