Basi di Dati e Sistemi Informativi I, Prove scritte AA 2005/06

Adriano Peron

Facoltà di Scienze M.F.N., Corso di Laurea in Informatica, Dipartimento di Scienze Fisiche,
Università di Napoli 'Federico II', Italy
E-mail: peron@na.infn.it

1 3 marzo 2007

Si consideri il seguente schema relazionale che descrive una collezione di alberi costruiti su un insieme comune di nodi:
$ALBERI(\underline{CodA}, Radice)$
$NODI(\underline{CodN}, Peso)$
$COMPNODI(CodA, CodN)$
$COMPARCHI(CodA, Padre, Figlio, Peso)$

$ALBERI$ descrive la collezione di alberi presenti ($Radice$ è un codice di nodo);
$NODI(\underline{CodN}, Peso)$, descrive la collezione di nodi condivisi su cui tutti gli alberi sono definiti ($Peso$ è un intero);
$COMPNODI$ descrive l'insieme dei nodi associati ad un albero;
$COMPARCHI$ descrive l'insieme degli archi associati ad un albero (a ciascun arco è associato un peso espresso da un valore intero;

Esercizio 11 (6 punti) *Si scriva una interrogazione in algebra relazionale che fornisca se valutata il codice degli alberi in cui ogni nodo (tranne le foglie) ha esattamente un discendente.*

Esercizio 12 (9 punti) *Si scriva una interrogazione in SQL che fornisca coppie di codici di alberi tali che il secondo elemento della coppia è un sottoalbero del primo elemento della coppia (ogni nodo e ogni arco del sottoalbero deve essere nodo e arco, rispettivamente, del primo albero).*

Esercizio 13 (9 punti) *Si scriva una procedura PLSQL che riceve in ingresso il codice di un albero ed il codice di un nodo e che restituisce la somma dei pesi incontrati negli archi dal nodo dato alla radice e la somma dei pesi incontrati nei nodi dal nodo alla radice.*

Esercizio 14 (9 punti) *Si esprima nel modo più adeguato il seguente insieme di vincoli:*

- *Ogni nodo di un albero ha al più un padre;*
- *La radice di un albero non ha padre;*
- *I nodi padre e figlio di un arco associato ad una albero devono essere nodi dell'albero;*
- *Quando viene rimosso un nodo da un grafo (cancellazione dalla tabella $COMPNODI$) devono essere rimossi tutti gli archi e tutti i nodi del sottoalbero radicato nel nodo (discendenti);*

2 2 febbraio 2007

Si consideri il seguente schema relazionale che descrive alcuni dati anagrafici:
$PERSONA(\underline{CF}, Nome, Cognome, DataN, DataM)$
$RESIDENZA(\underline{CF, DataI}, Via, Num, Citta, DataF)$
$GENITORI(\overline{CF}, CFMadre, CFPadre)$

$PERSONA$, fornisce informazioni anagrafiche quali il Codice fiscale (chiave) data di nascita e data di morte (attributo parziale nullo per persone vive); $RESIDENZA$ descrive lo storico delle residenze di ogni persona (la data di fine residenza è attributo parziale ed è in particolare nullo per la residenza corrente); $GENITORI$ associa ad ogni persona padre e madre (entrambi gli attributi posso essere parziali);

Esercizio 21 (6 punti) *Si scriva una interrogazione in algebra relazionale che fornisca se valutata nomi e cognomi delle persone che sono state residenti per tutta la vita nella stessa città.*

Esercizio 22 (9 punti) *Si scriva una interrogazione in SQL che fornisca nome e cognome delle persone che hano avuto la residenza esattamente nello stesso insieme di città del loro padre.*

Esercizio 23 (9 punti) *Si scriva una procedura PLSQL che riceve in ingresso il CF di una persona e restituisce una stringa di caratteri contenete il nome e cognome di tutti gli antenati in linea maschile (padre, nonno paterno, padre del nonno paterno etc.).*

