|  |  |  |
| --- | --- | --- |
| **Esercizio n.: 1.11a** | **Nome Esercizio**: WebChat DB | |
| **Obiettivo**: | | |
| **Descrizione**: Partendo dal modulo utenti già realizzato, realizzare lo schema concettuale e logico per una applicazione di chat per discussioni one-to-one e facoltativamente di gruppo | | |
| **Classe**:5 BM | **Cognome Nome**: Piccinini Simone | **Data**: 26/01/2024 |

1. **Analisi**

Per creare il database di una webchat è fondamentale curare l’aspetto utenti.

Un utente per partecipare ad una conversazione deve essere registrato con username e password, un utente può cercare un altro utente grazie al suo username ed inviargli un messaggio.

Possibili funzioni da implementare:

* Un utente può scegliere di non voler ricevere messaggi
* può uscire da una conversazione
* bloccare un gruppo o contatto.

Per prima cosa vediamo come gestire la memorizzazione di messaggi tra utenti.

Ci sarà un’entità utente con tutti gli attributi che lo interessano: codicefiscale, nome, cognome, email, passwordhash, numeroDiTelefono

* bannato: questo attributo serve per memorizzare se l’utente è stato bandito dall’applicazione
* durataBan: data in cui se l’utente è stato bannato verrà sbannato
* ultimaVoltaOnline: memorizza l’ultima volta che l’utente è stato online, se l’ultima volta che è stato online è meno di 5 minuti fa, allora viene considerato come online

Entità **messaggio**.

Quest’entità serve per memorizzare le informazioni e i contenuti di ogni messaggio, un utente invia molti messaggi, ogni messaggio è stato inviato da un solo utente.

Quindi sarà una associazione 1 a N.

In questo modo riesco a conoscere chi ha inviato un messaggio, ma per creare una chat serve anche qualcosa che colleghi N utenti ad N messaggi.

È possibile farlo con un’entità **conversazione**, questa entità deve essere connessa con utente e messaggio, perché una conversazione viene svolta da N persone che inviano N messaggi.

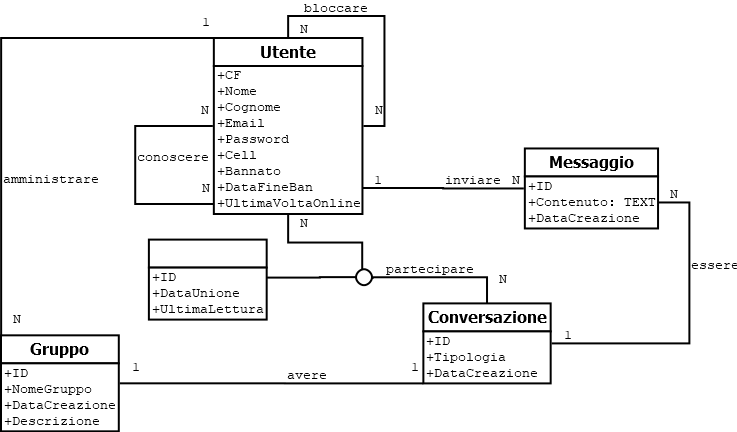
Un utente ha N conversazioni e una conversazione ha N utenti, un messaggio è in una conversazione ed una conversazione ha N messaggi.

Quindi la relazione tra utenti e conversazioni è molti a molti, con un’entità associativa avere.

L’entità conversazione distinguerà una chat privata da una chat di gruppo, l’entità conversazione avrà un campo “nomeGruppo” che se verrà popolato solo se la chat è di gruppo.

Un utente ha due entità molti a molti che servono per sapere quali utenti sono bloccati e quali invece sono amici.

