Basi di dati — 26 novembre 2018 — Prova parziale — Compito A

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore
`SELECT *` `FROM utenze JOIN persone ON titolare=CF`				

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`				

`SELECT CF, cognome, nome, count(*) AS numeroUtenze`
`FROM persone JOIN utenze ON CF = titolare`
`GROUP BY CF, cognome, nome`				

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice fiscale, nome e cognome delle persone che non hanno nessuna utenza

Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna utenza che abbia bollette, mostrare l'importo totale delle bollette stesse

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

Domanda 3 (20%)
Considerare le seguenti quattro relazioni su uno stesso schema:

(A)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	true

(B)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	false

(C)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	false
3	3000	1000	2200	false

(D)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	false
2	4000	1000	3000	false
3	3000	1000	2200	false

Considerare i tre vincoli di integrità mostrati nella tabella seguente e dire per ciascuno (con un sì o un no nelle celle corrispondenti), quali relazioni lo soddisfano e quali no:

	(A)	(B)	(C)	(D)
`CHECK ( ( ( Netto = StipLordo - Trattenute) AND (OK = 'true'))` `OR ((Netto <> StipLordo - Trattenute) AND (OK = 'false' ) ) )`				
`CHECK ( ( NOT (OK = 'true') ) OR ( Netto = StipLordo - Trattenute ) )`				
`CHECK ( NOT( Netto = StipLordo - Trattenute ) ) OR ( ( (OK = 'true') )`				

Basi di dati — 26 novembre 2018 — Prova parziale — Compito B

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`
	0	100000		

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`				

`SELECT codice, indirizzo, SUM(importo) AS totale`
`FROM utenze JOIN bollette ON codice = utenza`
`GROUP BY codice, indirizzo`
	U	U		

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice e indirizzo delle utenze per le quali non c'è nessuna bolletta

codice,indirizzo(Utenze)-
codice,utente (utenze join utenza= codice bollette)

Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna persona che sia titolare di utenze, mostrare il numero delle utenze di cui è titolare

```
select titolare, count(numero) as Utenzetotali from Bollette join utenze on utenza=codice
group by (utenza)
```

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

```
select titolare, sum(importo) as totale from Bollette join utenze on utenza=codice
group by (utenza)
```

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

```
create view TotalePagato as
select utenzaP, sum(importo) as Pagato
from utenze join bollette on codice = utenza
where datapagamento!=NULL
group by (utenza)
union
select utenza, 0 as Pagato from utenze
where utenza not in (utenze join bollette on codice = utenza  where datapagamento!=NULL);

create view TotaleDaPagare as
select utenzaD, sum(importo) as DaPagare
from utenze join bollette on codice = utenza
group by (utenza)
union
select utenza, 0 as DaPagare from utenze
where utenza not in (utenze join bollette on codice = utenza );

select utenzaP, Pagato, DaPagare, DaPagare-Pagato as Debito
from TotaleDaPagare join Totaledapagare on UtenzaD=UtenzaP
```

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

```
create view debito as
 select UtenzaP as UtenzaDeb, DaPagare -Pagato as debito
from TotaleDaPagare,TotalePagato
group by(utenzaP);

create view maxDabito as
select utenzaMaxDeb, max(debito) as DebitoMassimo;

select codice, titolare, indirizzo ,from
Utenze join debito on UtenzaDeb = Codice
join maxDebito on DebitoMassimo= debito



OPPURE
select codice,titolare ,indirizzo, MassimoDebito
from Debito
```

Domanda 3 (20%)
Considerare le seguenti quattro relazioni su uno stesso schema:

(A)

Stipendi				
ID	StipLordo	Ritenute	StipNetto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	true

(B)

Stipendi				
ID	StipLordo	Ritenute	StipNetto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	false

(C)

Stipendi				
ID	StipLordo	Ritenute	StipNetto	OK
1	3000	800	2200	true
2	4000	1000	3000	false
3	3000	1000	2200	false

(D)

Stipendi				
ID	StipLordo	Ritenute	StipNetto	OK
1	3000	800	2200	false
2	4000	1000	3000	false
3	3000	1000	2200	false

Considerare i tre vincoli di integrità mostrati nella tabella seguente e dire per ciascuno (con un sì o un no nelle celle corrispondenti), quali relazioni lo soddisfano e quali no:

	(A)	(B)	(C)	(D)
`CHECK ( ( ( StipNetto = StipLordo - Ritenute) AND (OK = 'true'))` `OR ((StipNetto <> StipLordo - Ritenute) AND (OK = 'false' ) ) )`	F	V	F	F
`CHECK ( ( NOT (OK = 'true') ) OR ( StipNetto = StipLordo - Ritenute ) )`	F	V	V	V
`CHECK ( NOT( StipNetto = StipLordo - Ritenute ) ) OR ( ( (OK = 'true') )`	V	V	F	F