Esercizio 24 (9 punti) *Si esprima nel modo più adeguato il seguente insieme di vincoli:*

- *Ogni persona ha al più un padre ed una madre;*
- *Quando viene cancellata una persona deve essere cancellata anche ogni associazione di parentela che la riguarda;*
- *Non si può essere residenti nello stesso tempo in due luoghi diversi;*
- *Quando viene fissata la data di morte viene anche chiusa in quella data la residenza.*

3 8 gennaio 2007

Si consideri il seguente schema relazionale che descrive la gestione dei piani di studio degli studenti:
$STUDENTE(\underline{Matricola}, Nome, Cognome)$
$PIANI(\underline{CodPiani}, Matricola, Data)$
$COMPOSIZIONE(\underline{CodPiano, CodEsame}, Anno)$
$ESAME(\underline{CodEsame}, Titolo, CFU, Tipo)$
$ANNI(CodEsame, Anno)$
$PROPED(CodEsame, CodEsameProP)$

$PIANI$, fornisce la data di registrazione del Piano di Studi e la matricola dello studente (chiave esterna); $COMPOSIZIONE$ descrive la composizione di un piano di studio (una riga per ogni esame presente nel piano di studi insieme all'anno di corso in cui è previsto); $ESAME$ descrive gli attibuti degli insegnamenti; l'attributo tipo può assumere i valori $OBBLIGATORIO$, $FACOLTATIVO$, $DEFAULT$; $ANNI$ descrive i possibili anni del corso di studi in cui un insegnamento può essere collocato; $PROPED$ descrive i legami di propedeuticità tra i corsi.

Esercizio 31 *(punti 8) Si scriva una espressione in algebra relazionale che, se valutata, fornisca il nome e cognome e la matricola degli studenti che hanno un piano di studi dove tutti gli insegnamenti presenti non hanno richiesta di propedeuticità.*

Esercizio 32 *(punti 9) Definire nel modo più appropriato e semplice il seguente insieme di vincoli: (a) lo stesso esame non deve essere associato più di una volta allo stesso piano di studi; (b) nella tabella $PROPED$ ad un esame possono essere associati solo esami effettivamente presenti nella tabella $ESAMI$; (c) se un esame è presente in un piano di studi devono essere presenti anche tutti gli esami propedeutici; (d) la somma dei CFU presenti in ogni anno (I, II e III) deve essere esattamente 60 CFU.*

Esercizio 33 *(***Punti 9***) Si scriva una procedura in PL\SQL che riceve in ingresso la matricola di uno studente e un nuovo codice di piano di studi (non ancora presente nella base di dati). La procedura crea un piano di studi (avente come codice quello passato per parametro) che contiene tutti gli esami obbligatori e tutti gli esami di default. Se un esame può essere collocato in anni diversi, nel piano di studi viene collocato nel primo anno possibile).*

Esercizio 34 *(***Punti 4***) Si dica se il seguente schedule è compatibile con il locking a due fasi (il pedice funge da identificativo della transazione)*
$R_2(x)W_3(x)c_3W_1(y)c_1R_2(y)W_2(z)c_2$. *Interpretando il pedice come timestamp si dica quali transazioni superano il controllo basato su timestamp.*
Si descriva la differenza tra locking a due fasi e locking a due fasi stretto indicando quali anomalie delle transazioni concorrenti permettono rispettivamente di evitare.

Esercizio 35 *(***Facoltativo. Punti 4***) Si scriva una vista che permette di presentare all'utente in un unico schema tutti i dati presenti in $ESAMI$ ed $ANNI$. Si scriva poi un trigger che permette di gestire (rimpiazzandola) una operazione di INSERT nella vista.*

4 7 settembre 2006

Si consideri il seguente schema relazionale che memorizza alberi con nodi ed archi etichettati:

$TREE(\underline{CodT}, Root)$
$NODI(\underline{CodN}, Label)$
$ARCHI(\underline{CodA}, Padre, Figlio, Label)$
$COMPN(CodT, CodN)$
$COMPA(CodT, CodA)$

$TREE$ fornisce l'elenco degli alberi (codice dell'albero e codice del nodo radice); $NODI$ fornisce l'elenco dei nodi (codice del nodo e dell'etichetta); $ARCHI$ fornisce l'elenco degli archi (codice dell'arco, codice del nodo padre, del nodo figlio ed etichetta); $COMPN$ fornisce l'associazione dei nodi agli alberi; $COMPA$ fornisce l'associazione degli archi agli alberi;

Si osservi che alberi diversi possono condividere nodi ed archi.

