# SUPERMERCATO

Troiano Franco matricola 178960
Stringini Davide matricola 179421

# Indice:

# 1.Requisiti:

**Problema:**

Si vuole realizzare una base di dati per l'organizzazione di un supermercato.
Sebbene quanto richiesto sia stato semplificato e non copra interamente tutti gli aspetti di gestione, il sistema realizzato rappresenterà una base reale e significativa.

Il supermercato è strutturato in una serie di reparti. Ogni reparto ha un nome, un responsabile e una serie di impiegati, ognuno con una mansione specifica. Di ogni impiegato vogliamo memorizzare tutti i principali dati anagrafici (nome, indirizzo, telefono, codice fiscale, data di nascita,...) oltre alla data di assunzione.
Ogni impiegato lavora in un determinato reparto ed ha un particolare livello (che ne determina lo stipendio insieme all'anzianità di servizio).
Il supermercato mette in vendita una serie di prodotti, dei quali vogliamo memorizzare nome, genere ("preparazione alimentare", "prodotto di profumeria",...), il prezzo al pubblico, il tempo massimo entro il quale il prodotto deve essere venduto (ad es. se il prodotto è deperibile, il tempo potrebbe essere di solo qualche giorno), il reparto in cui è posto in vendita, la quantità di prodotto presente in magazzino e la soglia minima (determinata dall'esperienza, o da un'apposita interrogazione) al di sotto della quale il prodotto va riordinato. A
ciascun prodotto viene inoltre assegnato un codice interno. Alcuni prodotti non sono acquistati direttamente dai fornitori, ma "assemblati" o "preparati" direttamente nel supermercato (ad esempio, delle confezioni regalo di profumi in offerta speciale, o un piatto precotto) a partire da una serie di materie prime (che a loro volta sono prodotti disponibili nel magazzino). Per questi prodotti "composti" vogliamo conoscere anche tutti i prodotti "ingredienti" e la quantità di essi necessaria alla loro preparazione.
La base di dati contiene una lista dei fornitori a cui il supermercato fa riferimento.
Per ogni fornitore vogliamo conoscere la ragione sociale, la partita IVA, la modalità di pagamento richiesta (es. bonifico a 60 giorni), oltre all'indirizzo e al recapito telefonico e di fax. Ogni fornitore può fornire, a un particolare prezzo, uno o più dei prodotti venduti dal supermercato, identificandoli con un codice interno (diverso da quello usato dal magazzino del supermercato!) da indicare negli ordini.
Per quel che riguarda i clienti del supermercato, alcuni di essi possono essere "fidelizzati" perché hanno richiesto una speciale tessera, che permette loro di accumulare punti con gli acquisti e ottenere sconti o regali.
I clienti con tessera sono registrati nella base di dati con i loro dati anagrafici e il numero di tessera, e a ciascuno è associato il numero di punti correntemente accumulati.
Per semplificare, supponiamo che i "premi" delle raccolte punti siano a loro volta prodotti in vendita nel supermercato. Per ciascun premio, oltre al prodotto associato, vogliamo conoscere il numero di punti necessari ad ottenerlo. I punti si ottengono acquistando particolari prodotti. E' necessario mantenere una lista dei prodotti che partecipano alla raccolta punti, unitamente al numero di punti ottenibili con il loro acquisto. Ogni volta che il cliente ritira un premio, vengono scalati i corrispondenti punti dalla sua tessera.
I registratori di cassa del nostro supermercato sono direttamente interfacciati con la base i dati (come succede molto spesso oggigiorno). Ogni volta che un prodotto viene venduto, la

base di dati registra i dati della vendita (numero di scontrino, prodotto venduto, prezzo al quale è stato venduto) unitamente al codice del cliente al quale è stato venduto, se questo è titolare di una tessera.
Contemporaneamente, vengono aggiornate la disponibilità del prodotto in magazzino e il numero di punti del cliente (se il prodotto in questione dava diritto a dei punti-raccolta).

**Le operazioni previste sulla base di dati sono:**

1. Fornire le istruzioni per la creazione del DB e degli oggetti che lo costituiscono.
2. Per ogni relazione individuata, fornire le istruzioni di inserimento, modifica ed eliminazione delle istanze.
3. Modifica del responsabile e degli impiegati di un reparto;
4. Determinazione delle vendite per un reparto in un particolare periodo;
5. Determinazione dei prodotti più venduti in un determinato reparto.
6. Modifica del prezzo di un prodotto;
7. Modifica dei dati riguardanti le scorte di prodotto disponibili;
8. Per i prodotti “composti”, specifica degli “ingredienti” e delle quantità necessarie alla preparazione;
9. Inclusione o esclusione di un prodotto dalla raccolta punti e indicazione del numero di punti raccolta forniti dal prodotto;
10. Determinazione dei prodotti sotto scorta;
11. Lista dei fornitori dai quali un determinato prodotto può essere acquistato, ordinati in base al prezzo richiesto;
12. Modifica del numero di punti necessari ad ottenere un determinato premio;
13. Verifica dei premi attualmente disponibili.
14. Inserimento/Modifica dei prodotti forniti da un fornitore.
15. Modifica del reparto di assegnazione di un impiegato e del suo livello.
16. Modifica del numero di punti assegnati al cliente;
17. Determinazione dei premi a cui un cliente ha diritto;
18. Ritiro di un premio da parte di un cliente;
19. Determinazione dei prodotti più acquistati da un cliente;
20. Determinazione della spesa totale effettuata da un cliente in un determinato periodo.

# 2.Introduzione:

Il problema in esame, come descritto dalla consegna sopra, consiste di realizzare una base di dati per un supermercato, noi cercheremo di seguire una linea molto vicina alla realtà facendo opportune assunzioni nei punti in cui il testo della consegna è: non chiaro, non esaustivo, vago o superficiale.
Quindi abbiamo eseguito le dovute disambiguazioni del testo per poi passare alla progettazione concettuale.
Abbiamo quindi evidenziato nel testo i concetti chiave e quelli che secondo noi hanno più rilevanza usando una strategia detta “mista”, cioè che consente di usare le strategie “Bottom-up”, “Inside-out”, “Top-down”, individuando per prima cosa i concetti più importanti e man man scendendo sempre più affondo nel particolare.
Successivamente tenendo conto di tutti i concetti e considerazioni, siamo passati alla realizzazione di un Modello E-R indicando: Entità con attributi e chiavi primarie e le relazioni con rispettivi attributi e cardinalità.
Ultimata così la progettazione concettuale siamo passati alla progettazione logica addentrandoci sempre più a fondo nei dettagli della base di dati. Abbiamo quindi proseguito con un 'ottimizzazione del Modello E-R analizzando ridondanze,eliminando le generalizzazioni, accorpando o partizionando entità e relazioni dove necessario e scegliendo in maniera definitiva le chiavi primarie e le chiavi esterne.

# 3.Evidenziazione dei concetti rilevanti nel testo:

Nell'analisi dei requisiti abbiamo evidenziato i concetti che a parer nostro sono più rilevanti, e in corsivo delle frasi o parole candidate ad essere attributi.

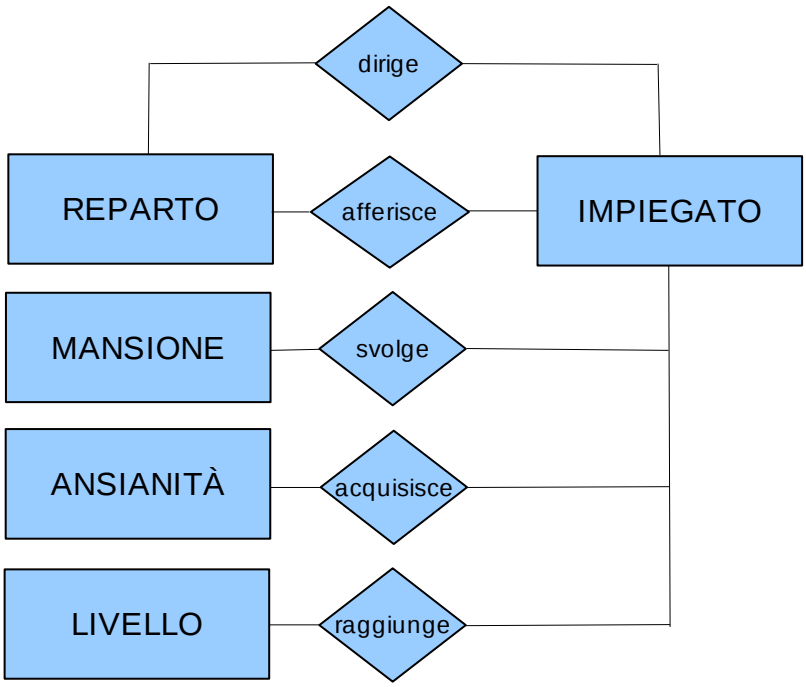
Il supermercato è strutturato in una serie di reparti. Ogni reparto ha un *nome*, un *responsabile* e una serie di impiegati, ognuno con una mansione specifica. Di ogni impiegato vogliamo memorizzare tutti i principali *dati anagrafici* (nome, indirizzo, telefono, codice fiscale, data di nascita,...) oltre alla data di assunzione. Ogni impiegato *lavora in un determinato reparto ed ha un particolare livello* (che ne determina lo stipendio insieme all’anzianità di servizio).
Il supermercato mette in vendita una serie di prodotti, dei quali vogliamo memorizzare *nome*, *genere* ("preparazione alimentare", "prodotto di profumeria",...), il *prezzo* al pubblico, il *tempo massimo entro il quale il prodotto deve essere venduto* (ad es. se il prodotto è deperibile, il tempo potrebbe essere di solo qualche giorno), *il reparto in cui è posto in vendita*, la *quantità di prodotto presente in magazzino* e la *soglia minima* (determinata dall’esperienza, o da un’apposita interrogazione) al di sotto della quale il prodotto va riordinato. A ciascun prodotto viene inoltre assegnato un *codice interno*. Alcuni prodotti non sono acquistati direttamente dai fornitori, ma “assemblati” o “preparati” direttamente nel supermercato (ad esempio, delle confezioni regalo di profumi in offerta speciale, o un piatto precotto) a partire da una serie di materie prime (che a loro volta sono prodotti disponibili nel magazzino). Per questi prodotti “composti” *vogliamo conoscere anche tutti i prodotti “ingredienti” e la quantità di essi necessaria alla loro preparazione.*

La base di dati contiene una lista dei fornitori a cui il supermercato fa riferimento.
Per ogni fornitore vogliamo conoscere la *ragione sociale*, la *partita IVA*, la *modalità di pagamento* richiesta (es. bonifico a 60 giorni), oltre *all'indirizzo* e al *recapito telefonico e di fax*. Ogni fornitore può fornire, a un particolare prezzo, uno o più dei prodotti venduti dal supermercato, identificandoli con un *codice interno* (diverso da quello usato dal magazzino del supermercato!) da indicare negli ordini.
Per quel che riguarda i clienti del supermercato, alcuni di essi possono essere "fidelizzati" perché hanno richiesto una speciale tessera, che permette loro di accumulare *punti* con gli acquisti e ottenere sconti o regali. I clienti con tessera sono registrati nella base di dati con i loro *dati anagrafici* e il *numero di tessera*, e a ciascuno è associato il *numero di punti correntemente accumulati.*
Per semplificare, supponiamo che i "premi" delle raccolte punti siano a loro volta prodotti in vendita nel supermercato. Per ciascun premio, oltre al prodotto associato, vogliamo conoscere il *numero di punti necessari ad ottenerlo*. I punti si ottengono acquistando particolari prodotti. E' necessario mantenere una lista dei prodotti che partecipano alla raccolta punti, unitamente al *numero di punti ottenibili* con il loro acquisto. Ogni volta che il cliente ritira un premio, vengono scalati i corrispondenti punti dalla sua tessera.
I registratori di cassa del nostro supermercato sono direttamente interfacciati con la base i dati (come succede molto spesso oggigiorno). Ogni volta che un prodotto viene venduto, la base di dati registra i *dati della vendita* (numero di scontrino, prodotto venduto, prezzo al quale è stato venduto) unitamente al *codice del cliente* al quale è stato venduto, se questo è titolare di una tessera. Contemporaneamente, vengono aggiornate la disponibilità del prodotto in magazzino e il numero di punti del cliente (se il prodotto in questione dava diritto a dei punti-raccolta).

# 4.Ristrutturazione dei requisiti:

1. **Reparto**: ogni reparto ha un nome , un responsabile e una serie di **impiegati**.

2. **Impiegato**: ognuno con una mansione specifica. Di ogni impiegato vogliamo memorizzare i dati anagrafici , data di assunzione. Ogni impiegato lavora in un determinato **reparto** ed ha un particolare livello (che ne determina lo stipendio insieme all'anzianità di servizio).
   Abbiamo deciso di inserire delle entità di appoggio per la mansione, il livello e l'anzianità dell'impiegato così da avere ben chiara la situazione con relative relazioni. Con l'inserimento di queste entità di appoggio abbiamo: una lista di reparti, di mansioni, livelli prestabiliti e anzianità. Quindi esemplifichiamo molte operazioni che sarebbero state molto difficili.

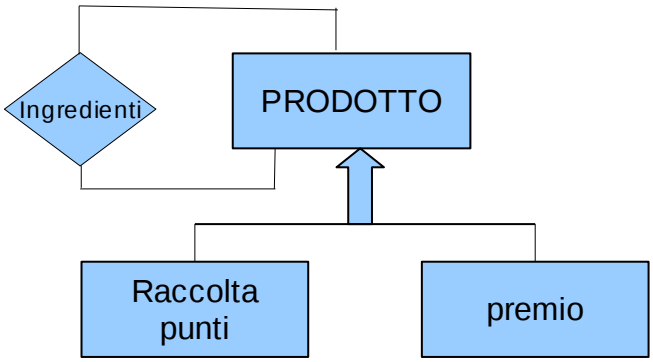



3. **Prodotto**: il supermercato mette in vendita una serie di prodotti , dei quali vogliamo memorizzare nome, genere , prezzo al pubblico , il tempo massimo entro il quale il prodotto deve essere venduto, il **reparto** in cui è posto in vendita, la quantità di prodotto presente in magazzino e la soglia minima al di sotto della quale il prodotto va riordinato. A ciascun prodotto viene associato un codice interno.