Basi di dati — 26 novembre 2018 — Prova parziale — Compito C

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`WHERE importo > 100`				

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`				

`SELECT CF, cognome, nome, count(*) AS numeroUtenze`
`FROM persone JOIN utenze ON CF = titolare`
`GROUP BY CF, cognome, nome`				

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice fiscale, nome e cognome delle persone che non hanno nessuna utenza

Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna utenza che abbia bollette, mostrare l'importo totale delle bollette stesse

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

Domanda 3 (20%)
Considerare le seguenti quattro relazioni su uno stesso schema:

(A)

Stipendi				
ID	Lordo	Imposte	StipNetto	Verifica
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	true

(B)

Stipendi				
ID	Lordo	Imposte	StipNetto	Verifica
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	false

(C)

Stipendi				
ID	Lordo	Imposte	StipNetto	Verifica
1	3000	800	2200	true
2	4000	1000	3000	false
3	3000	1000	2200	false

(D)

Stipendi				
ID	Lordo	Imposte	StipNetto	Verifica
1	3000	800	2200	false
2	4000	1000	3000	false
3	3000	1000	2200	false

Considerare i tre vincoli di integrità mostrati nella tabella seguente e dire per ciascuno (con un sì o un no nelle celle corrispondenti), quali relazioni lo soddisfano e quali no:

	(A)	(B)	(C)	(D)
`CHECK ( ( ( StipNetto = Lordo - Imposte) AND (Verifica = 'true')) OR ((StipNetto <> Lordo - Imposte) AND (Verifica = 'false' ) ) )`				
`CHECK ( ( NOT (Verifica = 'true') ) OR ( StipNetto = Lordo - Imposte ) )`				
`CHECK ( NOT( StipNetto = Lordo - Imposte ) ) OR ( ( (Verifica = 'true') )`				

Basi di dati — 26 novembre 2018 — Prova parziale — Compito D

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`WHERE importo > 100`				

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`				

`SELECT codice, indirizzo, SUM(importo) AS totale`
`FROM utenze JOIN bollette ON codice = utenza`
`GROUP BY codice, indirizzo`				

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice e indirizzo delle utenze per le quali non c'è nessuna bolletta

Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna persona che sia titolare di utenze, mostrare il numero delle utenze di cui è titolare

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

Domanda 3 (20%)
Considerare le seguenti quattro relazioni su uno stesso schema:

(A)

Stipendi				
ID	Lordo	Tasse	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	true

(B)

Stipendi				
ID	Lordo	Tasse	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	false

(C)

Stipendi				
ID	Lordo	Tasse	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	false
3	3000	1000	2200	false

(D)

Stipendi				
ID	Lordo	Tasse	Netto	OK
1	3000	800	2200	false
2	4000	1000	3000	false
3	3000	1000	2200	false

Considerare i tre vincoli di integrità mostrati nella tabella seguente e dire per ciascuno (con un sì o un no nelle celle corrispondenti), quali relazioni lo soddisfano e quali no:

	(A)	(B)	(C)	(D)
`CHECK ( ( ( Netto = Lordo - Tasse) AND (OK = 'true'))` `OR ((Netto <> Lordo - Tasse) AND (OK = 'false' ) ) )`				
`CHECK ( ( NOT (OK = 'true') ) OR ( Netto = Lordo - Tasse ) )`				
`CHECK ( NOT( Netto = Lordo - Tasse ) ) OR ( ( (OK = 'true') )`				

Basi di dati — 26 novembre 2018 — Prova parziale — Compito A

Possibili soluzioni

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore
`SELECT *` `FROM utenze JOIN persone ON titolare=CF`	U	20.000	U	20.000

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`
	0	0	B	100.000

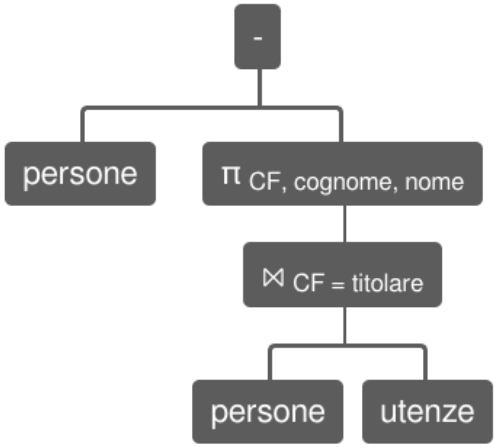
`SELECT CF, cognome, nome, count(*) AS numeroUtenze`
`FROM persone JOIN utenze ON CF = titolare`
`GROUP BY CF, cognome, nome`
	1	1	P	10.000

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice fiscale, nome e cognome delle persone che non hanno nessuna utenza



Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna utenza che abbia bollette, mostrare l'importo totale delle bollette stesse

```
SELECT utenza, sum(importo) AS importototale
FROM bollette
GROUP BY utenza
```

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

```
SELECT titolare as persona, sum(importo) AS importototale
FROM utenze join bollette on codice=utenza
GROUP BY titolare
```

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

```
CREATE OR REPLACE VIEW totaledapagare
AS SELECT codice as utenza, sum(importo) as importototale
   FROM utenze join bollette on codice=utenza
   GROUP BY codice
   UNION
   SELECT codice as utenza, 0 as importototale
   FROM utenze
   WHERE codice not in (SELECT utenza FROM bollette);


CREATE OR REPLACE VIEW totalepagato
AS SELECT codice as utenza, sum(importo) as importopagato
   FROM utenze join bollette on codice=utenza
   WHERE datapagamento IS NOT NULL
   GROUP BY codice
   UNION
   SELECT codice as utenza, 0 as importototale
   FROM utenze
   WHERE codice not in (SELECT utenza FROM bollette WHERE datapagamento IS NOT NULL)

SELECT td.utenza, importototale, importopagato, importototale-importopagato AS debito
FROM totaledapagare td join totalepagato tp on td.utenza=tp.utenza
```

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

```
CREATE OR REPLACE VIEW debiti
AS SELECT td.utenza, importototale, importopagato, importototale-importopagato AS debito
   FROM totaledapagare td join totalepagato tp on td.utenza=tp.utenza;

SELECT utenze.*, debito
FROM utenze join debiti on codice=utenza
WHERE debito = (SELECT max(debito) FROM debiti)
```

Domanda 3 (20%)
Considerare le seguenti quattro relazioni su uno stesso schema:

(A)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	true

(B)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	true
3	3000	1000	2200	false

(C)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	true
2	4000	1000	3000	false
3	3000	1000	2200	false

(D)

Stipendi				
ID	StipLordo	Trattenute	Netto	OK
1	3000	800	2200	false
2	4000	1000	3000	false
3	3000	1000	2200	false

Considerare i tre vincoli di integrità mostrati nella tabella seguente e dire per ciascuno (con un sì o un no nelle celle corrispondenti), quali relazioni lo soddisfano e quali no:

	(A)	(B)	(C)	(D)
`CHECK ( ( ( Netto = StipLordo - Trattenute) AND (OK = 'true'))` `OR ((Netto <> StipLordo - Trattenute) AND (OK = 'false' ) ) )`	**NO**	**SÌ**	**NO**	**NO**
`CHECK ( ( NOT (OK = 'true') ) OR ( Netto = StipLordo - Trattenute ) )`	**NO**	**SÌ**	**SÌ**	**SÌ**
`CHECK ( NOT( Netto = StipLordo - Trattenute ) ) OR ( ( (OK = 'true') ) )`	**SÌ**	**SÌ**	**NO**	**NO**

Basi di dati — 26 novembre 2018 — Prova parziale — Compito B

Possibili soluzioni

Tempo a disposizione: un'ora.

Cognome: _____ **Nome:** _____ **Matricola:** _____

Domanda 1 (20%) Considerare lo schema con le seguenti relazioni

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

con le seguenti cardinalità

- `persone`: cardinalità $P = 10.000$
- `utenze`: cardinalità $U = 20.000$
- `bollette`: cardinalità $B = 100.000$

Indicare la cardinalità del risultato di ciascuna delle seguenti interrogazioni SQL, specificando l'intervallo nel quale essa può variare; indicare simboli e numeri.

	Min simboli	Min valore	Max simboli	Max valore

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`
	0	0	B	100.000

`SELECT *`
`FROM bollette JOIN utenze ON utenza=codice`
`JOIN persone ON titolare=CF`
`WHERE importo > 100`
	0	0	B	100.000