Esercizio 41 *Si scriva una interrogazione in algebra relazionale che restituisca il codice degli alberi in cui tutte le foglie siano etichettate dalla etichetta A.*

Esercizio 42 *Si scriva una vista in SQL che per ogni albero fornisca: il numero delle foglie, il numero massimo di figli associato ai nodi, il numero di etichette distinte presenti negli archi.*

Esercizio 43 *Si scriva una funzione in PL\SQL che riceve in ingresso il nome di un albero ed il codice di un nodo appartenente all'albero. La procedura restituice una stringa contenete la concatenazione delle etichette dei nodi e degli archi incontrati nel percorso dal nodo passato in input alla radice.*

Esercizio 44 *Supponendo che le etichette dei nodi siano etichettate da interi, si scriva un Trigger che quando l'etichetta di un nodo viene variata (incrementata o decrementata) propaga la stessa variazione (lo stesso incremento o lo stesso decremento) a tutti i nodi figli.*

5 25 luglio 2006

Si consideri il seguente schema relazionale che descrive una base di dati per la gestione di linee ferroviarie:
$TRATTA(\underline{CodTratta}, StP, StA, Km)$
$TRENO(\underline{CodTreno}, Tipo, StP, StA)$
$PERCORRENZA(\underline{CodTreno, CodTratta}, Stop, OraP, OraA)$

$TRATTA$ contiene la descrizione delle tratte che collegano una stazione di partenza StP, una stazione di arrivo StA e la distanza tra le due stazioni Km. $TRENO$ contiene la descrizione dei treni che percorrono un insieme di tratte da una stazione di partenza ad una di arrivo. La sequenza delle tratte percorse da un treno viene descritta in $PERCORRENZA$: in particolare l'attributo booleano $Stop$ indica se il terno ferma nella stazione di arrivo (se vale $False$ il treno passa senza fermarsi nella stazione di arrivo della tratta); viene riportata inoltre l'ora di partenza e di arrivo relativa alla percorrenza della tratta.

Esercizio 51 *(***Punti 8***) Scrivere una espressione in algebra relazionale che se valutata fornisce il codice dei treni che percorrono percorsi con sole stazioni locali. Una stazione è locale se è destinazione di una ed una sola tratta e stazione di partenza per una e una sola tratta. Inoltre, si scriva una interrogazione che per i treni dell'interrogazione precedente fornisca la lunghezza complessiva del percorso.*

Esercizio 52 *(***Punti 8***) Si scriva una interrogazione in SQL che restituisca coppie di codici di treni che partono dalla stessa stazione, arrivano alla stessa stazione ma fanno un percorso diverso (sequenza di tratte diverse).*

Esercizio 53 *(***Punti 10***) Si scriva una procedura in PL\SQL (o alternativamente in Pascal o C) che riceve in ingresso una stazione e restituisce in uscita una stringa di caratteri che indica tutte le stazioni raggiungibili (connesse da una sequenza di tratte) e la lunghezza complessiva del percorso necessaria per raggiungerle. Per poter ottenere il risultato si faccia uso di una tabella temporanea $TEMP(StR, Km)$ dove ospitare i risultati parziali della procedura. Si assuma che la tabella sia già definita e che all'inizio della procedura vada solo svuotata. Si eviti di inserire duplicati nella tabella (righe per la stessa stazione). Si trascuri il fatto che una stazione potrebbe essere raggiungibile con percorsi diversi e lunghezze di percorso diverse.*