4. **Prodotto assemblato**: alcuni prodotti non sono acquistati direttamente dai fornitori ma "*assemblati*" o "*preparati*" direttamente nel supermercato a partire da una serie di materie prime (che a loro volta sono prodotti disponibili in magazzino). Per questi prodotti composti vogliamo conoscere anche tutti i prodotti "*ingredienti*" e la quantità di essi necessari alla loro preparazione.

5. **Premi**: sono a loro volta **prodotti** in vendita nel supermercato. Per ciascun premio , oltre al **prodotto** associato , vogliamo conoscere il numero di punti necessari per ottenerlo. I punti si ottengono acquistando particolari prodotti. É necessario mantenere una lista dei prodotti che partecipano alla raccolta punti , unitamente al numero di punti ottenibili con il loro acquisto. Ogni volta che un cliente ritira un premio, vengono scalati i corrispondenti punti dalla sua tessera.

6. **Raccolta punti**:i "premi" delle raccolte punti siano a loro volta prodotti in vendita nel supermercato. I punti si ottengono acquistando particolari prodotti. E' necessario mantenere una lista dei prodotti che partecipano alla raccolta punti, unitamente al numero di punti ottenibili con il loro acquisto.



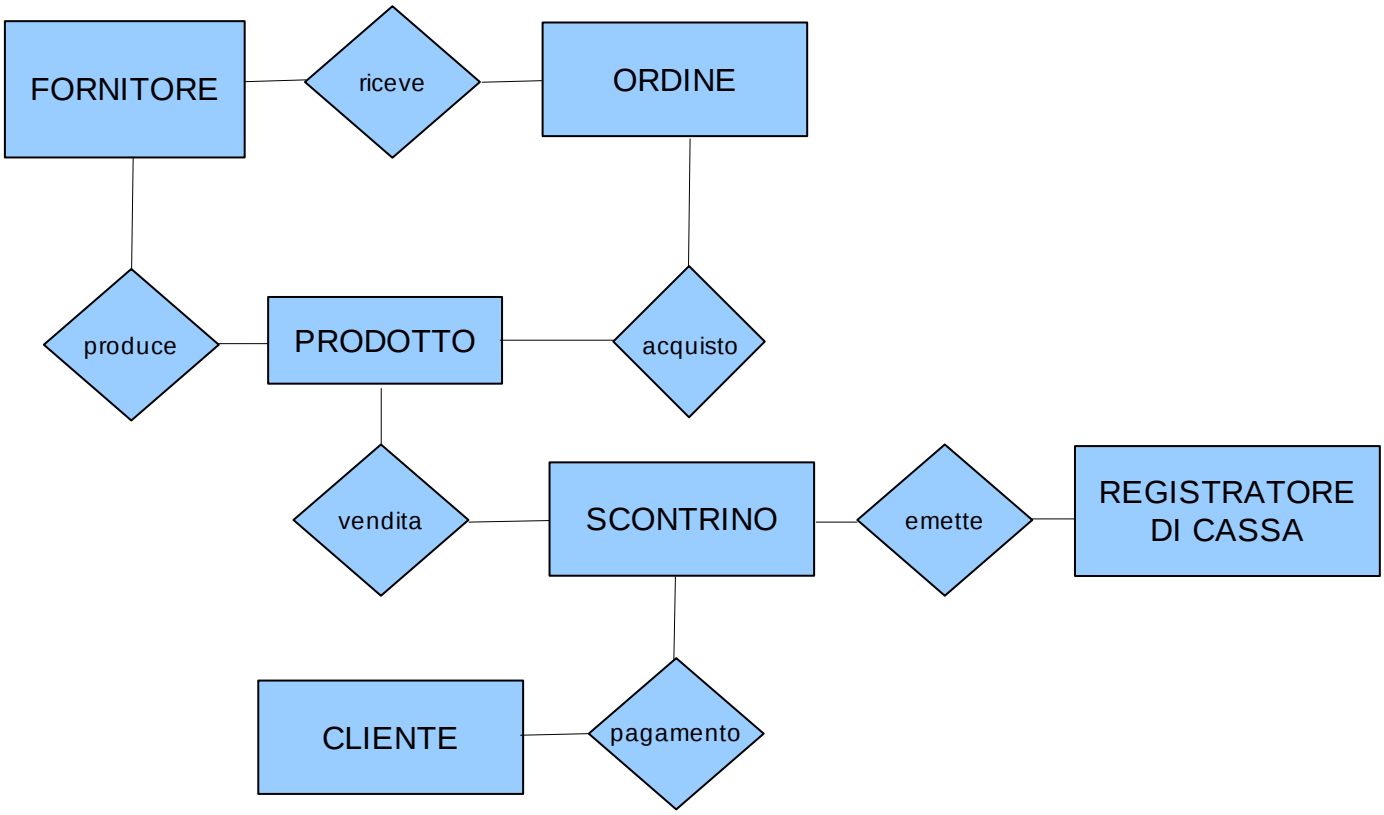

7. **Fornitori**: La base di dati contiene una lista di fornitori a cui il supermercato fa riferimento.
   Per ogni fornitore vogliamo conoscere la ragione sociale , la partita IVA , la modalitàdi pagamento richiesta , oltre all'indirizzo e al recapito telefonico e di fax. Ogni fornitore può fornire ad un particolare prezzo , uno o più **prodotti** venduti nel supermercato, identificandoli con un codice interno (diverso da quello del magazzino) da indicare negli ordini.

8. **Ordine**:Ogni fornitore può fornire, a un particolare prezzo, uno o più dei prodotti venduti dal supermercato, identificandoli con un codice interno (diverso da quello usato dal magazzino del supermercato!) da indicare negli ordini.

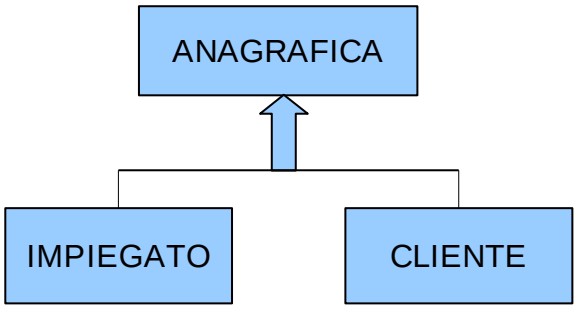
9. **Clienti**: alcuni di essi possono essere "*fidelizzati*" perché hanno richiesto una speciale tessera , che permette loro di accumulare punti con gli acquisti e ottenere sconti o regali. I clienti con tessera sono registrati nella base di dati con i loro dati anagrafici e il numero di tessera , e a ciascuno è associato il numero di punti correntemente accumulati.(da adesso in poi con clienti intendiamo quelli fidelizzati)

10. **Registratori di cassa**: sono direttamente interfacciati con la base di dati. Ogni volta che un **prodotto** viene venduto , la base di dati registra i dati della vendita , numero di scontrino , prodotto venduto , prezzo al quale è stato venduto , unitamente al codice del **cliente** al quale è stato venduto , se questo è titolare di una tessera. Contemporaneamente , vengono aggiornate la disponibilità del prodotto in magazzino e il numero di punti del **cliente** (se il **prodotto** in questione dava diritto a dei punti-raccolta).

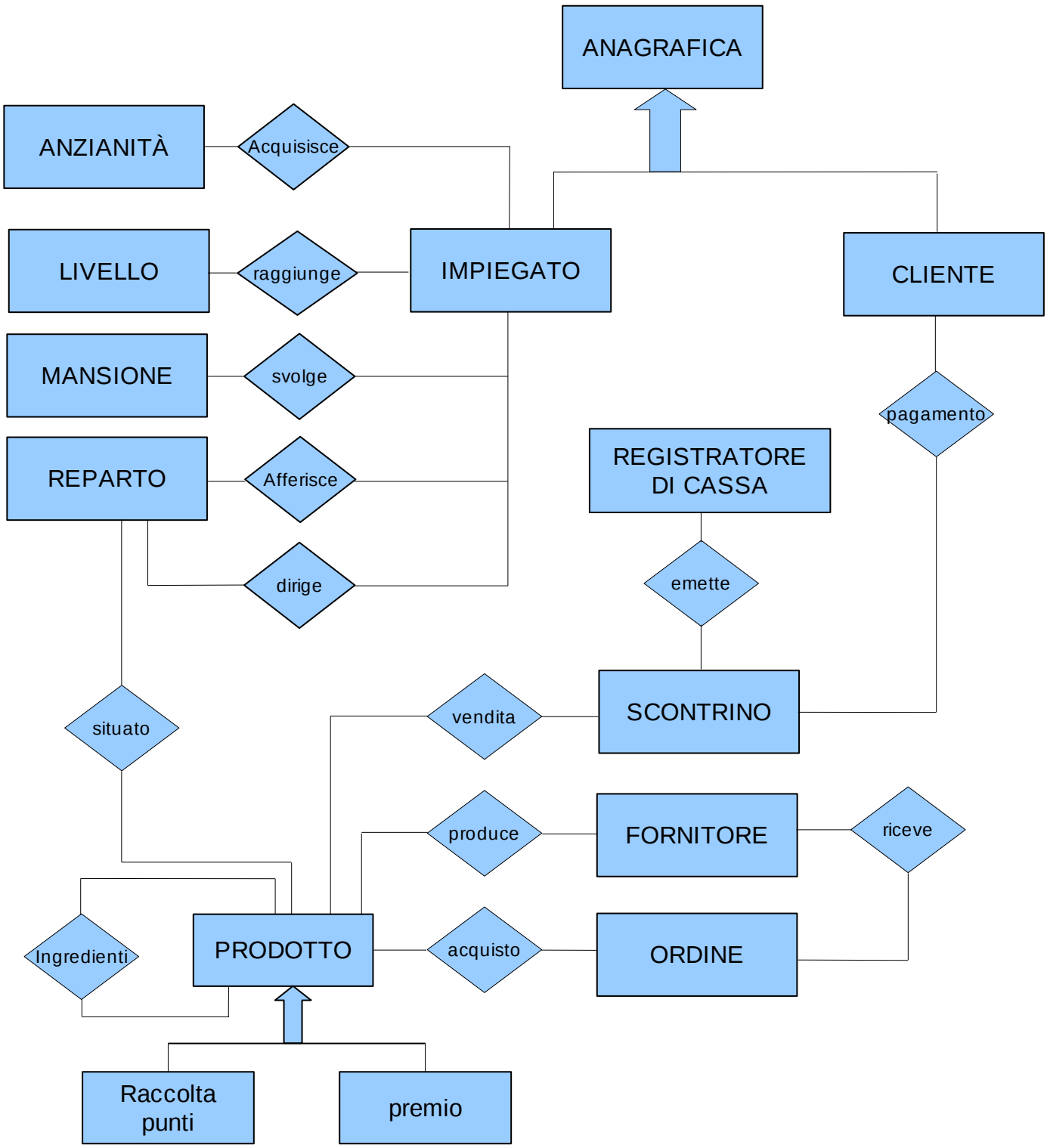
11. **Scontrino**:Ogni volta che un prodotto viene venduto, la base di dati registra i dati della vendita (numero di scontrino, prodotto venduto, prezzo al quale è stato venduto) unitamente al codice del cliente al quale è stato venduto, se questo è titolare di una tessera.




12. **Dati anagrafici**: Di ogni impiegato vogliamo memorizzare tutti i principali dati anagrafici (nome, indirizzo, telefono, codice fiscale, data di nascita,...) I clienti con tessera sono registrati nella base di dati con i loro dati anagrafici e...
abbiamo inserito questo nuova entità per registrare tutti i dati anagrafici delle persone, cioè in questo caso impiegati e clienti. Così da alleggerire anche le entità impiegato e cliente.



# 5.Schema concettuale con modello E-R:



# 6.Dettaglio oggetti:

- **ENTITÀ**: ***Reparto***
  **ID**: nome del reparto;
  **ATTRIBUTI**: nome del reparto;
  **ASSOCIAZIONI**: afferenza di impiegati al dato reparto , impiegato responsabile del reparto;
  **OPERAZIONI**: modifica del responsabile del reparto.
  Conviene mantenere l'entità reparto in quanto se eliminiamo questa entità dovremmo inserire ad ogni impiegato il nome del reparto in cui lavora ed avremo così delle ridondanze ed errori. In questo modo invece abbiamo una lista di reparti come è giusto che sia e se cambiamo il nome ad un reparto non dobbiamo modificare null'altro.
- **ENTITÀ**: ***anagrafica***;
  **ID**: Codice fiscale;
  **ATTRIBUTI**: nome, cognome, indirizzo, data di nascita, telefono, tipo(per distinguere tra impiegati e clienti);
  **ASSOCIAZIONI**: dati_impiegati, dati_clienti rispettivamente tra le entità cliente e impiegato;
  **OPERAZIONI**: cambio numero di telefono e\o indirizzo.
- **ENTITÀ**: ***Impiegato***;
  **ID**: Codice fiscale;(chiave esterna nella tabella *anagrafica*)
  **ATTRIBUTI**: data assunzione, stipendio;
  **ASSOCIAZIONI**: afferenza degli impiegati al reparto, impiegato responsabile del reparto, svolge una data mansione, acquisisce anzianità, raggiunge un dato livello, dati_impiegato con anagrafica;
  **OPERAZIONI**: cambio di afferenza di un impiegato ad un reparto , esempio se un impiegato cambia reparto, modifica del suo livello, aggiornamento della sua anzianità e quindi aggiornamento dello stipendio.
- **ENTITÀ**: ***mansione***;
  **ID**: mansione;
  **ATTRIBUTI**: mansione;
  **ASSOCIAZIONI**: svolge con impiegato;
  **OPERAZIONI**:modifica della mansione.
- **ENTITÀ**: ***anzianità***;
  **ID**: anzianità;
  **ATTRIBUTI**: anzianità, plus_stipendio;
  **ASSOCIAZIONI**: acquisisce con impiegato;
  **OPERAZIONI**:modifica del plus_stipendio (cioè la quantità di soldi che vanno aggiunti allo stipendio relativo al suo livello).
- **ENTITÀ**: ***livello***;
  **ID**: livello;
  **ATTRIBUTI**: livello, stipendio_base;
  **ASSOCIAZIONI**: raggiunge con impiegato;
  **OPERAZIONI**:modifica dello stipendio_base.

