Sistemi Informativi T
31 gennaio 2020

Tempo a disposizione: 2:30 ore

La consegna deve essere eseguita mediante l'apposito applicativo Web, facendo l'upload dei file specificati sul sito **http://esamix.labx** (solo per l'es. 1 la consegna è su carta)

N.B. Per superare la prova di SI-T è necessario totalizzare almeno 3 punti negli esercizi 1 e 2

1) Algebra relazionale (3 punti totali):

Consegnare le risposte su un foglio di carta, intestato con matricola, nome e cognome

Date le seguenti relazioni, disponibili nello schema **B16884** con dati fittizi di esempio:

```
FURGONI(Targa,Sede);
VIAGGI(IDV,Targa,Data,OraPartenza,OraArrivo,KmPercorsi),
      Targa REFERENCES FURGONI,
      UNIQUE(Targa,Data,OraPartenza);
SOSTE(IDV,OraInizio,NumMinuti),
      IDV REFERENCES VIAGGI;

-- OraPartenza, OraArrivo e OraInizio sono di tipo TIME (es. '12:32').
-- KmPercorsi e NumMinuti sono di tipo INT.
-- L'ora di inizio di una sosta è sempre compresa tra l'ora di partenza
-- e l'ora di arrivo.
-- Per semplicità si suppone che ogni viaggio inizi e termini in uno
-- stesso giorno.
```

si scrivano in algebra relazionale le seguenti interrogazioni:

1.1) [1 p.] Le targhe dei furgoni della sede di Bologna che nel 2020 hanno fatto un viaggio di almeno 500 km facendo almeno una sosta di più di 30 minuti

1.2) [2 p.] Le targhe dei furgoni che non hanno mai fatto 2 o più soste in uno stesso viaggio

---

2) SQL (5 punti totali)

Consegnare il file SQL.txt (o SQL.sql)

Con riferimento al DB dell'esercizio 1, si scrivano in SQL le seguenti interrogazioni:

2.1) [2 p.] Per ogni sede, il tempo medio delle soste dei propri furgoni, considerando solo i viaggi in cui c'è stata almeno una sosta di almeno 15 minuti

2.2) [3 p.] Per ogni anno, si determini la targa del furgone e l'identificativo del relativo viaggio in cui è stata massima la durata totale delle soste rispetto alla durata del viaggio

NB: Per l'uso delle funzioni SQL relative a date, orari e altro si consulti il file FunzioniSQL nella pagina del Lab

Sistemi Informativi T
31 gennaio 2020

3) Progettazione concettuale (6 punti)
Consegnare il file ER.lun

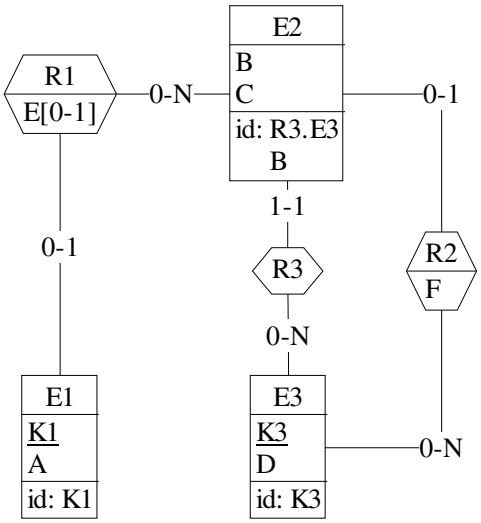
Il Museo di Arte Moderna e Ipermoderna (MAMI) ha istituito la vendita di biglietti online per la visita dei propri spazi e delle esposizioni che si tengono in questi. Ogni utente, registrato con username, password, nome e cognome, può comprare biglietti per sé o anche per altre persone, in quest'ultimo caso fornendo un nominativo (nome e cognome) ed un nickname univoco. I biglietti si possono anche acquistare all'ingresso del museo, ma in questo caso non hanno un nominativo. Tutti i biglietti hanno un codice numerico, che viene generato automaticamente all'atto della vendita, e un timestamp di emissione (quando il biglietto è stato generato e venduto).
Assieme a un biglietto ordinario si può acquistare un biglietto aggiuntivo per un'esposizione. Ogni esposizione ha un titolo, una descrizione, un prezzo, un periodo di svolgimento (data inizio e data fine) e l'elenco delle sale del MAMI presso cui si svolge. Di ogni sala, identificata da un numero o, alternativamente, da un nome (es.: la sala 15 si chiama "Duchamp") il sistema del MAMI mantiene informazioni sui metri quadri e sulla capienza (numero massimo di persone).

---

4) Progettazione logica (6 punti totali)
Consegnare i file SCHEMI.txt e TRIGGER.txt (o SCHEMI.sql e TRIGGER.sql)

Dato lo schema concettuale in figura e considerando che:

a) tutti gli attributi sono di tipo INT;
b) nessuna associazione viene tradotta separatamente;
c) un'istanza di E1 può essere associata, tramite R1 e R2, solo a un'istanza di E3 con D>A;
d) la somma dei valori di B delle istanze di E2 identificate da una stessa istanza di E3 è minore di 50;



4.1) [3 p.] Si progettino gli opportuni schemi relazionali e si definiscano tali schemi in DB2 (sul database SIT_STUD) mediante un file di script denominato SCHEMI.txt (o SCHEMI.sql)

4.2) [3 p.] Per i vincoli non esprimibili a livello di schema si predispongano opportuni **trigger che evitino inserimenti di singole tuple non corrette**, definiti in un file TRIGGER.txt (o TRIGGER.sql) e usando se necessario il simbolo '@' per terminare gli statement SQL (altrimenti ';')

IMPORTANTE:

- I file **NON** devono includere istruzioni di (dis)connessione al DB
- Per il punto 4.2), se necessario, si specifichino usando commenti SQL eventuali inserimenti di tipo transazionale (ossia, più INSERT nella stessa transazione)
- Si prega di attenersi scrupolosamente alle istruzioni relative ai nomi dei file (maiuscole incluse). **Il mancato rispetto delle istruzioni potrà comportare penalizzazioni di punteggio**