Soluzioni: nota metodologica

Nota: molti esercizi ammettono più di una soluzione possibile. Se la soluzione che trovate nelle pagine seguenti è diversa da quella che avete individuate voi, non significa necessariamente che la vostra soluzione non sia corretta. Non correte nello svolgere gli esercizi, provate a svolgere ogni esercizio senza guardare la soluzione, al termine confrontate criticamente la vostra soluzione con le soluzioni proposte nelle pagine seguenti (analizzando ogni minimo dettaglio). Questo "confronto" è fondamentale per l'apprendimento.
Se trovate imprecisioni, inesattezze, o se le soluzioni vi sembrano poco chiare, siete pregati di avvisare mirko.cesarini@unimib.it

Buon lavoro.

1) Soluzioni Montagna del delitto

1. select * from dati order by cognome;
2.
    a. select * from dati where partenza='10:55:00';
    b. select * from dati where capelli='biondi';
    c. select * from dati where misurascarpe=44;
    d. select * from dati where partenza='10:55:00' and capelli='biondi';
    e. select * from dati where partenza='10:55:00' and capelli='biondi' and misurascarpe=44;
3.
    a. select * from dati where partenza='08:10:00' and capelli='neri';
    b. select * from dati where ritorno='19:05:00' and capelli='neri';
    c. select * from dati where cognome='Agostini' and capelli='biondi';
4.
    a. select * from dati where partenza<'15:30:00' and capelli='castani';
    b. select * from dati where partenza<'15:30:00' and capelli='castani' and data='2001-07-07';
    c. select * from dati where partenza<'15:30:00' and capelli='castani' and data='2001-07-07' and misurascarpe=40;
    d. …
5.
    a. select * from dati where partenza='09:30:00' and data='2001-07-07' and (capelli='rossi' or capelli='castani') and misurascarpe >= 36 and ritorno='19:30:00';
    b. …

2) Soluzioni DB librerie

1. …
2. …
3. Visualizzate il contenuto della tabella libri
   SELECT * FROM libri;
4. Visualizzate solo il contenuto degli attributi titolo, tipo, prezzo della tabella libri
   SELECT titolo, tipo, prezzo FROM libri;
5. Visualizzate titolo e prezzo dei libri che costano meno di 10 euro
   SELECT titolo, prezzo FROM libri WHERE prezzo <10;
6. Visualizzate le informazioni dei libri di informatica (i libri con tipo uguale a 'CS'), CS sta per computer science
   SELECT * FROM libri WHERE tipo='CS';
7. Visualizzate contemporaneamente le informazioni sia dei libri di informatica sia dei libri di fiction (tipi: 'CS' e 'FIC')
   SELECT * FROM libri WHERE tipo='CS' or tipo='FIC';
8. Visualizzate per ogni libro, il titolo del libro e l'editore (nota: dovete effettuare il join tra le tabelle libri e le tabelle editori)
   SELECT titolo, nome FROM editori, libri WHERE editori.codice=libri.codice_editore;
   Nota: provate a vedere cosa succede se si omette la clausola WHERE (e la condizione)
9. Visualizzate per ogni libro il titolo, il prezzo e la descrizione del formato (vedi tabella formato)
   SELECT titolo, prezzo, descrizione FROM libri, formato WHERE formato.codice = libri.cod_format;
10. Stampate titolo del libro e cognome dell'autore che lo ha scritto (nota: dovete effettuare il join di 3 tabelle)
    SELECT titolo, cognome FROM libri as l, hascritto as h, autori as a WHERE a.codice_autore=h.codice_autore and h.codice_libro = l.codice;
11. Come il punto precedente, ma stampate solamente i libri scritti da Kafka
    SELECT titolo FROM libri as l, hascritto as h, autori as a WHERE a.codice_autore=h.codice_autore and h.codice_libro = l.codice and cognome = 'kafka';
12. Come il punto precedente, ma stampate solamente i libri scritti o da Kafka o da Agata Christie.
    SELECT titolo, cognome FROM libri as l, hascritto as h, autori as a WHERE a.codice_autore=h.codice_autore and h.codice_libro = l.codice and (a.cognome='kafka' or a.cognome='Christie');
13. Stampate il nome e la sede delle librerie dove è in vedita il libro 'dBASE Programming'
    SELECT nome, sede FROM negozi, scorte, libri WHERE negozi.codice_negozio=scorte.codice_negozio and scorte.codice_libro = libri.codice and libri.titolo='dBASE Programming';
14. …

