Basi di Dati e Sistemi Informativi I, 18-10-22

Silvio Barra, Mara Sangiovanni

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: silvio.barra@unina.it

Si consideri il seguente schema relazionale:

IMPIEGATO(*Matricola*, *Cognome*, *Stipendio*, **Dipartimento**)
DIPARTIMENTO(*Codice*, *Nome*, *Sede*, **Manager**)
PROGETTO(*Sigla*, *Nome*, *Bilancio*, **Responsabile**)
PARTECIPAZIONE(**Progetto**, **Impiegato**)

Esercizio 01 *Algebra Relazionale e Query SQL per le seguenti interrogazioni.*

1. Trovare matricola e cognome degli impiegati che guadagnano piu' di 50 milioni.
2. Trovare cognome e stipendio degli impiegati che lavorano a Roma.
3. Trovare cognome degli impiegati e nome del dipartimento in cui lavorano.
4. Trovare cognome degli impiegati che sono direttori di dipartimento.
5. Trovare i nomi dei progetti e i cognomi dei responsabili.
6. Trovare i nomi dei progetti con bilancio maggiore di 100K e i cognomi degli impiegati che lavorano su di essi.
7. Trovare il cognome degli impiegati che guadagnano piu' del loro direttore di dipartimento.
8. Trovare cognome dei direttori di dipartimento e dei responsabili di progetto.
9. Trovare nomi dei dipartimenti in cui lavorano impiegati che guadagnano piu' di 60K.
10. Trovare nomi dei dipartimenti in cui tutti gli impiegati guadagnano piu' di 60K.
11. Trovare cognome degli impiegati di stipendio massimo.
12. Trovare matricola e cognome degli impiegati che non lavorano a nessun progetto.
13. Trovare matricola e cognome degli impiegati che lavorano a piu' di un progetto.
14. Trovare matricola e cognome degli impiegati che lavorano a un solo progetto.
15. Trovare il cognome dei direttori che lavorano nello stesso dipartimento di cui sono direttori.

Basi di Dati e Sistemi Informativi I, 22 dicembre 2022 - PROVA A

Silvio Barra, Mara Sangiovanni

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: silvio.barra@unina.it, mara.sangiovanni@unina.it

Si consideri il seguente schema relazionale che descrive un tabellone nel corso di una partita ($codP$ è il codice della partita). Una partita ha una sequenza di mosse ($OrdMossa$ indica l'ordine della mossa). La tabella TAB contiene per ogni partita e mossa della partita una riga per ogni casella del tabellone (individuata dalla coppia X e Y di coordinate orizzontale e verticale). Tale riga indica il contenuto della casella ($Pezzo$) ad una specifica mossa della partita (se la casella è vuota allora $Pezzo$ è NULL). Quindi, per ogni partita e mossa si ha una descrizione completa del contenuto del tabellone. Nella tabella $MOSSA$ viene indicato lo spostamento di un pezzo da una casella ad un'altra casella. La tabella $ELIMINATO$ contengono i pezzi eliminati nelle varie mosse. Nella tabella $PEZZI$ viene memorizzata l'informazione sull'assegnazione dei pezzi ai giocatori nelle partite.

$PEZZI(codP, Giocatore, Pezzo)$
$TAB(CodP, OrdMossa, Pezzo, X, Y)$
$MOSSA(\underline{CodP, OrdMossa}, Giocatore, daX, daY, aX, aY, Pezzo)$
$ELIMINATO(CodP, OrdMossa, Pezzo)$

Esercizio 01 *(Punti 10) Si scriva un trigger che viene attivato all'inserimento di una mossa in una partita. L'effetto del trigger è il seguente:*

1. *Viene eliminato il pezzo che si trova (alla mossa precedente) eventualmente nella casella di destinazione della mossa;*