- **ENTITÀ**: ***Prodotto***;
  **ID**: codice prodotto;
  **ATTRIBUTI**:codice prodotto, nome prodotto, prezzo_vendità, scadenza, genere, numeri di pezzi(disponibili in magazzino), soglia minima (di pezzi da avere in magazzino), punti_conferiti, punti_necessari;
  **ASSOCIAZIONI**: vendita con l'entità scontrino, produce con fornitore, acquisto con ordine, situato con reparto;
  **OPERAZIONI**: modifica prezzo prodotto, determinazione delle vendite dei prodotti di un dato reparto in un periodo di tempo e del prodotto più venduto del reparto, modifica dei dati riguardanti le scorte di prodotto disponibili, inclusione o esclusione di un dato prodotto dalla raccolta punti e indicazione dei punti-raccolta conferiti dall'acquisto del prodotto,determinazione dei prodotti sotto scorta, vendita di un prodotto, determinazione della spesa totale effettuata da un cliente in un determinato periodo. Determinazione dei prodotti più acquistati da un cliente, ritiro di un premio da parte di un cliente.
- **ENTITÀ**: ***Cliente*** ("fidelizzato");
  **ID**: numero tessera;
  **ATTRIBUTI**: numero tessera, punti;
  **ASSOCIAZIONI**: pagamento di scontrini, dati_cliente con anagrafica;
  **OPERAZIONI**:ritira un premio che consiste nel decremento dei punti e nel decremento del numero di pezzi del prodotto prelevato come premio, modifica del numero di punti assegnati al cliente, determinazione dei premi a cui un cliente ha diritto.
- **ENTITÀ**: ***Fornitore***;
  **ID**: partita IVA;
  **ATTRIBUTI**: ragione sociale, partita IVA, pagamento, indirizzo, telefono, fax;
  **ASSOCIAZIONI**: produce con l'entità prodotto, riceve con l'entità ordine;
  **OPERAZIONI**:modifica dei suoi attributi, fornire prodotti.
- **ENTITÀ**: ***ordine***;
  **ID**: numero_ordine;
  **ATTRIBUTI**: numero_ordine, effettuato;
  **ASSOCIAZIONI**: acquisto con prodotto,riceve con fornitore;
  **OPERAZIONI**:ordine di un prodotto.
  Abbiamo inserito un attributo effettuato che ci consentirà che ci servirà in fase implementativa per attivare delle procedure.
- **ENTITÀ**: ***scontrino***;
  **ID**: numero_scontrino;
  **ATTRIBUTI**: numero_scontrino,data, tessera_cliente, pagato, totale;
  **ASSOCIAZIONI**:vendita con prodotto, emesso con registratore di cassa, pagamento con cliente
  **OPERAZIONI**:registrazione di tutti i dati di vendita del supermercato.
- **ENTITÀ**: ***registratore di cassa***;
  **ID**: numero_cassa;
  **ATTRIBUTI**: numero_cassa, nome_operatore;
  **ASSOCIAZIONI**: emette con scontrino;
  **OPERAZIONI**: emissione di scontrini..

- **RELAZIONE**: ***afferisce***;
  **TRA**: impiegato e reparto;
  **CARDINALITÀ**: un impiegato può afferire ad un solo reparto, un reparto può avere più impiegati ma almeno uno.
  Manteniamo questa lista in cui ogni impiegato è legato ad un solo reparto così anche se ad esempio cambiamo nome al reparto non dobbiamo cambiare a tutti gli impiegati il relativo reparto di afferenza quindi mantenendo questa relazione evitiamo ridondanze ed errori.
- **RELAZIONE**: ***dirige***;
  **TRA**: impiegato e reparto;
  **CARDINALITÀ**: un impiegato può essere responsabile di uno e un solo reparto,, un reparto può avere uno e un solo responsabile.
  Con questa relazione facciamo si che un reparto abbia un solo responsabile e un impiegato non può essere responsabile di più reparti eliminando tutti gli errori che ne scaturiscono.
- **RELAZIONE**: ***svolge***;
  **TRA**: impiegato e mansione;
  **CARDINALITÀ**: un impiegato può avere una sola mansione una mansione (come classe) può essere svolta da più impiegati.
- **RELAZIONE**: ***acquisisce***;
  **TRA**: impiegato e anzianità;
  **CARDINALITÀ**:un impiegato può avere una sola anzianità, una tale anzianità può essere acquisita da più impiegati.
- **RELAZIONE**: ***raggiunge***;
  **TRA**: impiegato e livello;
  **CARDINALITÀ**: un impiegato può raggiungere un solo livello, un dato livello può essere raggiunta da più impiegati.
- **RELAZIONE**: ***riceve***;
  **TRA**: fornitore e ordine;
  **CARDINALITÀ**: un fornitore può ricevere più ordini, un dato ordine può essere mandato ad un solo fornitore.
- **RELAZIONE**: ***produce***;
  **TRA**:fornitore e prodotto;
  **CARDINALITÀ**:un fornitore può produrre più prodotti, un dato prodotto può essere prodotto da più solo fornitore.
- **RELAZIONE**: ***situato***;
  **TRA**: prodotto e reparto;
  **CARDINALITÀ**: un prodotto può stare in un solo reparto, un reparto può avere più prodotti.
  Questa relazione fa si che: un prodotto si trovi in un unico reparto.
- **RELAZIONE**: ***assemblato***;
  **TRA**:tra prodotto e prodotto;
  **CARDINALITÀ**:un prodotto può avere da 2 a N ingredienti, mentre un prodotto può essere ingredienti di 0 o N prodotti.

- **RELAZIONE**: ***acquisto***;
  **TRA**:prodotto e ordine;
  **CARDINALITÀ**:un prodotto può essere acquistato dal supermercato da 0 a N volte,un ordine da 1 a N.
- **RELAZIONE**: ***vendita***;
  **TRA**:prodotto e scontrino;
  **CARDINALITÀ**:un prodotto può essere venduto da 0 a N volte, uno scontrino comprende da minimo 1 ad N prodotti.
- **RELAZIONE**: ***emette***;
  **TRA**:scontrino e registratore di cassa;
  **CARDINALITÀ**:uno scontrino può essere emesso da un solo registratore di cassa, mentre un registratore di cassa può emettere da 0 a N scontrini.
- **RELAZIONE**: ***pagamento***;
  **TRA**:scontrino e cliente;
  **CARDINALITÀ**:uno scontrino è pagato da un solo cliente, un cliente può pagare più scontrini da 0 a N.
- **RELAZIONE**: ***dati_impiegato****;*
  **TRA**:impiegato e anagrafica;
  **CARDINALITÀ**:un impiegato ha dati unici, in anagrafica possono o non possono esserci i dati di un tale impiegato.
- **RELAZIONE**: ***dati_cliente***;
  **TRA**:cliente e anagrafica;
  **CARDINALITÀ**:un cliente a dati unici, in anagrafica possono o meno esserci i dati di un dato cliente.

# 7.Formalizzazione dei vincoli:

**Regole di vincolo:**

**RV1** : nell'entità "Anagrafica" sono obligatori i campi "ID_CodiceFisc", "Nome", "Cognome", "Indirizzo", "DataNascita" e "Tipo"(che ci farà riconoscere se i dati sono di un impiegato o di un cliente);
**RV2** : nell'entità "Anagrafica" il campo "Tipo" appartiene al seguente dominio: "I" (impiegato) o "C" (cliente);
**RV3** : nell'entità "Livelli" si devono obbligatoriamente inserire i valori nei campi "ID_Livello" e "StipendioBase";
**RV4** : nell'entità "Anzianita" si devono obbligatoriamente inserire i valori nei campi "ID_Anzianita" e "StipendioAgg";
**RV5** : nell'entità "Mansioni" si devono obbligatoriamente inserire i valori nei campi "ID_Mansione" e "Descrizione";
**RV6** :nell'entità "Impiegati" si devono obbligatoriamente inserire i campi "ID_Impiegato", "ID_Reparto", "ID_Mansione", "ID_Livello","ID_Anzianita", "DataAssunzione", "Stipendio";
**RV7** : in "Impiegati" ci devono essere tuple relative a persone i cui dati sono registrati nell'entità "Anagrafica" che hanno come "Tipo" il record "I";
**RV8** : nell'entità "Impiegati" la data di assunzione non può essere successiva a quella odierna;
**RV9** : In "Reparti" bisogna obbligatoriamente inserire i valori nei campi "ID_Reparto" e "Nome";
**RV10** : Ogni reparto ha uno e un solo responsabile;
**RV11** : Due o più reparti non possono avere lo stesso responsabile;
**RV12** : nell'entità "Prodotti" vanno indicati i valori nei campi "ID_Prodotto","ID_Reparto", "Nome", "Genere", "PrezzoVendita", "Quantita", "Soglia", "Assemblato", "PuntiRaccolta" e "PuntiNecessari";
**RV13** : Assumiamo che non ci possono essere prodotti con nomi uguali;
**RV14** : La scadenza di un prodotto non può essere antecedente alla data odierna;
**RV15** : In "Prodotti" il campo "Assemblato" deve avere record appartenenti al dominio: "1" (si) e "0" (no);
**RV16** : I prodotti che non partecipano alla raccolta punti devono avere il campo "PuntiRaccolta" pari a 0;
**RV17** : I prodotti che non sono premi devono avere il campo "PuntiNecessari" pari a 0;
**RV18** : In "Prodotti_Assemblati" bisogna necessariamente indicare i campi "ID_Assemblato" e "ID_Ingrediente";
**RV19** : In "Prodotti_Assemblati" nel campo "ID_Assemblato" vanno inseriti soltanto record di prodotti con il campo "Assemblato" pari a "1";
**RV20** : In "Prodotti_Assemblati" nel campo "ID_Ingrediente" vanno inseriti soltanto record di prodotti con il campo "Assemblato" pari a "0"(assumiamo che un prodotto assemblato non possa avere come suoi ingredienti prodotti a loro volta assemblati);

**RV21** : nell'entità “Fornitori” si devono per forza indicare i campi “ID_PartitaIVA”, “RagioneSoc”, “Pagamento” e “Indirizzo”;
**RV22** : In “Prodotti_Forniti” si devono obbligatoriamente inserire i valori nei campi “ID_CodiceInt”, “ID_Fornitore”, “ID_Prodotto”, “PrezzoAcquisto”;
**RV23** : In “Ordini” bisogna inserire i valore nei campi “ID_Ordine”, “ID_Fornitore”, “Data”, “Consegnato”;
**RV24** : La data dell’ordine non può essere antecedente alla data corrente;
**RV25** : In “Dettaglio_Ordini” vanno necessariamente indicati i campi “ID_Ordine”, “ID_CodiceInt”, “ID_Prodotto”, “Quantita” e “PrezzoAcquisto”;
**RV26** : Gli ordini vanno fatti su prodotti che i fornitori vendono;
**RV27** : In “Cliente” bisogna indicare i campi “ID_Cliente”, “ID_Tessera”, “Data” e “Punti”;
**RV28** : In “Cliente” ci devono essere record relativi a persone i cui dati sono registrati in “Anagrafica” che hanno come “Tipo” il record “F”;
**RV29** : La data di registrazione di un cliente fidelizzato non può essere antecedente la data corrente;
**RV30** : In “RegistratoreDiCassa” vanno necessariamente inseriti i valori nei campi “ID_Cassa” e “Nome”;
**RV31** : In “Scontrino” vanno indicati obbligatoriamente i campi “ID_Scontrino”, “ID_Cassa”, “Data”, “Totale” e “Pagato”;
**RV32** : La data dello scontrino non può essere antecedente alla data corrente;
**RV33** : In “Dettaglio_Scontrini” si devono necessariamente inserire i record nei campi “ID_Scontrino”, “ID_Prodotto”, “Quantita” e “PrezzoVen”.

**Regole di derivazione:**
**RD1** : L’Anzianita si ricava facendo la differenza tra la data corrente e la data di assunzione;
**RD2** : L’importo dello stipendio si ottiene sommando StipendioBase(Livello) con Plus_Stipendio(Anzianita);
**RD3** : Il totale dello scontrino si calcola sommando i vari importi dei prodotti;
**RD4** : Gli importi dei prodotti si calcolano moltiplicando quantità per prezzo;
**RD5** : Le quantità di prodotti disponibili in magazzino vengono aggiornate non appena l’ordine viene consegnato;
**RD6** : Le quantità di prodotti disponibili in magazzino vengono aggiornate non appena avviene il pagamento dello scontrino;

**RD7** : I punti della tessera di un fidelizzato vengono aggiunti ogni volta che paga lo scontrino.
**RD8** : I punti della tessera di un fidelizzato vengono sottratti ogni volta che ritira un premio.
**RD9:** un prodotto assemblato deve avere almeno 2 ingredienti e non può avere come ingrediente se stesso;

# 8.Qualità dello schema concettuale:

Lo schema concettuale risulta:
CORRETTO: utilizza correttamente i costrutti dei diagrammi ER e non si sono individuati errori sintattici o semantici.
COMPLETO: il diagramma copre tutti i requisiti analizzati e tutte le operazioni definite.
LEGGIBILE: i nomi dei concetti espressi sono stati derivati direttamente dai requisiti cercando quindi di usare un lessico consistente. Anche da un punto di vista grafico il diagramma risulta esteticamente chiaro.
MINIMALE: non esistono cicli né altre ripetizioni.

