Corso di Basi di Dati 2014/2015

PROGETTO DI UNA BASE DI DATI PER LA GESTIONE DI UN NEGOZIO DI ELETTRODOMESTICI ED ELETTRONICA

MANTOVANI ANDREA

1070258

MASSIGNAN FABIO

1070541

Indice generale

1. Abstract

Si vuole realizzare una base di dati per la gestione di un piccolo negozio di elettrodomestici ed elettronica, che dia la possibilità di gestire disponibilità, acquisti e vendite dei prodotti, fidelizzazione con i clienti, rapporti con i fornitori, gestione fiscale e statistiche sulle vendite.
Le operazioni tipiche sono: la registrazione di un nuovo cliente, la vendita di prodotti, la catalogazione dei prodotti forniti dai diversi fornitori, l'applicazione degli sconti ai clienti che possiedono una tessera, e l'eliminazione della tessera qual ora il cliente non volesse più aderire alla raccolta punti.

2. Descrizione dei Requisiti

La classe principale di questa base di dati è il **Prodotto**, attorno al quale gira tutto il gestionale. Per ogni Prodotto, si vuole memorizzare il modello, che è identificativo, Prezzo di Vendita, Ricarico (guadagno in percentuale sul prezzo di vendita), giacenza (quantità presente all'interno del negozio), e IVA.
Ogni prodotto appartiene ad una **Sotto_Categoria**, che è identificata da un nome, ed è inglobata in una **Categoria** merciologica più ampia. Grazie a questa suddivisione, è possibile analizzare più facilmente come si comportano i clienti all'interno del negozio e capire quale tipologia di prodotto preferiscono.
Un Prodotto può essere fornito da diversi **Fornitori**: di ognuno interessano il codice interno del negozio, che li identifica univocamente, la Ragione Sociale, e i dati per il recapito.
Ogni spedizione di prodotti effettuata dal Fornitore è accompagnata da un **DDT** (Documento Di Trasporto) all'interno del quale vi è la lista dei prodotti forniti da quel DDT in una determinata Data. Quindi i prodotti forniti in un DDT sono identificati univocamente dal numero del Documento di Trasporto e dal Modello, in più vengono riferiti dal fornitore anche la marca, il ricarico ed il prezzo di acquisto applicato in quella data. Un Prodotto, infine, può essere identificato anche da un Univoco Prodotto del fornitore ovvero un codice interno utilizzato dal fornitore per identificare univocamente il prodotto nel proprio magazzino. Ne consegue che per un prodotto possono esserci più Univoco Prodotto associati ognuno al rispettivo fornitore. Questa e' stata una richiesta esplicita da parte del committente del database, in quanto, in fase di riordino del prodotto rende l'operazione più semplice.
Per quanto riguarda i **clienti** che frequentano il negozio, invece, si vogliono memorizzare i dati soltanto dei titolari di una tessera fedeltà, oppure, di quelli che richiedono la fattura e non lo scontrino fiscale. I dati da registrare per identificare i clienti sono Codice Fiscale, Nome, Cognome, Data di nascita, Numero di Telefono, Residenza e Partita Iva, se posseduta, necessaria per emettere la fattura.
Si vuole mantenere traccia di tutti gli **acquisti** effettuati, sia dei clienti non abituali,che degli abituali, e, per i clienti che posseggono una Tessera Fedeltà, si richiede anche lo storico del **movimento** dei punti (accumulati o scalati).
La **Tessera**, poi, e' identificata dal Numero Tessera, e permette di accumulare dei punti che, raggiunta una determinata soglia, potranno essere convertiti in uno sconto.
Infine per ogni **acquisto**: si vuole memorizzare l'Id Acquisto che e' identificativo, Data di acquisto, e l'eventuale sconto applicato. Contestualmente all'acquisto viene emessa la ricevuta, la quale può essere di due tipi:

- Scontrino: Il numero dello scontrino si resetta giornalmente
- Fattura: il numero della fattura si resetta annualmente

Dall'interfaccia web si vuole poter registrare un nuovo DDT o un acquisto con le relative selezioni dei prodotti acquistati o venduti, ricercare quali prodotti sono presenti all'interno del negozio, e visionare alcune statistiche sui dati che si stanno modellando. È richiesta anche un

ulteriore sezione che permetta ai clienti che posseggono una tessera di vedere i propri acquisti e lo stato dei movimenti dei loro punti, utilizzando per l'accesso il proprio codice fiscale ed il numero della tessera posseduta. Inoltre, se sono state inserite le credenziali di accesso dell'amministratore, è possibile visionare per ogni utente registrato gli acquisti effettuati ed il movimento dei punti, inserire i dati di un nuovo cliente o modificare quelli di uno già esistente, ad eccezione dei campi tessera se posseduta, e codice fiscale.

3. Progettazione Concettuale

3.1 Entità e Attributi

- Prodotto: elenco dei Prodotti contenuti all'interno del negozio
  - Modello: VARCHAR(20)
  - Marca: VARCHAR(20)
  - Prezzo Vendita: DECIMAL(8,2)
  - Ricarico: INTEGER
  - Giacenza: INTEGER(20)
  - IVA: INTEGER
- Sotto_Categoria: sotto categoria a cui appartengono i prodotti
  - Nome: VARCHAR(20)
- Categoria: insieme di Sotto Categorie
  - Nome: VARCHAR(20)
- Fornitore: fornisce i Prodotti
  - Codice: INTEGER
  - Indirizzo: VARCHAR(20)
  - Telefono: INTEGER
  - Partita: IVA INTEGER
  - Ragione Sociale: VARCHAR(20)
- DDT: identificativo della consegna di un insieme di prodotti da parte del fornitore
  - Data: DATETIME
  - Id_Trasporto: INTEGER(20)
- Cliente_Registrato: cliente abituale del negozio
  - Codice Fiscale: VARCHAR(20)
  - Nome: VARCHAR(20)
  - Cognome: VARCHAR(20)
  - Data di Nascita: DATETIME
  - Partita IVA: INTEGER
  - Telefono: INTEGER
  - Residenza
    - Citta VARCHAR(20)
    - Numero INTEGER
    - Via VARCHAR(20)

    E' presente anche la seguente generalizzazione parziale in quanto non tutti i clienti registrati sono tesserati
    - Tesserato: cliente abituale che intende accumulare punti per poi ricevere uno sconto
      - Numero Tessera: INTEGER
      - Punti: INTEGER

- Movimento: definisce se in un acquisto c'e' stato un accumulo o una sottrazione di punti
  - Punti: INTEGER

- Sconto: rapporto tra i punti e lo sconto effettivo da applicare all'acquisto
  - Totale: INTEGER

- Acquisto: definisce l'insieme di prodotti acquistati da un cliente
  - Id_Acquisto: INTEGER
  - Data: DATETIME

- Ricevuta: ricevuta emessa contestualmente all'acquisto
  - Data: DATETIME

    E' presente la seguente generalizzazione totale
    - Scontrino
      - NumeroScontrino: INTEGER
    - Fattura
      - NumeroFattura: INTEGER(20)

3.2 Relazioni

- *Cliente_Registrato – Acquisto*: Effettuato associa all'acquisto il cliente che lo ha effettuato
  - Ogni cliente effettua da 1 a N *Acquisto*, un *Acquisto* e' effettuato da un solo *Cliente_Registrato* oppure da un cliente non abituale
  - Molteplicità: N:1

- *Tesserato – Movimento*: Interessato se il cliente ha la tessera associa il movimento punti alla tessera
  - Ogni *Tesserato* effettua da 1 a N *Movimenti*, un *Movimento* e' effettuato da uno e un solo *Tesserato*
  - Molteplicità: N:1

