UNIVERSITÀ DEGLI STUDI
DI SALERNO

Corso di Laurea in Informatica, prof A.De Lucia,

a.a 2022/2023

Progetto di Ingegneria del software

Dati Persistenti

Partecipanti	Matricola
Antonio Michele Russo	0512109078

Revision History

Data	Versione	Descrizione	Autore
18/03/2023	1.0	Prima stesura del documento	Antonio Michele Russo
20/03/2023	1.1	Varie Modifiche	Antonio Michele Russo

Indice

1. GESTIONE DATI PERSISTENTI

1.1 Class Diagram

Di seguito riportiamo la parte del database che trova corrispondenza nel database utilizzato dal nostro sistema.

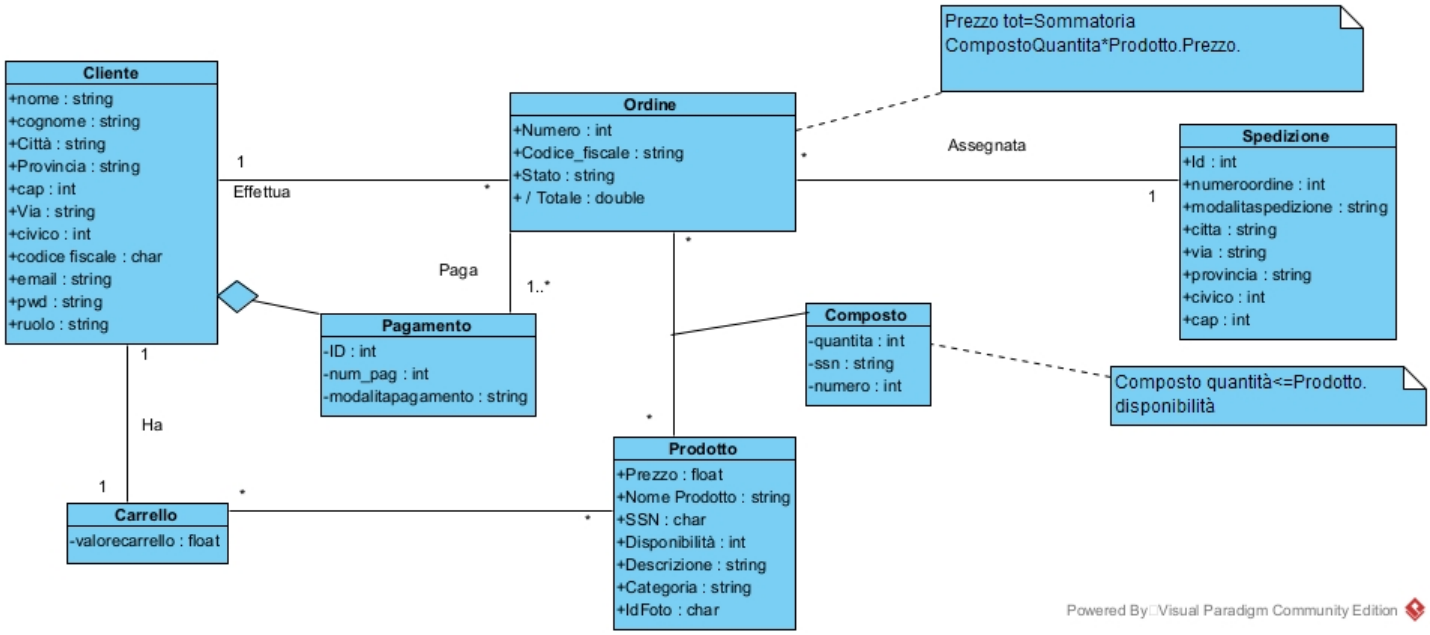
- Le informazioni delle varie tipologie di attori del sistema vengono memorizzate e quindi rappresentate dalla tabella **Utente** per la quale andiamo a definire: nome, cognome, indirizzo, città, provincia, numero civico, cap, codicefiscale, email, password, ruolo. Ciascun Utente Registrato può effettuare uno o più Ordini per questo è associato alla tabella Ordine. Un Utente Registrato possiede anche un Carrello, avremo quindi l'associazione con la tabella Carrello. Ad ogni Utente Registrato viene associato un Indirizzo di Spedizione e un Metodo di Pagamento che identificheremo con le omonime tabelle.

