UNIVERSITÀ DEGLI STUDI DI SALERNO

Corso di Laurea in Informatica, prof. A. De Lucia,
a.a 2021/2022
Progetto di Ingegneria del Software

Gestione dei Dati Persistenti

Partecipanti	Matricola
Marta Coiro	0512108154
Katia Buonocore	0512106528
Rita Cuccaro	0512109495

Revision History

Data	Versione	Descrizione	Autore
16/11/2021	1.0	Prima stesura del documento (Problem Statement)	Membri del team
30/11/2021	1.0	Requirement Analysis Document	Membri del team
06/12/2021	1.0	System Design Document	Membri del team
20/12/2021	1.0	Gestione Dati Persistenti_MusicConsole	Membri del team
27/12/2021	1.0	Object Design Document	Membri del team
15/01/2022	1.0	Test Plan	Membri del team
28/01/2022	1.0	Test Execution Report	Membri del team
18/02/2022	1.0	Test Summary Report	Membri del team

Indice

1. GESTIONE DATI PERSISTENTI

1.1 Class Diagram

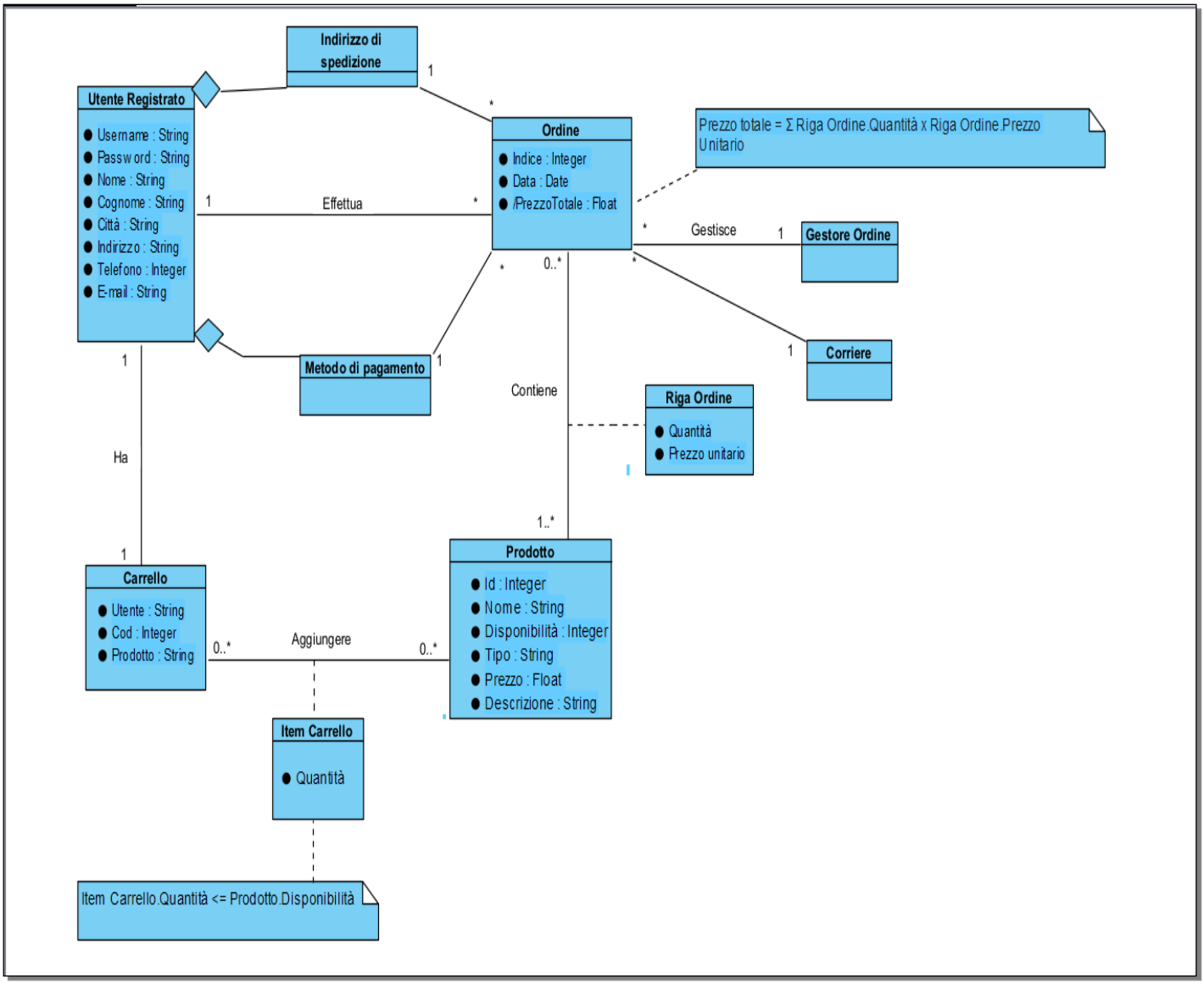
Di seguito riportiamo la parte del database che trova corrispondenza nel database utilizzato dal nostro sistema.