- *Acquisto – Movimento*: Genera un A*cquisto* se effettuato da un *Tesserato*, provoca un *Movimento* punti
  - Un *Acquisto* genera da 0 a 1 *Movimento*, un *Movimento* corrisponde esattamente ad un *Acquisto*
  - Molteplicità: 1:1

- *Movimento – Sconto*: Comporta se *Movimento* punti e' negativo vengono confrontati i Punti per capire quale sconto applicare
  - Un *Movimento Comporta* al più uno *Sconto*, un *Sconto* corrisponde esattamente ad un *Movimento*
  - Molteplicità: 1:1

- *Sconto – Acquisto*:Applicato viene A*pplicato* all'A*cquisto* lo S*conto* solo se M*ovimento* e' <0
  - Uno *Sconto* può essere A*pplicato* da 0 a N *Acquisti*, in un *Acquisto* puo' essere A*pplicato* da 0 a un solo *Sconto*
  - Molteplicità:N:1

- *Acquisto – Ricevuta*: Emissione ad ogni A*cquisto* viene emessa una R*icevuta*
  - Ad ogni *Acquisto* corrisponde una sola *Ricevuta*, ogni R*icevuta* corrisponde ad un solo *Acquisto*
  - Molteplicità: 1:1
- *Acquisto – Prodotto*: Riguarda contiene l'insieme di prodotti riferiti ad un *acquisto*
  - Un *Acquisto* riguarda almeno un *prodotto*, un *Prodotto* può essere contenuto in 0 o N *Acquisti*
  - Molteplicità: N:N
  - Attributi:
    - Prezzo DECIMAL(8,2)
    - Quantita INTEGER

- *Prodotto – DDT*: Fornito contiene l'insieme di prodotti forniti da un *DDT*
  - Un *Prodotto* può essere fornito da 1 a N *DDT*, ogni *DDT* contiene da 1 a N *Prodotti*
  - Molteplicità: N:N
  - Attributi:
    - Prodotto VARCHAR(20)
    - Marca VARCHAR(20)
    - Quantita INTEGER
    - Prezzo_Acquisto DECIMAL(8,2)
    - Ricarico INTEGER

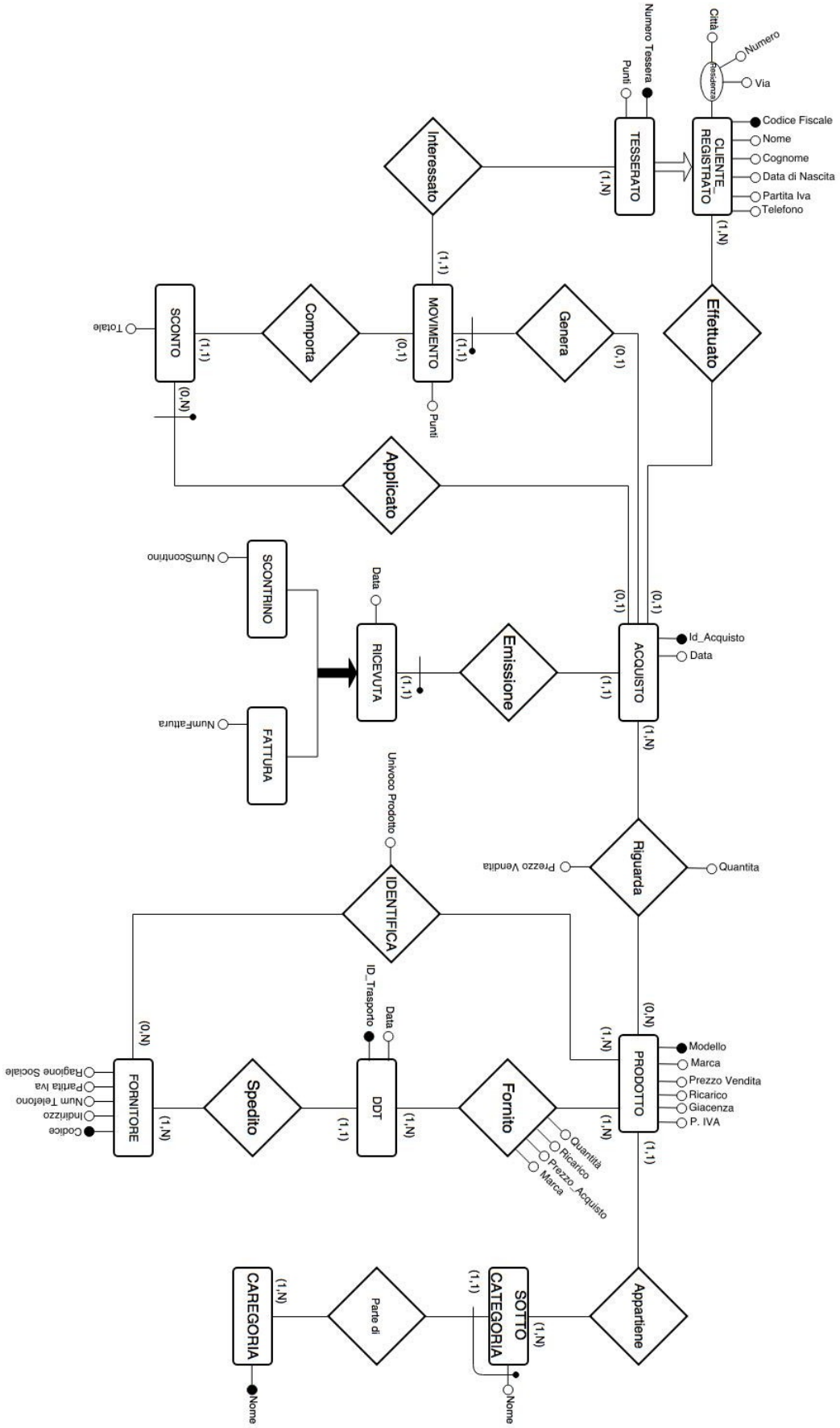
- *DDT – Fornitore*: Spedito legame che unisce il *DDT* al F*ornitore* che lo ha spedito
  - Un *DDT* viene spedito da un unico *Fornitore*, un *Fornitore* può spedire da 1 a N *DDT*
  - Molteplicità: 1:N

- *Prodotto – Fornitore*: Identifica associa al *prodotto* presente l'univoco prodotto definito dal *fornitore*
  - Un *Prodotto* viene identificato da 1 a N *Fornitori*, ogni *Fornitore* identifica da 0 a N *Prodotti*
  - Molteplicità: N:N
  - Attributi:
    - UnivocoProdotto INTEGER

- *Prodotto – Sotto_Categoria*: Appartiene
  - Un *Prodotto* appartiene ad una sola *Sotto_Categoria*, ad una *Sotto_Categoria* appartengono da 1 a N *Prodotti*
  - Molteplicità: 1:N

- *Sotto_Categoria – Categoria*: Parte_Di
  - Una *Sotto_Categoria* da parte di un'unica *Categoria*, in una *Categoria* ci sono da 1 a N *Sotto_Categorie*
  - Molteplicità: 1:N

3.3 Schema E-R iniziale



4. Progettazione Logica

4.1 Ristrutturazione

4.1.1 Analisi delle Ridondanze

Ci sono due casi di ridondanze che e' stato deciso di mantenere in quanto:

- L'attributo *Prezzo_Vendita* presente nell'associazione Riguarda, e' un dato derivabile da Prezzo_Vendita in Prodotto ma, nel corso del tempo, il prezzo di vendita può variare, quindi non esisterebbe più questa ridondanza in corrispondenza di un prodotto venduto in una data passata

- L'attributo *Giacenza* in Prodotto e' un dato derivabile dalla somma delle quantità in Fornito – somma delle quantità in Riguarda, ma in base agli inserimenti che verranno fatti sul database, e' più conveniente mantenere la ridondanza con un conseguente miglioramento dell'efficienza

In un caso invece è possibile eliminare la ridondanza:

- L'attributo Data in *Ricevuta* viene eliminato in quanto, essendo in relazione con A*cquisto* tramite l'associazione *Emissione* avente molteplicità 1 a 1, la Data è facilmente ricavabile.