2. *Vengono inserite righe per ogni casella del tabellone (le stesse caselle presenti alla mossa precedente) per mantenere la situazione corrente del tabellone (le caselle che non sono sorgente e destinazione della mossa mantengono la stessa situazione e il pezzo interessato dalla mossa si sposta dalla casella sorgente a quella della destinazione).*

Esercizio 02 *(Punti 8) Si scriva una funzione SQL che riceve in ingresso un codice di partita ed un pezzo. La funzione restituisce una stringa di caratteri contenente il cammino fatto dal pezzo nel tabellone nel corso della partita. Il cammino viene espresso come sequenza delle coordinate X,Y delle caselle separate da virgole.*
Esempio Stringa in output: X1,Y1;X2,Y2;.....

Esercizio 03 *(Punti 15) Si scriva una funzione SQL che riceve in ingresso una lista di codici di partite separate dal carattere separatore '@' e utilizzando SQL DINAMICO restituisce una lista di pezzi che sono stati eliminati in* ***tutte*** *le partite passate per parametro.*

Basi di Dati I, Esempio Seconda Prova Intercorso, 8/12/2022

Mara Sangiovanni

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: mara.sangiovanni@unina.it

Si consideri il seguente schema relazionale che descrive un frammento della base di dati per gestire le informazioni relative a viaggi in autostrada. Nella tabella tariffe sono indicati i costi dal casello di ingresso al casello di uscita per una determinata categoria di automobili. Un viaggio inizia da un casello in ingresso nel giorno dataI al tempo TempoI e finisce nel casello Uscita nel giorno DataF al tempo TempoF. Per i viaggi iniziati ma non ancora conclusi DataF, TempoF, Uscita e Tariffa sono NULL. La tabella CHECK contiene le rilevazioni dei tutor fatte al passaggio dell'automobile (identificata dalla targa) nel tragitto. Ogni tutor ha una velocità massima consentita descritta nella tabella $PCHECK$. Se non vi sono infrazioni il campo Infrazione ha valore NULL.

$AUTO(\underline{Targa}, CF\_P, Categoria)$
$TARIFFE(Ingresso, Uscita, KM, Categoria, Costo)$
$PCHECK(\underline{PuntoCheck}, VelocitaMax)$
$CHECK(PuntoCheck, Targa, Velocita, Data, Tempo, Infrazione)$
$VIAGGIO(\underline{CodV}, Targa, DataI, DataF, TempoI, TempoF, Ingresso, Uscita, Tariffa, KM)$

Esercizio 01 *(Punti 6), 15 minuti Si scriva il seguente trigger. Quando viene inserito un check per un viaggio si controlla se la velocità rilevata è superiore alla velocità massima. Se è superiore, si pone a TRUE il campo infrazione del CHECK.*

Esercizio 02 *(Punti 6, 15 minuti) Si scriva il seguente trigger. Quando viene aggiornato un viaggio esprimendo un valore per il casello di uscita si aggiornano anche gli attributi Km e Tariffa recuperando i valori dalla tabella TARIFFE (la tariffa dipende da ingresso, uscita e categoria dell'auto).*

Si consideri il seguente schema relazionale che descrive un frammento della base di dati per gestire gli accessi ad una biblioteca digitale di periodici. Le riviste sono identificate dal codice $ISNN$ e sono strutturate in fascicoli (identificati da $CodF$). Ogni fascicolo contiene articoli identificati dall'attibuto Doi. L'utente identificato dal codice fiscale CF ha un profilo associato $Codprofilo$ che regola le possibilità di accesso (numero di articoli consultati al giorno e al mese). In $ACCESSO$ ogni istanza memorizza l'accesso di un utente an un articolo. In $PAROLECHIAVE$ viene memorizzato un insieme di parole chiave significative per una rivista. Le parole chiave sono usate per descrivere astrattamente gli articoli in base al loro contenuto. L'associazione tra parole chiave e articoli viene memorizzata nella relazione $DESCRIZIONE$.

