# PROGETTO DI UNA BASE DI DATI

## *Gestione di un negozio online di prodotti elettronici*

*Realizzato da Claudio Canulla, Giovanni Lopopolo, Stefano Zhao*

### Sommario

# PROGETTAZIONE CONCETTUALE

## Raccolta dei Requisiti

Si vuole progettare una base di dati finalizzata a modellare l'organizzazione di un negozio online di prodotti elettronici.

L'**utente** può registrarsi con il proprio nome, cognome, mail e password e viene identificato da un Id_Utente.

Il Negozio viene rifornito di prodotti elettronici tramite fornitori, ogni fornitore può essere relativo ad uno o più prodotti.

Ogni fornitore è identificato dalla partita iva, nome, numero di telefono, indirizzo, e-mail e sede comprensiva di città e indirizzo.

Il **prodotto** è identificato da un Id_Prodotto e presenta prezzo, marchio e modello.

Ogni prodotto è fornito da un solo fornitore.

I prodotti, una volta ricevuti dai fornitori, vengono stipati in 5 **magazzini** posti in diversesedi, ogni magazzino è identificato con un Id_Magazzino costituito da una lettera alfabetica.

I prodotti vengono suddivisi per **categoria.**

Abbiamo 4 diverse categorie: Casalinghi, TV/Audio, Telefonia e Informatica, quest'ultima è suddivisa ulteriormente in Accessori, Pc e Software. Ciascun prodotto ha un'unica categoria.

Un utente può inserire nel **carrello**uno o più prodottise disponibili in uno dei magazzini, una volta che una quantità di prodotto è aggiunta nel carrello allora tale quantità è tolta dall'inventario.

Per la gestione del **pagamento,** l'utente deve inserire un numero di carta e ogni pagamento è identificato tramite un id numerico.Il pagamento riporta in oltre una data e l'importo composto dal costo totale dei prodotti più il costo della spedizione.

Un utente può effettuare diversi pagamenti all'interno del negozio e ciascun singolo pagamento è associato al solo utente che lo ha effettuato.

Il pagamento è associato al carrello composto dall'utente e una volta effettuato genere un ordine e una spedizione.

**Un ordine**, oltre all'id numerico, presenta il numero di articoli da cui è composto.

Il negozio non gestisce direttamente le spedizioni degli ordini ai clienti ma si affida a varie **ditte di spedizione** identificate dalla partita iva, nome, sede, mail e numero di telefono facoltativo.

Ogni **spedizione** coincide con un ordine ed è identificata da un Id_Spedizione (numerico) e ha una data e un costo di spedizione pari a 5 euro per ordine.

Qualora l'ordine si uguale o supere a 30 euro il costo di spedizione sarà a carico del negozio pertanto sarà nullo per l'utente.

# Analisi delle specifiche

*Frasi di carattere generale:*

Si vuole progettare una base dati finalizzata a modellare l'organizzazione di un negozio online di prodotti elettronici.

*Frasi relative gli utenti*:

L'**utente** può registrarsi con il proprio nome, cognome, mail e password e viene identificato da un Id_Utente.

*Frasi relative ai prodotti*:
Il **prodotto** è identificato da un Id_Prodotto e presenta un prezzo, marchio e modello.

Ogni prodotto è fornito da un solo fornitore.

*Frasi relative al carrello:*

Un utente può inserire nel **carrello**uno o più prodottise disponibili in uno dei magazzini, una volta che una quantità di prodotto è aggiunta nel carrello allora tale quantità è tolta dall'inventario.

*Frasi relative a categoria*:

I prodotti vengono suddivisi per **categoria.**

Abbiamo 4 diverse categorie: Casalinghi, TV/Audio, Telefonia e Informatica, quest'ultima è suddivisa ulteriormente in Accessori, Pc e Software. Ciascun prodotto ha un'unica categoria.

*Frasi relative ai fornitori*:

Il Negozio viene rifornito di prodotti elettronici tramite fornitori, ogni fornitore può essere relativo ad uno o più prodotti.

Ogni fornitore è identificato dalla partita iva, nome, numero di telefono, indirizzo, e-mail e sede comprensiva di città e indirizzo.

*Frasi relative ai magazzino*:
I prodotti, una volta ricevuti dai fornitori, vengono stipati in 5 **magazzini** posti in diversesedi, ogni magazzino è identificato con un Id_Magazzino costituito da una lettera alfabetica.

*Frasi relative al pagamento:*
Per la gestione del **pagamento,** l'utente deve inserire un numero di carta e ogni pagamento è identificato tramite un id numerico.Il pagamento riporta in oltre una data e l'importo composto dal costo totale dei prodotti più il costo della spedizione.

Un utente può effettuare diversi pagamenti all'interno del negozio e ciascun singolo pagamento è associato al solo utente che lo ha effettuato.

Il pagamento è associato al carrello composto dall'utente e una volta effettuato genere un ordine e una spedizione.

*Frasi relative agli ordini:*

**Un ordine**, oltre all'id numerico, presenta il numero di articoli da cui è composto.

*Frasi relative a ditta spedizione*:
Il negozio non gestisce direttamente la spedizione dell'ordine al cliente ma si affida a varie **ditte di spedizione** di terze parti identificate dalla partita iva, da un nome, una sede, una mail e un numero di telefono facoltativo.

*Frasi relative a spedizione:* Il negozio non gestisce direttamente le spedizioni degli ordini ai clienti ma si affida a varie **ditte di spedizione** identificate dalla partita iva, nome, sede, mail e numero di telefono facoltativo.

Ogni **spedizione** coincide con un ordine ed è identificata da un Id_Spedizione (numerico) e ha una data e un costo di spedizione pari a 5 euro per ordine.

Qualora l'ordine si uguale o supere a 30 euro il costo di spedizione sarà a carico del negozio pertanto sarà nullo per l'utente.

# Glossario dei termini principali

| Termine | Descrizione | Sinonimi | Termini Collegati |
|---|---|---|---|
| Utente | Persona che effettua acquisti di prodotti nel negozio online | Cliente | Spedizione, Ordine, Pagamento |
| Prodotti | Merce elettronica venduta dal negozio online | Merci | Fornitori, Magazzino, Categoria |
| Carrello | Composizione di più prodotti scelti dall'utente prima dell'acquisto | | Prodotto |
| Categoria | Tipologia a cui fa riferimento un prodotto | Informatica, Pc, Software, Accessori, Casalinghi, TV/ Audio, Telefonia | Prodotto |
| Fornitore | Ditta che rifornisce prodotti eletronici | | Prodotto |
| Magazzino | Deposito nel quale vengono stipati i prodotti | | Prodotto |
| Ordine | Raccolta di uno o più prodotti acquistati dall'utente in un unico pagamento | | Prodotti, Pagamento, Spedizione |
| Pagamento | Transazione di denaro effettuata da un utente a beneficio del negozio per l'acquisto dei prodotti in un dato carrello | Acquisto | Carrello, Ordine, Utente, Prodotto, Spedizione |
| Spedizione | Consegna dell'ordine all'utente da parte di una ditta di spedizione | | Ordine, Ditta spedizione |
| Ditta Spedizione | Ditta che gestisce le consegne degli ordini | | Spedizione, Ordine |