4.1.2 Eliminazione delle generalizzazioni

- La generalizzazione *Tesserato* di *Cliente_Registrato* viene promossa ad entità e diventa *Tessera* la quale è legata a *Cliente_Registrato* tramite l'associazione *Possiede* con molteplicità 1 a N.

- Le generalizzazioni di *Ricevuta* (*Scontrino* e *Fattura*) sono state eliminate e i loro attributi vengono assorbiti dall'entità padre. A questa entità viene aggiunto un nuovo attributo *Tipo* che serve per indicare quale tipologia di ricevuta si intende selezionare, e per la gestione dei numeri di Fattura e Scontrino che hanno due comportamenti diversi nella durata dell'anno, si è deciso di utilizzare un solo attributo *Numero*, e non più due divisi e la selezione di tale valore viene effettuata successivamente in modo automatizzato.

4.1.3 Partizionamento/Accorpamento di entità e associazioni

- E' stato deciso di partizionare l'entità *Fornitore* in modo tale da gestire i dati anagrafici dei fornitori, separatamente rispetto ai dati principali per l'identificazione. I dati anagrafici, quindi, sono stati trasferiti nella nuova entità *Anagrafica* che è collegata tramite l'associazione *Dati_Fornitore*.
  Questa scelta è stata fatta sulla base del fatto che la maggior parte degli accessi al database non richiedono i dati anagrafici, ma soltanto i dati utili per identificare quale fornitore ha fornito un prodotto.

- È stato deciso di creare un entità separata da Cliente_Registrato per storicizzare i dati di residenza dei clienti

4.1.4 Scelta degli identificatori principali

- Cliente_Registrato: per Cliente_Registrato è stato scelto come identificatore principale *Codice_Fiscale* in quanto è l'unico codice personale che può identificare univocamente un cliente

- Tessera: viene identificata da un *Numero* che è il numero scritto fisicamente sulla tessera

- Movimento: in un *Acquisto* può essere inserito un solo movimento di punti perciò *Acquisto* è la chiave di questa entità

- Acquisto: per la chiave primaria è stato scelto un *Id* auto incrementale. *Data* e *Cliente_Registrato* sarebbero stati sufficienti, ma avere un identificatore interno di dimensioni ridotte è preferibile ad uno esterno che comprende più attributi

- Sconto: la chiave primaria sono i *Punti* poiché solo determinati punti possono comportare uno sconto

- Ricevuta: in ogni acquisto viene emessa una ricevuta, perciò, come identificatore primario è stato scelto *Acquisto*

- Prodotto: l'unico dato che può essere univoco per identificare un prodotto è il *Modello*

- Fornitore: identificato da un *Codice* numerico interno del negozio

- Anagrafica: come per Ricevuta è identificata dal *Fornitore*

- Categoria: la chiave è proprio il *Nome* della categoria

- Sotto_Categoria: identificata da un *Id* per una più facile realizzazione delle query

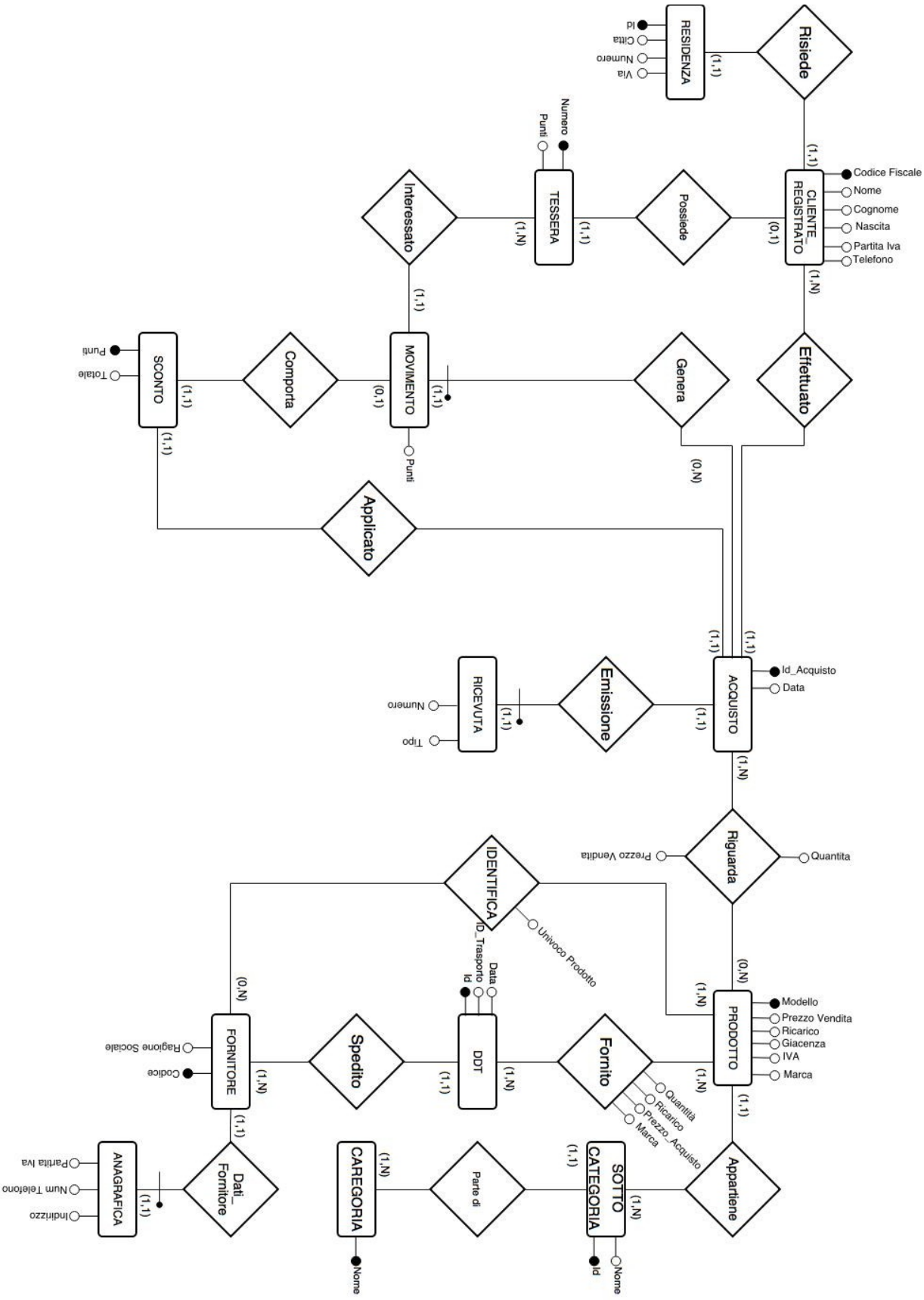
- DDT: come Sotto_Categoria, la chiave primaria è *Id*, altrimenti si sarebbe dovuto utilizzare l'insieme *ID_Trasporto*, *Data*, Fornitore che avrebbe portato ad una realizzazione più complessa delle altre tabelle; per questo motivo è stata utilizzata una chiave più sintetica.