# 9.Progettazione logica:
# stima dei volumi:

(consideriamo un supermercato di medie dimensioni)

| Concetto | Tipo | Volume |
|---|---|---|
| Reparto | E | 20 |
| Impiegati | E | 1'000 |
| Cliente | E | 50'000 |
| Anagrafica | E | 51'000 |
| Mansione | E | 8 |
| Anzianità | E | 30 |
| Livello | E | 10 |
| Prodotto | E | 500'000 |
| Fornitore | E | 100 |
| Ordine | E | 50 / giorno |
| Scontrino | E | 10'000 / giorno |
| Registratore di cassa | E | 20 |
| Afferisce | R | 1'000 |
| Dirige | R | 20 |
| Svolge | R | 1'000 |
| Acquisisce | R | 1'000 |
| Raggiunge | R | 1'000 |
| Riceve | R | 10 / giorno |
| Produce | R | 500'000 |
| Situato | R | 500'000 |
| Assemblato | R | 10'000 |
| Acquisto | R | 100 / giorno |
| Vendita | R | 10'000 al giorno |
| Emette | R | 10'000 al giorno |
| Pagamento | R | 10'000 al giorno |
| Dati impiegati | R | 1'000 |
| Dati clienti | R | 50'000 |

# Caratteristiche operazioni, operazioni di base:

1. **impiegato**:
    - inserimento e cancellazione di un nuovo impiegato;
    - modifica del livello, mansione, anzianità, stipendio;
    - modifica del telefono;

2. **reparto**:
    - inserimento cancellazione di un reparto;
    - modifica del suo responsabile;

3. **prodotto**:
    - inserimento cancellazione di un prodotto;
    - modifica del prezzo, numero di pezzi, soglia minima, numero di punti conferiti, scadenza;

4. **premio**:
    - inclusione o esclusione di un prodotto da essere o meno premio;
    - modifica dei punti necessari per avere il premio;
    - oltre alle operazioni di prodotto;

5. **assemblato**:
    - modifica dei prodotti ingredienti, e delle quantità di prodotti necessarie;
    - oltre alle operazioni di prodotto;

6. **cliente**:
    - registrazione di un nuovo cliente;
    - aggiornamento dei punti del cliente;

7. **catalogo**:
    - inserimento , cancellazione prodotti dei fornitori nel catalogo;
    - modifica del prezzo;

8. **fornitore**:
    - inserimento , cancellazione nuovi fornitori;
    - modifica di pagamento, indirizzo, telefono, fax.

# operazioni specifiche:

9. **acquisto**:un cliente effettua un acquisto e vengono registrati data, nome del prodotto, prezzo, numero della tessera del cliente se è fidelizzato, numero dello scontrino, quantità di tale prodotto.
   Contemporaneamente viene decrementato il numero di prodotti disponibili (del prodotto acquistato), vengono sommati i punti alla tessera del cliente se fidelizzato e se il prodotto dava luogo a punti, successivamente viene controllato se il numero di pezzi disponibili del dato prodotto è minore della soglia minima se cosi è , si riordina il prodotto;

10. **Ordine**:un prodotto va ordinato, viene scelto da un catalogo che il supermercato ha a disposizione e lo si ordina in una quantità maggiore o uguale alla soglia minima del prodotto;

11. **ritiro premio**: un cliente fidelizzato può richiedere di ritira un premio per il quale occorre un numero di punti minore o uguale a quelli che il cliente possiede;

Le operazioni dalla 1 alla 8 costano un solo accesso a memoria in quanto trovano dove modificare o inserire l'informazione e la inseriscono/modificano. Le operazioni 9, 10 e 11 invece richiedono più accessi a memoria.
L'operazione **acquisto** registra i dati della vendita poi accede al numero di punti del cliente (se fidelizzato) e li aggiorna , poi aggiorna anche la disponibilità del prodotto e se è minore della soglia minima si riordina.
Quindi a seconda di che situazione abbiamo davanti ci possono essere vari costi di questa operazione.
L'operazione **ordine** in pratica è solo una somma al numero di pezzi di un dato prodotto.
L'operazione **ritiro premio** consiste nel decremento dei punti del cliente e decrementare la disponibilità di quel prodotto che in questo caso è anche premio 2 accessi in memoria per modificare i suddetti parametri.

# 10.Ristrutturazione dello schema E-R

Siamo passati dunque alla fase di ristrutturazione dello schema concettuale sopra descritto andando a visionare i dettagli di tutti gli oggetti definendo chiavi primarie, eliminando le generalizzazioni e analizzando le ridondanze presenti.

## Risoluzione delle generalizzazioni:

(che non sono supportate) abbiamo risolto nel seguente modo (già intuibile dal dettaglio degli oggetti):

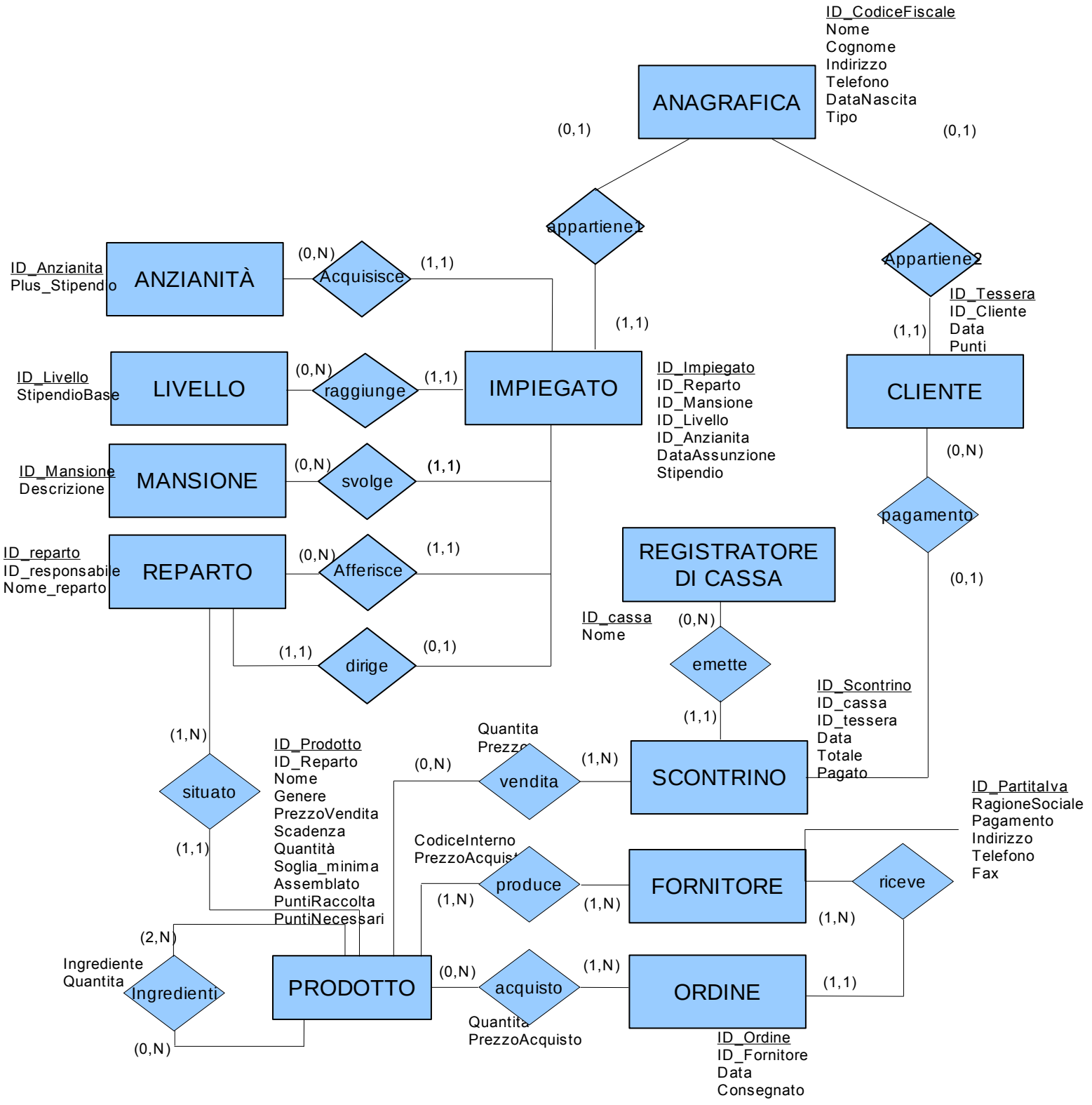
- La generalizzazione di anagrafica con **cliente** e **impiegato** come figli l'abbiamo risolta tramite 2 relazioni diverse: **dati_impiegato**, **dati_cliente** che legano rispettivamente queste due entità con l'entità padre **anagrafica** alla quale abbiamo inserito un attributo supplementare "*tipo*" che ci permette di distinguere tra i dati di impiegati e clienti.
- Mentre la generalizzazione tra **prodotto premi** e prodotti che appartengono alla **raccolta punti** (cioè che danno luogo a punti al momento del loro acquisto) l'abbiamo risolta inserendo nell'entità padre i 2 attributi aggiuntivi dei figli, "*punti_conferiti"* (prodotti per la raccolta punti), "*punti_necessari*"(premi), quindi quando uno dei due ha un valore valido l'altro attributo aggiuntivo è nullo.

## Altri accorgimenti:

- Abbiamo inserito attributi aggiuntivi alle entità ordine l'attributo "*effettuato*" che ci permetterà di caricare, aggiornare il numero di pezzi del prodotto ordinato nel magazzino del supermercato (implementato tramite un trigger).

- Invece abbiamo inserito l'attributo "*pagato*" all'entità scontrino che ci servirà per aggiornare le quantità di prodotto presenti nel magazzino di quel dato prodotto(implementato tramite un trigger).

# 11.Schema E-R:


ID_CodiceFiscale
Nome
Cognome
Indirizzo
Telefono
DataNascita
Tipo
ANAGRAFICA
(0,1)
(0,1)
appartiene1
Appartiene2
ID_Anzianita
Plus_Stipendio
ANZIANITÀ
(0,N)
Acquisisce
(1,1)
(1,1)
ID_Tessera
ID_Cliente
Data
Punti
(1,1)
ID_Livello
StipendioBase
LIVELLO
(0,N)
raggiunge
(1,1)
IMPIEGATO
ID_Impiegato
ID_Reparto
ID_Mansione
ID_Livello
ID_Anzianita
DataAssunzione
Stipendio
CLIENTE
ID_Mansione
Descrizione
MANSIONE
(0,N)
svolge
(1,1)
(0,N)
pagamento
ID_reparto
ID_responsabile
Nome_reparto
REPARTO
(0,N)
Afferisce
(1,1)
REGISTRATORE
DI CASSA
(0,1)
ID_cassa
Nome
(0,N)
(1,1)
dirige
(0,1)
emette
(1,1)
ID_Scontrino
ID_cassa
ID_tessera
Data
Totale
Pagato
(1,N)
situato
(1,1)
ID_Prodotto
ID_Reparto
Nome
Genere
PrezzoVendita
Scadenza
Quantità
Soglia_minima
Assemblato
PuntiRaccolta
PuntiNecessari
Quantita
Prezzo
(0,N)
vendita
(1,N)
SCONTRINO
ID_Partitalva
RagioneSociale
Pagamento
Indirizzo
Telefono
Fax
CodiceInterno
PrezzoAcquist
produce
(1,N)
(1,N)
FORNITORE
riceve
(1,N)
(2,N)
Ingrediente
Quantita
Ingredienti
(0,N)
PRODOTTO
(0,N)
acquisto
(1,N)
ORDINE
(1,1)
Quantita
PrezzoAcquisto
ID_Ordine
ID_Fornitore
Data
Consegnato

# 12.Traduzione nel modello relazionale

Le chiavi primarie sono evidenziate in giallo mentre le chiavi esterne sono evidenziate in verde.

- **REPARTI** (ID_Reparto, ID_Responsabile, Nome_Reparto)
- **ANAGRAFICA** (ID_CodiceFiscale, Nome, Cognome, Indirizzo, Telefono, DataNascita, Tipo)
- **IMPIEGATI** (ID_Impiegato, ID_Reparto, ID_Mansione, ID_Livello, ID_Anzianita, DataAssunzione, Stipendio)
- **MANSIONI** (ID_Mansione, Descrizione)
- **LIVELLI** (ID_Livello, StipendioBase)
- **ANZIANITA** (ID_Anzianita, Plus_Stipendio)
- **PRODOTTI** (ID_Prodotto, ID_Reparto, Nome, Genere, PrezzoVendita, Scadenza, Quantita, Soglia_minima, Assemblato, PuntiRaccolta, PuntiNecessari)
- **PRODOTTI_ASSEMBLATI** (ID_Assemblato, ID_Ingrediente, Quantita)
- **FORNITORI** (ID_PartitaIva, RagioneSoc, Pagamento, Indirizzo, Telefono, Fax)
- **PRODOTTI_FORNITI** (ID_CodiceInterno, ID_Fornitore, ID_Prodotto, PrezzoAcquisto)
- **ORDINI** (ID_Ordine, ID_Fornitore, Data, Consegnato)
- **DETTAGLIO_ORDINI** (ID_Ordine, ID_CodiceInterno, ID_Prodotto, Quantita, PrezzoAcquisto)
- **CLIENTI** (ID_Tessera, ID_Cliente, Data, Punti)
- **REGISTRATORE_DI_CASSA** (ID_Cassa, Nome)
- **SCONTRINI** (ID_Scontrino, ID_Cassa, ID_Tessera, Data, Totale, Pagato)
- **DETTAGLIO_SCONTRINI** (ID_Scontrino, ID_Prodotto, Quantita, PrezzoVen)

# 13.Descrizione Tabelle

**ANAGRAFICA** = Tabella contenente i dati anagrafici relativi alle persone che possono essere Clienti o Impiegati del Supermercato
- “ID_CodiceFisc”:Codice univoco identificante il codice fiscale delle persone
- “Nome”:Nome della persona
- “Cognome”:Cognome della persona
- “Indirizzo”:Indirizzo della persona
- “Telefono”: Numero di telefono della persona
- “DataNascita”: Data di nascita della persona
- “Tipo”: Campo indicante se Impiegato o Cliente

**LIVELLO** = Tabella contenente i Livelli raggiungibili dagli impiegati con il corrispondente Stipendio Base
- “ID_Livello”:Codice univoco identificante il livello
- “StipendioBase”: Stipendio Base relativo al livello

**ANZIANITA** = Tabella contenente le Anzianita’ raggiungibili dagli impiegati con il corrispondente Stipendio Addizionale
- “ID_Anzianita”:Codice univoco identificante l’anzianita’ di servizio
- “StipendioAgg”: Stipendio Addizionale relativo all’anzianita’ di servizio