# Operazioni

1. Aggiunta di un utente (100 volte al mese)
2. Aggiunta di un fornitore (2 volte al mese)
3. Aggiunta di una ditta di spedizioni (2 volte al mese)
4. Aggiunta di un prodotto in negozio (10 volte al mese)
5. Aggiunta di un carrello (100 volte al mese)
6. Aggiunta di un pagamento (300 volte al mese)
7. Aggiunta di una spedizione (300 volte al mese)//una spedizione per ogni ordine
8. Assegnazione di una spedizione ad un ditta di spedizione (300 volte al mese)
9. Aggiunta di un ordine (300 volte al mese)
10. Aggiunta di un prodotto al carrello (2(numero prodotti)* 300(numero carrelli) = 600 volte al mese)
11. Cancellazione di un utente (10 volte al mese)
12. Cancellazione di un fornitore (1 volte al mese)
13. Cancellazione di una ditta di spedizioni (1 volta al mese)
14. Cancellazione di un prodotto (5 volte al mese)
15. Cancellazione di un pagamento (30 volte al mese)
16. Cancellazione di un ordine (30 volte al mese)
17. Cancellazione di una spedizione (30 volte al mese)
18. Cancellazione di un prodotto dal carrello (50 volte al mese)
19. Modifica dati di un utente (20 volte al mese)
20. Modifica dati fornitore (3 volte al mese)
21. Modifica dati ditta di spedizione ( 3 volte al mese)
22. Modifica di prezzo di un prodotto (40 al mese)
23. Stampa lista degli utenti (1 volta al mese)
24. Stampa lista dei fornitori (1 volta al mese)
25. Stampa la lista delle Spedizioni assegnate ad una ditta di spedizioni (20 volte al mese)
26. Stampa l’inventario dei prodotti nei magazzini (1 volta al mese)

27. Stampa la lista dei pagamenti effettuati da un utente (3 volta al mese)
28. Stampa la lista dei prodotti di una categoria (1 volta al mese)
29. Stampa i prodotti con lo stesso marchio (1 volta al mese)
30. Rifornimento di un prodotto (300 volte al mese)

# Schema Entità-Relazione

Il punto di partenza per la realizzazione del modello concettuale sono i concetti chiave di Utente e Prodotto:

- Utente rappresenta tutti gli iscritti al sito del negozio
- Prodotto rappresenta tutti i prodotti attualmente disponibili nel negozio

Utilizzeremo una strategia di raffinamento ibrida infatti andremo prima a raffinare i concetti principali mediante una strategia Top-Down e poi andremo ad aggiungere altri concetti a macchia d'olio.

Lo scheletro iniziale è composto da due entità UTENTE e PRODOTTO legate dalla relazione Acquisto le cui occorrenze rappresentano i prodotti acquistati da ogni utente.


Poiché questo schema non rappresenta in maniera precisa tutte le relazioni tra le due entità coinvolte andremo a raffinare alcuni aspetti.

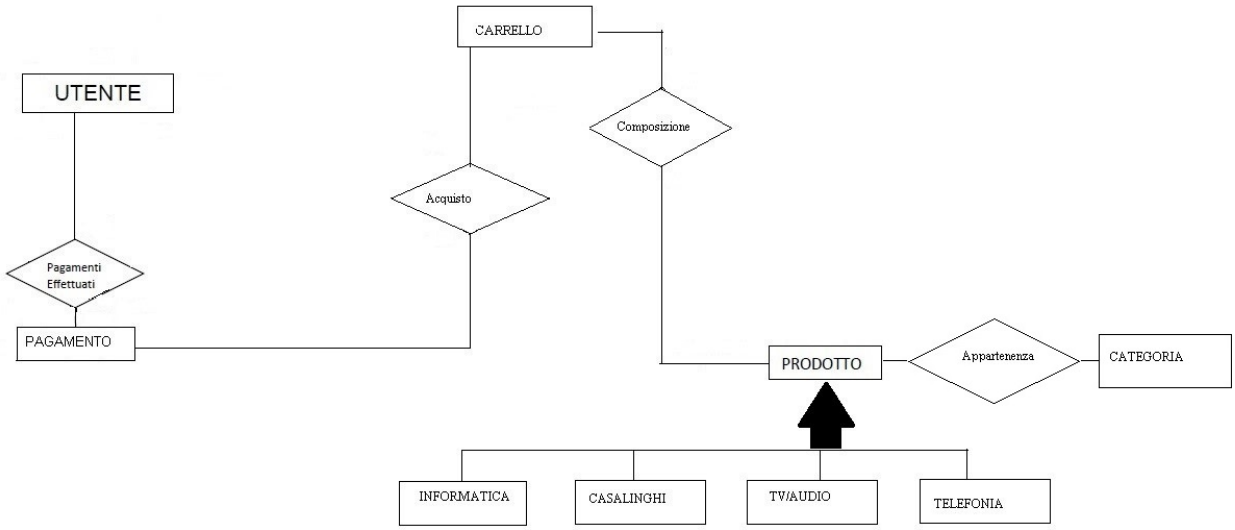
Per poter tener traccia di tutti i pagamenti effettuati da ogni utente inseriamo l'entità PAGAMENTO relazionata con utente tramite la relazione Pagamenti Effettuati.

Ogni pagamento corrisponde all'acquisto di un CARRELLO ovvero alla composizione di prodotti selezionati dall'utente.

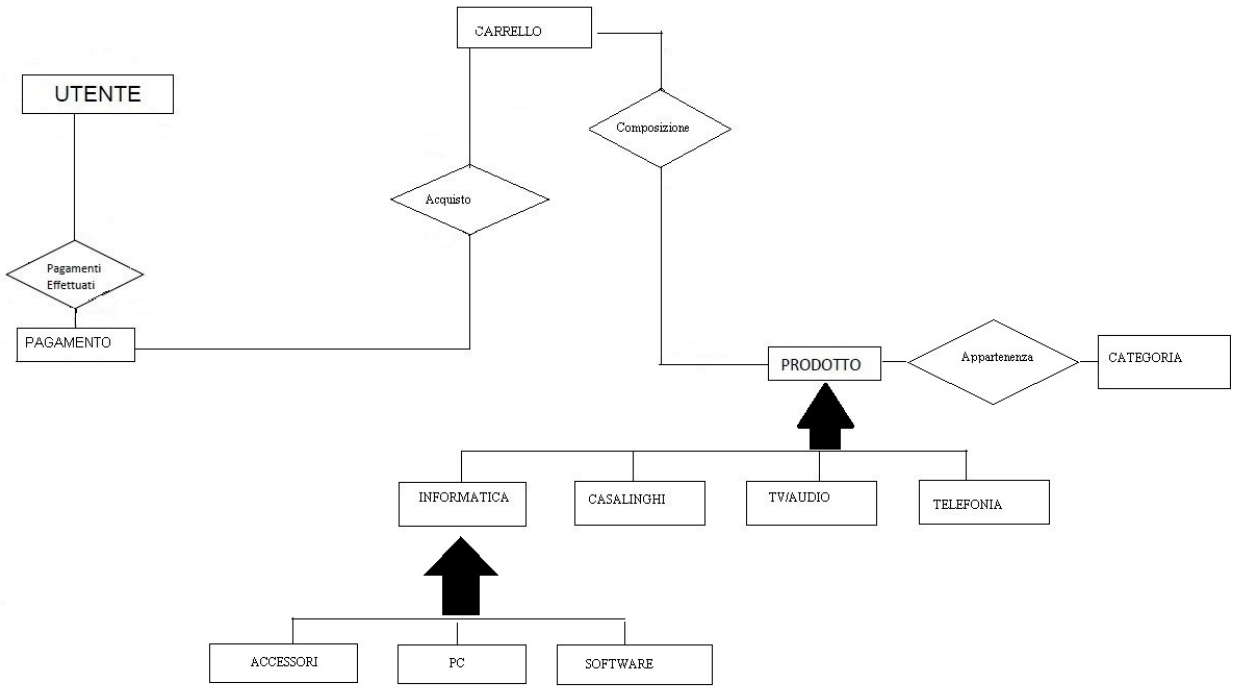
Abbiamo aggiunto l'entità CATEGORIA che rappresenta la lista delle tipologie di prodotti presenti nel negozio.

Aggiungiamo poi una generalizzazione totale ed esclusiva di Prodotto creando le entità INFORMATICA, CASALINGHI, TV/AUDIO e TELEFONIA le quali rappresentano le liste di prodotti presenti nelle categorie.

Lo schema è il seguente.

