INFORMATICA DOCUMENTALE

Laurea in Scienze della Comunicazione – Università degli Studi di Verona

VERIFICA SCRITTA DEL 5 SETTEMBRE 2013

matricola	cognome	nome	firma

A.1 + A.2 + A3	B.1	B.2	B.3	Totale

Istruzioni

- È vietato portare all'esame libri, eserciziari, appunti e dispense. Chiunque venga trovato in possesso di documentazione relativa al corso – anche se non attinente alle domande proposte – vedrà annullata la propria prova.
- Scrivere solo sui fogli distribuiti, cancellando le parti di brutta con un tratto di penna. Non separare questi fogli.
- Tempo a disposizione: 1 ora e 45 minuti.

A. Parte prima

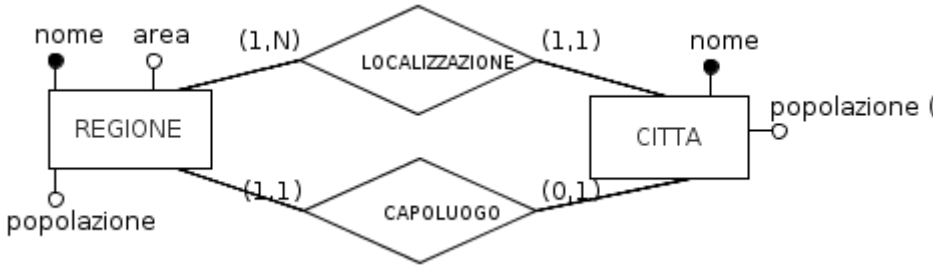
A.1. In al più 10 righe dire cosa significa che i DBMS devono poter gestire basi di dati grandi, persistenti e condivise?

A.2. Utilizzando il modello Entità-Relazione (ER) fare un esempio di relazione ricorsiva.

A.3. Nell'ambito del modello E-R, definire il concetto di identificatore esterno, spiegando quando è necessario con un esempio.

B. Parte seconda

B.1. Considerare il seguente schema ER.



B.1.a. In al più 3 righe, spiegare cosa significa (0,1) vicino all'entità CITTA.

B.1.b. Solo sulla base dell'ER sopra (cioè senza ulteriori vincoli e/o informazioni) è possibile avere due città con lo stesso nome in regioni diverse? Motivare brevemente (max 5 righe) la risposta e in caso di risposta negativa dire come si potrebbe modificare l'ER per renderlo possibile.

B.1.c. Ipotizzando che gli abitanti vivano solamente nelle città riportate, nello schema ER c'è qualche attributo ridondante? Se sì, spiegare vantaggi e svantaggi di avere o non avere tale/i attributo/i.

B.1.d. Se si volesse rappresentare anche il nome del sindaco dei capoluoghi di regione (solo dei capoluoghi) come si potrebbe modificare l'ER?

B.1.e. Tradurre lo schema ER in uno schema relazionale, indicando in quest'ultimo eventuali chiavi, vincoli di non nullità e vincoli di integrità referenziale.

B.2. Considerare il seguente schema relazionale che rappresenta una parte di un sistema per la gestione di gruppi musicali:

```
MUSICISTA(nome, cognome, data_nascita*)
GRUPPO_MUSICALE(nome, tipologia)
STRUMENTO(nome)
PARTECIPAZIONE(nome_musicista_, cognome_musicista, gruppo, strumento)
```

con vincoli di integrità referenziale tra gli attributi nome_musicista e cognome_musicista di PARTECIPAZIONE e i rispettivi attributi della relazione MUSICISTA, tra l'attributo gruppo di PARTECIPAZIONE e l'attributo nome della relazione GRUPPO_MUSICALE e tra l'attributo strumento di PARTECIPAZIONE e l'attributo nome di STRUMENTO, e la seguente istanza.

GRUPPO_MUSICALE	
nome	**tipologia**
Anaconda Fusion	Jazz
New Beatles	Rock
Rondò Veneziano	Classica

MUSICISTA		
nome	**cognome**	**data_nascita**
Claudio	Allevi	NULL
Enrico	Bentivoglio	1968-11-01
Uto	Ughi	NULL
Gianni	Compri	1954-04-24