**MANSIONE** = Tabella contenente le Mansioni svolte dagli impiegati
- “ID_Mansione”:Codice univoco identificante la mansione svolta
- “Descrizione”: Descrizione della mansione

**IMPIEGATO** = Tabella che associa gli Impiegati all’Anagrafica, al Reparto, alla Mansione, al Livello e all’Anzianita’
- “ID_Impiegato”: FK -> Indicante l’anagrafica corrispondente
- “ID_Reparto”:FK -> Indicante il reparto associato
- “ID_Mansione”: FK -> Indicante la mansione svolta
- “ID_Livello”:FK -> Indicante il livello economico
- “ID_Anzianita”:FK -> Indicante l’anzianita’ di servizio
- “DataAssunzione”:Data di assunzione
- “Stipendio”: Campo derivato dalla somma StipendioBase+StipendioAgg

**REPARTO** = Tabella contente i Reparti ed il relativo Responsabile
- “ID_Reparto”:Codice univoco identificante il reparto
- “Nome” :Nome del reparto
- “ID_Responsabile”: FK -> Indicante l’impiegato corrispondente

**PRODOTTO** = Tabella contene i dati relativi Prodotti presenti nel Supermercato
"ID_Prodotto":Codice univoco identificante il prodotto generato da trigger
"ID_Reparto": FK -> Indicante in che reparto si trova il prodotto
"Nome":Nome del prodotto
"Genere": Genere del prodotto
"PrezzoVen": Prezzo di vendita del prodotto
"Scadenza": Data di scadenza del prodotto
"Quantita": Quantita' presente in magazzino
"Soglia" :Quantita' minima sotto la quale effettuare il riordino
"Assemblato":S indica che e' un prodotto Assemblato da piu' Prodotti N indica che e' un prodotto semplice
"PuntiRaccolta":0 indica che il prodotto non partecipa alla Raccolta Punti
x indica il numero di punti che il prodotto fa accumulare
"PuntiNecessari":0 indica che il prodotto non e' un Premio x indica il numero di punti necessari per riceverlo come Premio

**PRODOTTI_ASSEMBLATI** = Tabella che associa i prodotti Assemblati con gli ingredienti
necessari a comporlo con le relative quantita'
"ID_Assemblato":FK -> Indicante il codice del prodotto assemblato
"ID_Ingrediente": FK -> Indicante il prodotto ingrediente che compone il prodotto Assemblato
"Quantita":Indica la quantita di prodotto ingrediente necessaria

**FORNITORE** = Tabella contenente i dati relativi ai Fornitori del Supermercato
"ID_PartitaIva": Codice univoco identificante la partita iva del fornitore
"RagioneSoc": Ragione Sociale del fornitore
"Indirizzo":Indirizzo del fornitore
"Telefono": Numero di telefono del fornitore
"Fax": Numero di fax del fornitore
"Pagamento":Metodo di pagamento relativo al fornitore

**ORDINE** = Tabella che associa gli Ordini ai Fornitori
"ID_Ordine":Codice univoco generato da trigger indicante il numero d'ordine
"ID_Fornitore":FK -> Indicante il fornitore a cui si effettua l'ordine
"Data": Data di emissione dell'ordine
"Consegnato":Indica se l'ordine e' stato consegnato dal Fornitore

**DETTAGLIO_ORDINI** = Tabella che associa agli Ordini i Codici Interni dei prodotti dei Fornitori ai Codici dei prodotti del Supermercato, con le quantita' ordinate e prezzo di acquisto

"ID_Ordine":Codice univoco generato da trigger indicante il numero d'ordine
"ID_CodiceInt": FK -> Indicante il codice interno del fornitore relativo al prodotto
"ID_Prodotto": FK -> Indicante il codice del supermercato relativo al prodotto
"Quantita":Numero di pezzi
"PrezzoAcq":Prezzo di acquisto di ogni singolo pezzo

**CASSE** = Tabella contenente le Casse presenti nel supermercato
"ID_Cassa": Codice univoco identificante la cassa del supermercato
"Nome": Nome della cassa

**FIDELIZZATO** = Tabella che associa i Fidelizzati all'Anagrafica
"ID_Tessera":Codice univoco identificante il numero di tessera del fidelizzato
"ID_Fidelizzato": FK -> Indicante l'anagrafica relativa al fidelizzato
"Data": Data di fidelizzazione
"PuntiAttuali": Punti accumulati presenti nella tessera del fidelizzato

**SCONTRINO** = Tabella che contiene gli Scontrini effettuati dalle Casse
"ID_Scontrino":Codice univoco identificante il numero dello scontrino emesso da una particolare cassa
"ID_Cassa" :FK -> Indicante quale cassa ha emesso lo scontrino
"ID_Tessera": FK -> Indicante l'eventuale tessera del fidelizzato
"Data": Data di emissione dello scontrino
"Totale": Campo derivato dalla somma di tutte le Quantita*PrezzoVen della tabella Dettaglio_Scontrini per lo stesso ID_Scontrino
"Pagato": Indica se lo scontrino e' stato pagato

**DETTAGLIO_SCONTRINI** = Tabella che associa ad ogni Scontrino i prodotti acquistati
"ID_Scontrino":FK -> Indicante a quale scontrino si riferiscono i prodotti
"ID_Prodotto":FK -> Indicante quale prodotto e' stato acquistato
"Quantita": Numero pezzi acquistati
"PrezzoVen": Prezzo di vendita di ogni pezzo