3) Soluzioni Foodstore

Nota, per ogni punto possono essere proposte diverse soluzioni, che fanno uso di diversi costrutti del linguaggio SQL

1. select * from customer where fname='Charles'; /* dal numero di tuple restituite è possibile risalire al numero di persone */
   select count(*) from customer where fname='Charles'; /*altra versione che fa uso delle funzioni di aggregazione */ Le query qua sopra possono essere ripetute cambiando nome
   select fname, count(*) from customer
   where fname='Charles' or fname='Patricia' or fname='Sharon'
   group by fname; /*versione che fa uso delle funzioni di aggregazione e mostra tutti i risultati assieme */
2. select * from customer as c, member_card as mc where c.card_type_id = mc.card_type_id and mc.description='Gold'; /*questa deve essere ripetuta per i diversi valori delle tessere*/
   select mc.description, count(*) from customer as c, member_card as mc where c.card_type_id = mc.card_type_id group by mc.card_type_id;
3. select * from bill as b, customer as c where b.customer_id = c.customer_id and c.fname='Clyde' and c.lname='Dixon';
   select c.customer_id, c.fname, c.lname, count(*) from bill as b, customer as c where b.customer_id = c.customer_id and ((c.fname='Bonnie' and c.lname='Emerson') or (c.fname='Clyde' and c.lname='Dixon')) group by c.customer_id; /* versione con le funzioni di aggregazione */
4. select * from bill as b, customer as c where b.customer_id = c.customer_id and b.total>307; /* Per estrarre gli scontrini superiori a 307 euro*/
   select * from bill as b, customer as c where b.customer_id = c.customer_id and b.total>307 and fname='Bradley' and lname='Doman';
   select c.customer_id, c.lname, c.fname, count(*) from bill as b, customer as c where b.customer_id = c.customer_id and b.total>307 group by c.customer_id; /*per vedere come sono distribuiti tra le persone gli scontrini maggiori di 307 dollari */
5. select c.lname, c.fname from bill as b, customer as c, item_in_bill as i, product as p where c.customer_id=b.customer_id and b.bill_id=i.bill_id and i.product_id=p.product_id and p.product_name='Great Muffins' order by c.lname; /* L'order by non era richiesto, si può aggiungere la clausola *group by c.customer_id* */
6. select c.lname, c.fname, p.product_name, b.date from bill as b, customer as c, item_in_bill as i, product as p, supplier as s where c.customer_id=b.customer_id and b.bill_id=i.bill_id and i.product_id=p.product_id and p.supplier_id=s.supplier_id and s.name='Bravo';
7. select p.product_id, p.product_name, sum(i.quantity) from product as p, item_in_bill as i where p.product_id=i.product_id and p.product_id <20 group by p.product_id;
8. /* query che mostra le quantità per ogni product_class_id */
   select pc.product_class_id, pc.product_subcategory, sum(i.quantity) from item_in_bill as i, product as p, product_class as pc where i.product_id = p.product_id and p.product_class_id=pc.product_class_id group by pc.product_class_id;
   /* query che mostra il ricavo totale per ogni product_class_id */
   select pc.product_class_id, pc.product_subcategory, sum(i.quantity*i.price) from item_in_bill as i, product as p, product_class as pc where i.product_id = p.product_id and p.product_class_id=pc.product_class_id group by pc.product_class_id;
9. …
10. …