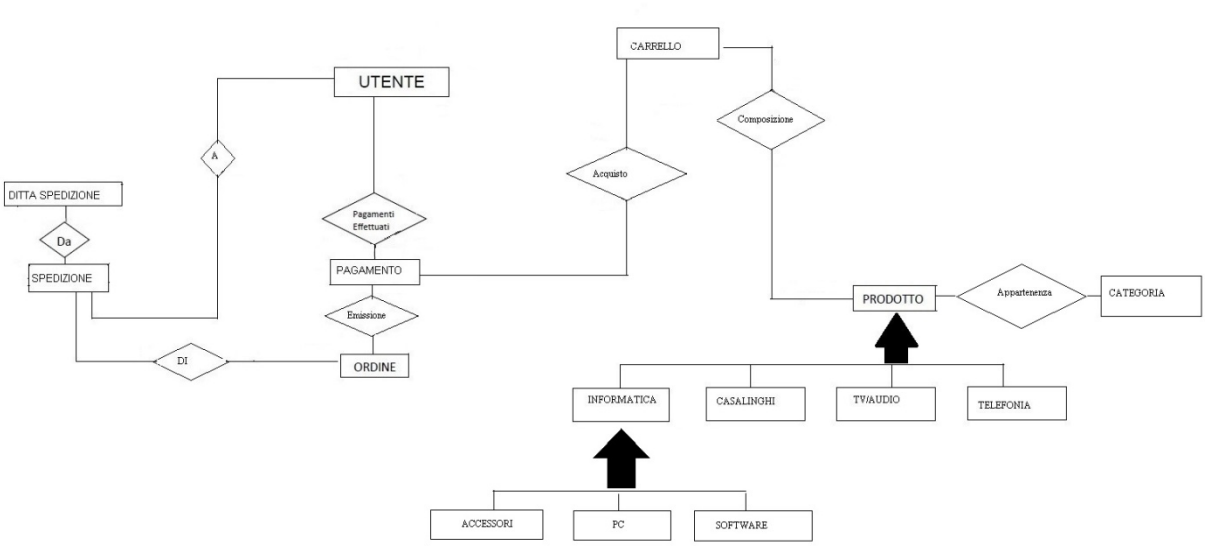



Allo stesso modo aggiungiamo la generalizzazione interna di INFORMATICA inserendo le entità ACCESSORI, PC e SOFTWARE.



Poiché ci viene richiesto nei requisiti di gestire ogni ordine, espandiamo lo schema  assumendo che l'entità ORDINE rappresenti un carrello acquistato da un utente, quindi corrispondente all' emissione di un pagamento.

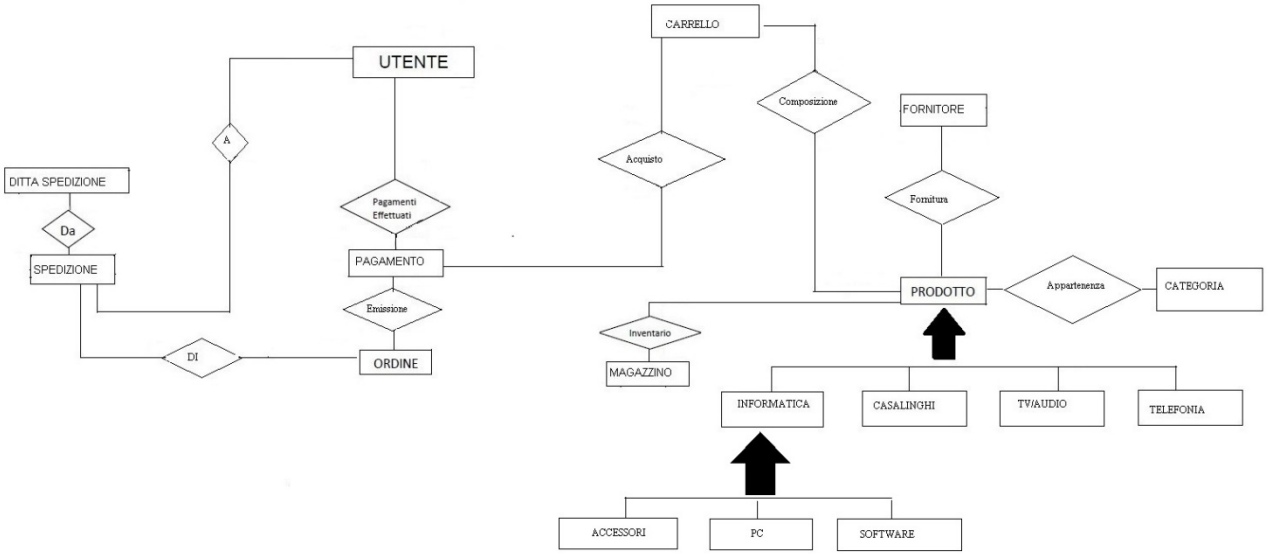
Un'ordine genera quindi una spedizione verso l'utente.




Espandiamo ulteriormente lo schema aggiungendo l'entità FORNITORI collegata a PRODOTTO mediante la relazione Fornitura che rappresenta i fornitori di ogni prodotto.

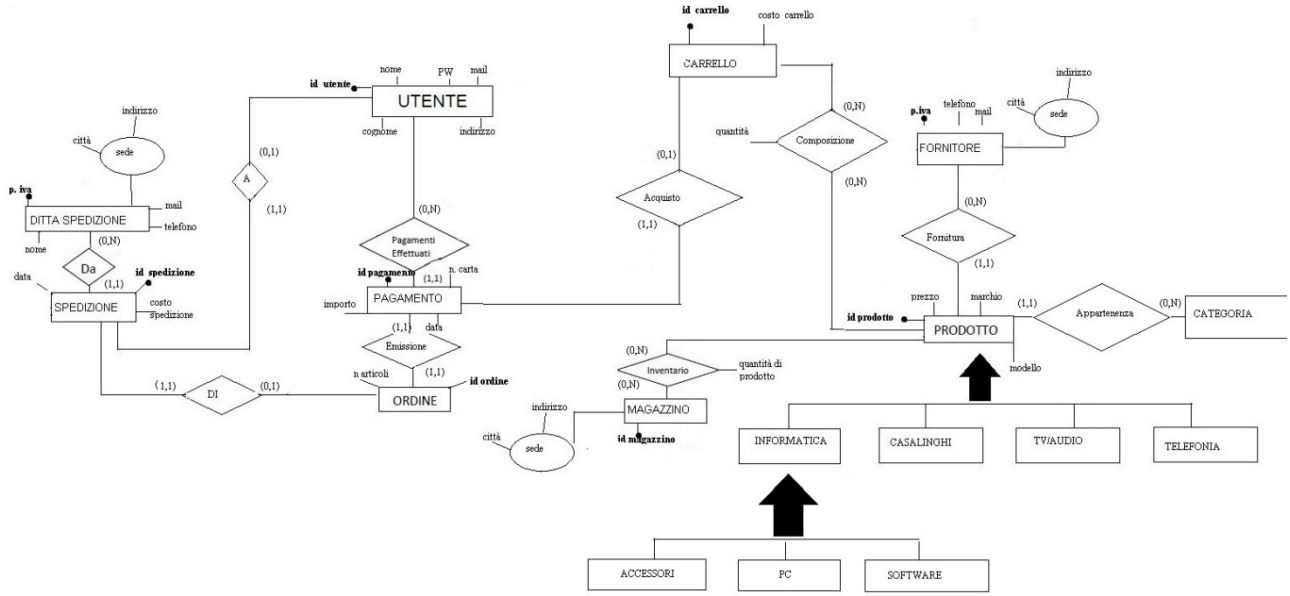
Ogni prodotto fornito viene quindi inventariato in uno dei magazzini del negozio.

Lo schema che abbiamo realizzato è il seguente.



Per terminare la fase di progettazione concettuale andiamo ad aggiungere informazioni mediante gli attributi e le cardinalità delle relazioni.

La maggior parte di queste informazioni sono ricavate dall'analisi delle specifiche, mentre alcuni attributi sono invece aggiunti per rendere più veritiera la descrizione della realtà rappresentata dallo schema.