4.2 Modello Relazionale

- Cliente_Registrato(Codice_Fiscale, Nome, Cognome, Nascita,P_IVA,Residenza,Telefono)
- Residenza(Id, Citta,Via,Numero)
- Tessera(Numero, Cliente_Registrato, Punti)
- Acquisto(Id,Data,Cliente_Registrato)
- Movimento(Acquisto, Tessera, Punti)
- Applicato(Acquisto, Sconto)
- Sconto(Punti, Totale)
- Ricevuta(Acquisto, Tipo, Numero)
- Prodotto(Modello, Marca,Prezzo_Vendita, Ricarico, Giacenza, IVA)
- Appartiene(Prodotto,Sotto_Categoria)
- Riguarda(Acquisto, Prodotto, Quantita, Prezzo_Vendita)
- DDT(Id, Id_Trasporto, Data, Fornitore)
- Fornito(Prodotto, DDT,Marca, Quantita,Prezzo_Acquisto,Ricarico)
- Fornitore(Codice, Ragione_Sociale)
- Identifica(Fornitore, Prodotto, Univoco_Prodotto)
- Anagrafica(Fornitore, P_IVA, Telefono,Indirizzo)
- Sotto_Categoria(Id,Categoria, Nome)
- Categoria(Nome)

NOTA: nell'associazione Applicato, sebbene sia una relazione con molteplicità 1 a N, come foreign key si è deciso di utilizzare soltanto Acquisto, in quanto Sconto, con il variare del tempo, potrebbe cambiare.

4.3 Schema E-R ristrutturato



5. Implementazione Schema Logico

5.1 Lista delle Tabelle

- Cliente_Registrato
  - *Codice_Fiscale* Character(16) *<<PK>>*
  - *Nome* Varchar(20)
  - *Cognome* Varchar(20)
  - *Nascita* Date
  - *P_Iva* Varchar(10)
  - *Residenza* Integer *<<FK (Residenza)>>*
  - *Telefono* Integer
- Residenza
  - *Id* Integer *<<PK>>*
  - *Citta* Varchar(20)
  - *Via* Varchar(20)
  - *N_Civico* Integer
- Tessera
  - *Numero* Integer *<<PK>>*
  - *Cliente_Registrato* Varchar(20) *<<FK (Cliente_Registrato)>>*
  - *Punti* Integer
- Acquisto
  - *Id* Integer Auto_Increment *<<PK>>*
  - *Data* Datetime
  - *Cliente_Registrato* Varchar(20) *<<FK (Cliente_Registrato)>>*
- Movimento
  - *Acquisto* Integer *<<PK>>*, *<<FK (Acquisto)>>*
  - *Tessera* Integer *<<FK (Tessera)>>*
  - *Punti* Integer
- Applicato
  - *Acquisto* Integer *<<PK>>*, *<<FK (Acquisto)>>*
  - *Sconto* Integer
- Sconto
  - *Punti* Integer *<<PK>>*
  - *Totale* Integer Not Null Unique
  -
- Ricevuta
  - *Acquisto* Integer *<<PK>>*, *<<FK (Acquisto)>>*
  - *Tipo* Character
  - *Numero* Integer
- Prodotto
  - *Modello* Varchar(20) *<<PK>>*
  - *Marca* Varchar(20)
  - *Prezzo_Vendita* Decimal(8,2)
  - *Ricarico* Integer
  - *Giacenza* Integer
  - *Iva* Integer Default 0

- Appartiene
  - *Prodotto* Varchar(20) *<<PK>>, <<FK (Prodotto)>>*
  - *Sotto_Categoria* Integer(20) *<<FK (Sotto_Categoria)>>*
- Riguarda
  - *Acquisto* Integer *<<FK (Acquisto)>>*
  - *Prodotto* Varchar(20) *<<FK (Prodotto)>>*
  - *Quantita* Integer
  - *Prezzo_Vendita* Decimal(8,2)
    <<PK(Acquisto, Prodotto)>>
- DDT
  - *Id* Integer Auto_Increment *<<PK>>*
  - *Id_Trasporto* Integer
  - *Data* Datetime Default Now()
  - Fornitore Integer *<<FK (Fornitore)>>*
- Fornito
  - *DDT* Integer *<<FK (DDT)>>*
  - *Prodotto* Varchar(20) *<<FK (Prodotto)>>*
  - *Marca* Varchar(20)
  - *Quantita* Integer
  - *Prezzo_Acquisto* Decimal(8,2)
  - *Ricarico* Integer
    <<PK(DDT, Prodotto)>>
- Fornitore
  - *Codice* Integer *<<PK>>*
  - *Ragione_Sociale* Varchar(20)
- Identifica
  - *Prodotto* Varchar(20) *<<PK>>, <<FK (Prodotto)>>*
  - *Fornitore* Integer *<<PK>>, <<FK (Fornitore)>>*
  - *Univoco_Prodotto* Integer
    <<PK(Fornitore, Prodotto)>>
- Anagrafica
  - *Fornitore* Integer *<<PK>>, <<FK (Fornitore)>>*
  - *P_Iva* Integer
  - *Indirizzo* Varchar(20)
  - *Telefono* Varchar(20)
- Categoria
  - *Nome* Varchar(20) *<<PK>>*
- Sotto_Categoria
  - *Id* Integer Auto_Increment *<<PK>>*
  - *Categoria* Varchar(20) *<<FK (Categoria)>>*
  - *Nome* Varchar(20)

