Basi di dati

A. Dato il seguente schema logico e i seguenti vincoli, si scriva il codice SQL che crea le tabelle corrispondenti. Si ricordi di creare le tabelle nell'ordine richiesto dal linguaggio SQL.

citta(<u>idCitta, nazione, </u>descrizione, numeroAbitanti)
voli(<u>partenza, arrivo, dataEOraPartenza,</u> intercontinetale, dataEOraArrivo, costoMedioVolo)
aeroporti(<u>id</u>, sigla, nome, idCitta, nazione)

Vincoli integrità referenziale
voli.partenza -> aeroporti.id
voli.arrivo-> aeroporti.id
aeroporti.idCitta, aeroporti.nazione -> citta.idCitta, citta.nazione

Altri vincoli e osservazioni
A parte le chiavi, tutti gli altri campi possono essere nulli.
Il nome dell'aeroporto è unico.
Il colonna intercontinentale contiene i valori "1" o "0".
La colonna descrizione contiene un file di testo lungo fino a 10k.
Il valore di dataEOraArrivo deve essere successivo a quello di dataEOraPartenza.
La colonna costoMedioVolo contiene valori numerici fino a 9.999,99. Il valore di default è 0,00.
La colonna sigla di aeroporto è una stringa di esattamente 3 caratteri. Ogni aeroporto ha una sigla diversa.

*

Sia dato il seguente schema relazionale

cittadino(<u>id</u>, nome, reddito, citta)
citta(<u>id</u>, nome, capolugo, regione, superfice)
regione(<u>id</u>,nome)

Vincoli integrità referenziale
cittadino.citta -> citta.id
citta.regione -> regione.id

B. Scrivere le seguenti interrogazioni in algebra relazionale con theta join e in calcolo su tuple

1. Trovare il nome della città capoluogo (capolugo=true) di maggiore superfice
2. Trovare l'id delle regioni per le quali l'archivio non contiene citta

C. **Scrivere le seguente interrogazioni sia in SQL**
(se necessario, definire delle viste)

3. Trovare tutti i cittadini che abitano in capoluoghi, restituendo il nome del cittadino, il nome dell'eventuale città in cui abita e il nome della regione. Fare attenzione a restituire anche i cittadini per i quali non è stata impostata la città o la regione.
4. Trovare le coppie di citta che appartengono alla stessa regione restituendo il nome di entrambe le città.
5. Trovare la media del reddito dei cittadini di ciascuna regione restituendo il nome della regione e la media
6. Trovare la somma del reddito dei cittadini di ciascuna città, restituendo l'id della città e la somma. Si restituisca solo le città che hanno più di cento cittadini e, nel calcolare la somma, si considerino solo i cittadini con un reddito minore a un milione.
7. Usando gli operatori insiemistici (senza usare le sottointerrogazioni), si trovi l'id delle regioni che non contengono città con superfice più grande di centomila.
8. Usando le sottointerrogazioni (senza usare gli operatori insiemistici), trovare il nome delle città in cui non c'è nessun cittadino con reddito maggiore di centomila.
9. Trovare il nome del cittadino con reddito maggiore.
10. Trovare la differenza nel numero di abitanti fra la citta con più abitanti e quella con meno abitanti

A.

```
create table citta(
	idCitta int not null,
	nazione varchar(20) not null,
	descrizione clob(10k),
	numeroAbitanti int,
	primary key(idCitta,nazione)
)
create table aeroporti(
	id serial not null primary key,
	sigla char(3) unique,
	nome varchar(20) unique,
	idCitta int,
	nazione varchar(20),
	foreign key(id,sigla) references citta(idCitta,nazione)
)

create table voli (
	partenza int not null references aeroporti(id),
	arrivo int not null references aeroporti(id),
	dataEOraPartenza timestamp not null,
	intercontinetale bit,
	dataEOraArrivo timestamp check (dataEOraArrivo> dataEOraPartenza)
	costoMedioVolo  decimal(6,2) default 0.00,
	primary key(partenza, arrivo, dataEOraPartenza)
)
```