- Le informazioni delle varie tipologie di attori del sistema vengono memorizzate e quindi rappresentate dalla tabella **Utente Registrato** per la quale andiamo a definire: username, password, nome, cognome, città, indirizzo, telefono ed e-mail. 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: indice, data, prezzo totale. Gli ordini verranno gestiti da un Gestore Ordine e associati ad uno specifico Corriere, infatti sono associati alle omonime tabelle. La tabella Ordine è associata alla tabella **Prodotto**.

- La tabella **Carrello** è costituita dai seguenti attributi: utente, cod, prodotto. Al carrello possono essere aggiunti nessuno o più prodotti, ed è associato alla tabella Prodotto.

- I prodotti che è possibile acquistare vengono memorizzati nella tabella **Prodotto** per la quale andiamo a specificare: id, nome, disponibilità, tipo, prezzo e descizione.


Indirizzo di spedizione
1
*
Utente Registrato
Username : String
Password : String
Nome : String
Cognome : String
Città : String
Indirizzo : String
Telefono : Integer
E-mail : String
Ordine
Indice : Integer
Data : Date
/PrezzoTotale : Float
Prezzo totale = Σ Riga Ordine.Quantità x Riga Ordine.Prezzo Unitario
1
Effettua
*
*
Gestisce
1
Gestore Ordine
*
0..*
*
1
Corriere
1
Metodo di pagamento
1
Contiene
Riga Ordine
Quantità
Prezzo unitario
Ha
1
1..*
1
Carrello
Utente : String
Cod : Integer
Prodotto : String
0..*
Aggiungere
0..*
Prodotto
Id : Integer
Nome : String
Disponibilità : Integer
Tipo : String
Prezzo : Float
Descrizione : String
Item Carrello
Quantità
Item Carrello.Quantità <= Prodotto.Disponibilità

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 in modo da fornire una visione d'insieme della sua struttura.

Utente Registrato(Username ↑, Password, Nome, Cognome, Città, Indirizzo, Telefono, E-mail)

Ordine(Indice ↑, Data, PrezzoTotale)

Carrello(Utente, Cod↑, Prodotto)

Prodotto(Id ↑, Nome, Disponibilità, Tipo, Prezzo, Descrizione)

1.3 Dettagli della struttura delle tabelle

Utente Registrato

Nome	Tipo	Vincoli	Key
Username	varchar(20)	NOT NULL	PRIMARY KEY
Password	varchar(8)	NOT NULL	
Nome	varchar(20)	NOT NULL	
Cognome	varchar(30)	NOT NULL	
Città	varchar(30)	NOT NULL	
Indirizzo	varchar(50)	NOT NULL	
Telefono	integer	NOT NULL	
E-mail	varchar(30)	NOT NULL	

Ordine

Nome	Tipo	Vincoli	Key
Indice	int	NOT NULL	PRIMARY KEY
Data	date	NOT NULL	
PrezzoTotale	float	NOT NULL	

Carrello

Nome	Tipo	Vincoli	Key
Utente	varchar(20)	NOT NULL	
Codice	integer	NOT NULL	PRIMARY KEY
Nome	varchar(50)	NOT NULL	
Prodotto	varchar(100)	NOT NULL	

Prodotto

Nome	Tipo	Vincoli	Key
Id	integer	NOT NULL	PRIMARY KEY
Nome	varchar(30)	NOT NULL	
Disponibilità	integer	NOT NULL	
Tipo	varchar(30)	NOT NULL	
Prezzo	float	NOT NULL	
Descrizione	varchar(80)	NOT NULL	

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. Inoltre, spesso un DBMS mantiene porzioni della base di dati in memoria centrale velocizzando in questo modo l'accesso ai dati. Infine, ogni interrogazione prima di essere eseguita viene ottimizzata scegliendo un piano efficiente di esecuzione sulla base degli indici esistenti.

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.