$RIVISTA(\underline{ISNN}, Titolo, Editore, Perodicita)$
$FASCICOLO(\underline{CodF}, Titolo, ISNN, Anno, Numero)$
$ARTICOLO(\underline{Doi}, CodF, Titolo, Autore, Sommario, PagI, PagF)$
$UTENTE(\underline{CF}, email, CodProfilo, Nome, Cognome, DataN)$
$ACCESSO(CF, Doi, Data)$
$PROFILO(\underline{CodProfilo}, Tipo, MaxGiorno, MaxMese)$
$PAROLECHIAVE(Parola, ISNN)$
$DESCRIZIONE(Parola, Doi)$

Esercizio 03 *(Punti 7, 20 minuti) Si implementi un trigger azionato quando viene inserito un nuovo articolo. Il trigger cerca la presenza nel sommario dell'articolo delle parole chiave associate alla rivista dell'articolo. Se viene trovata la presenza di una parola chiave questa viene memorizzata nella tabella $DESCRIZIONE$.*

Esercizio 04 *(Punti 7, 25 minuti) Si scriva una funzione in SQL DINAMICO che riceve in ingresso una stringa di parole chiave separate dal carattere +. La funzione restituisce la stringa di doi degli articoli a cui sono associate TUTTE le parole chiave nella stringa.*

Si consideri il seguente schema relazionale che descrive un frammento della base di dati per gestire depositi di immagini condivise tra utenti. Le foto sono contenute in album. Un album può contenere foto ed altri album (la struttura degli album è analoga a quella di un filesystem con file e cartelle). Ciascuna foto e ciscun album ha un proprietario (owner) espresso da un identificativo di utente. Nella tabella $ALBUM$ l'attributo $InAlbum$ è l'identificativo dell'album contenete (si assuma che un album alla radice della gerarchia di contenimento è contenuto nell'album di sistema di identificativo SYSALBUM). Gli attributi *Aggiunta* e *Rimossa* riportano la data di inserimento della foto e la eventuale data di rimozione (attributo parziale). Foto ed album possono avere dei tag associati. L'elenco dei tag ammissibili si trova nella tabella $HASHTAG$. I tag associati alle foto e agli album si trovano nelle tabelle $TAGFOTO$ e $TAGALBUM$ rispettivamente. La tablella $VISIBLE$ indica invece l'accessibilità dei file ai vari utenti. Il proprietario $CodProp$ dell'album $codA$ concede il diritto di visualizzazione a $CodUt$ su tutte le foto contenute in $codA$. Infine, nella tabella LOG vengono registrate tutte le operazioni fatte sulle foto dagli utenti: $codF$ è la foto su cui si fa l'operazione, $CodU$ l'utente che fa l'operazione ed $Operation$ il tipo di operazione (ad es. inserimento, cancellazione, visualizzazione).

$HASHTAG(parola)$
$FOTO(\underline{CodF}, uri, nome, titolo, owner, CodAlbum, Aggiunta, Rimossa)$
$ALBUM(\underline{CodA}, nome, titolo, owner, InAlbum)$
$UTENTE(\underline{CodU}, nome, cognome, email)$
$TAGFOTO(CodF, parola)$
$TAGALBUM(CodA, parola)$
$VISIBLE(CodProp, CodUt, CodA)$
$LOG(CodF, CodU, Time, Operation)$.

Esercizio 05 *(Punti 10) Si scriva una procedura PLSQL che riceve in ingresso l'identificativo di un album e che restituisce una stringa contenete tutti i tag associati all'album e agli album in esso contenuti (ad ogni livello di profondità) senza ripetizioni. Si consiglia di avvalersi di una tabella $TMP(CodA)$ (che si suppone già definita) dove memorizzare preventivamente l'albero degli album radicato nell'album passato come parametro.*

Esercizio 06 *(Punti 7) Usando SQL DINAMICO si scriva una funzione che riceve in ingresso una lista di tag separati dal carattere @ e che restituisce una stringa degli uri delle foto (separati da @) a cui sono associati tutti i tag passati per parametro.*