# 14.Implementazione Progetto

1) Fornire le istruzioni per la creazione del DB e degli oggetti che lo costituiscono

```
/* Codice SQL per la Creazione ed uso dello Schema Supermercato */

CREATE DATABASE Supermercato;
USE Supermercato;

/* Codice SQL per la Creazione della Tabella Anagrafica */
DROP TABLE IF EXISTS Supermercato.Anagrafica;
CREATE TABLE Anagrafica (
 ID_CodiceFisc CHAR(16) NOT NULL UNIQUE,
 Nome VARCHAR(15) NOT NULL,
 Cognome VARCHAR(20) NOT NULL,
 Indirizzo VARCHAR(60) NOT NULL,
 Telefono VARCHAR(15),
 DataNascita DATE NOT NULL,
 Tipo ENUM('I','F') NOT NULL,
 PRIMARY KEY (ID_CodiceFisc)
);

/* Codice SQL per la Creazione della Tabella Livello */
DROP TABLE IF EXISTS Supermercato.Livello;
CREATE TABLE Livello (
 ID_Livello INTEGER(2) UNSIGNED NOT NULL UNIQUE,
 StipendioBase DECIMAL(8,2) UNSIGNED NOT NULL,
 PRIMARY KEY (ID_Livello)
);
/* Codice SQL per la Creazione della Tabella Anzianita */
DROP TABLE IF EXISTS Supermercato.Anzianita;
CREATE TABLE Anzianita (
 ID_Anzianita INTEGER(2) UNSIGNED NOT NULL UNIQUE,
 StipendioAgg DECIMAL(6,2) UNSIGNED NOT NULL,
 PRIMARY KEY (ID_Anzianita)
);

/* Codice SQL per la Creazione della Tabella Mansione */
DROP TABLE IF EXISTS Supermercato.Mansione;
CREATE TABLE Mansione (
 ID_Mansione INTEGER(2) UNSIGNED NOT NULL UNIQUE,
 Descrizione VARCHAR(15) NOT NULL,
 PRIMARY KEY (ID_Mansione)
);
```

```
/* Codice SQL per la Creazione della Tabella Impiegato */
DROP TABLE IF EXISTS Supermercato.Impiegato;
CREATE TABLE Impiegato (
 ID_Impiegato CHAR(16) NOT NULL UNIQUE,
 ID_Reparto INTEGER(3) UNSIGNED DEFAULT 0,
 ID_Mansione INTEGER(2) UNSIGNED NOT NULL DEFAULT 0,
 ID_Livello INTEGER(2) UNSIGNED NOT NULL DEFAULT 1,
 ID_Anzianita INTEGER(2) UNSIGNED NOT NULL DEFAULT 0,
 DataAssunzione DATE NOT NULL,
 Stipendio DECIMAL(8,2) UNSIGNED NOT NULL,
 PRIMARY KEY (ID_Impiegato),
 FOREIGN KEY (ID_Impiegato) REFERENCES Anagrafica(ID_CodiceFisc)
 ON DELETE CASCADE
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Mansione) REFERENCES Mansioni(ID_Mansione)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Livello) REFERENCES Livelli(ID_Livello)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Anzianita) REFERENCES Anzianita(ID_Anzianita)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);
/*
Trigger che prima dell'inserimento di un impiegato:
1) Verifica che i dati che si inseriscono siano effettivamente di un impiegato (altrimenti
errore)
2) Verifica che la Data di Assunzione sia <= CURDATE() (altrimenti errore)
3) Calcola l'anzianita' in anni facendo YEAR(CURDATE-DataAssunzione)
*/
DROP TRIGGER IF EXISTS Supermercato.calcola_stipendio;
DELIMITER //
CREATE TRIGGER calcola_stipendio BEFORE INSERT ON Impiegato
FOR EACH ROW
begin
  if (SELECT Tipo FROM Anagrafica A WHERE
A.ID_CodiceFisc=NEW.ID_Impiegato)='I'
then
    if (NEW.DataAssunzione>CURDATE()) then
      SET NEW.DataAssunzione:=NULL;
    else
      SET NEW.ID_Anzianita:=
YEAR(FROM_DAYS(DATEDIFF(CURDATE(),NEW.DataAssunzione)));
      SET NEW.Stipendio:=
(SELECT StipendioBase FROM Livello L WHERE L.ID_Livello=NEW.ID_Livello) +
```

```
(SELECT StipendioAgg FROM Anzianita A WHERE
A.ID_Anzianita=NEW.ID_Anzianita);
  end if;
else
  SET NEW.ID_Impiegato:=NULL;
end if;

end;
//
DELIMITER ;

/*
Trigger che durante la modifica di un impiegato:
Ricalcola, l'anzianita e lo stipendio se necessario.
*/
DROP TRIGGER IF EXISTS Supermercato.upd_calcola_stipendio;
DELIMITER //
CREATE TRIGGER upd_calcola_stipendio BEFORE UPDATE ON Impiegato
FOR EACH ROW
begin
  if NEW.ID_Anzianita IS NOT NULL OR NEW.ID_Livello IS NOT NULL OR
NEW.Stipendio is
NOT NULL then
  SET NEW.ID_Anzianita:=
YEAR(FROM_DAYS(ABS(DATEDIFF(CURDATE(),NEW.DataAssunzione))));
  SET NEW.Stipendio:=
(SELECT StipendioBase FROM Livello L WHERE L.ID_Livello=NEW.ID_Livello) +
(SELECT StipendioAgg FROM Anzianita A WHERE
A.ID_Anzianita=NEW.ID_Anzianita);
  end if;
end;
//
DELIMITER ;

/* Codice SQL per la Creazione della Tabella Reparto */
DROP TABLE IF EXISTS Supermercato.Reparto;
CREATE TABLE Reparto (
 ID_Reparto INTEGER(3) UNSIGNED NOT NULL UNIQUE,
 Nome VARCHAR(20) NOT NULL,
 ID_Responsabile CHAR(16) UNIQUE,
 PRIMARY KEY (ID_Reparto),
 FOREIGN KEY (ID_Responsabile) REFERENCES Impiegati(ID_Impiegato)
 ON DELETE SET NULL
 ON UPDATE CASCADE
);
```

```
/* Codice SQL per aggiungere la FOREIGN KEY alla Tabella Impiegati */
ALTER TABLE Impiegato ADD FOREIGN KEY (ID_Reparto) REFERENCES
Reparto(ID_Reparto) ON DELETE NO ACTION ON UPDATE CASCADE;
/* Codice SQL per la Creazione della Tabella Prodotto */
DROP TABLE IF EXISTS Supermercato.Prodotto;
CREATE TABLE Prodotto (
 ID_Prodotto INTEGER(6) UNSIGNED NOT NULL UNIQUE,
 ID_Reparto INTEGER(3) UNSIGNED NOT NULL DEFAULT 0,
 Nome VARCHAR(20) NOT NULL UNIQUE,
 Genere VARCHAR(15) NOT NULL,
 PrezzoVen DECIMAL(6,2) UNSIGNED NOT NULL,
 Scadenza DATE DEFAULT NULL,
Quantita INTEGER(3) UNSIGNED NOT NULL DEFAULT 0,
 Soglia INTEGER(3) UNSIGNED NOT NULL DEFAULT 0,
 Assemblato ENUM('1','0') NOT NULL DEFAULT '0',
 PuntiRaccolta INTEGER(4) UNSIGNED NOT NULL DEFAULT 0,
 PuntiNecessari INTEGER(6) UNSIGNED NOT NULL DEFAULT 0,
 PRIMARY KEY (ID_Prodotto),
 FOREIGN KEY (ID_Reparto) REFERENCES Reparti(ID_Reparto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/*
Trigger sull'inserimento di nuovi prodotti che:
1) Verifica che la Data di Scadenza del prodotto sia >= CURDATE() (altrimenti errore e non
fa inserire il prodotto)
2) Verifica che il prodotto non dia Punti Raccolta e contemporaneamente sia un Premio
*/
DROP TRIGGER IF EXISTS Supermercato.check_ins_prodotti;
DELIMITER //
CREATE TRIGGER check_ins_prodotti BEFORE INSERT ON Prodotto
FOR EACH ROW
begin
  if NEW.Scadenza<CURDATE() then
    SET NEW.ID_Prodotto:=NULL;
  else
     if NEW.PuntiNecessari=NEW.PuntiRaccolta AND NEW.PuntiNecessari<>0 then
       SET NEW.PuntiNecessari:=NULL;
     end if;
  end if;
end;
//
DELIMITER ;
```

```
/* Codice SQL per la Creazione della Tabella Relaz_Prod_Assemb */
DROP TABLE IF EXISTS Supermercato.Rel_Assemblato;
CREATE TABLE Rel_Assemblato(
 ID_Assemblato INTEGER(6) UNSIGNED NOT NULL,
 ID_Ingrediente INTEGER(6) UNSIGNED NOT NULL,
 Quantita INTEGER(3) UNSIGNED NOT NULL DEFAULT 0,
 PRIMARY KEY (ID_Assemblato, ID_Ingrediente),
 FOREIGN KEY (ID_Assemblato) REFERENCES Prodotti(ID_Prodotto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Ingrediente) REFERENCES Prodotti(ID_Prodotto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/*
Trigger che verifica durante l'inserimento degli ingredienti di un assemblato:
1) l'ingrediente inserito non sia a sua volta assemblato(altrimenti errore)
2) il codice dell'assemblato inserito sia effettivamente un assemblato (altrimenti errore)
*/

DROP TRIGGER IF EXISTS Supermercato.controllo_assemblato;
DELIMITER //
CREATE TRIGGER controllo_assemblato BEFORE INSERT ON Rel_Assemblato
FOR EACH ROW
begin
  if (SELECT Assemblato FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Ingrediente)='1'
then
    SET NEW.ID_Ingrediente:=NULL;
  end if;
  if (SELECT Assemblato FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Assemblato)='0'
then
    SET NEW.ID_Assemblato:=NULL;
  end if;
end;
//
DELIMITER ;
```

```
/*
Trigger che verifica durante l'aggiornamento degli ingredienti di un assemblato:
1) l'ingrediente modificato non sia un assemblato egli stesso (altrimenti rest. errore)
2) il codice dell'assemblato inserito sia effettivamente un assemblato (altr. rest. errore)
*/
DROP TRIGGER IF EXISTS Supermercato.controllo_agg_assemblato;
DELIMITER //
CREATE TRIGGER controllo_agg_assemblato BEFORE UPDATE ON Rel_Assemblato
FOR EACH ROW
begin
  if NEW.ID_Assemblato IS NOT NULL then
    if (SELECT Assemblato FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Assemblato)='0'
then
      SET NEW.ID_Assemblato:=NULL;
    end if;
  end if;
  if NEW.ID_Ingrediente IS NOT NULL then
    if (SELECT Assemblato FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Ingrediente)='1'
then
      SET NEW.ID_Ingrediente:=NULL;
    end if;
  end if;
end;
//
DELIMITER ;

/* Codice SQL per la Creazione della Tabella Fornitore */
DROP TABLE IF EXISTS Supermercato.Fornitore;
CREATE TABLE Fornitore (
 ID_PartitaIva CHAR(11) NOT NULL UNIQUE,
 RagioneSoc VARCHAR(30) NOT NULL,
 Pagamento VARCHAR(20) NOT NULL,
 Indirizzo VARCHAR(30) NOT NULL,
 Telefono VARCHAR(15),
 Fax VARCHAR(15),
 PRIMARY KEY (ID_PartitaIva)
);

/* Codice SQL per la Creazione della Tabella Prodotto_Fornito */
DROP TABLE IF EXISTS Supermercato.Prodotti_Forniti ;
CREATE TABLE Prodotto_Fornito  (
 ID_CodiceInt INTEGER(6) UNSIGNED NOT NULL,
 ID_Fornitore CHAR(11) NOT NULL,
 ID_Prodotto INTEGER(6) UNSIGNED NOT NULL,
```

```
 PrezzoAcq DECIMAL(6,2) UNSIGNED NOT NULL DEFAULT 0,
 PRIMARY KEY (ID_CodiceInt, ID_Fornitore, ID_Prodotto),
 FOREIGN KEY (ID_Fornitore) REFERENCES Fornitore(ID_PartitaIva)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Prodotto) REFERENCES Prodotto(ID_Prodotto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/* Codice SQL per la Creazione della Tabella Ordine */
DROP TABLE IF EXISTS Supermercato.Ordine;
CREATE TABLE Ordine (
 ID_Ordine INTEGER(7) UNSIGNED NOT NULL,
 ID_Fornitore CHAR(11) NOT NULL,
 Data DATE NOT NULL,
 Consegnato ENUM('1','0') NOT NULL DEFAULT '0',
 PRIMARY KEY (ID_Ordine),
 FOREIGN KEY (ID_Fornitore) REFERENCES Fornitore(ID_PartitaIva)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/*
Trigger per verificare i dati prima dell'inserimento negli Ordini:
1) Verifica che la Data di inserimento dell'ordine non sia < CURDATE() (altr. errore)
*/
DROP TRIGGER IF EXISTS Supermercato.controllo_inserimento_ordini;
DELIMITER //
CREATE TRIGGER controllo_inserimento_ordini BEFORE INSERT ON Ordine
FOR EACH ROW
begin
  if NEW.Data<CURDATE() then
    SET NEW.Data:=NULL;
  end if;
end;
//
DELIMITER ;

/*
Trigger per verificare i dati prima dell'inserimento negli Ordini:
1) Verifica che la Data di inserimento dell'ordine non < CURDATE() (altr. errore)
2) Verifica se Consegnato='1' e prima era '0' effettua il caricamento dei prodotti nel
magazzino
*/
DROP TRIGGER IF EXISTS Supermercato.verifica_ordine;
```

```
DELIMITER //
CREATE TRIGGER verifica_ordine BEFORE UPDATE ON Ordine
FOR EACH ROW
begin
  if NEW.Data<CURDATE() then
    SET NEW.Data:=NULL;
  else
    if (NEW.Consegnato='1') AND (OLD.Consegnato='0') then
      if (NEW.ID_Ordine IS NOT NULL) then
        CALL carica_ordine(NEW.ID_Ordine);
      else
        CALL carica_ordine(OLD.ID_Ordine);
      end if;
    else
      if (NEW.Consegnato='0') AND (OLD.Consegnato='1') then
        SET NEW.Consegnato=NULL;
      end if;
    end if;
  end if;
end;
//
DELIMITER ;
/*
Procedura che effettua lo scaricamento dei prodottti dal magazzino per lo scontrino passato
nel parametro scontrino e restituisce il totale dello scontrino nel parametro totale
*/
DROP PROCEDURE IF EXISTS Supermercato.carica_ordine;
DELIMITER //
CREATE PROCEDURE carica_ordine (IN myordine INT)
begin
  DECLARE myprodotto INT;
  DECLARE myquantita INT;
  DECLARE done INT DEFAULT 0;
  DECLARE cursore CURSOR FOR
  SELECT ID_Prodotto, Quantita FROM Dettaglio_Ordini WHERE ID_Ordine=myordine;
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
  OPEN cursore;
  REPEAT
   FETCH cursore INTO myprodotto, myquantita;
   if NOT done then
     UPDATE Prodotti P SET P.Quantita:=P.Quantita+myquantita WHERE
P.ID_Prodotto=myprodotto;
   end if;
  UNTIL done END REPEAT;
  CLOSE cursore;
end //
```

```
DELIMITER ;
/* Codice SQL per la Creazione della Tabella Dettaglio_Ordini */
DROP TABLE IF EXISTS Supermercato.Dettaglio_Ordini;
CREATE TABLE Dettaglio_Ordini (
 ID_Ordine INTEGER(7) UNSIGNED NOT NULL,
 ID_CodiceInt INTEGER(6) UNSIGNED NOT NULL,
 ID_Prodotto INTEGER(6) UNSIGNED NOT NULL,
 Quantita INTEGER(3) UNSIGNED NOT NULL DEFAULT 0,
 PrezzoAcq DECIMAL(6,2) UNSIGNED NOT NULL DEFAULT 0,
 PRIMARY KEY (ID_Ordine, ID_CodiceInt),
 FOREIGN KEY (ID_Ordine) REFERENCES Ordini(ID_Ordine)
 ON DELETE CASCADE
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_CodiceInt) REFERENCES Relaz_Prod_Forn(ID_CodiceInt)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Prodotto) REFERENCES Prodotti(ID_Prodotto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/*
Trigger per verificare i dati prima dell'inserimento dei dettagli ordini:
1) Verifica che ID_Prodotto sia in relazione a CodiceInt del Fornitore nella Tabella
Prodotti_Forniti
2) Verifica che il prodotto non sia assemblato (altr. errore)
3) Copia Prezzo Acquisto dalla Tabella Prodotti_Forniti

*/

DROP TRIGGER IF EXISTS Supermercato.ins_dettaglio_ordini;
DELIMITER //
CREATE TRIGGER ins_dettaglio_ordini BEFORE INSERT ON Dettaglio_Ordini
FOR EACH ROW
begin
  DECLARE id INTEGER(7) UNSIGNED;
  SET id:= (SELECT ID_Prodotto FROM Prodotti_Forniti RPF WHERE
RPF.ID_Fornitore=(SELECT ID_Fornitore FROM Ordine O WHERE
O.ID_Ordine=NEW.ID_Ordine) AND RPF.ID_CodiceInt=NEW.ID_CodiceInt);
  if id IS NULL OR id<>NEW.ID_Prodotto then
    SET NEW.ID_Prodotto:=NULL;
  else
     if (SELECT Assemblato FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Prodotto)='1'
then
      SET NEW.ID_Prodotto:=NULL;
```

```
    else
      SET NEW.PrezzoAcq:=(SELECT PrezzoAcq FROM Prodotti_Forniti RPF WHERE
RPF.ID_Fornitore=(SELECT ID_Fornitore FROM Ordine O WHERE
O.ID_Ordine=NEW.ID_Ordine) AND RPF.ID_CodiceInt=NEW.ID_CodiceInt);
    end if;
  end if;
end;
//
DELIMITER ;
/* Codice SQL per la Creazione della Tabella Fidelizzato */
DROP TABLE IF EXISTS Supermercato.Fidelizzato;
CREATE TABLE Fidelizzato (
 ID_Tessera INTEGER(6) UNSIGNED NOT NULL AUTO_INCREMENT,
 ID_Fidelizzato CHAR(16) NOT NULL UNIQUE,
 Data DATE NOT NULL,
 PuntiAttuali INTEGER(7) UNSIGNED NOT NULL DEFAULT 0,
 PRIMARY KEY (ID_Tessera),
 FOREIGN KEY (ID_Fidelizzato) REFERENCES Anagrafica(ID_CodiceFisc)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);
/*
Trigger che durante l'inserimento di un fidelizzato:
1) Verifica che i dati che si inseriscono siano effettivamente di un fidelizzato (altr. errore)
2) Verifica che la Data di Fidelizzazione sia <= CURDATE() (altr. errore)
*/
DROP TRIGGER IF EXISTS Supermercato.controllo_inserimento_fidelizzato;
DELIMITER //
CREATE TRIGGER controllo_inserimento_fidelizzato BEFORE INSERT ON Fidelizzato
FOR EACH ROW
begin
  if NOT (SELECT Tipo FROM Anagrafica A WHERE
A.ID_CodiceFisc=NEW.ID_Fidelizzato)='F' then
     SET NEW.ID_Fidelizzato:=NULL;
  end if;
  if (NEW.Data>CURDATE()) then
     SET NEW.Data:=NULL;
  end if;
end;
//
DELIMITER ;
/* Codice SQL per la Creazione della Tabella Casse */
DROP TABLE IF EXISTS Supermercato.Casse;
CREATE TABLE Casse (
 ID_Cassa INTEGER(2) UNSIGNED NOT NULL,
 Nome VARCHAR(20) NOT NULL,
```

```
 PRIMARY KEY (ID_Cassa)
 );

/* Codice SQL per la Creazione della Tabella Scontrino */
DROP TABLE IF EXISTS Supermercato.Scontrino;
CREATE TABLE Scontrino (
 ID_Scontrino INTEGER(8) UNSIGNED NOT NULL AUTO_INCREMENT,
 ID_Cassa INTEGER(2) UNSIGNED NOT NULL,
 ID_Tessera INTEGER(6) UNSIGNED DEFAULT 0,
 Data DATE NOT NULL,
 Totale DECIMAL(7,2) UNSIGNED NOT NULL,
 Pagato ENUM('1','0') NOT NULL DEFAULT '0',
 PRIMARY KEY (ID_Scontrino),
 FOREIGN KEY (ID_Cassa) REFERENCES Casse(ID_Cassa)
 ON DELETE NO ACTION
 ON UPDATE CASCADE,
 FOREIGN KEY (ID_Tessera) REFERENCES Fidelizzati(ID_Tessera)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);

/*
Trigger eseguito prima dell'inserimento degli scontrini che genera un nuovo ID_Scontrino:
se
non ci sono scontrini nella Tabella ID_Scontrino = 1 altrimenti ID_Scontrino =
MAX(ID_Scontrino)+1.
Inoltre controlla che la data dello scontrino sia uguale a quella attuale.
*/
DROP TRIGGER IF EXISTS Supermercato.inc_id_scontrino;
DELIMITER //
CREATE TRIGGER inc_id_scontrino BEFORE INSERT ON Scontrino
FOR EACH ROW
begin
  SET NEW.ID_Scontrino:= (SELECT MAX(ID_Scontrino) FROM Scontrino);
  if NEW.ID_Scontrino IS NULL then
    SET NEW.ID_Scontrino:=1;
  else
    SET NEW.ID_Scontrino:=NEW.ID_Scontrino+1;
  end if;
  if NEW.Data<>CURDATE() then
    SET NEW.Data:=NULL;
  end if;
end;
//
DELIMITER ;
/*
```

