Basi di Dati

Progetto A.A. 2023/2024

TITOLO DEL PROGETTO

Matricola

Nome e Cognome

Indice

Tutto il testo su sfondo grigio, all'interno di questo template, deve essere eliminato prima della consegna. Viene utilizzato per fornire informazioni sulla corretta compilazione del report di progetto.

Non modificare il formato del documento:
- Carattere: Times New Roman, 12pt
- Dimensione pagina: A4
- Margini: superiore/inferiore 2,5cm, sinistro/destro: 1,9cm

- **Descrizione del Minimondo**

	Inserire all'interno di questo riquadro la specifica così come è stata fornita. Riportare nella colonna a sinistra la numerazione delle righe. Questi numeri dovranno essere utilizzati per riferirsi al testo nelle sezioni successive.
1	Bacheca elettronica di annunci
2	
3	Si vuole realizzare un sistema informativo per la gestione di una bacheca elettronica di
4	annunci. Tale bacheca permette agli utenti del sistema di inserire annunci per la vendita di
5	materiale usato, di scambiare messaggi tra di loro (in maniera privata) per accordarsi sulla
6	vendita/consegna dell'oggetto, o di inserire domande (in maniera pubblica) sull'oggetto.
7	
8	Un utente del sistema si registra scegliendo un username univoco, inserendo tutte le sue
9	informazioni anagrafiche, indicando un indirizzo di residenza ed eventualmente un indirizzo
10	di fatturazione, un numero arbitrario di recapiti (telefono, cellulare, email) indicandone uno
11	come mezzo di comunicazione preferito.
12	
13	I gestori del servizio possono creare una gerarchia di categorie per gli annunci. Un utente,
14	per creare un annuncio, seleziona una categoria e scrive una descrizione dell'oggetto.
15	Quando un oggetto inserito in bacheca è stato venduto, l'utente lo indica come tale e questo
16	non viene più visualizzato nella bacheca pubblica.
17	
18	Un utente del sistema, una volta letto e scelto un annuncio, può decidere di inserire un
19	commento pubblico o di inviare un messaggio privato all'utente che ha inserito l'annuncio.
20	Similmente, un utente può "seguire" uno degli annunci, venendo così informato ogni volta
21	che su questo viene effettuata una modifica (ad esempio, viene inserita una nuova nota).
22	
23	I gestori del servizio possono generare un report indicante in forma aggregata per ciascun
24	utente del sistema quale percentuale di annunci pubblicati sono stati contrassegnati come
25	venduti.

- # Analisi dei Requisiti

Lo scopo di questa sezione è raffinare la specifica fornita, andando ad effettuare un'operazione preliminare di disambiguazione.

- ## Identificazione dei termini ambigui e correzioni possibili

Compilare la seguente tabella, facendo riferimento alla specifica del minimondo di riferimento precedentemente indicata. Individuare i termini ambigui nella specifica (indicando la linea in cui essi compaiono), indicare il nuovo termine che si intende adottare nella specifica, ed indicare il motivo del cambiamento che si propone.

Linea	Termine	Nuovo termine	Motivo correzione
5	Materiale usato	oggetti	Nel testo il termine 'materiale usato' viene citato più volte utilizzando 'oggetto' per avere un termine univoco usiamo oggetto.
4	domande	Commenti	I termini domande e commenti, usati più volte nel testo si riferiscono ad un unico commento, quindi per evitare equivoci utilizziamo commenti.

Specifica disambiguata

Riportare in questo riquadro la specifica di progetto corretta, applicando le disambiguazioni proposte.

Bacheca elettronica di annunci

Si vuole realizzare un sistema informativo per la gestione di una bacheca elettronica di annunci. Tale bacheca permette agli utenti del sistema di inserire annunci per la vendita di oggetti, di scambiare messaggi tra di loro (in maniera privata) per accordarsi sulla vendita/consegna dell'oggetto, o di inserire commenti (in maniera pubblica) sull'oggetto.

Un utente del sistema si registra scegliendo un username univoco, inserendo tutte le sue informazioni anagrafiche, indicando un indirizzo di residenza ed eventualmente un indirizzo di fatturazione, un numero arbitrario di recapiti (telefono, cellulare, email) indicandone uno come mezzo di comunicazione preferito.

I gestori del servizio possono creare una gerarchia di categorie per gli annunci. Un utente, per creare un annuncio, seleziona una categoria e scrive una descrizione dell'oggetto. Quando un oggetto inserito in bacheca è stato venduto, l'utente lo indica come tale e questo non viene più visualizzato nella bacheca pubblica.

