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.

squadre(id, nome, citta, PIVA, mediaPunti, entrate, uscite, totale)
partite(squadraCasa, squadraOspite, data, idArbitro, nazioneArbitro, numeroSpettatori,filmato)
arbitri(id, nazione, nome, cognome, internazionale)

Vincoli integrità referenziale
partite.squadraCasa -> squadre.id
partite.squadraOspite -> squadre.id
partite.idArbitro, partite.nazioneArbitro -> arbitri.id, arbitri.nazione

Altri vincoli e osservazioni
A parte le chiavi, tutti gli altri campi possono essere nulli.
Il nome di ogni squadra è unico
La colonna mediaPunti contiene valori numerici in virgola mobile.
Per il filmato si memorizza un file binario fino a 100M.
Le colonne entrate, uscite e totale contengono valori fino a 999.999.999,99.
La colonna totale deve essere uguale alle differenza fra entrate e uscite.
La partita IVA (PIVA) è una sequenza di caratteri lunga esattamente 11 caratteri ed è unica.
La colonna internazionale contiene uno dei valori "0" o "1". Il valore di default è "0".

```
CREATE TABLE arbitri(
id int not null,
nazione varchar(20) not null,
nome varchar(20),
cognome varchar(20),
internazionale bit default 0,
primary key(id,nazione))

CREATE TABLE squadre(
id serial not null primary key,
nome varchar(20) unique,
citta varchar(20),
PIVA char(11)  unique,
mediapunti float,
entrate decimal(11,2),
uscite decimal(11,2),
totale decimal(11,2) check(totale=entrate-uscite))

CREATE TABLE partite (
squadracasa int not null references squadre(id),
squadraospite in not null references squadre(id),
data date not null,
idArbitro int,
nazioneArbitro varchar(20),
numeroSpettatori int,
filmato blob(100M),
primary key(squadracasa,squadraospite,data),
foreign key(idarbitro,nazionearbitro) refrences arbitri(id,nazione))
```

Sia dato il seguente schema relazionale

giocatori(<u>id</u>, nome, gol, nazione, squadra)
squadre(<u>id</u>, nome, nazione, allenatore)
allenatori(id, nome)

Vincoli integrità referenziale
giocatori.squadra -> squadre.id
squadre.allenatore -> allenatori.id

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

1. Trovare il primo nome di un allenatore in ordine lessicografico ascendente, cioè il primo quando i nomi sono ordinati dalla A alla Z.

$allenatori2 = \delta_{id2,nome2<-id,nome}(allenatori)$
$allenatoriNonPrimo = \pi_{id,nome}(allenatori \bowtie_{nome>nome2} allenatori2)$
$\pi_{nome}(\pi_{id,nome}(allenatori) - allenatoriNonPrimo\ )$

{a.nome |a(allenatori) | ¬∃ a2(allenatori) a2.nome<a.nome}

2. Trovare l'id delle squadre che non hanno giocatori.

$\pi_{id}(squadra) - \delta_{id<-squadra}(\pi_{squadra}(giocatori)\ )$

{s.id |s(squadra) | ¬∃ g(giocatori) g.squadra=s.id}

3. Trovare l'id delle squadre che hanno almeno due giocatori della stessa nazionalità.

$giocatori2 = \delta_{id2,nome2\ldots<-id,nome\ldots}(giocatori)$
$squadreConDueGiocatori = \pi\_squadra\ (giocatori \bowtie \_(id \neq id2 \wedge squadra = squadra2 \wedge nazione = nazione2)\ giocatori2)$

{s.id |s(squadra) | s.id=g.squadra ∧
∃ g1(giocatore), ∃ g2(giocatore)
g1.id≠g2.id ∧g1.squadra=g2.squadra2∧g1.nazione=g2.nazione }

oppure

{s.id |s(squadra) ,g1(giocatore), g2(giocatore) |
s.id=g.squadra ∧ g1.id≠g2.id
∧g1.squadra=g2.squadra2∧g1.nazione=g2.nazione }

4. Trovare l'id delle squadre che non hanno almeno due giocatori della stessa nazionalità.

Avendo definito squadreConDueGiocatori come nell'esercizio precedente

$\pi_{id}(squadra) - \delta_{id<-squadra}(\text{squadreConDueGiocatori}\ )$

{s.id |s(squadra) | ¬∃ g1(giocatore) ∃ g2(giocatore)
g1.id≠g2.id ∧g1.squadra=g2.squadra2∧g1.nazione=g2.nazione}

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

5. Trovare tutti i giocatori italiani (nazione=Italia), restituendo il nome del giocatore, il nome dell'eventuale squadra in cui gioca e il nome dell'allenatore. Fare attenzione a restituire anche i giocatori senza contratto e le squadre senza allenatore.

```
SELECT giocatori.nome, squadre.nome, allenatore.nome
FROM giocatori LEFT JOIN squadre ON giocatori.squadra=giocatori.id
        LEFT JOIN allenatori ON squadre.allenatori=allenatori.id
WHERE giocatori.nazione='Italia'
```

6. Trovare le coppie dei giocatori che hanno segnato gli stessi gol, restituendo il nome di entrambi i giocatori

```
SELECT g1.nome, g2.nome
FROM giocatori g1, giocatori g2
WHERE g1.gol=g2.gol AND g1.id>g2.id
```