4) Soluzioni Foodmart parte 1

1. Visualizzate product_id, nome prodotto e la data delle vendite contenute nella tabella sales_fact_1998. La data deve essere visualizzata in un formato nel quale si riesca a distinguere giorno, mese e anno; Il nome del prodotto e la data vanno recuperate da tabelle diversa da sales_fact_1998. I risultati devono essere ordinati per data e per nome prodotto.
   select p.product_id, p.product_name, t.the_date from sales_fact_1998 as s, time_by_day as t, product as p where s.time_id=t.time_id and s.product_id=p.product_id order by t.the_date, p.product_name;
2. Visualizzate il product_id, il time_id e il guadagno conseguito effettuando le vendite descritte nella tabella sales_fact_1998 …
   *select time_id, product_id, (store_sales-store_cost)*unit_sales as guadagno from sales_fact_1998 order by time_id, product_id;*
3. Stampate l'elenco dei prodotti venduti nel corso dell'anno. Nella tabella risultato deve apparire il nome per esteso del prodotto (se volete anche l'ID), inoltre ogni prodotto anche se venduto più volte deve apparire una sola volta nel risultato. (Consiglio: dovete usare la GROUP BY).
   select product_name, p.product_id from sales_fact_1998 as s, product as p where s.product_id=p.product_id group by p.product_id order by p.product_name;
4. Per ogni prodotto, calcolate il totale dei costi di acquisto (ogni prodotto anche se venduto più volte, deve apparire una sola volta nella tabella risultato) …
   *select product_name, p.product_id, sum(store_cost*unit_sales) from sales_fact_1998 as s, product as p where s.product_id=p.product_id group by p.product_id;*
5. Per ogni prodotto calcolate il totale dei guadagni (ogni prodotto anche se venduto più volte, deve apparire una sola volta).
   *select product_name, p.product_id, sum((store_sales-store_cost)*unit_sales) from sales_fact_1998 as s, product as p where s.product_id=p.product_id group by p.product_id;*
6. Ordinate l'elenco in ordine decrescente in base al guadagno
   *select product_name, p.product_id, sum((store_sales-store_cost)*unit_sales) as gain from sales_fact_1998 as s, product as p where s.product_id=p.product_id group by p.product_id order by gain DESC;*
7. Individuate i supermercati che hanno venduto più quantità di prodotti (create una lista in ordine decrescente).
   select store_name, s.store_id, sum(unit_sales) as qt from sales_fact_1998 as s, store as t where s.store_id=t.store_id group by s.store_id order by qt desc;
8. Individuate i supermercati che hanno venduto la minor quantità di prodotti (create una lista in ordine crescente).
   select store_name, s.store_id, sum(unit_sales) as qt from sales_fact_1998 as s, store as t where s.store_id=t.store_id group by s.store_id order by qt;
9. Individuate le tipologie di prodotto che hanno fatto guadagnare di più (per le tipologie fate riferimento all'attributo product_category della tabella product_class; create una lista in ordine decrescente).
   *select product_category, sum((store_sales-store_cost)*unit_sales) as gain from sales_fact_1998 as s, product as p, product_class as pc where s.product_id=p.product_id and p.product_class_id=pc.product_class_id group by product_category order by gain DESC;*
10. Individuate i prodotti più venduti (in termini di quantità), nella regione del 'Central West' (attributo sales_region della tabella region).
    select p.product_id, product_name, sum(unit_sales) as qt from region as r, store as st, sales_fact_1998 as s, product as p where r.region_id=st.region_id and s.store_id=st.store_id and s.product_id=p.product_id and sales_region='Central West' group by p.product_id order by qt desc;