Basi di Dati I, 13 dicembre 2022

Mara Sangiovanni

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: mara.sangiovanni@unina.it

Si consideri il seguente schema relazionale che descrive i dati di gestione in un sistema per l'accesso parallelo. L'intuizione è che una transazione (sequenza atomica di operazioni) prima di poter operare su una risorsa deve chiedere l'autorizzazione all'accesso (in lettura o scrittura) e solo quando la risorsa le viene assegnata può operare per leggere o modificare il valore della risorsa. Le operazioni possono essere $CREA$ (crea una nuova risorsa), CANCELLA (cancella una risorsa) MODIFICA (modifica il valore della risorsa) COMMIT (chiusura della transazione) ABORT (annulla una transazione). Le risorse condivise son identificate dall'attributo $CodRisorsa$ e lo stato indica: se sono libere $UNLOCK$, in uso in lettura $R-LOCK$ o in uso in scrittura $W-LOCK$. Le transazioni identificate da $CodTransazione$ inoltrano richieste di operazioni che sono registrate nella tabella $RICHIESTE$. Ogni richiesta ha un tempo di inoltro, un tipo di accesso (o R-LOCK o W-LOCK) e la risorsa richiesta. Nella tabella ASSEGNAZIONE sono memorizzate le assegnazioni correnti delle risorse alle transazioni. Nella tabella LOG vengono invece registrate le operazioni svolte dalle transazioni sulle risorse a loro assegnate. In particolare, per ogni operazione $MODIFICA$ si registra il valore della risorsa prima e dopo l'operazione; per operazione CANCELLA si esprime solo $ValorePrima$; per CREA si esprime solo il valore $ValoreDopo$; e per COMMIT ed ABORT e CHECK non si esprime nessun valore (NULL). In aggiunta per l'operazione CHECK non si esprime neppure il codice della transazione (il record di CHECK rappresenta un punto di controllo del sistema). Per tutte le operazioni è riportato un timestamp, cioè una marca temporale univoca nel sistema che tiene traccia della sequenza delle operazioni.

$LOG(\underline{Cod}, Operazione, CodRisorsa, ValorePrima, ValoreDopo, CodTransazione, Timestamp)$
$RISORSA(\underline{CodRisorsa}, Locazione, Valore, Stato)$
$RICHIESTE(CodTransazione, Tempo, tipoAccesso, CodRisorsa)$
$ASSEGNAZIONE(CodTransazione, Tempo, CodRisorsa, tipoAccesso)$

Esercizio 01 *(Punti 8) Si implementi il seguente trigger. Quando una transazione registra una operazione di ABORT sul log, tutte le scritture fatte dalla transazione e riportate sul LOG devono essere annullate in ordine inverso a quelle in cui sono state fatte. Per annullare le scritture si deve consultare il log e si deve assegnare ad ogni risorsa scritta dalla transizione il valore ValorePrima riportato nel LOG. Inoltre, le risorse assegnate alla transazione devono tornare libere: si rimuovono le assegnazioni alla transazione e lo stato della risorsa assume valore UNLOCK.*

Esercizio 02 *(Punti 8) Si scriva una funzione con parametro intero Tout che per tutte le transazioni T1 che sono in attesa per una risorsa per un tempo superiore a Tout (differenza tra il tempo di registrazione della richiesta e il tempo corrente) controlli se ci sia un deadlock (cioè se esiste un'altra transazione T2 che occupa la risorsa richiesta e la transazione T2 richiede una risorsa asegnata alla transazione T1). La funzione restituisce una stringa coi codici delle transazioni T1 in deadlock cosi trovate.*

[redacted]

Basi di Dati I, 19 Gennaio 2023

Mara Sangiovanni, Silvio Barra

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: mara.sangiovanni@unina.it, silvio.barra@unina.it