1. **Schema Concettuale** (ER o UML)



1. **Schema Logico**

Utente: ID:INT,Nome:VARCHAR(50),Cognome: VARCHAR(50), Email:VARCHAR(50),Password:CHAR(255), Cell:Varchar(13), Bannato:TINYINT(1),DataFineBan:DATETIME, UltimaVoltaOnline:DATETIME

Messaggio = ID:INT,Contenuto:TEXT,DataCreazione:DATETIME, FK(CFUtente:INT, IDConversazione:INT)

Conversazione: ID:INT, Tipologia:CHAR(1),DataCreazione:DATETIME

Partecipare:ID:INT, DataUnione:DATETIME, UltimaLettura:DATETIME, FK(CFUtente:INT, IDConversazione:INT)

Gruppo:ID:INT, NomeGruppo:VARCHAR(50), Descrizione:VARCHAR(150), DataCreazione:DATETIME, FK(Amministratore:INT);

Conoscere: ID:INT, Date:DATE, StatoRischiesta:CHAR(1),FK(IDRichiedente:INT, IDConosciuto:INT),

Bloccare: ID:INT, Date:DATE, FK(IDBloccante:INT, IDBloccato:INT),

1. **Normalizzazione**

**1NF** (Schema Logico nella Prima Forma Normale)

Già in 1 FN

**2NF** (Schema Logico nella Seconda Forma Normale)

Già in 2 FN

**3NF** (Schema Logico nella Terza Forma Normale)

Già in 3 FN

1. **Implementazione DB e Popolazione tabelle**

**CREATE TABLE Utenti (**

**ID INT PRIMARY KEY,**

**Nome VARCHAR(50),**

**Cognome VARCHAR(50),**

**Email VARCHAR(50),**

**Password CHAR(255),**

**Cell VARCHAR(13),**

**Bannato TINYINT(1),**

**DataFineBan DATETIME,**

**UltimaVoltaOnline DATETIME**

**);**

**-- Creazione della tabella Conversazione**

**CREATE TABLE Conversazioni (**

**ID INT PRIMARY KEY,**

**Tipologia CHAR(1),**

**DataCreazione DATETIME**

**);**

**-- Creazione della tabella Messaggio**

**CREATE TABLE Messaggi (**

**ID INT PRIMARY KEY,**

**Contenuto TEXT,**

**DataCreazione DATETIME,**

**CFUtente INT,**

**IDConversazione INT,**

**FOREIGN KEY (CFUtente) REFERENCES Utenti(ID),**

**FOREIGN KEY (IDConversazione) REFERENCES Conversazioni(ID)**

**);**

**-- Creazione della tabella Partecipare**

**CREATE TABLE Partecipare (**

**ID INT PRIMARY KEY,**

**DataUnione DATETIME,**

**UltimaLettura DATETIME,**

**CFUtente INT,**

**IDConversazione INT,**

**FOREIGN KEY (CFUtente) REFERENCES Utenti(ID),**

**FOREIGN KEY (IDConversazione) REFERENCES Conversazioni(ID)**

**);**

**-- Creazione della tabella Gruppo**

**CREATE TABLE Gruppi (**

**ID INT PRIMARY KEY,**

**NomeGruppo VARCHAR(50),**

**Descrizione VARCHAR(150),**

**DataCreazione DATETIME,**

**Amministratore INT,**

**FOREIGN KEY (Amministratore) REFERENCES Utenti(ID)**

**);**

**-- Creazione della tabella Conoscere**

**CREATE TABLE Conoscere (**

**ID INT PRIMARY KEY,**

**Date DATE,**

**StatoRischiesta CHAR(1),**

**IDRichiedente INT,**

**IDConosciuto INT,**

**FOREIGN KEY (IDRichiedente) REFERENCES Utenti(ID),**

**FOREIGN KEY (IDConosciuto) REFERENCES Utenti(ID)**

**);**

**-- Creazione della tabella Bloccare**

**CREATE TABLE Bloccare (**

**ID INT PRIMARY KEY,**

**Date DATE,**

**IDBloccante INT,**

**IDBloccato INT,**

**FOREIGN KEY (IDBloccante) REFERENCES Utenti(ID),**

**FOREIGN KEY (IDBloccato) REFERENCES Utenti(ID)**

**);**

1. **Query** (in SQL)

[DML e QL]