6. Implementazione della Base di Dati

6.1 Creazione delle Tabelle

```
1. DROP TABLE IF EXISTS Cliente_Registrato;
2. DROP TABLE IF EXISTS Residenza;
3. DROP TABLE IF EXISTS Tessera;
4. DROP TABLE IF EXISTS Acquisto;
5. DROP TABLE IF EXISTS Movimento;
6. DROP TABLE IF EXISTS Applicato;
7. DROP TABLE IF EXISTS Sconto;
8. DROP TABLE IF EXISTS Ricevuta;
9. DROP TABLE IF EXISTS Prodotto;
10.DROP TABLE IF EXISTS Appartiene;
11.DROP TABLE IF EXISTS Riguarda;
12.DROP TABLE IF EXISTS DDT;
13.DROP TABLE IF EXISTS Fornito;
14.DROP TABLE IF EXISTS Fornitore;
15.DROP TABLE IF EXISTS Identifica;
16.DROP TABLE IF EXISTS Anagrafica;
17.DROP TABLE IF EXISTS Categoria;
18.DROP TABLE IF EXISTS Sotto_Categoria;
19.
20.CREATE TABLE Cliente_Registrato (
21.    Codice_Fiscale CHARACTER(16) PRIMARY KEY,
22.    Nome VARCHAR(20),
23.    Cognome VARCHAR(20),
24.    Nascita DATE,
25.    P_IVA VARCHAR(10),
26.    Residenza INTEGER,
27.    Telefono INTEGER,
28.    FOREIGN KEY( Residenza ) REFERENCES Residenza( ID )
29.)engine=innoDB;
30.
31.CREATE TABLE Residenza (
32.    Id INTEGER PRIMARY KEY,
33.    Citta VARCHAR(20),
34.    Via VARCHAR(20),
35.    N_Civico INTEGER
36.)engine=innoDB;
37.
38.CREATE TABLE Tessera (
39.    Numero INTEGER PRIMARY KEY,
40.    Cliente_Registrato VARCHAR(20),
41.    PUNTI INTEGER,
42.    FOREIGN KEY( Cliente_Registrato ) REFERENCES
   Cliente_Registrato( Codice_Fiscale )
43.)engine=innoDB;
44.
45.CREATE TABLE Acquisto (
46.    Id INTEGER AUTO_INCREMENT PRIMARY KEY,
47.    Data DATETIME,
48.    Cliente_Registrato VARCHAR(20),
49.    FOREIGN KEY(Cliente_Registrato)REFERENCES
   Cliente_Registrato(Codice_Fiscale)
50.)engine=innoDB;
51.
52.
53.
```

```
54.CREATE TABLE Movimento (
55.    Acquisto INTEGER PRIMARY KEY,
56.    Tessera INTEGER,
57.    Punti INTEGER,
58.    FOREIGN KEY( Acquisto ) REFERENCES Acquisto(Id),
59.    FOREIGN KEY( Tessera ) REFERENCES Tessera( Numero ) ON DELETE CASCADE
60.)engine=innoDB;
61.
62.CREATE TABLE Applicato (
63.    Acquisto INTEGER PRIMARY KEY,
64.    Sconto INTEGER,
65.    FOREIGN KEY( Acquisto ) REFERENCES Acquisto( Id )
66.)engine=innoDB;
67.
68.CREATE TABLE Sconto (
69.    Punti INTEGER PRIMARY KEY,
70.    Totale INTEGER NOT NULL UNIQUE
71.)engine=innoDB;
72.
73.CREATE TABLE Ricevuta (
74.    Acquisto INTEGER PRIMARY KEY,
75.    Tipo CHARACTER,
76.    Numero INTEGER,
77.    FOREIGN KEY( Acquisto ) REFERENCES Acquisto( Id )
78.)engine=innoDB;
79.
80.CREATE TABLE Prodotto (
81.    Modello VARCHAR(20) PRIMARY KEY,
82.    Marca VARCHAR(20),
83.    Prezzo_Vendita DECIMAL(8,2),
84.    Ricarico INTEGER,
85.    Giacenza INTEGER,
86.    IVA INTEGER default 0
87.)engine=innoDB;
88.
89.CREATE TABLE Appartiene (
90.    Prodotto VARCHAR(20),
91.    Sotto_Categoria INTEGER(20),
92.    PRIMARY KEY(Prodotto),
93.    FOREIGN KEY(Sotto_Categoria ) REFERENCES Sotto_Categoria(Id),
94.    FOREIGN KEY(Prodotto) REFERENCES Prodotto(Modello)
95.)engine=innoDB;
96.
97.CREATE TABLE Riguarda (
98.      Acquisto INTEGER,
99.Prodotto VARCHAR(20),
100.         Quantita INTEGER,
101.         Prezzo_Vendita DECIMAL(8,2),
102.         PRIMARY KEY(Acquisto, Prodotto),
103.         FOREIGN KEY(Acquisto ) REFERENCES Acquisto(Id),
104.         FOREIGN KEY(Prodotto ) REFERENCES Prodotto(Modello)
105.     )engine=innoDB;
106.
107.     CREATE TABLE DDT (
108.         Id INTEGER AUTO_INCREMENT PRIMARY KEY,
109.         ID_Trasporto INTEGER,
110.         Data DATETIME default NOW(),
111.         Fornitore INTEGER
112.     )engine=innoDB;
113.
114.
```

```
115.    CREATE TABLE Fornito (
116.        DDT INTEGER,
117.        Prodotto VARCHAR(20),
118.        Marca VARCHAR(20),
119.        Quantita INTEGER,
120.        Prezzo_Acquisto DECIMAL(8,2),
121.        Ricarico INTEGER,
122.        PRIMARY KEY( DDT, Prodotto ),
123.        FOREIGN KEY( DDT ) REFERENCES DDT( Id ),
124.        FOREIGN KEY( Prodotto ) REFERENCES Prodotto( Modello )
125.    )engine=innoDB;
126.
127.    CREATE TABLE Fornitore (
128.        Codice INTEGER PRIMARY KEY,
129.        Ragione_Sociale VARCHAR(20)
130.    )engine=innoDB;
131.
132.    CREATE TABLE Identifica (
133.        Prodotto VARCHAR(20),
134.        Fornitore INTEGER,
135.        Univoco_Prodotto INTEGER,
136.        PRIMARY KEY( Fornitore, Prodotto ),
137.        FOREIGN KEY( Prodotto ) REFERENCES Prodotto( Modello ),
138.        FOREIGN KEY( Fornitore ) REFERENCES Fornitore( Codice )
139.    )engine=innoDB;
140.
141.    CREATE TABLE Anagrafica (
142.        Fornitore INTEGER PRIMARY KEY,
143.        P_IVA INTEGER,
144.        Indirizzo VARCHAR(20),
145.        Telefono VARCHAR(20),
146.        FOREIGN KEY(Fornitore) REFERENCES Fornitore(Codice)
147.    )engine=innoDB;
148.
149.    CREATE TABLE Categoria (
150.        Nome VARCHAR(20) PRIMARY KEY
151.    )engine=innoDB;
152.
153.    CREATE TABLE Sotto_Categoria (
154.        Id INTEGER AUTO_INCREMENT PRIMARY KEY,
155.        Categoria VARCHAR(20),
156.        Nome VARCHAR(20),
157.        FOREIGN KEY(Categoria) REFERENCES Categoria(Nome)
158.    )engine=innoDB;
```

6.2 Query

Seguono alcuni esempi di query implementate

1. Query che restituisce i dati dei clienti registrati che hanno acquistato almeno un prodotto per ogni categoria merciologica presente all'interno del negozio

```
SELECT *
FROM Cliente_Registrato R
WHERE NOT EXISTS
      (
       SELECT Nome
       FROM Categoria
       WHERE Categoria.Nome NOT IN
            (
             SELECT SC1.Categoria
             FROM Acquisto A1, Riguarda R1,
                  Appartiene App1, Sotto_Categoria SC1
              WHERE A1.Cliente_Registrato = R.Codice_Fiscale AND
                    A1.Id = R1.Acquisto AND
                    R1.Prodotto = App1.Prodotto AND
                    App1.Sotto_Categoria = SC1.Id
            )
             );
```

Output

```
+------------------+--------+-----------+------------+-------+-----------+------------+
| Codice_Fiscale   | Nome   | Cognome   | Nascita    | P_IVA | Residenza | Telefono   |
+------------------+--------+-----------+------------+-------+-----------+------------+
| MNTNDR94P17L407J | Andrea | Mantovani | 1994-09-17 | NULL  |        11 | 3406936174 |
+------------------+--------+-----------+------------+-------+-----------+------------+
```

2. Determinare quali delle 5 sotto_categorie più vendute nello scorso mese, sono ancora le più vendute nel mese corrente

```
select PASS.Nome
FROM( select SC.Nome, SUM(R.Quantita) as Massimo
      from Sotto_Categoria SC, Appartiene A, Riguarda R, Acquisto Acq
      where SC.Id=A.Sotto_Categoria AND
            A.Prodotto=R.Prodotto AND
            Acq.Id=R.Acquisto AND
            MONTH(Acq.Data)=MONTH(NOW()) AND
            YEAR(Acq.Data)=YEAR(NOW())
      Group by SC.Nome
            Order By Massimo DESC
            LIMIT 5
          ) CORR JOIN
                (
                 select SC1.Nome, SUM(R1.Quantita) as Massimo
                 from Sotto_Categoria SC1,
                      Appartiene A1, Riguarda R1, Acquisto Acq1
                 where SC1.Id=A1.Sotto_Categoria AND
                       A1.Prodotto=R1.Prodotto AND
                       Acq1.Id=R1.Acquisto AND
                       MONTH(Acq1.Data)= MONTH(NOW())-1 AND
                       YEAR(Acq1.Data)=YEAR(NOW())
                 Group by SC1.Nome
                 Order By Massimo DESC
                 LIMIT 5
                ) PASS ON CORR.Nome=PASS.Nome;
```