Trigger eseguito prima dell'update dello scontrino che se Pagato='1' chiama la procedura che
scarica i prodotti venduti e calcola il totale dello scontrino e calcola il totale dei punti raccolta aggiornando le rispettive Tabelle
*/
```
DROP TRIGGER IF EXISTS Supermercato.up_id_scontrino;
DELIMITER //
CREATE TRIGGER up_id_scontrino BEFORE UPDATE ON Scontrino
FOR EACH ROW
begin
  DECLARE xpunti INT;
  DECLARE xtessera INT;
  SET xpunti:=0;
  SET xtessera:=0;
  if (NEW.ID_Tessera IS NOT NULL) then
    SET xtessera:=NEW.ID_Tessera;
  else
    if (OLD.ID_Tessera IS NOT NULL) then
      SET xtessera:=OLD.ID_Tessera;
    end if;
  end if;
  if (NEW.Pagato='1') AND (OLD.Pagato='0') then
    if (NEW.ID_Scontrino IS NOT NULL) then
      CALL scarica_scontrino(NEW.ID_Scontrino,NEW.Totale, xpunti);
    else
      CALL scarica_scontrino(OLD.ID_Scontrino,NEW.Totale, xpunti);
    end if;

   if (xtessera<>0) AND (xpunti<>0) then
     UPDATE Fidelizzati F SET F.PuntiAttuali:=F.PuntiAttuali+xpunti WHERE
F.ID_Tessera=xtessera;
   end if;
  else
   if (NEW.Pagato='0') AND (OLD.Pagato='1') then
     SET NEW.Pagato:=NULL;
   end if;
  end if;
end;
//
DELIMITER ;
/* Codice SQL per la Creazione della Tabella Dettaglio_Scontrini */
DROP TABLE IF EXISTS Supermercato.Dettaglio_Scontrini;
CREATE TABLE Dettaglio_Scontrini (
 ID_Scontrino INTEGER(8) UNSIGNED NOT NULL,
 ID_Prodotto INTEGER(6) UNSIGNED NOT NULL,
 Quantita INTEGER(2) UNSIGNED NOT NULL,
```

```
 PrezzoVen DECIMAL(6,2) UNSIGNED NOT NULL,
 PRIMARY KEY (ID_Scontrino, ID_Prodotto),
 FOREIGN KEY (ID_Scontrino) REFERENCES Scontrini(ID_Scontrino)
 ON DELETE NO ACTION
 ON UPDATE NO ACTION,
 FOREIGN KEY (ID_Prodotto) REFERENCES Prodotto(ID_Prodotto)
 ON DELETE NO ACTION
 ON UPDATE CASCADE
);
/*
Trigger che durante l'inserimento di un Dettaglio Scontrino:
1) Setta il prezzo di vendita del prodotto come scritto nella Tabella Prodotto
*/
DROP TRIGGER IF EXISTS supermercato.ins_dettaglio_scontrini;
DELIMITER //
CREATE TRIGGER ins_dettaglio_scontrini BEFORE INSERT ON Dettaglio_Scontrini
FOR EACH ROW
begin
  SET NEW.PrezzoVen:=(SELECT PrezzoVen FROM Prodotto P WHERE
P.ID_Prodotto=NEW.ID_Prodotto);
end;
//
DELIMITER ;

/*
Procedura che effettua lo scaricamento dei prodottti dal magazzino per lo scontrino passato
nel
parametro myscontrino e restituisce il totale dello scontrino nel parametro mytotale e
restituisce
i puntiraccolta totali nel parametro mypunti
*/
DROP PROCEDURE IF EXISTS Supermercato.scarica_scontrino;
DELIMITER //
CREATE PROCEDURE scarica_scontrino (IN myscontrino INT, OUT mytotale
DECIMAL(6,2),
OUT mypunti INT)
begin
  DECLARE myprodotto INT;
  DECLARE myquantita INT;
  DECLARE myprezzo DECIMAL(6,2) UNSIGNED DEFAULT 0;
  DECLARE myraccolta INT;
  DECLARE done INT DEFAULT 0;
  DECLARE cursore CURSOR FOR
  SELECT DS.ID_Prodotto, DS.Quantita, DS.PrezzoVen, P.PuntiRaccolta FROM
Dettaglio_Scontrini DS, Prodotto P WHERE DS.ID_Scontrino=myscontrino AND
DS.ID_Prodotto=P.ID_Prodotto;
```

```
  DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
  SET mytotale:=0;
  SET mypunti:=0;
  OPEN cursore;
  REPEAT
   FETCH cursore INTO myprodotto, myquantita, myprezzo, myraccolta;
   if NOT done then
     UPDATE Prodotto SET Quantita:=Quantita-myquantita WHERE
ID_Prodotto=myprodotto;
     SET mytotale:=mytotale+myquantita*myprezzo;
     SET mypunti:=mypunti+myquantita*myraccolta;
   end if;
  UNTIL done END REPEAT;
  CLOSE cursore;
end
//
DELIMITER ;
```

# Inserimento dei valori nella Base Dati

```
/* Insert Livelli */
INSERT INTO Livello VALUES (1,600);
INSERT INTO Livello VALUES (2,800);
INSERT INTO Livello VALUES (3,1000);
INSERT INTO Livello VALUES (4,1200);
INSERT INTO Livello VALUES (5,1500);
INSERT INTO Livello VALUES (6,2000);
INSERT INTO Livello VALUES (7,25000);
INSERT INTO Livello VALUES (8,5000);
INSERT INTO Livello VALUES (9,10000);
INSERT INTO Livello VALUES (10,20000);
/* Insert Anzianita */
INSERT INTO Anzianita VALUES (0,0);
INSERT INTO Anzianita VALUES (1,10);
INSERT INTO Anzianita VALUES (2,20);
INSERT INTO Anzianita VALUES (3,30);
INSERT INTO Anzianita VALUES (4,40);
INSERT INTO Anzianita VALUES (5,50);
INSERT INTO Anzianita VALUES (6,100);
INSERT INTO Anzianita VALUES (7,120);
INSERT INTO Anzianita VALUES (8,150);
INSERT INTO Anzianita VALUES (9,180);
INSERT INTO Anzianita VALUES (10,200);
INSERT INTO Anzianita VALUES (11,230);
INSERT INTO Anzianita VALUES (12,250);
INSERT INTO Anzianita VALUES (13,280);
INSERT INTO Anzianita VALUES (14,300);
INSERT INTO Anzianita VALUES (15,320);
INSERT INTO Anzianita VALUES (16,350);
INSERT INTO Anzianita VALUES (17,380);
INSERT INTO Anzianita VALUES (18,400);
INSERT INTO Anzianita VALUES (19,420);
INSERT INTO Anzianita VALUES (20,450);
INSERT INTO Anzianita VALUES (21,480);
INSERT INTO Anzianita VALUES (22,500);
INSERT INTO Anzianita VALUES (23,520);
INSERT INTO Anzianita VALUES (24,550);
INSERT INTO Anzianita VALUES (25,580);
INSERT INTO Anzianita VALUES (26,600);
INSERT INTO Anzianita VALUES (27,620);
INSERT INTO Anzianita VALUES (28,650);
INSERT INTO Anzianita VALUES (29,680);
INSERT INTO Anzianita VALUES (30,700);
```

```
/* Insert Mansioni */
INSERT INTO Mansione VALUES (0,'SCARICO MERCI');
INSERT INTO Mansione VALUES (1,'MAGAZZINIERE');
INSERT INTO Mansione VALUES (2,'COMMESSO REPARTO');
INSERT INTO Mansione VALUES (3,'CAPO REPARTO');
INSERT INTO Mansione VALUES (4,'ADDETTO CASSA');
INSERT INTO Mansione VALUES (5,'ADDETTO PULIZIE');
INSERT INTO Mansione VALUES (6,'VICE DIRETTORE');
INSERT INTO Mansione VALUES (7,'DIRETTORE');
/* Insert Reparti */
INSERT INTO Reparto VALUES (0,'MACELLARIA',NULL);
INSERT INTO Reparto VALUES (1,'PANETTERIA',NULL);
INSERT INTO Reparto VALUES (2,'ELETTRODOMESTICI',NULL);
INSERT INTO Reparto VALUES (3,'TELEFONIA',NULL);
INSERT INTO Reparto VALUES (4,'FRUTTA E VERDURA',NULL);
INSERT INTO Reparto VALUES (5,'INFORMATICA',NULL);

/* Insert Casse */
INSERT INTO Casse VALUES (1,'Cassa 1');
INSERT INTO Casse VALUES (2,'Cassa 2');
INSERT INTO Casse VALUES (3,'Cassa 3');
INSERT INTO Casse VALUES (4,'Cassa 4');
INSERT INTO Casse VALUES (5,'Cassa 5');

/* Insert Anagrafica Impiegati */
INSERT INTO Anagrafica VALUES ('DKRI5839FKDIRKM4','Marco','Pace','Via
astignano Pianella PE','085972581',19880901,'I');
INSERT INTO Anagrafica VALUES ('DJVCKEI45ID4KDMF','Luca','Pietrangelo','Via
colle vecchio 43 Pianella PE','085972543',19880127,'I');
INSERT INTO Anagrafica VALUES ('TRNFNC88D13G482S','Franco','Troiano','C\da
Pratelle 30 Pianella PE','085971989',19880413,'I');
INSERT INTO Anagrafica VALUES ('DVDSTR88DJSU57RH','Davide','Stringini','Via di
prova 43 Avezzano AQ','0862764839',19880101,'I');

/* Insert Anagrafica Fidelizzati */
INSERT INTO Anagrafica VALUES ('BBBCCC77E10F004X','Di giovanni' , 'Babbione' ,
'Via Lucifero 13','0111/11112188',19770510,'F');
INSERT INTO Anagrafica VALUES ('DDDFNC72F15G005F','Franco','Daddede','Piazza
Farnese 15','0222/22223244',19720315,'F');
INSERT INTO Anagrafica VALUES ('FFNNRC75G25H004F','Enrica','Fafone','Piazza
Mazzini 35','0333/33333244',19750725,'F');
INSERT INTO Anagrafica VALUES ('LMBMGZ69B04F234E','Maria
Grazia','Lombardi','Via da Qui 89','0444/9999900',19690204,'F');
```

```
/* Insert Impiegati */
INSERT INTO Impiegato VALUES (’DKRI5839FKDIRKM4’,2,3,2,0,20050205,0);
INSERT INTO Impiegato VALUES (’9DJVCKEI45ID4KDM’,2,3,2,0,20060304,0);
INSERT INTO Impiegato VALUES (’FRTRNFNC88D13G48’,2,3,3,0,19990122,0);
INSERT INTO Impiegato VALUES (’SDVDSTR88DJSU57R’,3,3,2,0,20000520,0);


/* Insert Fidelizzati */
INSERT INTO Fidelizzato (ID_Fidelizzato, Data) VALUES
(’BBBCCC77E10F004X’,CURDATE());
INSERT INTO Fidelizzato (ID_Fidelizzato, Data) VALUES
(’DDDFNC72F15G005F’,CURDATE());
INSERT INTO Fidelizzato (ID_Fidelizzato, Data) VALUES
(’FFNNRC75G25H004F’,CURDATE());
INSERT INTO Fidelizzato (ID_Fidelizzato, Data) VALUES
(’LMBMGZ69B04F234E’,CURDATE());

/* Insert Prodotto normali */
INSERT INTO Prodotto VALUES (1, 1, ’Cavolo’, ’Alimentari’, 2.5, 20090224, 200, 500,
’0’, 10, 0);
INSERT INTO Prodotto VALUES (2, 1, ’Insalata’, ’Alimentari’, 2.1, 20090222, 300, 100,
’0’, 0, 100);
INSERT INTO Prodotto VALUES (3, 1, ’Acqua’, ’Alimentari’, 0.4, 20090501, 350, 450,
’0’, 20, 0);
INSERT INTO Prodotto VALUES (4, 1, ’Lievito’, ’Alimentari’, 1.2, 20090502, 350, 50,
’0’, 5, 0);
INSERT INTO Prodotto VALUES (5, 4, ’Salmone’, ’Pesce’, 7.5, 20090228, 350, 50, ’0’,
40, 0);
INSERT INTO Prodotto VALUES (6, 1, ’Mele’, ’Frutta’, 1.5, 20090228, 350, 50, ’0’, 5, 0);
INSERT INTO Prodotto VALUES (7, 1, ’Pere’, ’Frutta’, 1, 20090228, 350, 50, ’0’, 0, 200);
INSERT INTO Prodotto VALUES (8, 2, ’Farina’, ’Alimentari’, 3, 20101230, 350, 50, ’0’,
3, 0);
INSERT INTO Prodotto VALUES (9, 2, ’Sale’, ’Alimentari’, 3, NULL, 350, 50, ’0’, 2, 0);

INSERT INTO Prodotto VALUES (10, 4, ’Pesce spada’, ’Pesce’, 5.99, 20090228, 40, 20,
’0’, 100, 0);
INSERT INTO Prodotto VALUES (11, 8, ’Gelato esotico’, ’Surgelati’, 2.5, NULL, 480,
150, ’0’, 0, 0);
/* Insert Prodotto assemblati */
INSERT INTO Prodotto VALUES (100, 2, ’Pizza’, ’Alimentari’, 3, CURDATE(), 350, 50,
’1’, 0, 0);
INSERT INTO Prodotto VALUES (110, 2, ’Pane’, ’Alimentari’, 3, CURDATE(), 350, 50,
’1’, 0, 0);
/* Insert ingredienti Prodotti assemblati */
INSERT INTO Prodotti_Assemblati VALUES (100, 3, 1);
INSERT INTO Prodotti_Assemblati VALUES (100, 4, 1);
```

```
INSERT INTO Prodotti_Assemblati VALUES (100, 5, 1);
INSERT INTO Prodotti_Assemblati VALUES (100, 8, 1);
INSERT INTO Prodotti_Assemblati VALUES (100, 9, 1);
INSERT INTO Prodotti_Assemblati VALUES (110, 4, 2);
INSERT INTO Prodotti_Assemblati VALUES (110, 3, 1);
INSERT INTO Prodotti_Assemblati VALUES (110, 8, 1);
/* Insert Scontrino */
INSERT INTO Scontrino VALUES (0, 1, 1, CURDATE(), 0, '0');
INSERT INTO Scontrino VALUES (0, 1, NULL, CURDATE(), 0, '0');
INSERT INTO Scontrino VALUES (0, 5, NULL, CURDATE(), 0, '0');
INSERT INTO Scontrino VALUES (0, 2, NULL, CURDATE(), 0, '0');
/* Insert dettaglio scontrini */
INSERT INTO Dettaglio_Scontrini VALUES (1, 1, 100, 0);
INSERT INTO Dettaglio_Scontrini VALUES (1, 2, 57, 0);
INSERT INTO Dettaglio_Scontrini VALUES (1, 3, 48, 0);
INSERT INTO Dettaglio_Scontrini VALUES (1, 10, 2, 0);
INSERT INTO Dettaglio_Scontrini VALUES (2, 1, 50, 0);
INSERT INTO Dettaglio_Scontrini VALUES (2, 2, 100, 0);
INSERT INTO Dettaglio_Scontrini VALUES (3, 3, 200, 0);
INSERT INTO Dettaglio_Scontrini VALUES (3, 7, 40, 0);
INSERT INTO Dettaglio_Scontrini VALUES (3, 2, 10, 0);
INSERT INTO Dettaglio_Scontrini VALUES (3, 11, 1, 0);
INSERT INTO Dettaglio_Scontrini VALUES (4, 1, 10, 0);
INSERT INTO Dettaglio_Scontrini VALUES (4, 9, 10, 0);
/* Insert Fornitori */
INSERT INTO Fornitore VALUES ('00049872134', 'Oasi', 'Rate 3 mesi', 'Via Pari 17,
Pescara
66023', '085/479865', '085/479860');
INSERT INTO Fornitore VALUES ('00057772134', 'Pirati', 'Contanti', 'Via Piazza 157,
Pescara 66023', '085/457865', '085/433160');
INSERT INTO Fornitore VALUES ('00046812134', 'Sprint', 'Giroconto', 'Via Milano 97,
Pescara 66023', '085/411105', '085/479870');
INSERT INTO Fornitore VALUES ('00149872174', 'Zippo', 'Rate 12 mesi', 'Via Bari 64,
Pescara 66023', '085/479321', '085/479009');
/* Insert Prodotti_Forniti */
INSERT INTO Prodotti_Forniti VALUES (1, '00049872134', 3, 2.4);
INSERT INTO Prodotti_Forniti VALUES (2, '00049872134', 7, 2.8);
INSERT INTO Prodotti_Forniti VALUES (3, '00049872134', 8, 1.4);
INSERT INTO Prodotti_Forniti VALUES (1, '00057772134', 3, 2.1);
INSERT INTO Prodotti_Forniti VALUES (1, '00046812134', 3, 1.7);
INSERT INTO Prodotti_Forniti VALUES (2, '00046812134', 5, 2.4);
/* Insert Ordini */
INSERT INTO Ordine VALUES (1, '00046812134', CURDATE(),'0');
INSERT INTO Ordine VALUES (2, '00046812134', 20090223,'0');
INSERT INTO Ordine VALUES (3, '00057772134', CURDATE(),'0');
INSERT INTO Ordine VALUES (4, '00049872134', 20090425,'0');
```