## Vincoli non esprimibili

Una spedizione si paga soltanto se il totale del costo dei prodotti è inferiore a 30€

# Dizionario dei dati (Entità)

| Entità | Identificatore | Attributi | Descrizione |
|---|---|---|---|
| Utente | Seriale | nome, pw, mail, città, cognome, indirizzo | Persona che effettua acquisti |
| Ditta spedizioni | P_iva | Mail, sede, telefono, nome | Ditta che gestisce le consegne |
| Spedizione | Codice | data, costo_spedizione | Consegna dell'ordine a un utente |
| Pagamento | Numero | importo, n_carta, data | Transazione di denaro a favore del Negozio |
| Ordine | Numero | n.articoli | Prodotto/i pagati in un pagamento |
| Carrello | Numero | costo_carello | Composizione dei prodotti scelti da un utente prima dell'acquisto |
| Prodotto | Numero | Prezzo, marchio, modello | Merce del negozio |
| Magazzino | Nome | Sede | Deposito in qui vengono stipati i prodotti |
| Categoria | Nome | | Tipologia a cui fa riferimento un prodotto |
| Fornitore | P_iva | Indirizzo, mail | Ditta che fornisce prodotti |
| Informatica | Numero | | Insieme dei prodotto informatica |
| Casalinghi | Numero | | Insieme dei prodotto dedicati alla casa |
| Tv/Audio | Numero | | Insieme dei prodotto Tv/Audio |
| Telefonia | Numero | | Insieme dei prodotto Telefonici |
| Pc | Numero | | Insieme dei PC |
| Software | Numero | | Insieme dei Software |
| Accessori | Numero | | Insieme degli Accessori |

# Dizionario dei dati (Relazioni)

| Relazioni | Identificatori | Attributi | Descrizione |
|---|---|---|---|
| Da | Ditta_spedizione, Spedizione | | Associazione di una spedizione a una ditta |
| Di | Spedizione, Ordine | | Associazione di una spedizione a un ordine |
| A | Spedizione, Utente | | Associazione di una spedizione a un utente |
| Pagamenti Effettuati | Utente, Pagamento | | Associazione di un pagamento a un utente |
| Emissione | Pagamento, Ordine | | Emissione di un ordine dopo un pagamento |
| Acquisto | Carrello, Pagamento | | Acquisto di un carrello |
| Composzione | Carrello, Prodotto | Quantità | Insieme dei prodotti in un carrello |
| Fornitura | Fornitore, Prodotto | | Fornitura di un fornitore |
| Appartenenza | Prodotto, Categoria | | Appartenenza di un prodotto a una categoria |
| Inventario | Prodotto, Magazzino | Quantità di prodotto | Insieme di tutti i prodotti di tutti i magazzini |

# PROGETTAZIONE LOGICA

## Tavola dei volumi

Ipotizziamo che il database possa essere utilizzato per almeno 10 anni e sviluppiamo quindi la tavola dei volumi.

| Concetto | Tipo | Volume |
|---|---|---|
| Utente | E | 12000 |
| Prodotto | E | 1200 |
| Carrello | E | 60000 |
| Pagamento | E | 36000 |
| Fornitore | E | 240 |
| Ordine | E | 36000 |
| Categoria | E | 4 |
| Informatica | E | 300 |
| Casalinghi | E | 300 |
| TV/Audio | E | 300 |
| Telefonia | E | 300 |
| Pc | E | 100 |
| Software | E | 100 |
| Accessori | E | 100 |
| Magazzino | E | 5 |
| Ditta spedizione | E | 240 |
| Spedizione | E | 36000 |
| Fornitura | R | 1200 |
| Pagamenti Effettuati | R | 36000 |
| Emissione | R | 36000 |
| Da (Ditta spedizioni - spedizione) | R | 36000 |
| Di (Ordine - Spedizioni) | R | 36000 |
| A (Utente-Spedizione) | R | 36000 |
| Acquisto | R | 36000 |
| Composizione | R | 720000(60000 Carelli * 1200 Prodotti) |

| Concetto | Tipo | Volume |
|---|---|---|
| Appartenenza | R | 1200 |
| Inventario | R | 6000 (5 Magazzini * 1200 Prodotto) |

# TAVOLA DELLE OPERAZIONI

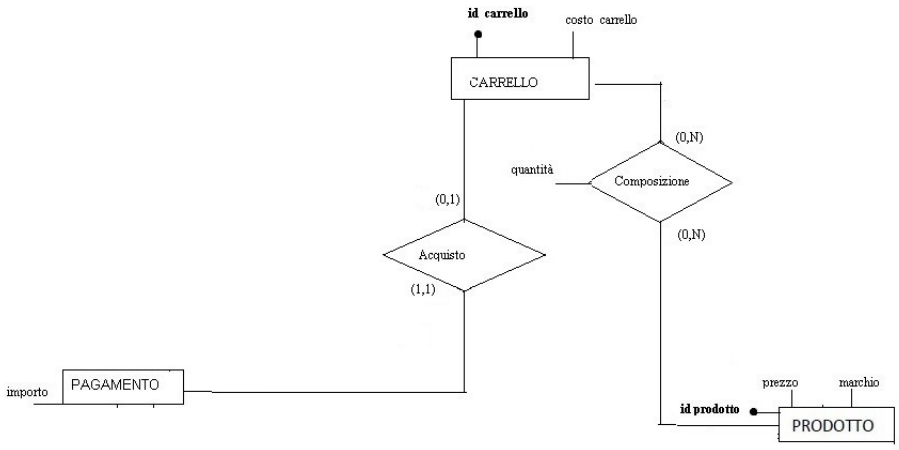
| Operazioni | Tipo | Frequenza |
|---|---|---|
| 1. Aggiunta di un utente | I | 100 volte al mese |
| 2. Aggiunta di un fornitore | I | 2 volte al mese |
| 3. Aggiunta di una ditta di spedizione | I | 2 volte al mese |
| 4. Aggiunta di un prodotto in negozio | I | 10 volte al mese |
| 5. Aggiunta di un Carrello | I | 100 volte al mese |
| 6. Aggiunta di un pagamento | I | 300 volta al mese |
| 7. Aggiunta di una spedizione | B | 300 volte al mese |
| 8. Assegnazione di una spedizione ad una ditta di spedizione | I | 300 volte al mese |
| 9. Aggiunta un ordine | B | 300 volte al mese |
| 10. Aggiunta di un prodotto al carrello | I | (2(numero prodotti in media)* 300(numero ordini) = 600 volta al mese |
| 11. Cancellazione di un utente | I | 10 volta al mese |
| 12. Cancellazione di un fornitore | I | 1 volte al mese |
| 13. Cancellazione di una ditta di spedizione | I | 1 volte al mese |
| 14. Cancellazione di un prodotto | I | 5 volte al mese |
| 15. Cancellazione di un pagamento | I | 30 volte al mese |
| 16. Cancellazione di un ordine | B | 30 volte al mese |
| 17. Cancellazione di una spedizione | B | 30 volte al mese |
| 18. Cancellazione di un prodotto dal carrello | I | 50 volte al mese |
| 19. Modifica dati di un utente | I | 20 volte al mese |
| 20. Modifica dati di un fornitore | I | 3 volte al mese |
| 21. Modifica dati ditta di spedizione | I | 3 volte al mese |
| 22. Modifica del prezzo di un prodotto | I | 40 volte al mese |
| 23. Stampa lista degli utenti | B | 1 volta al mese |
| 24. Stampa lista dei fornitori | B | 1 volte al mese |
| 25. Stampa la lista delle spedizioni assegnate ad una ditta di spedizioni | I | 20 volte al mese |
| 26. Stampa l'inventario dei prodotti nei magazzini | B | 1 volte al mese |

| Operazioni | Tipo | Frequenza |
|---|---|---|
| 27. Stampa la lista dei pagamenti effettuati da un utente | I | 3 volta al mese |
| 28. Stampa la lista dei prodotti di una categoria | I | 1 volta al mese |
| 29. Stampa la lista dei prodotti con un dato marchio | I | 1 volta al mese |
| 30. Rifornimento di un prodotto | I | 300 volte al mese |

# ANALISI DELLE RIDONDANZE

Una possibile ridondanza è quelle che riguarda il costo del carrello nella relazione carrello, questo è infatti utilizzato per il calcolo dell'importo in Pagamento ma è possibile calcolare il costo del Carrello senza doverlo memorizzare passando per la relazione Composizione e l'entità Prodotto.

