Capitolo 4

Esercizio 4.1

Ordinare i seguenti domini in base al valore massimo rappresentable, supponendo che `integer` abbia una rappresentazione a 32 bit e `smallint` a 16 bit: `numeric(12,4)`, `decimal(10)`, `decimal(9)`, `integer`, `smallint`, `decimal(6,1)`.

Soluzione:

Dominio	Valore Massimo
1. `decimal(10)`	9999999999
2. `integer`	4294967296
3. `decimal(9)`	999999999
4. `numeric(12,4)`	99999999.9999
5. `decimal(6,1)`	99999.9
6. `smallint`	65536

Esercizio 4.2

Definire un attributo che permetta di rappresentare stringhe di lunghezza massima pari a 256 caratteri, su cui non sono ammessi valori nulli e con valore di default "sconosciuto".

Soluzione:

```
Create domain STRING as character varying (256) default 'sconosciuto'
not null
```

Esercizio 4.3

Dare le definizioni SQL delle tre tabelle

```
FONDISTA(Nome, Nazione, Età)
GAREGGIA(NomeFondista, NomeGara, Piazzamento)
GARA(Nome, Luogo, Nazione, Lunghezza)
```

rappresentando in particolare i vincoli di foreign key della tabella GAREGGIA.

Soluzione:

```
Create Table FONDISTA
(
Nome     character(20)      primary key,
Nazione  character(30),
Età      smallint
)

Create table GARA
(
Nome     character(20)     primary key,
Luogo    character(20),
Nazione  character(20),
Lunghezza         integer
)

Create table GAREGGIA
(
NomeFondista      character(20)   references FONDISTA(Nome),
NomeGara          character(20),
Piazzamento       smallint,
primary key (NomeFondista, NomeGara),
foreign key (NomeGara)    references GARA(Nome)
)
```

Esercizio 4.4

Dare le definizioni SQL delle tabelle

```
AUTORE (Nome, Cognome, DataNascita, Nazionalità)
LIBRO (TitoloLibro, NomeAutore, CognomeAutore, Lingua)
```

Per il vincolo foreign key specificare una politica di `cascade` sulla cancellazione e di `set null` sulle modifiche.

Soluzione:

```
Create table AUTORE
(
Nome              character(20),
Cognome           character(20),
DataNascita       date,
Nazionalità       character(20),
primary key(Nome, Cognome)
)

Create table LIBRO
(
TitoloLibro       character(30)     primary key,
NomeAutore        character(20),
CognomeAutore     character(20),
Lingua            character(20),
foreign key (NomeAutore, CognomeAutore)
                  references AUTORE(Nome, Cognome)
                  on delete cascade
                  on update set NULL
)
```

Esercizio 4.5

Dato lo schema dell'esercizio precedente, spiegare cosa può capitare con l'esecuzione dei seguenti comandi di aggiornamento:

```
delete from AUTORE where Cognome = 'Rossi'

update LIBRO set NomeAutore= 'Umberto'
          where CognomeAutore = 'Eco'

insert into AUTORE(Nome,Cognome) values('Antonio','Bianchi')

update AUTORE set Nome = 'Italo' where Cognome = 'Calvino'
```

Soluzione:

1. Il comando cancella dalla tabella AUTORE tutte le tuple con Cognome = 'Rossi'. A causa della politica `cascade` anche le tuple di LIBRO con CognomeAutore = 'Rossi' verranno eliminate.

2. Il comando non è corretto: Nome e Cognome sono attributi della tabella AUTORE e non della tabella LIBRO
3. Il comando aggiunge una nuova tupla alla tabella AUTORE. Non ha alcun effetto sulla tabella LIBRO
4. Le tuple di AUTORE con Cognome = Calvino vengono aggiornate a Nome = Italo. A causa della politica `set null` gli attributi NomeAutore e CognomeAutore delle tuple di Libro con CognomeAutore = Calvino vengono posti a `NULL`.