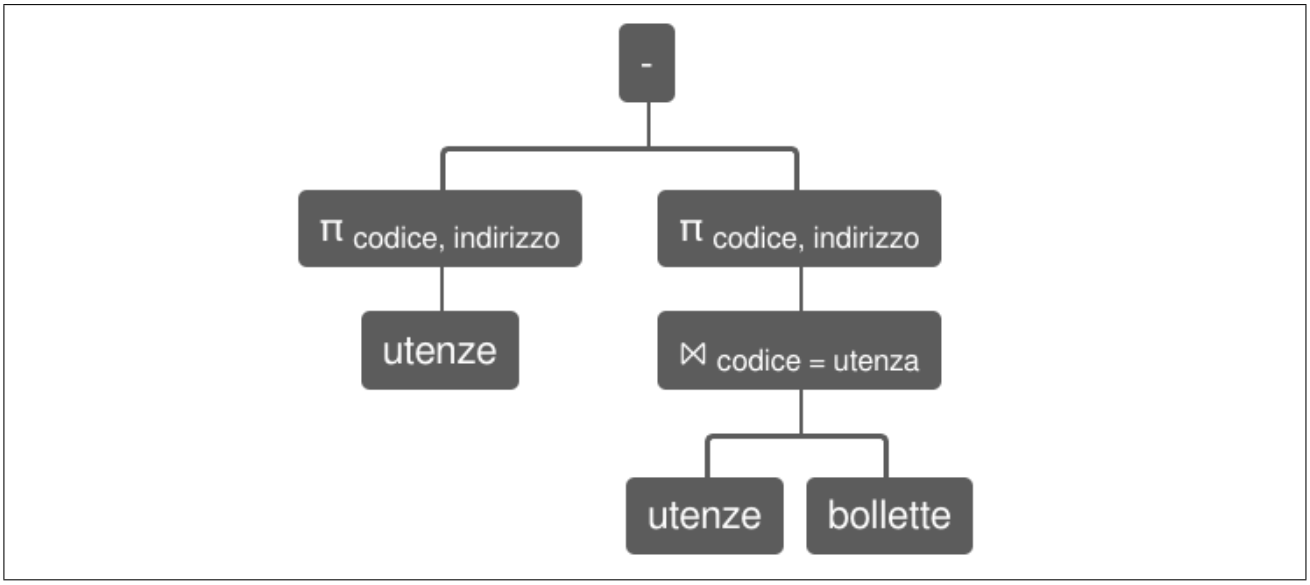
`SELECT codice, indirizzo, SUM(importo) AS totale`
`FROM utenze JOIN bollette ON codice = utenza`
`GROUP BY codice, indirizzo`
	1	1	U	20.000

Domanda 2 (60%) Considerare nuovamente lo schema utilizzato nella domanda precedente

```
CREATE TABLE persone (  CF  text NOT NULL PRIMARY KEY,
                        cognome text,
                        nome text );
CREATE TABLE utenze (   codice integer NOT NULL PRIMARY KEY,
                        titolare text NOT NULL REFERENCES persone,
                        indirizzo text NOT NULL);
CREATE TABLE bollette ( numero integer NOT NULL PRIMARY KEY,
                        utenza integer NOT NULL REFERENCES utenze ,
                        data date NOT NULL,
                        importo integer NOT NULL,
                        datapagamento date );
```

Formulare la seguente interrogazione in **algebra relazionale**

1. Mostrare codice e indirizzo delle utenze per le quali non c'è nessuna bolletta



Formulare le seguenti interrogazioni in **SQL**

2. Per ciascuna persona che sia titolare di utenze, mostrare il numero delle utenze di cui è titolare

```
SELECT titolare, count(*) AS numeroutenze
FROM utenze
GROUP BY titolare
```

3. Per ciascuna persona che sia titolare di utenze che hanno bollette, mostrare l'importo totale delle bollette di tali utenze.

```
SELECT titolare as persona, sum(importo) AS importototale
FROM utenze join bollette on codice=utenza
GROUP BY titolare
```

4. Per ciascuna utenza, mostrare il totale delle bollette da pagare e il totale delle bollette pagate e l'eventuale "debito", cioè la differenza fra il totale da pagare e il totale pagato. Considerare anche le utenze senza bollette (con totale da pagare pari zero) e le utenze senza bollette pagate (con totale pagato pari a zero)

```
CREATE OR REPLACE VIEW totaledapagare
AS SELECT codice as utenza, sum(importo) as importototale
   FROM utenze join bollette on codice=utenza
   GROUP BY codice
   UNION
   SELECT codice as utenza, 0 as importototale
   FROM utenze
   WHERE codice not in (SELECT utenza FROM bollette);


CREATE OR REPLACE VIEW totalepagato
AS SELECT codice as utenza, sum(importo) as importopagato
   FROM utenze join bollette on codice=utenza
   WHERE datapagamento IS NOT NULL
   GROUP BY codice
   UNION
   SELECT codice as utenza, 0 as importototale
   FROM utenze
   WHERE codice not in (SELECT utenza FROM bollette WHERE datapagamento IS NOT NULL)

SELECT td.utenza, importototale, importopagato, importototale-importopagato AS debito
FROM totaledapagare td join totalepagato tp on td.utenza=tp.utenza
```

5. Mostrare le informazioni sull'utenza per la quale è massimo il "debito" di cui all'interrogazione precedente

```
CREATE OR REPLACE VIEW debiti
AS SELECT td.utenza, importototale, importopagato, importototale-importopagato AS debito
   FROM totaledapagare td join totalepagato tp on td.utenza=tp.utenza;

SELECT utenze.*, debito
FROM utenze join debiti on codice=utenza
WHERE debito = (SELECT max(debito) FROM debiti)
```