SCHEMA CON RIDONDANZA:




Analizziamo e confrontiamo gli indici di prestazione nei due casi con e senza ridondanza, le operazioni che coinvolgono le entità e le relazioni nello schema sono le seguenti:

OP6: Aggiunta di un pagamento (300 volte al mese)

OP10: Aggiunta di un prodotto al carrello (1000 volte al mese)

OP18: Cancellazione di un prodotto dal carrello (50 volte al mese)

In presenza di ridondanza ho le seguenti tavole degli accessi:

OP6:

| Concetto | Costrutto | Accessi |
|---|---|---|
| Pagamento | Entità | 1 Scrittura |
| Acquisto | Relazione | 1 Scrittura |
| Carrello | Entità | 1 Lettura |

OP10:

| Concetto | Costrutto | Accessi |
|---|---|---|
| Prodotto | Entità | 1 Lettura |
| Carrello | Entità | 1 Lettura |
| Composizione | Relazione | 1 Scrittura |
| Composizione | Relazione | 1 Lettura |
| Carrello | Entità | 1 Scrittura |

OP18:

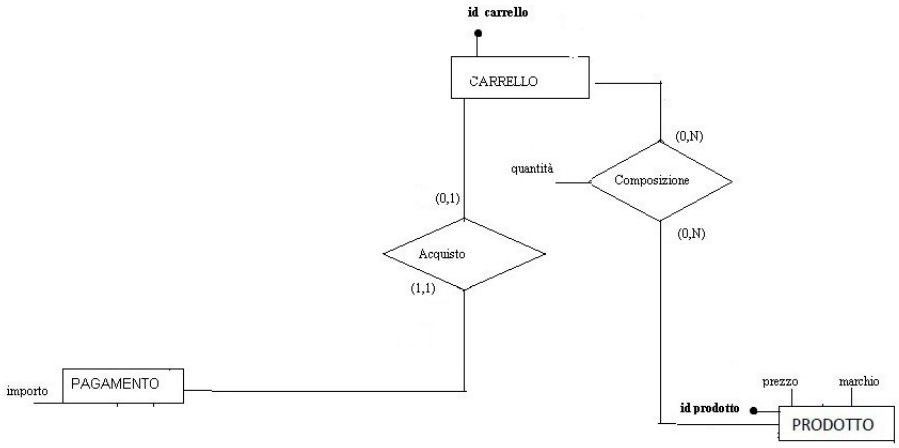
| Concetto | Costrutto | Accessi |
|---|---|---|
| Prodotto | Entità | 1 Lettura |
| Carrello | Entità | 1 Lettura |
| Composizione | Relazione | 1 Scrittura |
| Composizione | Relazione | 1 Lettura |
| Carrello | Entità | 1 Scrittura |

Considerando che il peso della scrittura è doppio rispetto al peso della lettura abbiamo che :

(2S + 1L) * 300 +(2S + 3L) * 1000 + (2S + 3L) *50 = (5L) * 300 + (7L) * 1000 + (7L) * 50 = 1500 + 7000 + 350 = 8850letture al mese

Poiché il numero di carrelli è 60000 e lo spazio per memorizzare un singolo costo del carrello è di 2 byte avremo uno spreco di memoria pari a 120000 Byte ovvero 120 Kbyte

Valutiamo ora il costo senza ridondanza:




OP6:

| Concetto | Costrutto | Accessi |
|---|---|---|
| Carrello | Entità | 1 Lettura |
| Composizione | Relazione | 1 Lettura |
| Pagamento | Entità | 1 Scrittura |

OP10:

| Concetto | Costrutto | Accessi |
|---|---|---|
| Prodotto | Entità | 1 Lettura |
| Carrello | Entità | 1 Lettura |
| Composizione | Relazione | 1 Scrittura |

OP18:

| Concetto | Costrutto | Accessi |
|---|---|---|
| Prodotto | Entità | 1 Lettura |
| Carrello | Entità | 1 Lettura |
| Composizione | Relazione | 1 Scrittura |

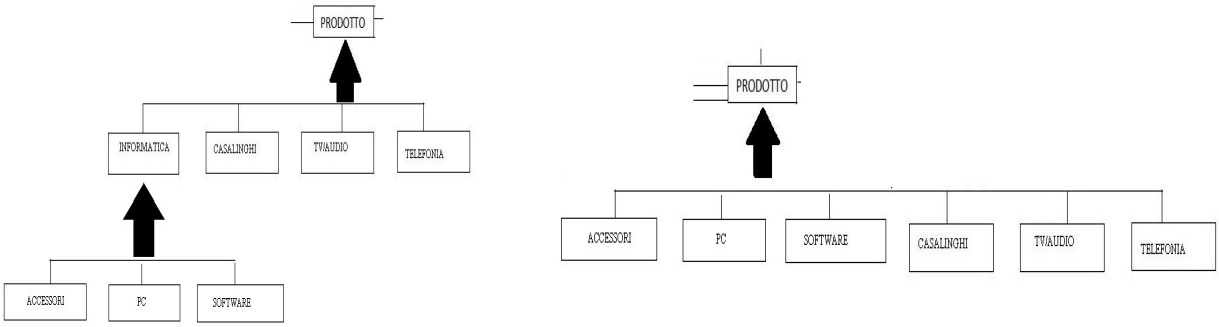
Considerando che il peso della scrittura è doppio rispetto al peso della lettura abbiamo che :

((1S + 2L) * 300) * 2 + (1S + 2L) * 1000 + (1S + 2L) * 50 = (4L) * 300 + (4L) *1000 + (4L) * 50 = 2400 + 4000 + 200 = 6600 letture al mese

Gli accessi senza ridondanza sono minori rispetto a quelli con la ridondanza poiché la modifica del costo carrello avviene ogni volta che un utente vuole inserire o togliere un elemento dal carrello, inoltre eliminando l'attributo costo carrello si risparmiano 120Kbyte di spazio.

## ELIMINAZIONE DELLE GENERIZZAZIONI

Lo schema finale ottenuto presenta una macro generalizzazione su più livelli riguardante l'entità prodotto, iniziamo con l'analisi della generalizzazione interna ovvero quella riguardante l'entità informatica, abbiamo deciso di optare per l'eliminazione dell'entità padre poiché nell'operazioni è d'interesse distinguere solo le entità figlie, il procedimento è mostrato nel seguente schema.



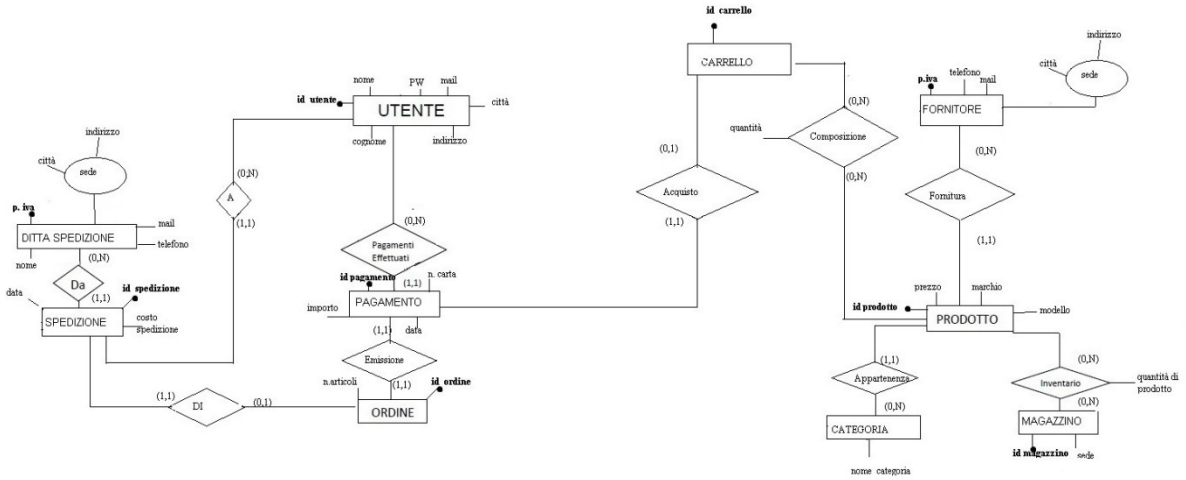

Procedendo abbiamo deciso che non era d'interesse distinguere tra le entità figlie ma era sufficiente utilizzare l'attributo categoria già presente nell'entità prodotto per distinguere correttamente le varie categorie di prodotto, attraverso una **SELECT** possiamo distinguere i vari prodotti appartenenti a differenti categorie.