/* Insert Dettaglio Ordini */
INSERT INTO Dettaglio_Ordini VALUES (1, 1, 3, 500, 0);
INSERT INTO Dettaglio_Ordini VALUES (1, 2, 5, 350, 0);
INSERT INTO Dettaglio_Ordini VALUES (3, 1, 3, 400, 0);
INSERT INTO Dettaglio_Ordini VALUES (4, 1, 3, 500, 0);
INSERT INTO Dettaglio_Ordini VALUES (4, 2, 7, 700, 0);

2) Per ogni relazione individuata, fornire le istruzioni di inserimento, modifica ed eliminazione delle istanze.
INSERT INTO Livello VALUES (x,y);
UPDATE Livello SET A1=x,A2=y WHERE condition;
DELETE FROM Livello WHERE condition;
INSERT INTO Anzianita VALUES (x,y);
UPDATE Anzianita SET A1=x,A2=y WHERE condition;
DELETE FROM Anzianita WHERE condition;
INSERT INTO Mansione VALUES (x,'string');
UPDATE Mansione SET A1=x,A2=y WHERE condition;
DELETE FROM Mansione WHERE condition;
INSERT INTO Reparto VALUES (x,'string1','string2');
UPDATE Reparto SET A1=x,... WHERE condition;
DELETE FROM Reparto WHERE condition;
INSERT INTO Casse VALUES (x,'string');
UPDATE Casse SET A1=x,... WHERE condition;
DELETE FROM Casse WHERE condition;
INSERT INTO Anagrafica VALUES ('string',...);
UPDATE Anagrafica SET A1=x,... WHERE condition;
DELETE FROM Anagrafica WHERE condition;
INSERT INTO Impiegato VALUES ('string',...);
UPDATE Impiegato SET A1=x,... WHERE condition;
DELETE FROM Impiegato WHERE condition;
INSERT INTO Fidelizzato (A1, A2) VALUES ('string',date);
UPDATE Fidelizzato SET A1=x,... WHERE condition;
DELETE FROM Fidelizzato WHERE condition;
INSERT INTO Prodotto VALUES (x,...);
UPDATE Prodotto SET A1=x,... WHERE condition;
DELETE FROM Prodotto WHERE condition;
INSERT INTO Prodotto_Assemblato VALUES (x,...);
UPDATE Prodotto_Assemblato SET A1=x,... WHERE condition;
DELETE FROM Prodotto_Assemblato WHERE condition;
INSERT INTO Scontrino VALUES (x, ...);
UPDATE Scontrino SET A1=x,... WHERE condition;
DELETE FROM Scontrino WHERE condition;
INSERT INTO Dettaglio_Scontrini VALUES (x,...);
UPDATE Dettaglio_Scontrini SET A1=x,... WHERE condition;
DELETE FROM Dettaglio_Scontrini WHERE condition;
INSERT INTO Fornitore VALUES ('string', ...);

```
UPDATE Fornitore SET A1='string',... WHERE condition;
DELETE FROM Fornitore WHERE condition;
INSERT INTO Prodotti_Forniti VALUES (x, ...);
UPDATE Prodotti_Forniti SET A1=x,... WHERE condition;
DELETE FROM Prodotti_Forniti WHERE condition;
INSERT INTO Ordine VALUES (x,...);
UPDATE Ordine SET A1=x,... WHERE condition;
DELETE FROM Ordine WHERE condition;
INSERT INTO Dettaglio_Ordini VALUES (x,...);
UPDATE Dettaglio_Ordini SET A1=x,... WHERE condition;
DELETE FROM Dettaglio_Ordini WHERE condition;
```

3) Modifica del responsabile e degli impiegati di un reparto:

```
UPDATE Reparto SET ID_Responsabile='SPRCLD71H16D773Z' WHERE
ID_Reparto=4;
UPDATE Impiegato SET ID_Reparto = 4 WHERE ID_Impiegato =
'SPRCLD71H16D773Z';
UPDATE Impiegato SET ID_Reparto = 4 WHERE ID_Impiegato =
'PCEMRC88T13G482J';
UPDATE Impiegati SET ID_Reparto = 4 WHERE ID_Impiegato =
'LMBMCH79D05E111E';
```

4) Determinazione delle vendite per un reparto in un particolare periodo:

ID_Reparto=1

```
SELECT P.Nome, SUM(D.Quantita) AS Quantita_Vendute
FROM Dettaglio_Scontrini D, Scontrino S, Prodotto P, Reparto R
WHERE D.ID_Scontrino = S.ID_Scontrino AND
 D.ID_Prodotto = P.ID_Prodotto AND
 P.ID_Reparto = R.ID_Reparto AND
 S.DATA BETWEEN 20080101 AND 20091231 AND
 R.ID_REPARTO = 1
 GROUP BY P.Nome;
```

| Nome | Quantita_Vendute |
|---|---|
| Acqua | 248 |
| Cavolo | 160 |
| Insalata | 167 |
| Pere | 40 |

5) Determinazione dei Prodotti più venduti in un determinato reparto:

ID_Reparto=1

```
SELECT P.Nome, sum(D.Quantita) as Pezzi_Venduti
FROM Dettaglio_Scontrini D, Prodotto P, Reparto R
WHERE D.ID_Prodotto = P.ID_Prodotto AND
 P.ID_Reparto = R.ID_Reparto AND
```

```
 R.ID_Reparto = 1
GROUP BY P.ID_Prodotto
ORDER BY sum(D.Quantita) DESC;
Nome            Pezzi_Venduti
--------        -------------------
Acqua            248
Insalata         167
Cavolo           160
Pere            40
```

6) Modifica del prezzo di un prodotto:
ID_Prodotto=3
UPDATE Prodotto SET PrezzoVen = 2.5 WHERE ID_Prodotto = 3;

7) Modifica dei dati riguardanti le scorte di prodotto disponibili:
ID_Prodotto=3
UPDATE Prodotto SET Quantita=500 WHERE ID_Prodotto=3;

8) Per i prodotti “composti”, specifica degli “ingredienti” e delle quantità necessarie :

```
SELECT P.Nome AS Prodotto_Assemblato, B.Nome AS Ingredienti, B.Quantita AS
Quantita
FROM Prodotti_Assemblati R, Prodotto P, Prodotto B
WHERE R.ID_Assemblato = P.ID_Prodotto AND
R.ID_Ingrediente = B.ID_Prodotto
ORDER BY P.Nome;
Prodotto_Assemblati              Ingredienti     Quantita
-------------------------        --------------- ------------
Pane                             Farina          350
Pane                             Acqua           500
Pane                             Lievito         350
Pizza                            Salmone         350
Pizza                            Farina          350
Pizza                            Acqua           500
Pizza                            Sale            350
Pizza                            Lievito         350
```

9) Inclusione o esclusione di un prodotto dalla raccolta punti e indicazione del numero di puntiraccolta forniti dal prodotto:
ID_Prodotto=11
UPDATE Prodotto SET PuntiRaccolta = 50 WHERE ID_PRODOTTO = 11;
UPDATE Prodotto SET PuntiRaccolta = 0 WHERE ID_PRODOTTO = 11;

10) Determinazione dei prodotti sotto scorta:

```
SELECT Nome, Genere, Quantita, Soglia
FROM Prodotto
```

```
WHERE Quantita < Soglia;
Nome            Genere          Quantita        Soglia
--------        ----------      -----------     --------
Cavolo          Alimentari      200             500
```

11) Lista dei fornitori dai quali un determinato prodotto può essere acquistato, ordinati in base al prezzo richiesto;

```
SELECT ID_PartitaIva, RagioneSoc, PrezzoAcq
FROM Prodotti_Forniti R, Fornitori F
WHERE R.ID_Fornitore = F.ID_PartitaIva AND
 R.ID_Prodotto = 3
ORDER BY PrezzoAcq;
ID_PartitaIva           RagioneSoc      PrezzoAcq
-----------------       --------------  -------------
00046812134             Sprint          1.70
00057772134             Pirati          2.10
00049872134             Oasi            2.40
```

12) Modifica del numero di punti necessari ad ottenere un determinato premio:

```
UPDATE Prodotto SET PuntiNecessari = 1500 WHERE ID_Prodotto = 11;
```

13) Verifica dei premi attualmente disponibili:

```
SELECT Nome, PuntiNecessari, Quantita
FROM Prodotto
WHERE PuntiNecessari>0 AND Quantita>0;
Nome            PuntiNecessari          Quantita
Insalata        100                     300
Pere            200                     350
Gelato esotico  1500                    480
```

14) Inserimento/Modifica dei Prodotto forniti da un fornitore:

```
INSERT INTO Prodotti_Forniti VALUES (4, ’00049872134’, 7, 1.2);
UPDATE Prodotti_Forniti SET ID_Prodotto = 9 WHERE ID_Fornitore = ’00049872134’
AND
ID_CodiceInt = 4;
```

15) Modifica del reparto di assegnazione di un impiegato e del suo livello:

```
UPDATE Impiegato SET ID_Reparto = 2, ID_Livello = 3 WHERE ID_Impiegato =
’SPRCLD71H16D773Z’;
```

16) Modifica del numero di punti assegnati al cliente:

```
UPDATE Fidelizzato SET PuntiAttuali = PuntiAttuali + 100 WHERE ID_Fidelizzato =
’BBBCCC77E10F004X’;
```

17) Determinazione dei premi a cui un cliente ha diritto:
Per poter rispondere a questa query, dobbiamo attivare le procedure di scarico scontrino attivabili solo quando uno scontrino e' Pagato, quindi effettueremo prima un pagamento di uno scontrino in modo che vengano caricati i punti del fidelizzato
ID_FIDELIZZATO=1
UPDATE Scontrino SET Pagato='1' WHERE ID_Scontrino=1;
SELECT P.Nome, P.PuntiNecessari
FROM Fidelizzato F, Prodotto P
WHERE P.PuntiNecessari <= F.PuntiAttuali AND P.PuntiNecessari<>0
ORDER BY P.PuntiNecessari;

| Nome | PuntiNecessari |
|---|---|
| Insalata | 100 |
| Pere | 200 |
| Gelato esotico | 1500 |

18) Ritiro di un premio da parte di un cliente:
ID_PRODOTTO=2
ID_FIDELIZZAT0=1
Per realizzare quest'aggiornamento usiamo una procedura
CALL ritira_premio(2,1);

```
DROP PROCEDURE IF EXISTS Supermercato.ritira_premio;
DELIMITER //
CREATE PROCEDURE ritira_premio (IN mypremio INT, IN mytessera INT)
begin
 DECLARE xpunti INT;
 SET xpunti=0;
  UPDATE Prodotto SET Quantita=Quantita-1 WHERE ID_Prodotto=mypremio;
  SELECT PuntiNecessari INTO xpunti FROM Prodotto WHERE ID_Prodotto=mypremio;
  UPDATE Fidelizzati SET PuntiAttuali=PuntiAttuali-xpunti WHERE
ID_Tessera=mytessera;
end //
DELIMITER ;
```

19) Determinazione dei prodotti più acquistati da un cliente:
ID_TESSERA=1

```
SELECT P.Nome, sum(D.Quantita) AS Pezzi_Acquistati
FROM Scontrino S, Dettaglio_Scontrini D, Prodotto P
WHERE D.ID_Scontrino = S.ID_Scontrino AND
  D.ID_Prodotto = P.ID_Prodotto AND
  S.ID_Tessera = 1
GROUP BY P.ID_Prodotto
ORDER BY sum(D.Quantita) DESC;
```

| Nome | Pezzi_Acquistati |
|---|---|
| Cavolo | 100 |
| Insalata | 57 |
| Acqua | 48 |
| Pesce spada | 2 |

20) Determinazione della spesa totale effettuata da un cliente in un determinato periodo:
ID_FIDELIZZATO=1

```
SELECT A.Nome, SUM(S.Totale) AS Tot_spesa
FROM Scontrino S, Anagrafica A, Fidelizzato F
WHERE S.Data BETWEEN 20080101 AND 20091231 AND
  S.ID_Tessera = 1 AND
  S.ID_Tessera = F.ID_Tessera AND
  F.ID_Fidelizzato = A.ID_CodiceFisc;
```

```
Nome          Tot_spesa
--------      -------------
Di Giovanni     400.88
```