Esercizio 4.6

Date le definizioni:

```
create domain Dominio as integer default 10
create table Tabella (Attributo Dominio default 5)
```

indicare cosa avviene in seguito ai comandi:

```
1. alter table Tabella alter column Attributo drop default
2. alter domain Dominio drop default
3. drop domain Dominio
```

Soluzione:

1. Il comando cancella il valore di default di Attributo. Il valore di default dopo il comando sarà quello impostato in Dominio, ossia 10.
2. Il comando cancella il valore di default di Dominio. Dopo l'operazione il valore di default sarà `NULL`
3. Il comando cancella l'intero dominio Domain. In Tabella il dominio di Attributo diventerà `integer`

Esercizio 4.7

Dato il seguente schema:

```
AEROPORTO(Città, Nazione,NumPiste)
VOLO(IdVolo,GiornoSett,CittàPart,OraPart,
                  CittàArr,OraArr,TipoAereo)
AEREO(TipoAereo,NumPasseggeri,QtaMerci)
```

scrivere le interrogazioni SQL che permettono di determinare:

1. Le città con un aeroporto di cui non è noto il numero di piste;
2. Le nazioni da cui parte e arriva il volo con codice AZ274;
3. I tipi di aereo usati nei voli che partono da Torino;
4. I tipi di aereo e il corrispondente numero di passeggeri per i tipi di aereo usati nei voli che partono da Torino. Se la descrizione dell'aereo non è disponibile, visualizzare solamente il tipo;
5. Le città da cui partono voli internazionali;
6. Le città da cui partono voli diretti a Bologna, ordinate alfabeticamente;
7. Il numero di voli internazionali che partono il giovedì da Napoli;
8. Il numero di voli internazionali che partono ogni settimana da città italiane (farlo in due modi, facendo comparire o meno nel risultato gli aeroporti senza voli internazionali);
9. Le città francesi da cui partono più di venti voli alla settimana diretti in Italia;

10. Gli aeroporti italiani che hanno solo voli interni. Rappresentare questa interrogazione in quattro modi:
    a) con operatori insiemistici;
    b) con un interrogazione nidificata con l'operatore `not in`;
    c) con un interrogazione nidificata con l'operatore `not exists`;
    d) con l'outer join e l'operatore di conteggio
11. Esprimere l'interrogazione pure in algebra relazionale;
12. Le città che sono servite dall'aereo caratterizzato dal massimo numero di passeggeri;
13. Il massimo numero di passeggeri che possono arrivare in un aeroporto italiano dalla Francia il Giovedì (se vi sono più voli, si devono sommare i passeggeri)

Soluzione:

```
1.      select Città
        from AEROPORTO
        where NumPiste is NULL

2.      select A1.Nazione, A2.Nazione
        from AEROPORTO as A1 join VOLO on A1.Città=CittàArr
        join AEROPORTO as A2 on CittàPart=A2.Città
        where IdVolo= 'AZ274'

3.      select TipoAereo
        from VOLO
        where CittàPart='Torino'

4.      select VOLO.TipoAereo, NumPasseggeri
        from VOLO left join AEREO
        on  VOLO.TipoAereo=aereo.TipoAereo
        where CittàPart= 'Torino'

5.      select CittàPart
        from AEROPORTO as A1 join VOLO on CittàPart=A1.Città
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione <> A2.Nazione

6.      select CittàPart
        from VOLO
        where CittàArr= 'Bologna'
        order by CittàPart

7.      select count(*)
        from VOLO join AEROPORTO on CittàArr=Città
        where CittàPart = 'Napoli' and Nazione <> 'Italia' and
        GiornoSett= 'Giovedì'

8.  a.  select count(*), CittàPart
        from AEROPORTO as A1 join VOLO  on A1.Città=CittàPart
        join AEROPORTO as A2 on CittàArr=A2.Città
        where A1.Nazione='Italia' and A2.Nazione <> ' Italia'
        group by CittàPart
```

b.
```
select count(CittàArr)
from AEROPORTO as A1 join VOLO  on A1.Città=CittàPart
join AEROPORTO as A2 on CittàArr=A2.Città
where A1.Nazione='Italia' and A2.Nazione <> ' Italia'
group by CittàPart
```