## ACCORPAMENTO/PARTIZIONAMENTO DI RELATIONSHIP

Lo schema non presenta possibili accorpamenti e partizionamenti per le relationship

## SCELTA DEI IDENTIFICATORI PRIMARI

Com'è possibile notare dello schema sottostante tutte le tabelle hanno un unico identificatore interno che diviene automaticamente l'identificatore primario




Era tuttavia possibile optare per non inserire l'attributo id ordine della relazione ORDINE e creare un identificatore primario esterno con l'entità PAGAMENTO.

# MODELLO RELAZIONALE

**Traduzione Di Entità**

• UTENTE (Id_utente, Nome, Cognome, Password, Email, Indirizzo, Città)

• PAGAMENTO (Id_pagamento, N_carta, Importo, Data, Utente, Ordine, Carrello)

Con vincolo d'integrità referenziale tra l'attributo Utente e la relazione Utente, vincolo d'integrità referenziale tra l'attributo Ordine e la relazione Ordine e vincolo d'integrità referenziale tra attributo Carrello e la relazione Carrello.

• ORDINE (Id_ordine, N_articoli, Spedizione)

Con vincolo d'integrità referenziale tra l'attributo Spedizione e la relazione Spedizione

• SPEDIZIONE (Id_spedizione, Costo_spedizione, Data, Ditta_spedizione, Utente)

Con vincolo d'integrità referenziale tra l'attributo Ditta spedizione e la relazione Ditta spedizione e vincolo d'integrità referenziale tra l'attributo Utente e la relazione Utente

• DITTA_SPEDIZIONE (P_Iva, Nome, Email, Telefono, Città, Indirizzo)

• CARRELLO (Id_carrello)

• PRODOTTO (Id_prodotto, Prezzo, Categoria, Marchio,Modello, Fornitore)

Con vincolo d'integrità referenziale tra l'attributo Fornitore e la relazione Fornitore e vincolo d'integrità referenziale tra l'attributo Categoria e la relazione Categoria

• FORNITORE(P_Iva, Nome, Email, Telefono, Città, Indirizzo)

• CATEGORIA (Nome_categoria)

• MAGAZZINO (Id_magazzino, Città, Indirizzo)

**Traduzione Di Relazioni**

• COMPOSIZIONE (Carrello, Prodotto, Quantità)

Con vincolo d'integrità referenziale tra l'attributo Carrello e la relazione Carrello e vincolo d'integrità referenziale tra l'attributo Prodotto e la relazione Prodotto

• INVENTARIO (Prodotto, Magazzino, Quantità)

Con vincolo d'integrità referenziale tra l'attributo Prodotto e la relazione Prodotto e vincolo d'integrità referenziale tra l'attributo Magazzino e la relazione Magazzino.

# IMPLEMENTAZIONE DELLE OPERAZIONI

## Creazione delle tabelle

**CREATE TABLE "CARRELLO"**
(
"Id_Carrello" integer NOT NULL,
  CONSTRAINT "Carrello_pkey" PRIMARY KEY ("Id_Carrello")
)

**CREATE TABLE "CATEGORIA"**
(
"Nome_Categoria" character varying(3) NOT NULL,
  CONSTRAINT "Categoria_pkey" PRIMARY KEY ("Nome_Categoria")
)

**CREATE TABLE "COMPOSIZIONE"**
(
"Carrello" integer NOT NULL,
"Prodotto" integer NOT NULL,
"Quantità" integer,
  CONSTRAINT "Composizione_pkey" PRIMARY KEY ("Carrello", "Prodotto"),
  CONSTRAINT "Composizione_Carrello_fkey" FOREIGN KEY ("Carrello")
      REFERENCES "CARRELLO" ("Id_Carrello") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE CASCADE,
  CONSTRAINT "Composizione_Prodotto_fkey" FOREIGN KEY ("Prodotto")
      REFERENCES "PRODOTTO" ("Id_Prodotto") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE CASCADE
)

**CREATE TABLE "DITTA SPEDIZIONE"**
(
"P_Iva" character(11) NOT NULL,
"Nome" character varying NOT NULL,
"Mail" character varying,
"Telefono" integer,
"Città" character varying,
"Indirizzo" character varying,
  CONSTRAINT "Ditte_Spedizione_pkey" PRIMARY KEY ("P_Iva")
)

**CREATE TABLE "FORNITORE"**
(
"P_Iva" character(11) NOT NULL,
"Nome_Fornitore" character varying(32) NOT NULL,

```
"Telefono" integer,
"Mail" character varying,
"Città" character varying,
"Indirizzo" character varying,
  CONSTRAINT "Fornitore_pkey" PRIMARY KEY ("P_Iva"),
  CONSTRAINT "FORNITORE_Mail_key" UNIQUE ("Mail")
)
```

**CREATE TABLE "INVENTARIO"**

```
(
"Prodotto" integer NOT NULL,
"Quantità" integer,
"Magazzino" character(1) NOT NULL,
  CONSTRAINT "Inventario_pkey" PRIMARY KEY ("Prodotto", "Magazzino"),
  CONSTRAINT "Inventario_Magazziono_fkey" FOREIGN KEY ("Magazzino")
      REFERENCES "MAGAZZINO" ("Id_Magazzino") MATCH SIMPLE
      ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT "Inventario_Prodotto_fkey" FOREIGN KEY ("Prodotto")
      REFERENCES "PRODOTTO" ("Id_Prodotto") MATCH SIMPLE
      ON UPDATE CASCADE ON DELETE CASCADE
)
```

**CREATE TABLE "MAGAZZINO"**

```
(
"Id_Magazzino" character(1) NOT NULL,
"Città" character varying(16),
"Indirizzo" character varying(16),
  CONSTRAINT "Magazzino_pkey" PRIMARY KEY ("Id_Magazzino")
)
```

**CREATE TABLE "ORDINE"**

```
CREATE TABLE "ORDINE"

(

  "Id_Ordine" integer NOT NULL DEFAULT 1,

  "N_Articoli" integer,

  "Spedizione" integer,

  "Pagamento" integer,

  CONSTRAINT "Ordine_pkey" PRIMARY KEY ("Id_Ordine"),

  CONSTRAINT "Ordine_Pagamento_fkey" FOREIGN KEY ("Pagamento")

      REFERENCES "PAGAMENTO" ("Id_Pagamento") MATCH SIMPLE
```

```
    ON UPDATE CASCADE ON DELETE CASCADE
)
WITH (
  OIDS=FALSE
);
ALTER TABLE "ORDINE"
  OWNER TO postgres;
```