Output:

```
+---------------+
| Nome          |
+---------------+
| FrigoriferiBI |
| Tv            |
| Asciugatrice  |
| Fotografia    |
+---------------+
```

3. Dire quali clienti hanno acquistato un prodotto una sola volta ad il relativo prodotto

```
1. Select C.Codice_Fiscale, C.Nome, C.Cognome, R.Prodotto
2. From Cliente_Registrato C, Acquisto A, Riguarda R
3. Where C.Codice_Fiscale=A.Cliente_Registrato AND
4.       A.Id=R.Acquisto AND C.Codice_Fiscale<>'0' AND
5.       NOT EXISTS (select A1.Id, A1.Cliente_Registrato, R1.Prodotto
6.                   from Acquisto A1, Riguarda R1
7.                   where A1.Id=R1.Acquisto AND
8.                         A1.Cliente_Registrato=A.Cliente_Registrato AND
9.                         A1.Id<>A.Id AND
10.                              R.Prodotto=R1.Prodotto
11.                       );
```

```
+------------------+---------+-----------+----------------+
| Codice_Fiscale   | Nome    | Cognome   | Prodotto       |
+------------------+---------+-----------+----------------+
| MNTNDR94P17L407J | Andrea  | Mantovani | GIO48          |
| MNTNDR94P17L407J | Andrea  | Mantovani | LOP-25687      |
| MNTNDR94P17L407J | Andrea  | Mantovani | x5DIP          |
| MNTNDR94P17L407J | Andrea  | Mantovani | P880           |
| MNTNDR94P17L407J | Andrea  | Mantovani | SCALD526       |
| MSSFBA94B60L840Y | Fabio   | Massignan | KUL15A60       |
| MSSFBA94B60L840Y | Fabio   | Massignan | UE32J500       |
| MSSFBA94B60L840Y | Fabio   | Massignan | ARL6501APIU    |
| MSSFBA94B60L840Y | Fabio   | Massignan | GIO48          |
| MSSFBA94B60L840Y | Fabio   | Massignan | LOP-25687      |
| MSSFBA94B60L840Y | Fabio   | Massignan | SCALD526       |
| ZCCGCM94D04Z526U | Giacomo | Zecchin   | Compaq_LA2006x |
+------------------+---------+-----------+----------------+
```

4. Cliente che ha effettuato il maggior numero di acquisti utilizzando gli sconti

```
1. Drop view if exists AcquistiScontatiPerCliente;
2. create view AcquistiScontatiPerCliente AS
3.       select A.Cliente_Registrato, count(*) as NumAcquisti
4.       from Acquisto A Right JOIN Applicato Ap ON A.Id=Ap.Acquisto
5.       where A.Cliente_Registrato<>'0'
6.       Group By A.Cliente_Registrato;
7.
8. Select C.Codice_Fiscale, C.Nome, C.Cognome, Acq.NumAcquisti
9. from AcquistiScontatiPerCliente Acq, Cliente_Registrato C
10.      Where Acq.Cliente_Registrato=C.Codice_Fiscale AND
11.            Acq.NumAcquisti >=ALL(Select Acq1.NumAcquisti
12.                                  From AcquistiScontatiPerCliente Acq1);
```

Output:

```
+------------------+--------+-----------+-------------+
| Codice_Fiscale   | Nome   | Cognome   | NumAcquisti |
+------------------+--------+-----------+-------------+
| MNTNDR94P17L407J | Andrea | Mantovani |           3 |
+------------------+--------+-----------+-------------+
```

5. Tutti gli acquisti effettuati dal cliente avente codice fiscale: 'MSSFBA94B60L840Y' nel mese corrente

```
1. drop view if exists AcquistiCliente;
2. Create view AcquistiCliente as
3.       select *
4.       From Acquisto
5.       where Cliente_Registrato='MSSFBA94B60L840Y' AND
6.             MONTH(DATA)=MOUNTH(NOW());
7.
8. drop view if exists ProdottiAcquisto;
9. Create view ProdottiAcquisto as
10.            Select  R.Prodotto, R.Quantita, R.Prezzo_Vendita, A.Data
11.            From AcquistiCliente A, Riguarda R
12.            Where A.Id=R.Acquisto;
13.
14.      Select  PA.Prodotto, P.Marca, PA.Quantita,
15.              PA.Prezzo_Vendita, PA.Data
16.      From Prodotto P, ProdottiAcquisto PA
17.      Where PA.Prodotto=P.Modello;
```

Output:

Prodotto	Marca	Quantita	Prezzo_Vendita	Data
SCALD526	CAT	1	108.36	2015-07-03 15:42:46
KFR562	LG	1	219.11	2015-07-05 16:04:51
RNLE24	ASUS	3	657.33	2015-07-06 16:12:38
MI200KY	MIELE	1	1095.55	2015-07-07 09:49:21

6. Query che restituisce quanti prodotti e l'importo totale di vendita sono stati venduti per ogni Sotto_Categoria dal 1 Giugno al 26 Giugno

```
1. Select SC.Nome as SottoCategoria,
2.        COALESCE(SUM(Prezzo_Vendita),0) as Prezzo,
3.        COALESCE(SUM(Quantita),0) as Quantita
4. From((Sotto_Categoria SC
5.         LEFT JOIN Appartiene A ON SC.Id=A.Sotto_Categoria)
6.             LEFT JOIN Riguarda R ON A.Prodotto=R.Prodotto)
7.                   LEFT JOIN Acquisto Ac ON R.Acquisto=Ac.Id
8. Where DATE(Ac.Data)>=DATE('2015-06-01') AND
9.       DATE(Ac.Data)<=DATE('2015-06-26')
10.      Group By SC.Nome;
```

Output:

SottoCategoria	Prezzo	Quantita
Asciugatrice	120.18	8
Fotografia	327.36	5
FrigoriferiBI	1323.90	58
Radio/HiFi	1.01	1
Smartphone	46.12	2
Tv	1791.99	26

6.3 Procedure e Funzioni

1. Funzione che dato in ingresso l'Id di un acquisto, restituisce il totale di quell'Acquisto

```
1. DELIMITER |
2. DROP FUNCTION IF EXISTS TotaleAcquisto|
3. CREATE FUNCTION TotaleAcquisto (Id_Acq INTEGER) RETURNS DECIMAL(8,2)
4. BEGIN
5.       DECLARE Totale DECIMAL(8,2);
6.       SELECT SUM(R.Prezzo_Vendita) INTO Totale
7.       FROM Riguarda as R
8.       WHERE R.Acquisto=Id_Acq;
9. RETURN Totale;
10.      END |
11.      DELIMITER;
```