Esercizio 54 *(***Punti 6***) Si scriva uno schema Entità Associazioni per lo schema logico proposto. Si definiscano usando SQL le tre tabelle corrispondenti allo schema logico indicando obbligatoriamente almeno i vincoli di integrità referenziale. A discrezione si suggeriscano altri vincoli ragionevoli per il problema in esame e li si codifichi opportunamente. (il punteggio previsto può essere incrementato in presenza di una adeguata implementazione di vincoli aggiuntivi).*

6 23 giugno 2006

Si consideri il seguente schema relazionale che descrive una base di dati per la programmazione di film e passaggi di spot pubblicitari in un cinema multisala:
$FILM(\underline{CodFilm}, Titolo, Regista, Anno, Durata)$
$PROIEZF(\underline{CodProiez}, CodSala, CodFilm, Data, Ora)$
$SPOT(\underline{CodSpot}, Titolo, Prodotto, Durata)$
$PROIEZS(\underline{CodSpot, CodProiez})$.

$FILM$ contiene la descrizione dei Film da proiettare. $PROIEZ$ contiene la descrizione delle proiezioni previste per un film (il codice del film da proiettare, la sala in cui viene proiettato, la data della proiezione e l'ora di inizio). $SPOT$ contiene la descrizione degli spot pubblicitari da associare alle proiezioni. $PROIEZS$ collega gli spot alle proiezioni (più spot possono essere associati alla medesima proiezione).

Esercizio 61 *(***Punti 8***) Scrivere una espressione in algebra relazionale che se valutata fornisce il titolo dei film nei quali (nel corso delle varie proiezioni del film) sono passati tutti gli spot descritti nella tabella $SPOT$.*

Esercizio 62 *(***Punti 7***) Definire nel modo più appropriato il seguente insieme di vicoli: (a) lo stesso spot non deve essere associato più di una volta alla medesima proiezione; (b) gli spot possono essere associati solo a proiezioni effettivamente presenti nella tabella $PROIEZF$ (c) nella tabella sono presenti solo le proiezioni relative alla data corrente o a date successive alla data corrente (d) per ogni proiezione, la durata complessiva di tutti gli spot ad essa associati deve essere inferiore o uguale ad una costante fissata K.*

Esercizio 63 *(***Punti 10***) Si scriva una procedura in PL\SQL che riceve in ingresso Il codice di un file, il codice di uno spot, il numero di passaggi N previsto per lo spot, una data iniziale per i passaggi dello spot. La procedura associa (inserendo le righe necessarie nella tabella $PROIEZF$, se possibile) lo spot indicato a N proiezioni del film indicato già programmate (presenti nella tabella $PROIEZF$) seguendo i seguenti criteri (1) lo spot non deve essere già associato a quella proiezione, (2) l'inserimento dello spot non deve produrre la violazione del vincolo (d) dell'esercizio precedente, (3) sono preferibili le proiezioni che hanno la data fornita dal parametro o quelle immediatamente successive (non vanno considerate quelle antecedenti).*

Esercizio 64 *(***Punti 5***) Si dica se il seguente schedule è compatibile con il locking a due fasi (il pedice funge da identificativo della transazione)*
$R_2(x)W_3(x)c_3W_1(y)c_1R_2(y)W_2(z)c_2$. *Interpretando il pedice come timestamp si dica quali transazioni superano il controllo basato su timestamp.*
Si descriva la differenza tra locking a due fasi e locking a due fasi stretto indicando quali anomalie delle transazioni concorrenti permettono rispettivamente di evitare.

Esercizio 65 *(***Facoltativo. Punti 5***) Si scriva una vista che permette di presentare all'utente in un unico schema i dati riguardanti il film (almeno il titolo), delle proiezioni e degli spot (almeno il titolo dello spot) associati alle proiezioni. Si scriva poi un trigger che permette di gestire (rimpiazzandola) una operazione di INSERT nella vista.*