**CREATE TABLE "PAGAMENTO"**

```
(
"N_carta" integer NOT NULL,
"Utente" integer NOT NULL,
"Ordine" integer NOT NULL DEFAULT 1,
"Data_Pagamento" date,
"Id_Pagamento" integer NOT NULL DEFAULT 1,
"Carrello" integer NOT NULL,
"Importo" double precision,
  CONSTRAINT "Pagamento_pkey" PRIMARY KEY ("Id_Pagamento"),
  CONSTRAINT "Pagamento_Carrello_fkey" FOREIGN KEY ("Carrello")
      REFERENCES "CARRELLO" ("Id_Carrello") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE NO ACTION,
  CONSTRAINT "Pagamento_Utente" FOREIGN KEY ("Utente")
      REFERENCES "UTENTE" ("Id_Utente") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE CASCADE
)
```

**CREATE TABLE "PRODOTTO"**

```
(
"Id_Prodotto" integer NOT NULL,
"Prezzo" double precision NOT NULL,
"Categoria" character(3),
"Marchio" character varying,
"Fornitore" character(11),
"Modello" character varying,
  CONSTRAINT "Prodotto_pkey" PRIMARY KEY ("Id_Prodotto"),
  CONSTRAINT "Prodotto_Categoria_fkey" FOREIGN KEY ("Categoria")
      REFERENCES "CATEGORIA" ("Nome_Categoria") MATCH SIMPLE
      ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT "Prodotto_Fornitore_fkey" FOREIGN KEY ("Fornitore")
      REFERENCES "FORNITORE" ("P_Iva") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE CASCADE
)
```

```
CREATE TABLE "SPEDIZIONE"
(
"Id_Spedizione" integer NOT NULL,
"Data_Spedizione" date,
"Costo_Spedizione" double precision,
"Ditta_di_Spedizione" character(11),
"Ordine" integer NOT NULL,
"Utente" integer NOT NULL,
  CONSTRAINT "Spedizione_pkey" PRIMARY KEY ("Id_Spedizione"),
  CONSTRAINT "Spedizione_Ditta" FOREIGN KEY ("Ditta_di_Spedizione")
      REFERENCES "DITTA SPEDIZIONE" ("P_Iva") MATCH SIMPLE
      ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT "Spedizione_Ordine_fkey" FOREIGN KEY ("Ordine")
      REFERENCES "ORDINE" ("Id_Ordine") MATCH SIMPLE
      ON UPDATE CASCADE ON DELETE CASCADE,
  CONSTRAINT "Spedizione_Utente_fkey" FOREIGN KEY ("Utente")
      REFERENCES "UTENTE" ("Id_Utente") MATCH SIMPLE
      ON UPDATE NO ACTION ON DELETE CASCADE
)
```

```
CREATE TABLE "UTENTE"
(
"Id_Utente" integer NOT NULL,
"Nome_Utente" character varying NOT NULL,
"Cognome_Utente" character varying NOT NULL,
"Password" character varying NOT NULL,
"Mail" character varying,
"Indirizzo" character varying,
"Città" character varying,
"N_Pagamenti" integer DEFAULT 0,
  CONSTRAINT "Utente_pkey" PRIMARY KEY ("Id_Utente"),
  CONSTRAINT "UTENTE_Mail_key" UNIQUE ("Mail")
)
```

# Creazione delle query

1. **Aggiunta di un utente:**

   INSERT INTO "UTENTE" ("Id_Utente", "Nome_Utente", "Cognome_Utente", "Password", "Mail","Città"," "Indirizzo")

   VALUES (integer, 'character', 'character', 'character', 'character','character','character');

2. **Aggiunta di un fornitore:**

   INSERT INTO "FORNITORE" ("P_Iva", "Nome_Fornitore","Telefono", "Mail", "Città","Indirizzo")

   VALUES ('character (11)',Character varying,integer, Character varying, Character varying,Character varying);

3. **Aggiunta di una ditta di spedizioni :**

   INSERT INTO "DITTA SPEDIZIONE" ("P_Iva", "Nome","Telefono", "Mail", "Città","Indirizzo")

   VALUES ('character (11)',Character varying,integer, Character varying, Character varying,Character varying);Aggiunta di un prodotto :

```
CREATE OR REPLACE FUNCTION "Aggiungi Prodotto In Negozio"(
    id_prodotto integer,
    prezzo double precision,
    categoria character varying,
    marchio character varying,
    modello character varying,
    fornitore character varying)
  RETURNS void AS
$BODY$
BEGIN
INSERT INTO
"PRODOTTO"("Id_Prodotto","Prezzo","Categoria","Marchio","Modello","Fornitore")
     VALUES (id_prodotto,prezzo,categoria,marchio,modello,fornitore);
INSERT INTO "INVENTARIO" ("Prodotto","Magazzino", "Quantità")
     VALUES (id_prodotto,'E',0);

END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Aggiungi Prodotto In Negozio"(integer, double precision, character
varying, character varying, character varying, character varying)
  OWNER TO postgres;
```

4. **Aggiunta di un carrello :**

l'aggiunta di un carrello è possibile automatizzarla tramite la seguente funzione

```
CREATE OR REPLACE FUNCTION "Aggiungi Carrello"()
  RETURNS void AS
$BODY$DECLARE id_carrello integer := (SELECT (max("Id_Carrello")+1) FROM
"CARRELLO");

BEGIN
IF id_carrello IS NULL THEN
    id_carrello= 1;
END IF;
INSERT INTO "CARRELLO" (
    "Id_Carrello")
VALUES (id_carrello);
END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Aggiungi Carrello"()
  OWNER TO postgres;
```

5. **Aggiunta di un prodotto al carrello:**

```
CREATE OR REPLACE FUNCTION "Aggiungi_Prodotto_Nel_Carrello"(

   id_carrello integer,

   id_prodotto integer,

   quantita integer)

  RETURNS void AS

$BODY$


DECLARE magazzino character := 'Z';

BEGIN

IF quantita <= (SELECT "Quantità" FROM "INVENTARIO" WHERE id_prodotto = "Prodotto"
AND "Magazzino"='A') THEN

    magazzino = 'A';


ELSIF quantita <= (SELECT "Quantità" FROM "INVENTARIO" WHERE id_prodotto =
"Prodotto" AND "Magazzino"='B') THEN

    magazzino = 'B';


ELSIF quantita <= (SELECT "Quantità" FROM "INVENTARIO" WHERE id_prodotto =
"Prodotto" AND "Magazzino"='C') THEN
```

```
    magazzino = 'C';


ELSIF quantita <= (SELECT "Quantità" FROM "INVENTARIO" WHERE id_prodotto =
"Prodotto" AND "Magazzino"='D') THEN
    magazzino = 'D';


ELSIF quantita <= (SELECT "Quantità" FROM "INVENTARIO" WHERE id_prodotto =
"Prodotto" AND "Magazzino"='E') THEN
    magazzino = 'E';


END IF;


IF magazzino <>'Z' THEN
INSERT INTO "COMPOSIZIONE" (
        "Carrello", "Prodotto", "Quantità")
VALUES (id_carrello, id_prodotto, quantita);


UPDATE "INVENTARIO"
    SET "Quantità"= "Quantità" - quantita
    WHERE "Prodotto"=id_prodotto AND "Magazzino"= magazzino;
ELSE
    RAISE 'PRODOTTO NON DISPONIBILE';
END IF;


END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Aggiungi_Prodotto_Nel_Carrello"(integer, integer, integer)
OWNER TO postgres;
```