STRUMENTO
nome
Pianoforte
Sassofono
Violino
Viola
Basso elettrico

PARTECIPAZIONE			
nome_musicista	**cognome_musicista**	**gruppo**	**strumento**
Claudio	Allevi	Rondò Veneziano	Pianoforte
Enrico	Bentivoglio	Rondò Veneziano	Sassofono
Claudio	Allevi	New Beatles	Pianoforte
Enrico	Bentivoglio	Anaconda Fusion	Sassofono
Uto	Ughi	Rondò Veneziano	Violino
Claudio	Allevi	Rondò Veneziano	Viola
Gianni	Compri	Anaconda Fusion	Basso elettrico

B.2.a. Scrivere i comandi SQL per creare le tre tabelle sopra riportate (incluse eventuali chiavi, vincoli di non nullità, vincoli di unicità e vincoli di integrità referenziale). Tenere conto che deve essere evitato di cancellare un musicista o un gruppo, a cui fa riferimento almeno una tupla di PARTECIPAZIONE, mentre se viene eliminato uno strumento in STRUMENTO vengono eliminate i record in PARTECIPAZIONE, che vi facevano riferimento. In caso di modifica del nome o del cognome di un musicista, di un orchestra oppure del nome di uno strumento, tali modifiche si devono riflettere in automatico anche in PARTECIPAZIONE.

B.2.b. Cosa restituisce la seguente interrogazione? (scrivere la tabella risultante)

```
SELECT strumento, COUNT(*) AS numero FROM partecipazione
WHERE strumento <> 'Basso elettrico' GROUP BY strumento ORDER BY strumento;
```

B.2.c. Cosa restituisce la seguente interrogazione? (scrivere la tabella risultante)

```
SELECT cognome_musicista AS musicista
FROM partecipazione INNER JOIN strumento ON partecipazione.strumento = strumento.nome
WHERE gruppo = 'Rondò Veneziano' AND nome_musicista LIKE '%o'
GROUP BY cognome_musicista HAVING COUNT(*) > 1;
```

B.2.d. Scrivere il comando SQL per aggiornare nell'intero database il nome dello strumento “Sassofono” in “Sax”.

B.2.e. Dire se il seguente comando è corretto e in tal caso scrivere quanti record avranno dopo la sua esecuzione le tabelle STRUMENTO e PARTECIPAZIONE sull'istanza sopra riportata, oppure se non lo è spiegare il motivo (max 3 righe).

```
DELETE FROM strumento WHERE nome IN (SELECT DISTINCT strumento FROM partecipazione
WHERE nome_musicista = 'Claudio');
```

B.3. Si consideri il seguente schema SQL, che rappresenta un semplice sistema di gestione ticket. Il sistema prevede degli utenti dei progetti e delle tipologie di attività/servizio. Per richiedere un intervento si apre un ticket, dove si specifica per un certo progetto l'attività richiesta. Il ticket può in seguito essere preso in carico da un utente (incaricato), che, una volta completato il compito (l'attività), chiude il ticket, impostando chiuso a TRUE.

```
CREATE TABLE utenti (
    username VARCHAR(15) PRIMARY KEY,
    password VARCHAR(15) NOT NULL
);
CREATE TABLE progetti (
    nome VARCHAR(32) PRIMARY KEY,
    citta VARCHAR(32)
);
CREATE TABLE attivita (
    descrizione VARCHAR PRIMARY KEY,
    costo INT NOT NULL
);
CREATE TABLE ticket (
    codice INT PRIMARY KEY,
    progetto VARCHAR(32),
    attivita VARCHAR,
    incaricato VARCHAR(15),
    chiuso BOOL DEFAULT FALSE,
    FOREIGN KEY(progetto) REFERENCES progetti(nome) ON UPDATE CASCADE ON DELETE SET
NULL,
    FOREIGN KEY(attivita) REFERENCES attivita(descrizione) ON UPDATE CASCADE ON DELETE
SET NULL,
    FOREIGN KEY(incaricato) REFERENCES utenti(username) ON UPDATE CASCADE ON DELETE
SET NULL
);
```

scrivere i comandi SQL che permettono di:

B.3.a. Ottenere il numero totale dei ticket chiusi dall'utente "Luigi".

B.3.b. Ottenere per ciascun incaricato la somma dei costi delle sue attività non ancora chiuse, ordinate per somma decrescente.

B.3.c. Ottenere i nomi non ripetuti di tutti gli utenti che hanno già chiuso ticket per progetti su Verona.

B.3.d. Aggiungere il ticket con codice 15 per il progetto 'Livenet Univr' per l'attività di 'Addestramento', che costa 100, tenendo conto che non sono ancora presenti nel database né tale progetto né tale attività

B.3.e. Rimuovere i ticket relativi ad attività che appaiono per più di 2 volte nella tabella ticket stessa.