Basi di dati I — Prova di autovalutazione 30 ottobre 2014

La prova verrà discussa in aula, prevedibilmente giovedì 6 novembre. Si consiglia di svolgerlo "simulando l'esame," sulla carta e senza ausilio di libri e appunti. Si consiglia poi di eseguire le interrogazioni SQL su un DBMS. Gli studenti interessati a sostenere le prove parziali **debbono** consegnare su Moodle le soluzioni e i risultati dei test (mostrando anche le basi di dati di esempio utilizzate).

Domanda 1 Considerare la base di dati relazionale definita per mezzo delle seguenti istruzioni:

```
create table studenti (matricola numeric not null primary key,
                       cognome char(20) not null,
                       nome char(20) not null,
                       età  numeric);
create table corsi (codice numeric not null primary key,
                       titolo char(20) not null,
                       CFU numeric not null);
create table esami (corso numeric not null references corsi(codice),
                       studente numeric not null references studenti(matricola),
                       data date not null,
                       voto numeric not null,
                       primary key (corso, studente));
```

Supponendo che le relative relazioni abbiano rispettivamente le cardinalità $S = 10.000$ (studenti), $C = 1.000$ (corsi) e $E = 40.000$ (esami), indicare le cardinalità minime e massime (in simboli e numeri) dei risultati delle seguenti interrogazioni:

	Min (simboli)	Max (simboli)	Min (valore)	Max (valore)
`SELECT matricola, codice` `FROM studenti, corsi`	$S \times C$	$S \times C$	10.000.000	10.000.000

`SELECT *`
`FROM studenti, esami`
`WHERE matricola = studente`
	E	E	40.000	40.000

`SELECT matricola, codice`
`FROM studenti, esami, corsi`
`WHERE matricola = studente`
`  AND corso = codice`
	E	E	40.000	40.000

Domanda 2 Con riferimento alla base di dati usata nella domanda precedente formulare le seguenti interrogazioni in algebra relazionale:

1. trovare matricole e cognomi degli studenti che hanno preso almeno un trenta

$$\pi_{matricola,cognome}(\text{studenti} \rhd\lhd_{matricola=studente} \sigma_{voto=30}(\text{esami}))$$

2. trovare le matricole degli studenti che hanno sostenuto almeno due esami

$$\pi_{studente}(\sigma_{corso \neq corso'}(\text{esami} \bowtie_{studente=studente'} \rho_{X \leftarrow X'}(\text{esami})))$$

Domanda 3 Con riferimento alla base di dati usata nelle domande precedenti, formulare le seguenti interrogazioni in SQL

1. trovare codici e titoli di corsi nei cui esami è stato assegnato almeno un trenta

```
SELECT DISTINCT codice, titolo
FROM corsi JOIN esami ON codice = corso
WHERE voto = 30
```

2. trovare le coppie di studenti (mostrare le sole matricole) per i quali uno dei due ha riportato un voto più alto in tutti gli esami superati da entrambi.

```
SELECT e1.studente, e2.studente
FROM esami e1, esami e2
WHERE e1.voto > e2.voto
AND e1.studente <> e2.studente
AND e1.corso = e2.corso
AND NOT EXISTS (                SELECT *
                                FROM esami e3, esami e4
                                WHERE e3.corso = e4.corso
                                AND e3.studente = e1.studente
                                AND e4.studente = e2.studente
                                AND e3.voto <= e4.voto )
```

3. trovare lo studente con la media più alta; mostrare i dati dello studente, la media in questione e il numero di esami superati

```
CREATE VIEW MediaVoti AS SELECT studente, AVG(voto) AS media, COUNT(*) AS numEsami
FROM esami
GROUP BY studente

SELECT studente, media, numEsami
FROM MediaVoti, studenti
WHERE studente = matricola
AND media = (SELECT MAX(media)
                  FROM MediaVoti)
```

oppure

```
SELECT matricola, cognome, nome, AVG(voto), COUNT(*)
FROM esami join studenti on studente = matricola
GROUP BY matricola, cognome, nome
HAVING AVG(voto) >= ALL
  (SELECT AVG(voto)
  FROM esami
  GROUP BY studente)
```

Domanda 4 Con riferimento al seguente schema di base di dati:

CITTÀ(Nome, Regione, Abitanti)
ATTRAVERSAMENTI(Città, Fiume)
FIUMI(Fiume, Lunghezza)

formulare, in algebra relazionale e in SQL, le seguenti interrogazioni:

1. visualizzare nome, regione e abitanti per le città che (i) hanno più di 50.000 abitanti e (ii) sono attraversate dal Po o dall'Adige;

$$\pi_{nome,regione,abitanti}(\sigma_{abitanti>50000 \wedge (fiume="Po" \vee fiume="Adige")}(\text{Citta} \bowtie_{nome=citta} \text{Attraversamenti}))$$

```
select distinct citta.*
from citta join attraversamenti on nome = citta
where abitanti > 50000
   and (fiume ='Po' or Fiume='Adige');
```

2. trovare le città che sono attraversate da (almeno) due fiumi, visualizzando il nome della città e quello del più lungo di tali fiumi (supponendo per semplicità che nessuna città sia attraversata da più di due fiumi)

$$\text{AttrFiume} := \text{Fiume} \bowtie_{fiume=f} (\rho_{f \leftarrow fiume}(\text{Attraversamenti}))$$

$$\pi_{citta,fiume}(\sigma_{lunghezza>lunghezza'}(\text{AttrFiume} \bowtie_{citta=citta'} \rho_{X \leftarrow X'}(\text{AttrFiume})))$$

```
create view attrfiume as
   select citta, f.fiume, lunghezza
   from attraversamenti a join fiumi f on a.fiume=f.fiume;

select distinct a1.citta, a1.fiume
from   attrfiume a1 join attrfiume a2 on a1.citta =a2.citta
where  a1.lunghezza > a2.lunghezza;
```

3. stessa interrogazione precedente, ma senza l'ipotesi semplificativa

$$\pi_{citta,fiume}(\sigma_{lunghezza>lunghezza'}(\text{AttrFiume} \bowtie_{citta=citta'} \rho_{X \leftarrow X'}(\text{AttrFiume}))) -$$

$$\pi_{citta,fiume}(\sigma_{lunghezza<lunghezza'}(\text{AttrFiume} \bowtie_{citta=citta'} \rho_{X \leftarrow X'}(\text{AttrFiume})))$$

```
select distinct a1.citta, a1.fiume
from attrfiume a1 join attrfiume a2 on a1.citta = a2.citta
where a1.fiume <> a2.fiume
except
select distinct a1.citta, a1.fiume
from attrfiume a1 join attrfiume a2 on a1.citta = a2.citta
where a1.lunghezza < a2.lunghezza
```

oppure

```
select distinct a1.citta, a1.fiume
from attrfiume a1 join attrfiume a2 on a1.citta = a2.citta
where a1.fiume <> a2.fiume
  and not exists (select *
                  from attrfiume a2
                  where a1.citta = a2.citta
                    and a2.lunghezza > a1.lunghezza);
```