9.
```
select CittàPart
from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
join AEROPORTO as A2 on CittàArr=A2.Città
where A1.Nazione='Francia' and A2.Nazione= 'Italia'
group by CittàPart
Having count(*) >20
```

10. a.
```
select CittàPart
from VOLO join AEROPORTO on CittàPart=Città
where Nazione = 'Italia'
except
select CittàPart
from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
join AEROPORTO as A2 on CittàArr=A2.Città
where (A1.Nazione=' Italia ' and A2.Nazione<>' Italia ' )
```

b.
```
select CittàPart
from VOLO join AEROPORTO on CittàPart=Città
where Nazione= 'Italia' and CittàPart not in
( select CittàPart
from AEROPORTO as A1 join VOLO on
A1.Città=CittàPart join AEROPORTO as A2 on CittàArr=A2.Città
where  A1.Nazione='Italia' and
A2.Nazione<> 'Italia'  )
```

c.
```
select CittàPart
from VOLO join AEROPORTO as A1 on CittàPart=Città
where Nazione= 'Italia' and
not exists ( select * from VOLO join AEROPORTO as A2
on A2.Città=CittàArr
where A1.Città=CittàPart and
A2.Nazione<>'Italia'  )
```

d.
```
select CittàPart
from AEROPORTO as A1 join VOLO on
A1.Città=CittàPart left join AEROPORTO as A2 on
(CittàArr=A2.Città and A2.Nazione='Italia')
where A1.Nazione='Italia'
group by CittàPart
having count (district A2.Nazione)= 1 )
```

e. $\Pi_{\text{Cittàpart}}\ \sigma_{\text{Nazione='Italia'}}(\text{AEROPORTO} \rhd\lhd_{\text{Città=CittàPart}} \text{VOLO})$
$-$
$\Pi_{\text{CittàPart}}\ \sigma_{\text{Nazione='Italia'}}\ (\text{AEROPORTO} \rhd\lhd_{\text{Città=CittàPart}} \text{VOLO} \rhd\lhd_{\text{CittàArr=CittàK}}\ \rho_{\text{CittàK,NazioneK,nK}\leftarrow\text{Città,Nazione,NumPiste}}(\sigma_{\text{Nazione}\neq\text{'Italia'}}(\text{AEROPORTO})))$

11.
```
select CittàPart
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri=      (select
                          max( NumPasseggeri )from AEREO )
union
select CittàArr
from VOLO join AEREO on VOLO.TipoAereo=AEREO.TipoAereo
where NumPasseggeri=      (select max( NumPasseggeri )
                           from AEREO)
```

12.
```
create view Passeggeri(Numero)
as select sum ( NumPasseggeri )
from AEROPORTO as A1 join VOLO on A1.Città=CittàPart
        join AEROPORTO as A2 on A2.Città=CittàArr
        join AEREO on VOLO.TipoAereo=Aereo.TipoAereo
where A1.Nazione='Francia' and A2.Nazione='Italia'
    and GiornoSett='Giovedì'
group by A2.Città

select max(Numero)
from Passeggeri
```

Esercizio 4.8

Dato il seguente schema:

```
DISCO(NroSerie, TitoloAlbum,Anno,Prezzo)
CONTIENE(NroSerieDisco,CodiceReg,NroProg)
ESECUZIONE(CodiceReg,TitoloCanz,Anno)
AUTORE(Nome,TitoloCanzone)
CANTANTE(NomeCantante,CodiceReg)
```

formulare le interrogazioni SQL che permettono di determinare:

1. I cantautori (persone che hanno cantato e scritto la stessa canzone) il cui nome inizia per 'D';
2. I titoli dei dischi che contengono canzoni di cui non si conosce l'anno di registrazione;
3. I pezzi del disco con numero di serie 78574, ordinati per numero progressivo, con indicazione degli interpreti per i pezzi che hanno associato un cantante;
4. Gli autori e i cantanti puri, ovvero autori che non hanno mai registrato una canzone e cantanti che non hanno mai scritto una canzone;
5. I cantanti del disco che contiene il maggior numero di canzoni;
6. Gli autori solisti di "collezioni di successi" (dischi in cui tutte le canzoni sono di un solo cantante e in cui almeno tre registrazioni sono di anni precedenti la pubblicazione del disco);
7. I cantanti che non hanno mai registrato una canzone come solisti;
8. I cantanti che non hanno mai inciso un disco in cui comparissero come unici cantanti;
9. I cantanti che hanno sempre registrato canzoni come solisti.