2. Procedura che gestisce l'azzeramento del numero degli scontrini (giornalmente) e del numero delle fatture (Annualmente)

```
1. DROP PROCEDURE IF EXISTS prRicevuta;
2. DELIMITER |
3. CREATE PROCEDURE prRicevuta (Id_Acq INTEGER, DataRic DATETIME, Tipo
   CHARACTER)
4. BEGIN
5.       declare NumeroRic INTEGER;
6.       declare ultimaData DATETIME;
7.       select Data into ultimaData
8.       from Acquisto
9.       order by Data DESC
10.           LIMIT 1, 1;
11.
12.           IF (Id_Acq>1)
13.           THEN
14.                 IF(Tipo='F')
15.                 THEN
16.                       IF(YEAR(DataRic)>YEAR(ultimaData))
17.                       THEN
18.                             set NumeroRic=1;
19.                       ELSE
20.                             Select A.Massimo into NumeroRic
21.                             From (select MAX(Numero) as Massimo,
22.                                         R.Tipo, Acq.Data
23.                                   From Ricevuta R, Acquisto Acq
24.                                   Where R.Tipo='F' AND
25.                                   Acq.Id=R.Acquisto AND
26.                                   YEAR(Acq.Data)=YEAR(DataRic)) A;
27.                             set NumeroRic=NumeroRic+1;
28.                       END IF;
29.                 ELSEIF(Tipo='S')
30.                 THEN
31.                       IF(DATE(DataRic)>DATE(ultimaData))
32.                       THEN
33.                             set NumeroRic=1;
34.                       ELSE
35.                             Select A.Massimo into NumeroRic
36.                             From (select MAX(Numero) as Massimo,
37.                                         R.Tipo, Acq.Data
38.                                   From Ricevuta R, Acquisto Acq
39.                                   Where R.Tipo='S' AND
40.                                   Acq.Id=R.Acquisto AND
41.                                   DATE(Acq.Data)=DATE(DataRic)) A;
42.                             set NumeroRic=NumeroRic+1;
43.                       END IF;
```

```
                END IF;
            ELSE
                set NumeroRic=1;
            END IF;
            insert into Ricevuta VALUES (Id_Acq, Tipo, NumeroRic);
      END |
      DELIMITER ;
```

3. Procedura che controlla se il cliente che ha effettuato l'acquisto è tesserato, e in caso affermativo: se non sono stati scalati dei punti, effettua il calcolo di quanti punti devono essere assegnati inserendoli nella tabella Movimento, altrimenti inserisce i punti scalati negativi

```
DROP PROCEDURE IF EXISTS AddPunti;
DELIMITER |
CREATE PROCEDURE AddPunti(Cliente Varchar(20),Acquisto INT,PuntiScalati
INT)
BEGIN
      Declare NumTessera INT;
      Declare Totale INT;
      Declare PuntiPresenti INTEGER;
      select T.Punti into PuntiPresenti
      from Tessera T
      where T.Cliente_Registrato=Cliente;

            IF EXISTS(SELECT T.Cliente_Registrato
                      FROM Tessera as T
                      WHERE Cliente=T.Cliente_Registrato)
            THEN
                  Select T.Numero INTO NumTessera
                  From Tessera as T
                  WHERE Cliente=T.Cliente_Registrato;
                  IF(PuntiScalati=0 OR PuntiPresenti<(PuntiScalati*-1))
                  THEN
                        Select TRUNCATE(TotaleAcquisto(Acquisto)/10,0)
                                        into Totale;
                        INSERT INTO Movimento VALUES (Acquisto,
                                                      NumTessera, Totale);
                  ELSEIF(PuntiScalati<0)
                  THEN
                       INSERT INTO Movimento VALUES(Acquisto,
                                                    NumTessera,PuntiScalati);
                  END IF;
            END IF;
      END |
      DELIMITER ;
```

4. Procedura per l'associazione di un prodotto alla relativa Sotto_Categoria

```
DELIMITER |
DROP PROCEDURE IF EXISTS AssegnaSottoCat|
CREATE PROCEDURE AssegnaSottoCat (Modello VARCHAR(20), SottoCat INTEGER)
BEGIN
      IF NOT EXISTS(SELECT Ap.Prodotto
                    FROM Appartiene AS Ap
                    WHERE Ap.Prodotto=Modello)
      THEN
            INSERT INTO Appartiene (Prodotto, Sotto_Categoria)
                   VALUES (Modello,SottoCat);
            END IF;
      END |
      DELIMITER ;
```

5. Procedura per l'associazione di un prodotto al relativo univoco del fornitore

```
DELIMITER |
DROP PROCEDURE IF EXISTS IdentificaUnivoco|
CREATE PROCEDURE IdentificaUnivoco(Prodotto VARCHAR(20),
                                   Fornitore INTEGER,Univoco INTEGER)
BEGIN
      IF NOT EXISTS(SELECT I.Prodotto, I.Fornitore
                    FROM Identifica AS I
                    WHERE I.Prodotto=Prodotto AND
                          I.Fornitore=Fornitore)
           THEN
                 INSERT INTO Identifica (Prodotto, Fornitore,
                                         Univoco_Prodotto)
                       VALUES (Prodotto, Fornitore, Univoco);
           END IF;
      END |
      DELIMITER ;
```

6. Procedura che inserisce su Riguarda che prodotto è stato acquistato in un dato acquisto, in che quantità e a quale prezzo, in più vene tenuta aggiornata anche la giacenza a magazzino

```
DELIMITER |
DROP PROCEDURE IF EXISTS ProdottiAcquisto|
CREATE PROCEDURE ProdottiAcquisto (Acquisto INTEGER,
                                   Modello VARCHAR(20),
                                   AggQuantita INTEGER,
                                   Prezzo_Vendita DECIMAL(8,2))
BEGIN
      DECLARE Prezzo DECIMAL(8,2);
      IF((SELECT P.Giacenza
                FROM Prodotto as P
                WHERE P.Modello=Modello)>=AggQuantita)
           THEN
                 IF (Prezzo_Vendita = 0)
                 THEN
                       set Prezzo=(SELECT P.Prezzo_Vendita
                                   FROM Prodotto as P
                                   WHERE P.Modello=Modello);
                 ELSE
                       set Prezzo=Prezzo_Vendita;
                 END IF;
                 IF EXISTS(SELECT R.Acquisto, R.Prodotto
                           FROM Riguarda as R
                           WHERE R.Acquisto=Acquisto AND
                           R.Prodotto=Modello)
                 THEN
                       UPDATE Riguarda as R SET
                              R.Quantita=R.Quantita+AggQuantita,
                     R.Prezzo_Vendita=R.Prezzo_Vendita+Prezzo*AggQuantita)
                        WHERE R.Acquisto=Acquisto AND
                              R.Prodotto=Modello;
                 ELSE
                       INSERT INTO Riguarda

      VALUES(Acquisto,Modello,AggQuantita,Prezzo*AggQuantita);
                 END IF;
                 UPDATE Prodotto as P Set Giacenza=Giacenza-AggQuantita
                 WHERE P.Modello=Modello;
           END IF;
      END |
      DELIMITER ;
```

7. Procedura che tiene aggiornati il numero di punti totali della tessera di un determinato cliente Registrato

```
1. DROP Procedure IF EXISTS PuntiTessera;
2. DELIMITER |
3. CREATE Procedure PuntiTessera(PuntiMovim INTEGER,
4.                               TesseraMovim INTEGER)
5. BEGIN
6.       UPDATE Tessera SET Punti=Punti+PuntiMovim
7.       WHERE Numero=TesseraMovim;
8. END |
9. DELIMITER ;
```