7. Trovare la somma totale dei gol fatti dalle squadre di ciascun allenatore restituendo il nome dell'allenatore e il numero di gol

```
SELECT SUM(gol), allenatori.nome
FROM giocatori, squadre, allenatori
WHERE giocatori.squadra=squadre.id AND squadre.allenatore=allenatori.id
GROUP BY allenatori.id, allenatori.nome
```

8. Trovare la media goal fatti dai giocatori di ciascuna nazione, restituendo la nazione e la media. Si restituisca solo quelle nazioni che hanno più di tre giocatori e si consideri, nel calcolare la media, solo i giocatori con più di 2 gol

```
SELECT avg(gol), nazione
FROM giocatori
WHERE gol>2
GROUP nazione
HAVING count(*) >3
```

9. Usando gli operatori insiemistici (senza usare le sottointerrogazioni), si trovi l'id degli allenatori senza squadra nell'archivio

```
SELECT id
FROM allenatori
EXCEPT
SELECT allenatore AS id
FROM squadre
```

10. Usando le sottointerrogazioni (senza usare gli operatori insiemistici), trovare il nome delle squadre in cui non giocano giocatori stranieri (di nazionalità diversa da quella della squadra).

```
SELECT nome
FROM squadre
WHERE NOT EXIST (SELECT *
                 FROM giocatori
                 WHERE giocatore.nazione <> squadra.nazione
                       AND giocatore.quadra=squadra.id)
----


SELECT nome
FROM squadre
WHERE squadra.id <> all  (SELECT id
                 FROM giocatori, squadre
                 WHERE giocatore.nazione <> squadra.nazione
                       AND giocatore.quadra=squadra.id)
```

11. Trovare il nome del giocatore che ha fatto più gol

```
SELECT nome
FROM giocatori
WHERE gol=(SELECT max(gol) FROM giocatori)
--- oppure -----------
CREATE VIEV maxgol(gol)
AS SELECT max(gol) FROM giocatori

SELECT nome
FROM giocatori, maxgol
WHERE giocatori.gol=maxgol.gol
```

12. Trovare la differenza di gol fra la squadra che ha fatto più gol e quella che ne ha fatti di meno

```
CREATE VIEW  gol_sq(id,gol)
AS      SELECT id, SUM(gol)
        FROM squadre, giocatori
        WHERE squadre.id=giocatori.squadra
        GROUP BY squadre.id

SELECT MAX(gol)-MIN(gol)
FROM gol_sq
```

D. 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.

persone(id,nome,cognome,CF)
vendite(venditore, compratore, idImmobile, provinciaImmobile, data, prezzo, contratto)
immobili(id, provincia, dimensione, commerciale, annoCostruzione, annoUltimaRistrutturazione)

Vincoli integrità referenziale
vendite.venditore -> prersone.id
vendite.compratore -> prersone.id
vendite.idImmobile, vendite.provinciaImmobile -> immobili.id, immobili.provincia

Altri vincoli e osservazioni
A parte le chiavi, tutti gli altri campi possono essere nulli.
Il codice fiscale (CF) è una sequenza di caratteri lunga esattamente 16 caratteri ed è unico.
Il prezzo di vendita è un valore fino a 99.999.999,99.
La colonna dimensione contiene valori numerici in virgola mobile.
La colonna commerciale contiene i valori "1" o "0". Il default è 0.
La colonna contratto contiene un file binario grande fino a 10M.
La provincia è codificata esattamente con due caratteri.
L'anno della colonna annoUltimaRistrutturazione deve essere successiva alla colonna annoCostruzione

Sia dato il seguente schema relazionale

piloti(id,nome,vittorie,auto,nazione)
caseAuto(id,nome,nazione,budget)
nazione(id,nome,continente)

Vincoli integrità referenziale
piloti.auto -> auto.id
piloti.nazione -> nazione.id
caseAuto.nazione ->nazione.id

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

1. Trovare il nome del pilota con più vittorie
2. Trovare l'id delle nazioni europee (continente=europa) senza case automobilistiche

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

3. Trovare le coppie di case automobilistiche che appartengono alla stessa nazione, restituendo il nome di entrambe le case.
4. Trovare tutti i piloti con più di tre vittorie, restituendo il nome del pilota, il nome dell'eventuale casa automobilistica con cui corre e il nome della nazione della casa. Fare attenzione a restituire anche i piloti senza auto e le case automobilistiche per le quali non sia stata impostata la nazione.
5. Trovare la somma totale delle vittorie di ciascuna nazione restituendo il nome della nazione e il numero delle vittorie
6. Trovare la media delle vittorie dei piloti di ogni nazione, restituendo l'id della nazione e la media. Si restituisca solo quelle nazioni che hanno più di due piloti e si consideri, nel calcolare la media, solo i piloti con più di due vittorie.
7. Usando le sottointerrogazioni (senza usare gli operatori insiemistici), trovare il nome delle case automobilistiche per le quali non ci sono piloti con nazionalità diversa da quella della stesa casa automobilistica.
8. Usando gli operatori insiemistici (senza usare le sottointerrogazioni), si trovi l'id delle case automobilistiche senza piloti.
9. Trovare la differenza di vittorie fra la casa automobilistica con maggior numero di vittorie e quella con minor numero di vittorie
10. Trovare il nome del pilota che ha più vittorie