11. Individuate i prodotti più venduti (in termini di quantità), negli USA (attributo sales_country della tabella region).
    select product_name, p.product_id, sum(unit_sales) as qt from region r, store st, sales_fact_1998 as s, product as p where r.region_id=st.region_id and s.store_id=st.store_id and s.product_id=p.product_id and sales_country='USA' group by p.product_id order by qt desc;
12. Calcolate il guadagno totale di tutti i prodotti venduti nei mesi di aprile e maggio 1998.
    *select sum((store_sales-store_cost)*unit_sales) from sales_fact_1998 as s, time_by_day as t where s.time_id=t.time_id and (the_month='April' or the_month='May');* ***oppure*** *select sum((store_sales-store_cost)*unit_sales) from sales_fact_1998 where time_id >=822 and time_id<=882;* Nota: per i valori di time_id, controllate il contenuto della tabella time_by_day
13. Individuate i clienti che hanno speso di meno in tutto il 1998.
    *select c.customer_id, lname, fname, sum(store_sales*unit_sales) as expense from sales_fact_1998 as s, customer as c where s.customer_id = c.customer_id and time_id >=732 and time_id<=1096 group by c.customer_id order by expense;*
14. Individuate il cliente che ha speso di più in tutto il 1998.
    *select c.customer_id, lname, fname, sum(store_sales*unit_sales) as expense from sales_fact_1998 as s, customer as c where s.customer_id = c.customer_id and time_id >=732 and time_id<=1096 group by c.customer_id order by expense desc;*
15. Individuate la città (attributo sales_city della tabella region) nella quale sono stati incassati più soldi nel 1998 (incasso=ricavo).
    *select sales_city, sum(store_sales*unit_sales) as col from region r, store st, sales_fact_1998 as s where r.region_id=st.region_id and s.store_id=st.store_id group by sales_city order by col desc;*
16. Individuate le 5 sales_region in cui la catena di supermercati ha guadagnato di più.
    *select sales_region, sum((store_sales-store_cost)*unit_sales) as gain from region r, store st, sales_fact_1998 as s where r.region_id=st.region_id and s.store_id=st.store_id group by sales_region order by gain desc;*

5) Soluzioni Foodmart parte 2

1. SELECT * FROM sales_fact_1998 as s where s.store_cost>=s.store_sales;
   Non ci sono prodotti sottocosto
2. select promotion_name, sum((s.store_sales-s.store_cost)*unit_sales) as guadagno from sales_fact_1998 as s, promotion as pr where s.promotion_id=pr.promotion_id and promotion_name<>'No Promotion' group by s.promotion_id order by guadagno desc;
3. select s.product_id, p.product_name,
   sum((store_sales - store_cost)*unit_sales) / sum(store_sales*unit_sales) as margine
   from sales_fact_1998 as s, product as p
   where s.product_id = p.product_id
   group by s.product_id
   order by margine desc;
4. select pc.product_category,
   sum((store_sales - store_cost)*unit_sales) / sum(store_sales*unit_sales) as margine
   from sales_fact_1998 as sf, product as pr, product_class as pc
   where sf.product_id = pr.product_id and pr.product_class_id = pc.product_class_id
   group by pc.product_category
   order by margine desc;
5. select pc.product_department,
   sum((store_sales - store_cost)*unit_sales) / sum(store_sales*unit_sales) as margine
   from sales_fact_1998 as sf, product as pr, product_class as pc
   where sf.product_id = pr.product_id and pr.product_class_id = pc.product_class_id
   group by pc.product_department
   order by margine desc;
6. select pc.product_department,
   sum((store_sales - store_cost)*unit_sales) / sum(store_sales*unit_sales) as margine
   from sales_fact_1998 as sf, product as pr, product_class as pc
   where sf.product_id = pr.product_id and pr.product_class_id = pc.product_class_id
   group by pc.product_department
   having margine > 0.6
   order by margine desc;