Un utente del sistema, una volta letto e scelto un annuncio, può decidere di inserire un commento pubblico o di inviare un messaggio privato all'utente che ha inserito l'annuncio. Similmente, un utente può "seguire" uno degli annunci, venendo così informato ogni volta che su questo viene effettuata una modifica (ad esempio, viene inserita una nuova nota).

I gestori del servizio possono generare un report indicante in forma aggregata per ciascun utente del sistema quale percentuale di annunci pubblicati sono stati contrassegnati come venduti.

- ## Glossario dei Termini

Realizzare un dizionario dei termini, compilando la tabella qui sotto, a partire dalle specifiche precedentemente disambiguate

Termine	Descrizione	Sinonimi	Collegamenti
Gestori del servizio	È colui che si occupa della gestione, dell'amministrazione e della manutenzione della bacheca elettronica.	Amministratori	Report, Categoria
Bacheca elettronica	È uno spazio online dove è possibile acquistare o vendere materiale.		Annunci
Annuncio	un'azione commerciale per vendere un determinato prodotto.		Utente, Categoria
username	Nome per farsi identificare da un utente di una determinata piattaforma		
Materiale usato	Merce privata pronta ad essere messa in vendita	oggetto	Annunci, Categoria
categoria	Suddivisone che si ottiene classificando secondo determinati canoni e criteri		Annunci, Gestore dei servizi
report	Resoconto dettagliato delle vendite di un utente del sistema		Gestore dei servizi

- ## Raggruppamento dei requisiti in insiemi omogenei

Per ciascun elemento “più importante” della specifica (riportata anche nel glossario precedente), estrapolare dalla specifica disambiguata le frasi ad esso associate. Compilare una tabella separata per ciascun elemento individuato.

Frasi relative a gestore dei servizi
I gestori del servizio possono creare una gerarchia di categorie per gli annunci.

Frasi relative a bacheca elettronica
Si vuole realizzare un sistema informativo per la gestione di una bacheca elettronica di annunci. Quando un oggetto inserito in bacheca è stato venduto, l’utente lo indica come tale e questo non viene più visualizzato nella bacheca pubblica.

Frasi relative a annuncio
Tale bacheca permette agli utenti del sistema di inserire annunci per la vendita di materiale usato,

Frasi relative a username
Un utente del sistema si registra scegliendo un username univoco,

Frasi relative a materiale usato
Tale bacheca permette agli utenti del sistema di inserire annunci per la vendita di materiale usato,

Frasi relative a categoria
Un utente, per creare un annuncio, seleziona una categoria e scrive una descrizione dell’oggetto.