PRIMA PARTE: Esercizi 1, 2, 3, 4 e 5
SECONDA PARTE: Esercizi 6, 7, e 8
APPELLO: Tutti

Si consideri il seguente schema relazionale che descrive un frammento della base di dati per gestire i prestiti in una biblioteca. Di un *LIBRO* (descrizione del libro) possono esserci più copie fisiche, memorizzate nella tabella *ESEMPLARE*. L'attributo booleano *Prestito* indica se l'esemplare è disponibile per il prestito. L'attributo booleano *Consultazione* indica se l'esemplare è disponibile per la consultazione in loco. Il prestito riguarda esclusivamente le copie fisiche; le prenotazioni sono registrate nella tabella *PRENOTAZIONE*: ogni prenotazione è relativa ad un libro (non alla sua copia fisica). Quando un esemplare del libro è disponibile si può effettuare il prestito. Ogni *UTENTE* ha un *PROFILO* che regola la durata dei prestiti e il massimo numero di esemplari che possono essere richiesti in prestito. I prestiti sono registrati nella tabella *PRESTITO*: ogni prestito ha una data di effettuazione, una data di scadenza, una data di restituzione e una data di sollecito (rispettivamente *DataE*, *DataS* , *DataR* e *DataSol*). La data di restituzione e la data di sollecito sono settate a *NULL* se il libro non è stato ancora restituito e se il prestito non è ancora scaduto, rispettivamente.

LIBRO(*ISBN*, *Titolo*, *Editore*, *Anno*, *Descrizione*)
ESEMPLARE(*ISBN*, *CodiceBarre*, *Collocazione*, *Prestito*, *Consultazione*)
UTENTE(*CF*, *CodProfilo*, *Nome*, *Cognome*, *DataN*)
PROFILO(*CodProfilo*, *MaxDurata*, *MaxPrestito*)
PRESTITO(*CodPrestito*, *CodiceBarre*, *Utente*, *DataE*, *DataS*, *DataR*, *Sollecito*)
PRENOTAZIONE(*CodPrenotazione*, *ISBN*, *Utente*, *Data*)

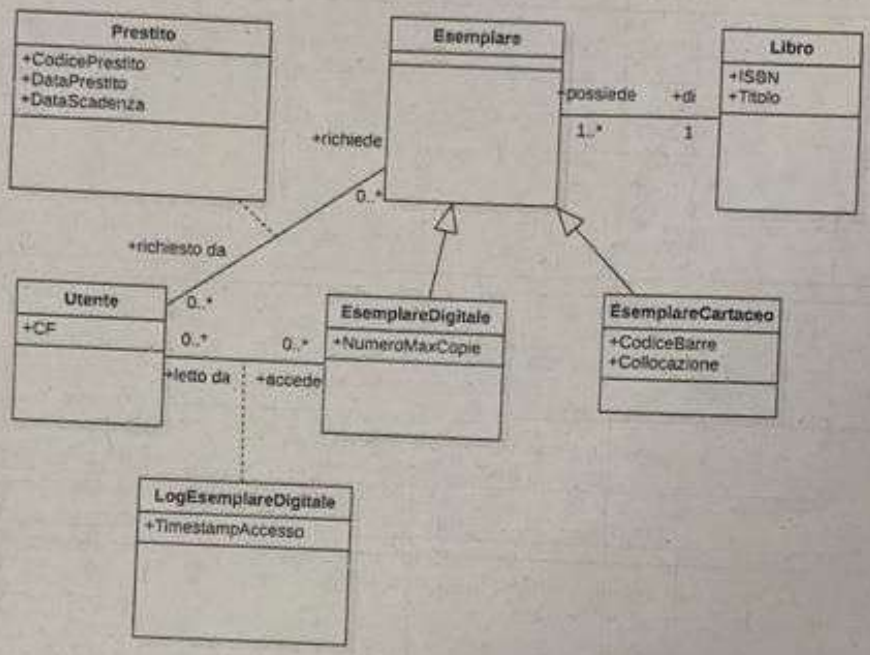
- **Esercizio 01** *(Punti 8) Si scriva una espressione in algebra relazionale che, se valutata, restituisca il nome degli utenti che non hanno mai restituito i prestiti in ritardo.*