7. (Viene riportato il risultato del punto c che comprende anche i punti a e b)
```
select me.store_id, me.vendita, cp.vendita, me.vendita/cp.vendita, me.store_name from
(
select st.store_id, sum(store_sales*unit_sales) as vendita, st.store_name
from sales_fact_1998 as sf, product as pr, store as st, product_class as pc
where sf.product_id = pr.product_id and sf.store_id = st.store_id and
pr.product_class_id = pc.product_class_id and
pc.product_department = 'Meat'
group by  st.store_id
) as me,
(
select sf.store_id, sum(store_sales*unit_sales) as vendita
from sales_fact_1998 as sf, product as pr, product_class as pc
where sf.product_id = pr.product_id and
pr.product_class_id = pc.product_class_id and
pc.product_department = 'Canned products'
group by sf.store_id
) as cp
where me.store_id = cp.store_id;
```
8.
```
select st.store_country, st.store_state, sum(s.store_sales*s.unit_sales) as ricavo from
sales_fact_1998 as s, store as st where s.store_id=st.store_id group by st.store_country,
store_state order by ricavo desc;
```
9.
```
SELECT p.product_id, product_name, sum((s.store_sales-s.store_cost)*s.unit_sales) as gain
FROM sales_fact_1998 as s, product as p where p.product_id=s.product_id group by
p.product_id order by gain desc;
```
10.
```
select st.store_id, sum((s.store_sales-s.store_cost)*s.unit_sales)/sum(s.unit_sales) as ratio from
sales_fact_1998 as s, store as st where s.store_id=st.store_id and st.store_type='Supermarket'
group by store_id order by ratio desc;
```
11.
```
select cu.member_card,
sum((store_sales - store_cost)*unit_sales) as utile
from sales_fact_1998 as sf, customer as cu
where sf.customer_id = cu.customer_id
group by cu.member_card
order by utile desc;
```
12. Vengono visualizzate le soluzioni del solo punto c), che reassume anche le soluzioni dei punti a) e b)
```
select sin.store_id, sin.store_type, sin.product_department, sin.margine,
gen.margine as marg_settore
from
(
select st.store_type, pc.product_department,
sum((store_sales-store_cost)*unit_sales)/sum(store_sales*unit_sales) as margine
from sales_fact_1998 as sf, store as st, product as pr, product_class as pc
where sf.store_id = st.store_id and sf.product_id = pr.product_id and
pr.product_class_id = pc.product_class_id
group by st.store_type, pc.product_department
) as gen,
(
select st.store_id, st.store_type, pc.product_department,
sum((store_sales-store_cost)*unit_sales)/sum(store_sales*unit_sales) as margine
from sales_fact_1998 as sf, store as st, product as pr, product_class as pc
where sf.store_id = st.store_id and sf.product_id = pr.product_id and
pr.product_class_id = pc.product_class_id
```

```
group by st.store_id, pc.product_department
) as sin
where sin.store_type = gen.store_type and
sin.product_department = gen.product_department
and sin.margine < (gen.margine * 0.96);
```

13. Vengono visualizzate le soluzioni del solo punto c), che reassume anche le soluzioni dei punti a) e b)

```
select tot.product_id, q1.qt as qt_q1, tot.qt as qt_tot, q1.qt / tot.qt as ratio from
(
 select p.product_id, sum(unit_sales) as qt from sales_fact_1998 as sf, time_by_day as tbd,
product as p
 where sf.time_id=tbd.time_id and sf.product_id=p.product_id and tbd.quarter='Q1'
 group by p.product_id
) as q1,
(
 select p.product_id, sum(unit_sales) as qt from sales_fact_1998 as sf, time_by_day as tbd,
product as p
 where sf.time_id=tbd.time_id and sf.product_id=p.product_id
 group by p.product_id
) as tot
where q1.product_id=tot.product_id
order by tot.product_id;
```

14. Vengono visualizzate le soluzioni del solo punto c), che reassume anche le soluzioni dei punti a) e b)

```
select sf.customer_id, count(*) as n_transazioni_maggiori, ag2.totn as n_transazioni,
count(*) / ag2.totn as indicatore
from
sales_fact_1998 as sf,
(
 select customer_id, avg(unit_sales) as avgqt
 from sales_fact_1998
 group by customer_id
) as ag1,
(
 select customer_id, count(*) as totn
 from sales_fact_1998
 group by customer_id
) as ag2
where sf.customer_id=ag1.customer_id and sf.customer_id=ag2.customer_id
and sf.unit_sales > ag1.avgqt
group by sf.customer_id
order by sf.customer_id;
```