Soluzione:

1.
```
select NomeCantante
from  CANTANTE join ESECUZIONE on
CANTANTE.CodiceReg=ESECUZIONE.CodiceReg
join AUTORE on ESECUZIONE.TitoloCanz=AUTORE.TitoloCanzone
where Nome=NomeCantante and Nome like 'd%'
```

2.
```
select TitoloAlbum
from DISCO join CONTIENE
        on DISCO.NroSerie=CONTIENE.NroSerieDisco
        join ESECUZIONE on
        CONTIENE.CodiceReg=ESECUZIONE.CodiceReg
where ESECUZIONE.anno is NULL
```

3.
```
select NroProg, TitoloCanz, NomeCantante
from (CONTIENE left join CANTANTE on
        CONTIENE.NroSerieDisco=CANTANTE.CodiceReg)
        join ESECUZIONE
        on CONTIENE.codiceReg= ESECUZIONE.CodiceReg
where NroSerieDisco=78574
order by NroProg
```

4.
```
select Nome
from AUTORE
where Nome not in
        ( select NomeCantante
          from CANTANTE )
union
select NomeCantante
from CANTANTE
where NomeCantante not in
      ( select Nome
                  from AUTORE )
```

5.
```
create view DiscoNumerato (NroSerieDisco,NumCanzoni)
as select  NroSerieDisco , count(*)
from CONTIENE
group by NumSerieDisco

select NomeCantante
from CANTANTE join CONTIENE on
                CANTANTE.CodiceReg=CONTIENE.CodiceReg
                join DiscoNumerato on

CONTIENE.NroSerieDisco=DiscoNumerato.NroSerieDisco
where NumCanzoni= ( select max (NumCanzoni)
                            from DiscoNumerato)
```

6.
```
select NroSerie
from DISCO
where NroSerie not in
( select NroSerieDisco
  from CONTIENE join CANTANTE as S1 on
  CONTIENE.CodiceReg=S1.CodiceReg
  join CANTANTE as S2 on
  CONTIENE.CodiceReg =S2.CodiceReg
  where   S1.NomeCantante<>S2.NomeCantante )
   and NroSerie in
 ( select NroSerieDisco
  from (CONTIENE join ESECUZIONE on
   CodiceReg= CodiceReg ) join DISCO on
  DISCO.NroSerie=CONTIENE.NroSerieDisco)
   where ESECUZIONE.Anno<DISCO.Anno
  group by NroSerieDisco
  having count(*) >=3 )
```

7.
```
select distinct NomeCantante
from CANTANTE
where NomeC not in
     ( select S1.NomeCantante
       from CANTANTE as S1
       where CodiceReg not in
      ( select CodiceReg
        from CANTANTE S2
        where S2.NomeCantante <> S1.NomeCantante ) )
```

8.
```
Create view CantantiUnDisco (NomeCantante) as
select NomeCantante
from CONTIENE join CANTANTE on
         CONTIENE.CodiceReg=CANTANTE.CodiceReg
where NroSerieDisco not in
     (  select NroSerieDisco
        from CONTIENE join CANTANTE as S1 on
                      CONTIENE.CodiceReg=S1.CodiceReg
                       join CANTANTE as S2 on
                      CodiceReg=S2.CodiceReg
        where S1.NomeCantante<>S2.NomeCantante  )

select NomeCantante
from CANTANTE
where NomeCantante not in (select NomeCantante
                           from CantantiUnDisco)
```

9.
```
select NomeCantante
from CANTANTE
where NomeCantante not in
( select S1.NomeCantante
from CANTANTE as S1 join ESECUZIONE on
CodiceReg=S1.CodiceReg join CANTANTE as S2 on
CodiceReg=S2.CodiceReg )
where S1.NomeCantante<> S2.NomeCantante )
```

Esercizio 4.9

Dare una sequenza di comandi di aggiornamento che modifichi l'attributo Stipendio della tabella Impiegato, aumentando del 10% gli stipendi sotto i 30 mila € diminuendo del 5% gli stipendi sopra i 30 mila € (gli stipendi di 30.000 € rimangono invariati).