- **Esercizio 02** *(Punti 8) Si scriva una interrogazione in SQL che, se valutata, fornisce il TITOLO del libro che ha avuto nell'anno 2022 il maggior numero di prestiti (si intende per numero di prestiti di un libro il numero di prestiti di tutti i suoi esemplari).*

- **Esercizio 03** *(Punti 7) Si implementino nel modo più adeguato i seguenti vincoli:*

  1. *Un utente non può avere più prestiti in corso (esemplari non ancora restituiti) di quanto previsto dal suo profilo.*
  2. *Se è presente un sollecito la restituzione è stata fatta dopo la data di scadenza.*
  3. *Un utente non può avere più di un profilo associato*

Esercizio 04 *(Punti 8)*



Si consideri la bozza di Class Diagram in figura.

- *Effettuare la ristrutturazione e motivarne il processo;*
- *Definire lo schema logico;*

Esercizio 05 *(Punti 7) Si implementino i seguenti trigger*

- *il primo trigger viene attivato quando viene inserita la data di sollecito per un prestito. L'effetto del trigger è che tutte le prenotazioni di quell'utente vengo automaticamente cancellate.*
- *il secondo trigger viene attivato ad ogni richiesta di prestito. L'effetto è che bisogna controllare se l'utente ha una prenotazione per quel libro. In tal caso la prenotazione corrispondente va rimossa.*

Esercizio 06 *(Punti 10) Si scriva una procedura che per ogni utente e per ogni anno estragga il numero dei prestiti e il numero delle prenotazioni. Si presupponga l'esistenza di una tabella* $STATISTICHE$*, inizialmente vuota, con il seguente schema:*
$STATISTICHE(CF, Anno, NumPrestiti, NumPren)$

Esercizio 07 *(Punti 14) Si scriva una funzione che riceve in ingresso una stringa contenente delle parole separate tra loro dal simbolo* $-$*. Si scriva una interrogazione in SQL dinamico che recupera i libri in cui* $TUTTE$ *le parole della stringa compaiono nella descrizione del Libro. La funzione restituisce una stringa contenente i titoli dei libri recuperati separati dal simbolo* ;.

APPELLO 27_04_2023.pdf

Basi di Dati, 27 aprile 2023

Mara Sangiovanni, Silvio Barra

DIETI, Corso di Laurea in Informatica, Università di Napoli "Federico II", Italy
E-mail: mara.sangiovanni@unina.it, silvio.barra@unina.it

Tempo: 2,5 ore.
Si consideri il seguente schema relazionale che descrive un frammento della base di dati per gestire una collezione di lavori scientifici (articoli) pubblicati su rivista. La tabella *ARTICOLI* contiene tutti articoli: *DOI* è l'identificativo dell'articolo; *ISSN* è l'identificativo della rivista. Gli autori dei lavori sono identificati dall'attributo *ORCID*. Nella tabella *SCRIVE* si codifica l'associazione tra articoli e gli autori; l'attributo *Ordine* indica la posizione dell'autore nella lista degli autori di un lavoro. Nella tabella *BIBLIOGRAFIA* viene tenuta traccia delle citazioni tra articoli: *DOI* è l'identificativo del lavoro che cita un articolo in bibliografia e *DOICitato* è l'identificativo del lavoro citato. La tabella *KEYWORD* tiene traccia delle parole chiave (*Parola*) associate a un articolo.