B. Scrivere le seguenti interrogazioni in algebra relazionale con theta join e in calcolo su tuple

1. Trovare il nome della città capoluogo (capolugo=true) di maggiore superfice

   capoluoghi=$\pi_{id,nome,superfice}(\sigma_{citta=capoluo} \quad (citta))$
   capoluoghi2=$\delta_{id2,nome2,cuperifice2< \quad ,nome,superfice}(capoluoghi)$
   cittaNonMaggiore =$\pi_{id,nome}$ (capoluogi $\bowtie_{superfice<superfice2}$ capoluoghi2)
   π_{nome} ($\pi_{id,nome}$ (citta) -$cittaNonMaggiore$))

   {c.nome | c(citta)| c.capoluogo=true $\wedge$ $\neg\exists$ c2(citta) c2.superfice>c.superfice $\wedge$ c2.capoluogo=true }

2. Trovare l'id delle regioni per le quali l'archivio non contiene citta

$$\pi_{id}(regione) - \delta_{id<-region} \ (\pi_{regione}(citta))$$

   {r.id | r(regione)| $\neg\exists$ s(citta) s.regione=r.id }

C. **Scrivere le seguente interrogazioni sia in SQL**
   (se necessario, definire delle viste)

3. Trovare tutti i cittadini che abitano in capoluoghi, restituendo il nome del cittadino, il nome dell'eventuale città in cui abita e il nome della regione. Fare attenzione a restituire anche i cittadini per i quali non è stata impostata la città o la regione.

```
select cittadino.nome,citta.nome,regione.nome
from cittadino left join citta on cittadino.citta=citta.id and capoluogo=true
   left join regione on citta.regione=regione.id
```

4. Trovare le coppie di citta che appartengono alla stessa regione restituendo il nome di entrambe le città.

```
select c1.nome as "citta 1", c2.nome "citta 2"
from citta  as c1, citta as  c2
where c1.regione=c2.regione and c1.id>c2.id
```

5. Trovare la media del reddito dei cittadini di ciascuna regione restituendo il nome della regione e la media

```
select regione.nome, avg(reddito) as "reddito medio"
from cittadino, citta, regione
where cittadino.citta=citta.id
        and citta.regione=regione.id
group by regione.id, regione.nome
```

6. Trovare la somma del reddito dei cittadini di ciascuna città, restituendo l'id della città e la somma. Si restituisca solo le città che hanno più di cento cittadini e, nel calcolare la somma, si considerino solo i cittadini con un reddito minore di un milione.

```
select citta.nome, sum(reddito) as "reddito medio"
from cittadino, citta, regione
where cittadino.citta=citta.id
     and reddito <1000000
group by citta.id, citta.nome
having count(*)>100
```

7. Usando gli operatori insiemistici (senza usare le sottointerrogazioni), si trovi l'id delle regioni che non contengono città con superfice più grande di centomila.

```
select regione.id
from regione
except
select regione.id
from regione, citta
where citta.regione=regione.id and citta.superfice>100000
```

8. Usando le sottointerrogazioni (senza usare gli operatori insiemistici), trovare il nome delle città in cui non c'è nessun cittadino con reddito maggiore di centomila.

```
select citta.nome
from citta
where not exists(select *
                 from cittadino
                 where cittadino.citta=citta.id
                      and cittadino.reddito>100000)
```

9. Trovare il nome del cittadino con reddito maggiore.

```
select nome
from cittadino
where reddito =(select max(reddito)
                from cittadino
                )
```

10. Trovare la differenza nel numero di abitanti fra la citta con più abitanti e quella con meno abitanti

```
create view numeroabitanti(id,numero)  as
 select citta.id, count(*)
 from citta, cittadino
 where cittadino.citta=citta.id
 group by citta.id

select max(numero)-min(numero) as "differenza max-min"
from numeroabitanti
```