6.4 Trigger

1. Trigger che per ogni prodotto fornito, lo inserisce nella tabella prodotto aggiornando la quantità, ed il prezzo di vendita soltanto se il ricarico è maggiore di quello precedente, se il prodotto è presente, altrimenti inserisce normalmente il prodotto ed i relativi attributi

```
DROP TRIGGER IF EXISTS AddProdotto;
DELIMITER |
CREATE TRIGGER AddProdotto
BEFORE INSERT ON Fornito
FOR EACH ROW
BEGIN
      IF EXISTS(SELECT P.Modello
                FROM Prodotto AS P
                WHERE P.Modello=New.Prodotto)
      THEN
            IF((SELECT P.Ricarico
                FROM Prodotto AS P
                WHERE P.Modello=NEW.Prodotto)<New.Ricarico)
            THEN
                  UPDATE Prodotto
                  SET Prezzo_Vendita=NEW.Prezzo_Acquisto*(1+(NEW.Ricarico/100)),
                      Ricarico=NEW.Ricarico
                  WHERE Modello=New.Prodotto;
            END IF;
            UPDATE Prodotto
            SET Giacenza=Giacenza+New.Quantita
            WHERE Modello=New.Prodotto;
      ELSE
            INSERT INTO Prodotto(Modello,Marca,Prezzo_Vendita,Ricarico,Giacenza)
            VALUES( New.Prodotto,
                    New.Marca,
                    New.Prezzo_Acquisto*(1+(New.Ricarico/100)),
                    New.Ricarico,
                    New.Quantita);
      END IF;
END |
DELIMITER ;
```

2. Trigger che mantiene aggiornati i punti accumulati e, se vengono scalati, inserisce nella relazione Applicato in quale acquisto è stato applicato uno sconto

```
DROP TRIGGER IF EXISTS ApplicaSconto;
DELIMITER $$
CREATE TRIGGER ApplicaSconto
```

```
AFTER INSERT ON Movimento
FOR EACH ROW
BEGIN
      CALL PuntiTessera(New.Punti, New.Tessera);
      IF(New.Punti<0)
      THEN
            INSERT INTO Applicato (Select NEW.Acquisto, Totale
                                   From Sconto
                                   Where ABS(NEW.Punti)=Punti );
      END IF;
END$$
DELIMITER ;
```

3. Trigger che calcola il prezzo di vendita aggiungendo l'iva

```
DROP TRIGGER IF EXISTS upIva;
DELIMITER |
CREATE TRIGGER upIva
BEFORE Update ON Prodotto
FOR EACH ROW
BEGIN
      IF(Old.Iva<>New.IVA)
      THEN
            set New.Prezzo_Vendita=New.Prezzo_Vendita*(1+(NEW.IVA/100));
      END IF;
END |
DELIMITER ;
```

7. Interfaccia web

7.1 acquisti_admin.php

è uno snippet del file *acquisti_admin.php* all'interno del quale è possibile effettuare la ricerca di tutti gli acquisti effettuati dai clienti. Si ha la possibilità di ordinare i risultati per data, prezzo, ed anche restituire gli acquisti di un cliente che vogliamo ricercare. Tutto questo grazie all'integrabilità di mysql su php.

```
<?php
    OpenTable();

    $cliente = "";
    if( isset( $_GET["Cliente"] ) && preg_match( $_REG_CODICE, $_GET["Cliente"] ) === 1 )
      { $cliente = $_GET["Cliente"]; }

    $query_cliente=($cliente=="") ? TRUE : "A.Cliente_Registrato='$cliente'";

    mysql_query( "Create or replace view ProdottiAcquisto as
                  Select A.Cliente_Registrato, R.Prodotto, R.Quantita, R.Prezzo_Vendita, A.Data
                  From Acquisto A, Riguarda R
                  Where A.Id=R.Acquisto AND
                  $query_cliente;", ConnectDB() );

    $condition = "";

    $prezzoStart = ( isset( $_GET["PrezzoStart"] ) &&
                     preg_match($_REG_NUMERO,$_GET["PrezzoStart"])===1) ? $_GET["PrezzoStart"] : "";
    $prezzoEnd = ( isset( $_GET["PrezzoEnd"] ) &&
                   preg_match( $_REG_NUMERO, $_GET["PrezzoEnd"]) === 1) ? $_GET["PrezzoEnd"] : "";
    $dataStart = ( isset( $_GET["DataStart"] ) &&
                   preg_match( $_REG_DATA, $_GET["DataStart"]) === 1 ) ? $_GET["DataStart"] : "";
    $dataEnd = ( isset( $_GET["DataEnd"] ) &&
                   preg_match( $_REG_DATA, $_GET["DataEnd"]) === 1 ) ? $_GET["DataEnd"] : "";
```

```php
    if( $prezzoStart != "" ) $condition = "AND PA.Prezzo_Vendita >= $prezzoStart ";
    if( $prezzoEnd != "" ) $condition .= "AND PA.Prezzo_Vendita <= $prezzoEnd ";

    if( $dataStart != "" ) $condition .= "AND PA.Data >= STR_TO_DATE( '$dataStart', '%Y-%m-%d' ) ";
    if( $dataEnd != "" ) $condition .= "AND PA.Data <= STR_TO_DATE( '$dataEnd', '%Y-%m-%d' ) ";

    $condition .= "ORDER BY ";

    if( isset( $_GET["PrezzoOrder"] ) )
    {
        $order = $_GET["PrezzoOrder"];
        $condition .= "PA.Prezzo_Vendita $order, ";
    }

    $condition .= "PA.Cliente_Registrato ASC";

    $ris_acquisti = mysql_query( "Select PA.Cliente_Registrato, PA.Prodotto, P.Marca,
                                        PA.Quantita, PA.Prezzo_Vendita, PA.Data
                                        From Prodotto P, ProdottiAcquisto PA
                                        Where PA.Prodotto=P.Modello
                                        $condition;", ConnectDB() );
    $color = "#CFCFCF";

    while( $row = mysql_fetch_row( $ris_acquisti ) )
    {
        echo "<tr bgcolor=$color>";
        echo "<td width=200px> $row[0] </td>";
        echo "<td width=170px> $row[1] </td>";
        echo "<td width=170px> $row[2] </td>";
        echo "<td width=100px> $row[3] </td>";
        echo "<td width=100px> $row[4] </td>";
        echo "<td> $row[5] </td>";
        echo "</tr>";
        $color = ( $color == "#CFCFCF" ) ? "#F3F3F3" : "#CFCFCF";
    }

    mysql_query( "drop view if exists AcquistiCliente;", ConnectDB() );
    mysql_query( "drop view if exists Prodotti;", ConnectDB() );

    CloseTable();
?>
```

8. Note

- Nella sezione Acquisti vi sono due aspetti da chiarire:
  - In fase di selezione del Cliente Registrato, è possibile anche non inserire alcun valore. In tal caso viene considerato un cliente generale che va a rappresentare tutti i clienti non abituali frequentanti il negozio. Nel database questo cliente è storicizzato con i valori: Codice_Fiscale='0' e i resto dei campi sono a NULL. Tutti gli acquisti effettuati da questo cliente fittizio non vengono considerati nelle statistiche riguardanti i clienti registrati.
  - In fase di selezione del prodotto venduto, oltre ai campi Prodotto e Quantità, vi è un ulteriore campo Prezzo Vendita che è opzionale. Se viene lasciato vuoto, come prezzo viene passato il valore corrispondente presente nella tabella prodotti, altrimenti può essere cambiato manualmente. Tale modifica è limitata solo a quell'acquisto.

- Nella sezione Ricerca è possibile visualizzare i prodotti all'interno del negozio per:
  - Categoria, selezionandola dal menu a sinistra
  - Sotto_Categoria, selezionandola dal menu a tendina della rispettiva Categoria
  - Modello, scrivendo il modello del prodotto che si vuole ricercare
  - Univoco prodotto, ma prima è necessario selezionare a quale fornitore è riferito