6. **Aggiunta di un pagamento :**

```
CREATE OR REPLACE FUNCTION "Aggiungi_Pagamento"(
    carta integer,
    utente integer,
    carrello integer,
    data_pagamento date)
  RETURNS void AS
$BODY$
DECLARE id_pagamento integer := (SELECT (max("Id_Pagamento")+1) FROM
"PAGAMENTO");

DECLARE id_ordine integer := (SELECT (max("Id_Ordine")+1) FROM "ORDINE");

DECLARE id_spedizione integer := (SELECT (max("Id_Spedizione")+1) FROM
"SPEDIZIONE");

DECLARE importo double precision := (SELECT sum("Prezzo"*"Quantità") FROM
"COMPOSIZIONE","PRODOTTO" WHERE carrello="Carrello" AND "Prodotto" =
"Id_Prodotto");

DECLARE costo_sped double precision := 0.0;

DECLARE n_articoli integer := (SELECT sum("Quantità") FROM "COMPOSIZIONE" WHERE
carrello="Carrello");


BEGIN
IF id_pagamento IS NULL THEN
    id_pagamento = 10000000;
END IF;

IF id_ordine IS NULL THEN
    id_ordine= 50000000;
END IF;

IF id_spedizione IS NULL THEN
    id_spedizione = 90000000;
END IF;

IF importo <= 30.0 THEN
    costo_sped = 5.0;
    importo = importo + costo_sped;
END IF;

IF importo<> 0 THEN
```

```
    INSERT INTO "PAGAMENTO"(
                "N_carta", "Utente", "Ordine",
"Data_Pagamento","Id_Pagamento","Carrello","Importo")
        VALUES ( carta, utente, id_ordine,
                data_pagamento,id_pagamento,carrello,importo);

    Update "UTENTE"
    SET "N_Pagamenti" = "N_Pagamenti" + 1
    Where "Id_Utente"=utente;

    INSERT INTO "ORDINE" (
                "Id_Ordine", "Spedizione", "N_Articoli","Pagamento")
    VALUES (id_ordine, id_spedizione,n_articoli,id_pagamento);



    INSERT INTO "SPEDIZIONE" (
                "Id_Spedizione","Ordine","Utente","Costo_Spedizione")
    VALUES (id_spedizione, id_ordine, utente,costo_sped);

    DELETE FROM "COMPOSIZIONE"
    WHERE "Carrello"=carrello;
ELSE
    RAISE 'CARRELLO VUOTO';
END IF;

END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Aggiungi_Pagamento"(integer, integer, integer, date)
  OWNER TO postgres;
```

10. **Assegnazione di una spedizione ad un ditta di spedizioni :**

```
CREATE OR REPLACE FUNCTION "Assegana Spedizione a Ditta"(
    id_spedizione integer,
    data_sped date,
    p_iva_ditta character)
  RETURNS void AS
$BODY$BEGIN
UPDATE "SPEDIZIONE"
SET "Ditta_di_Spedizione" =p_iva_ditta,
     "Data_Spedizione" = data_sped
WHERE "Id_Spedizione" =id_spedizione;
end $BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Assegana Spedizione a Ditta"(integer, date, character)
  OWNER TO postgres;
```

11. **Cancellazione di un utente:**

```
Delete FROM "UTENTE"
WHERE "Id_Utente" =integer
```

12. **Cancellazione di un fornitore :**

```
Delete FROM "FORNITORE"
WHERE "P_Iva" = 'character'
```

13. **Cancellazione di una ditta di spedizioni :**

```
DELETE FROM "DITTA SPEDIZIONE"
WHERE "P_Iva" = 'character'
```

14. **Cancellazione di un prodotto:**

```
DELETE FROM "PRODOTTO"
WHERE "id_prodotto" = character
```

15. **Cancellazione di un pagamento :**

```
DELETE FROM "PAGAMENTO"
WHERE "id_pagamento" = integer
```

**16. Cancellazione di una spedizione :**

```
DELETE FROM "SPEDIZIONE"
WHERE "id_spedizione" = integer
```

**17. Cancellazione di un ordine :**

```
DELETE FROM "ORDINE"
WHERE "id_ordine" = integer
```

18. **Rimozione di un prodotto dal carrello:**

```
CREATE OR REPLACE FUNCTION "Rimuovi dal carrello"(
    id_carrello integer,
    id_prodotto integer)
  RETURNS void AS
$BODY$
DECLARE quantità integer :=(SELECT "Quantità" FROM "COMPOSIZIONE" WHERE
"Carrello"=id_carrello AND "Prodotto"=id_prodotto);

BEGIN
IF (SELECT "Prodotto" FROM "COMPOSIZIONE" WHERE "Prodotto" =id_prodotto AND
"Carrello" = id_carrello) IS NOT NULL THEN
     IF quantità = 1 THEN
      DELETE FROM "COMPOSIZIONE"
     WHERE "Carrello"=id_carrello AND "Prodotto"=id_prodotto;
     ELSE
     UPDATE "COMPOSIZIONE"
     SET "Quantità"= quantità -1
     WHERE "Carrello"=id_carrello AND "Prodotto"=id_prodotto;

     end if;
     UPDATE "INVENTARIO"
               SET "Quantità"= "Quantità" +1
               WHERE "Prodotto"=id_prodotto AND "Magazzino"= 'E';

ELSE
     RAISE 'NULLA DA RIMUOVERE';
end if;
END
$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Rimuovi dal carrello"(integer, integer)
  OWNER TO postgres;
```

**19. Modifica dati di un utente:**

```
UPDATE "UTENTE"

        UPDATE "UTENTE"
        SET   "Nome_Utente" ='Paolo' ,
           "Cognome_Utente" ='Paolini',
           "Password" ='Pallino',
           "Mail"= 'Paolo.Paolini@mail.com',
           "Indirizzo" ='Viale marconi 3',
            "Città" ='Spoleto'

           WHERE "Id_Utente"= integer;
```

**20. Modifica dati fornitore :**

```
        UPDATE "FORNITORE"

        SET  "Nome_Fornitore"= character varying ,

                  "Telefono"= integer,

                  "Mail" =character varying,

                  "Città" =character varying,

                  "Indirizzo"= character varying

           WHERE "P_Iva” = character varying;
```

**21. Modifica dati ditta di spedizione :**

```
      UPDATE "DITTA  SPEDIZIONE"

      SET   "Nome"= character varying ,

              "Telefono"= integer,

               "Mail" =character varying,

              "Città" =character varying,

              "Indirizzo"= character varying

           WHERE "P_Iva” = character varying;
```

22. **Modifica di prezzo di un prodotto :**

```
UPDATE "PRODOTTO"

SET         "Prezzo"= double

WHERE "Id_Prodotto” = integer;
```

23. **Stampa la lista degli utenti :**

```
Select UTENTE*
Form UTENTE
```

24. **Stampa la lista dei fornitori :**

```
Select "FORNITORE".*
From "FORNITORE"
```

25. **Stampa la lista delle Spedizioni assegnate ad una ditta di spedizioni :**

```
Select "Id_Spedizione"

From   "SPEDIZIONE"
Where  "Ditta_di_Spedizione" =  ‘character varying’
```

26. **Stampa l’inventario dei prodotti nei magazzini**

```
Select "INVENTARIO".*
From  "INVENTARIO"
```

27. **Stampa la lista dei pagamenti effettuati da un utente:**

```
SELECT "Id_Pagamento"

FROM "PAGAMENTO"

WHERE   "Utente" = integer
```

28. **Stampa la lista dei prodotti di una categoria:**

```
SELECT "PRODOTTO".*
FROM "PRODOTTO"
WHERE  "Categoria" =  'character (3)'
```

29. **Stampa la lista dei prodotti con lo stesso marchio :**

```
SELECT "PRODOTTO".*
FROM "PRODOTTO"
```

```
WHERE  "Marchio" =  'character varyng'
```



**30. Rifornimento di un prodotto:**

```
CREATE OR REPLACE FUNCTION "Rifornimento di un prodotto"(
    prodotto integer,
    magazzino character,
    "quantità" integer)
  RETURNS void AS
$BODY$BEGIN
     IF (SELECT "Prodotto","Magazzino" FROM "INVENTARIO" WHERE magazzino =
"Magazzino" and prodotto= "Prodotto") IS NOT NULL THEN
     UPDATE "INVENTARIO"
               SET "Quantità" = "Quantità" +quantità
               Where "Prodotto"=prodotto AND "Magazzino"=magazzino;
     ELSE
     INSERT INTO "INVENTARIO"("Prodotto", "Magazzino", "Quantità")
               VALUES ( prodotto, magazzino, quantità);
     END IF;
END$BODY$
  LANGUAGE plpgsql VOLATILE
  COST 100;
ALTER FUNCTION "Rifornimento di un prodotto"(integer, character, integer)
  OWNER TO postgres;
```