AUTORE(<u>*ORCID*</u>, *Nome*, *Cognome*, *Affiliazione*)
SCRIVE(<u>*DOI*, *ORCID*</u>, *Ordine*)
ARTICOLO(<u>*DOI*</u>, *Titolo*, *Anno*, *pagI*, *pagF*, *NomeRivista*, *ISSN*, *Numero*, *Fascicolo*)
BIBLIOGRAFIA(<u>*DOI*, *DOICitato*</u>)
KEYWORDS(<u>*DOI*, *Parola*</u>)

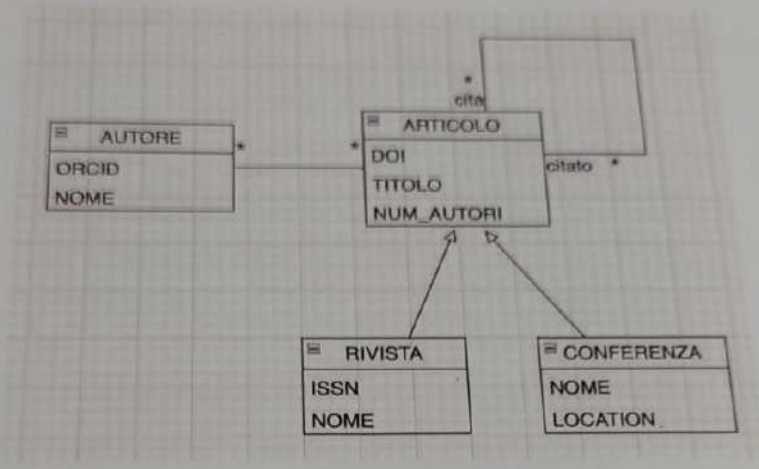
Esercizio 01 *(Punti 6) Si scriva una interrogazione in algebra relazionale che, se valutata, restituisca ORCID, nome e cognome degli autori che non hanno MAI scritto articoli in collaborazione (sempre autori unici dei loro articoli).*

Esercizio 02 *(Punti 9) Si scriva in SQL una interrogazione che restituisca coppie di identificativi di articoli (DOI1, DOI2) che rispettano il seguente criterio: le parole chiave associate a DOI2 sono TUTTE associate anche a DOI1.*

Esercizio 03 *(Punti 5) Si implementino nel modo più opportuno i seguenti vincoli:*

1. *Due autori di uno stesso articolo devono avere valori di ordine diverso;*
2. *Con riferimento alla tabella BIBLIOGRAFIA, l'anno di pubblicazione del lavoro identificato da DOI deve essere uguale o successivo all'anno di pubblicazione del lavoro identificato da DoiCitato;*
3. *Un articolo non deve citare più di una volta un altro articolo.*

Esercizio 04 *(Punti 6) Si consideri la bozza di Class Diagram in figura.*



- *Effettuare la ristrutturazione e motivarne il processo;*
- *Disegnare lo schema concettuale ristrutturato;*
- *Definire lo schema logico.*

Esercizio 05 *(Punti 9) Si scriva una funzione PLSQL che riceve l'ORCID di un autore e restituisca il suo indice di Hirsch (h-index). L'h-index di un autore è* **n** *se è autore di almeno* **n** *articoli, ognuno dei quali ha ricevuto almeno* **n** *citazioni.*
(SUGGERIMENTO: ordinare gli articoli per numero decrescente di citazioni).

Basi di Dati, 14 giugno 2023

Mara Sangiovanni, Silvio Barra

DIETI, Corso di Laurea in Informatica, Università di Napoli 'Federico II', Italy
E-mail: mara.sangiovanni@unina.it, silvio.barra@unina.it

Tempo: 2,5 ore.

Si consideri il seguente schema relazionale che descrive alcuni dati anagrafici: $PERSONA$, 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. Una persona può avere molte residenze, di cui una sola corrente (la residenza corrente è quella che ha NULL associato a $DataF$) ed alcune concluse (La data di fine residenza è presente). $GENITORI$ associa ad ogni persona padre e madre (entrambi gli attributi posso essere parziali). $FAMIGLIA$ descrive i nuclei famigliari e $CFCapo$ è l'identificativo del capo famiglia. $COMPFAMIGLIA$ indica la composizione della famiglia.