- La tabella **Ordine** va a specificare un determinato ordine effettuato da un Utente Registrato. Per tale tabella andiamo a specificare i seguenti attributi: numero, Codice_fiscale dell'utente, stato e Totale

- La tabella Ordine è associata alla tabella **Prodotto**, pagamento e spedizione

- I prodotti che è possibile acquistare vengono memorizzati nella tabella **Prodotto** per la quale andiamo a specificare: prezzo, nome prodotto, ssn, disponibilità, descrizione, categoria, idfoto

- Nella tabella pagamento sarà presente un id dell'ordine di riferimento, il num pagamento, modalità pagamento



- Nella tabella composto, sarà presente l'attributo quantità che specifica la quantità di prodotti presente nell'ordine, l'ssn e il numero a cui è riferito

1.2 Mapping del Database

In questo documento si è preferito non riportare il diagramma ER in quanto questo
può essere facilmente dedotto dal precedente class diagram. Riportiamo dunque direttamente il mapping del database .

```
CREATE TABLE utente (
```

```
nome    varchar(50)  not NULL,
cognome varchar(50) not NULL,
indirizzo varchar(30)        not NULL,
citta varchar(50) not NULL,
provincia varchar(50) not NULL,
ncivico int(4) not null,
cap int(5) not null,
codice_fiscale     varchar(16)         not NULL,
email    varchar(35)  not NULL,
pwd      varchar(255)   not NULL,
ruolo              varchar(255) not NULL,

primary key (codice_fiscale)


);
CREATE TABLE ordine (
numero   int   not NULL AUTO_INCREMENT ,
codice_fiscale    varchar(16) not NULL,
totale        double not NULL,
stato        varchar(35) not NULL,
 primary key (numero),

 foreign key (codice_fiscale) references utente(codice_fiscale)
 ON DELETE CASCADE

);
```

```
CREATE TABLE prodotto (
 prezzo     float        not NULL,
 nomep       varchar(10) not NULL,
 SSN        varchar(9)     not NULL,
 disponibilità int(5)   not NULL,
 descrizione varchar(50) not NULL,
 categ varchar(10) not null,
 idfoto varchar(30) not null,
  primary key (SSN),
  foreign key (categ) references categoria(categ)
```

```
 ON delete cascade
);


CREATE TABLE composto (
quantita int not null,
SSN      varchar(9) not NULL,
numero   int not NULL,
foreign key (SSN) references prodotto(SSN),
foreign key (numero) references ordine(numero)
ON delete cascade
);




CREATE TABLE amministratore (
 nome      varchar(35) not NULL,
 cognome varchar (35) not NULL,
  email    varchar(35)  not NULL,
  pwd      varchar(255)   not NULL,
  ruolo            varchar(255) not NULL,

  primary key (email)

);
```

```
CREATE TABLE spedizione (
 ID         int     not NULL auto_increment,
 numero   int not NULL,
 modalitaspedizione varchar(15) not null,
 costospedizione int not null,
 citta varchar(50) not null,
 provincia varchar(50) not null,
 cap int(5) not null,
  via varchar(50) not null,
  civico int(5) not null,
 primary key (ID),
  foreign key (numero) references ordine(numero)
);

DROP TABLE IF EXISTS pagamento;
CREATE TABLE pagamento (
 ID         int     not NULL auto_increment,
 num_pag      int  not NULL,
 modalitapagamento varchar(15) not null,
 primary key (ID),
 foreign key (num_pag) references spedizione(ID)
 );
```

1.4 Motivazioni

Si è scelto di utilizzare un DBMS di tipo relazionale poiché esso permette di accedere
in modo semplice ed efficiente ad una base di dati mantenendone la consistenza, la privatezza e l'affidabilità.
I vantaggi dell'utilizzo di un DBMS sono i seguenti:

1.2.1. Accesso ai dati tramite linguaggio SQL

Tale linguaggio permette la creazione delle strutture che contengono i dati, l'inserimento, la cancellazione, l'aggiornamento dei dati e il recupero delle informazioni dalla base di dati.

1.2.2 Accesso efficiente ai dati

Un DBMS ha molti modi per ottimizzare l'accesso all'informazione. La base di dati è solitamente memorizzata in memoria secondaria (disco rigido). Un DBMS permette di creare dei file ausiliari (indici) che permettono l'accesso veloce ai dati su disco.

1.2.3 Indipendenza dei dati

Un DBMS permette di accedere ai dati logici indipendentemente della loro rappresentazione fisica. Quest'ultima può cambiare senza che i metodi di accesso ai
dati logici debbano essere modificati. Si parla di indipendenza fisica dei dati.

1.2.4. Accesso concorrente ai dati

Un DBMS permette a più utenti di accedere contemporaneamente alla base di dati.
Più utenti possono accedere nello stesso istante a dati diversi. Inoltre, un DBMS fa in modo che l'accesso concorrente agli stessi dati non generi anomalie, cioè
inconsistenza nello stato della base di dati rispetto alla realtà modellata.