Soluzione:

```
update Impiegato set Stipendio= Stipendio*0.5
where Stipendio < 30000

update Impiegato set  Stipedio= Stipendio*0.95
where Stipendio > 30000

update Impiegato set Stipendio= Stipendio*2.2
where Stipendio < 15000
```

Esercizio 4.10

Definire sulla tabella Impiegato il vincolo che il dipartimento Amministrazione abbia meno di 100 dipendenti, con uno stipendio medio superiore ai 40 mila €.

Soluzione:

```
check (100 >= ( select count(*)
                 from Impiegato
                where Dipartimento='Amministrazione' )
      and 40000 <= ( select avg(Stipendio)
                     from Impiegato
                     where
                     Dipartimento='Amministrazione'))
```

Esercizio 4.11

Definire a livello di schema il vincolo che il massimo degli stipendi degli impiegati di dipartimenti con sede a Firenze sia minore dello stipendio di tutti gli impiegati del dipartimento Direzione.

Soluzione:

```
create assertion ControlloSalari
  check ( not exists(     select *
                          from Impiegato join Dipartimento on
                          Impiegato.Dipartimento=Dipartimento.Nome
                          where Dipartimento.Città='Firenze' and
                          Stipendio > (    select min(Stipendio)
                                           from Impiegato
                                           where
                                           Dipartimento='Direzione')
         )          )
```

Esercizio 4.12

Definire una vista che mostra per ogni dipartimento il valore medio degli stipendi superiori alla media del dipartimento

Soluzione:

```
create view SalariSopraMedia (Dipartimento,Stipendio) as
select Dipartimento, avg(Stipendio)
from I
where Stipendio > ( select avg(Stipendio)
                           from Impiegato as I )
                    where I.Dipartimento=I.Dipartimento )
group by Dipartimento
```

Esercizio 4.13

Tramite la definizione di una vista permettere all'utente "`Carlo`" di accedere al contenuto di `Impiegato`, escludendo l'attributo S`tipendio`.

Soluzione:

Ipotizzando la tabella Impiegato

```
Impiegato(Codice, Nome, Cognome, Stipendio, Dipartimento)
create view ImpiegatoRistretto
(Codice,Nome,Cognome,Dipartimento) as
select  Codice,Nome,Cognome,Dipartimento
from Impiegato

grant select on ImpiegatoRistretto to Carlo
```

Esercizio 4.14

Descrivere l'effetto delle seguenti istruzioni: quali autorizzazioni sono presenti dopo ciascuna istruzione? (ciascuna linea è preceduta dal nome dall'utente che esegue il comando)

```
Stefano:        grant select on Table1 to Paolo,Riccardo
                        with grant option
Paolo:          grant select on Table1 to Piero
Riccardo:       grant select on Table1 to Piero with grant option
Stefano:        revoke select on Table1 from Paolo cascade
Piero:          grant select on Table1 to Paolo
Stefano:        revoke select on Table1 from Riccardo cascade
```

Soluzione:

1. Stefano concede a Paolo e a Riccardo l'autorizzazione di `select` e di concedere a loro volta l'autorizzazione
2. Paolo concede a Piero l'autorizzazione di `select`
3. Riccardo concede a Piero l'autorizzazione di `select` e di `grant`. Ora Piero ha 2 diverse autorizzazioni sulla tabella.
4. Stefano revoca l'autorizzazione data a Paolo. A causa dell'attributo `cascade` anche Piero perde le autorizzazioni concesse da Paolo ma continua ad avere quella concessa da Riccardo.
5. Ora Paolo può di nuovo accedere alla tabella grazie all'autorizzazione concessa da Piero
6. Stefano revoca l'autorizzazione di Riccardo e tramite `cascade` anche di Piero e di Paolo. Ora solo Stefano ha autorizzazioni sulla tabella.