$PERSONA(\underline{CF}, Nome, Cognome, DataN, DataM)$
$RESIDENZA(\underline{CF, DataI}, Via, Num, Citta, DataF)$
$GENITORI(\underline{CF}, CFMadre, CFPadre)$
$FAMIGLIA(\underline{codFamiglia}, CFCapo)$
$COMPFAMIGLIA(\underline{codFamiglia, codMembro})$

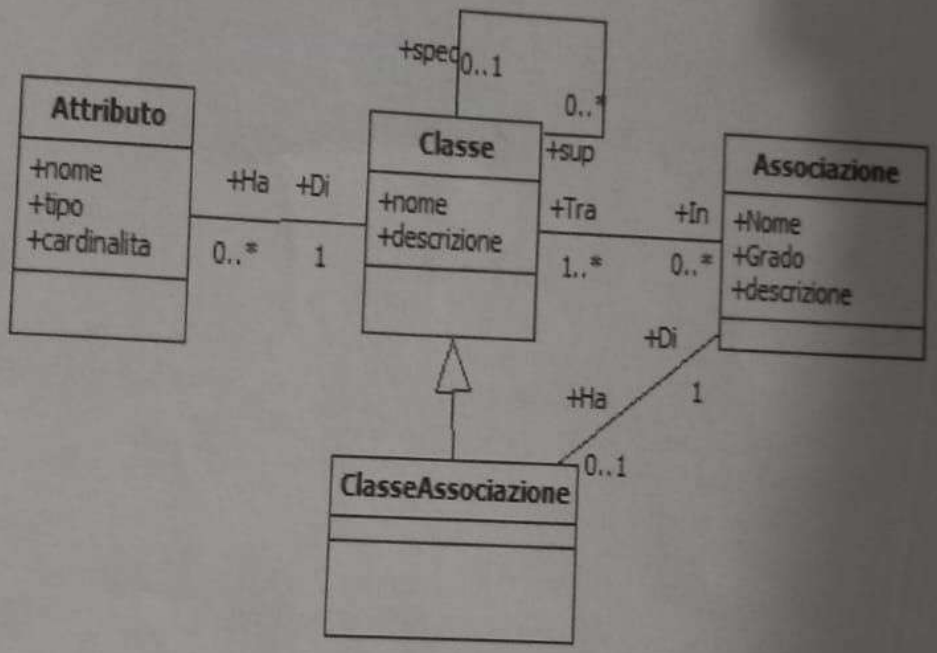
Esercizio 01 *(Punti 7) Si scriva una interrogazione in algebra relazionale che, se valutata, fornisca il Nome ed il Cognome di coloro che, in ogni anno, hanno cambiato residenza il più alto numero di volte.*

Esercizio 02 *(Punti 7) Si scriva una interrogazione in SQL che fornisca Nome e Cognome delle persone che sono state residenti per tutta la vita ESCLUSIVAMENTE nella stessa città.*

Esercizio 03 *(Punti 6) Si implementino nel modo più opportuno i seguenti vincoli (o trigger), motivando opportunamente le scelte fatte:*

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

Esercizio 04 *(Punti 6) Si consideri la bozza di Class Diagram in figura.*



- *Effettuare la ristrutturazione e motivarne il processo;*
- *Disegnare lo schema concettuale ristrutturato;*
- *Definire lo schema logico.*

Esercizio 05 *(Punti 8) Usando SQL dinamico si scriva una funzione che riceve in ingresso una stringa contenente codici fiscali separati da ";" Per OGNUNO DEI CODICI FISCALI IN INPUT la funzione restituisce una stringa così fatta: Codice fiscale, Nome, Cognome ed età di tutti componenti del gruppo familiare ordinati per età DECRESCENTE (i valori sono separati da spazi, ogni componente da ",", i gruppi familiari da ";").*