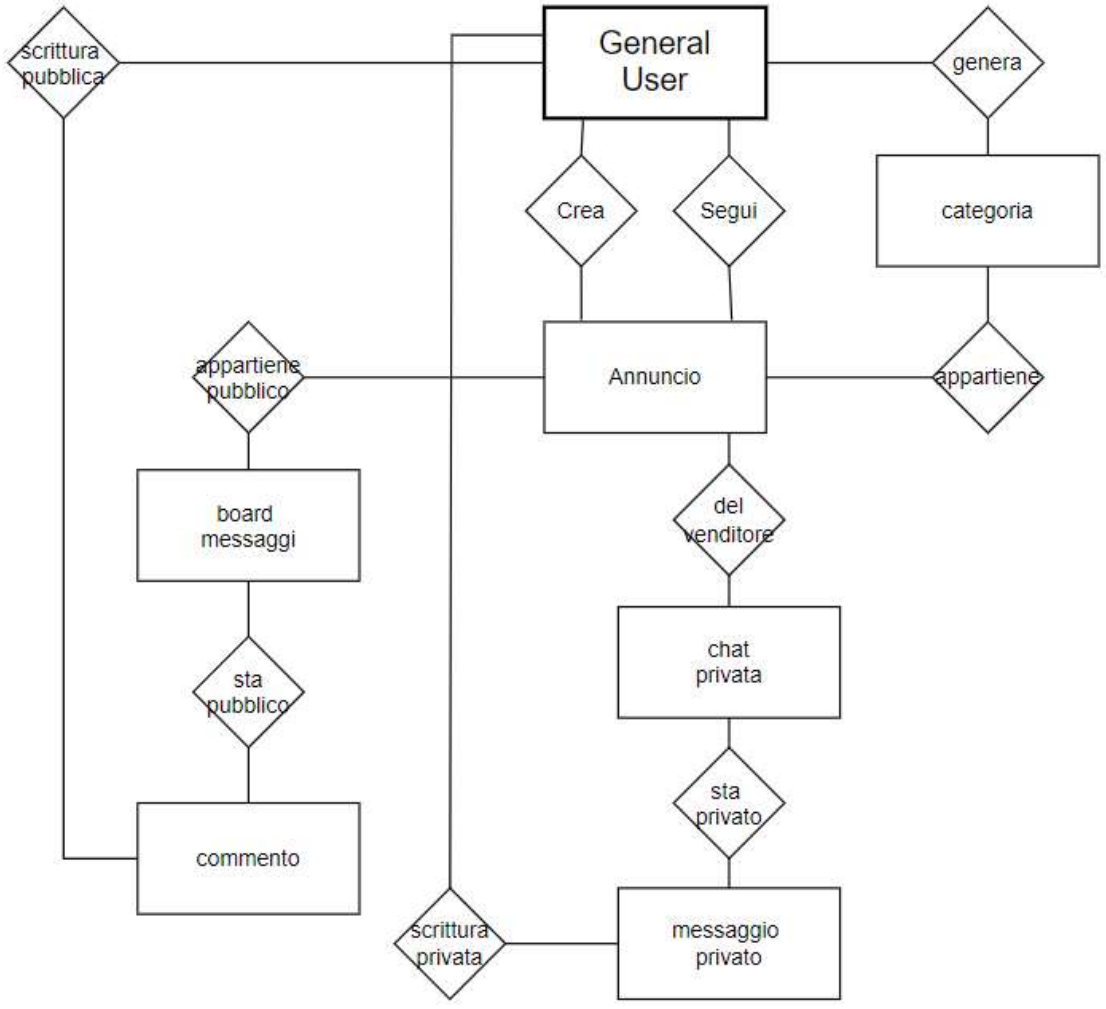
Frasi relative a report
I gestori del servizio possono generare un report indicante in forma aggregata per ciascun utente del sistema quale percentuale di annunci pubblicati sono stati contrassegnati come venduti.

- # Progettazione concettuale

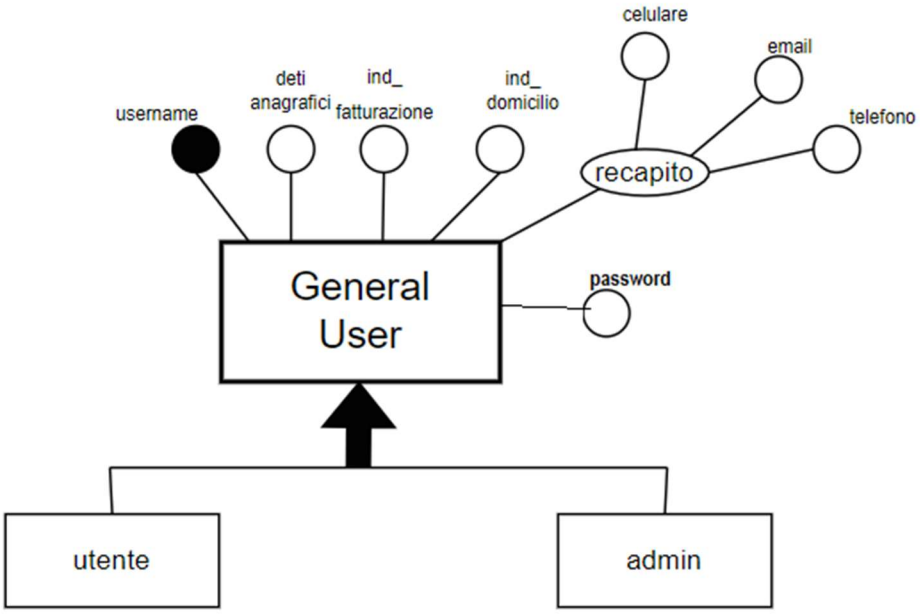
- ## Costruzione dello schema E-R

In questa sezione è necessario riportare tutti passi seguiti per la costruzione dello schema E-R finale, a partire dalle specifiche raccolte ed organizzate nel capitolo precedente. Non è richiesto un procedimento specifico: si può adottare una strategia top-down, bottom-up, a macchia d'olio o mista. L'importante è descrivere e commentare tutti i passi della costruzione, andando anche ad inserire "schemi parziali" utilizzati nel processo.

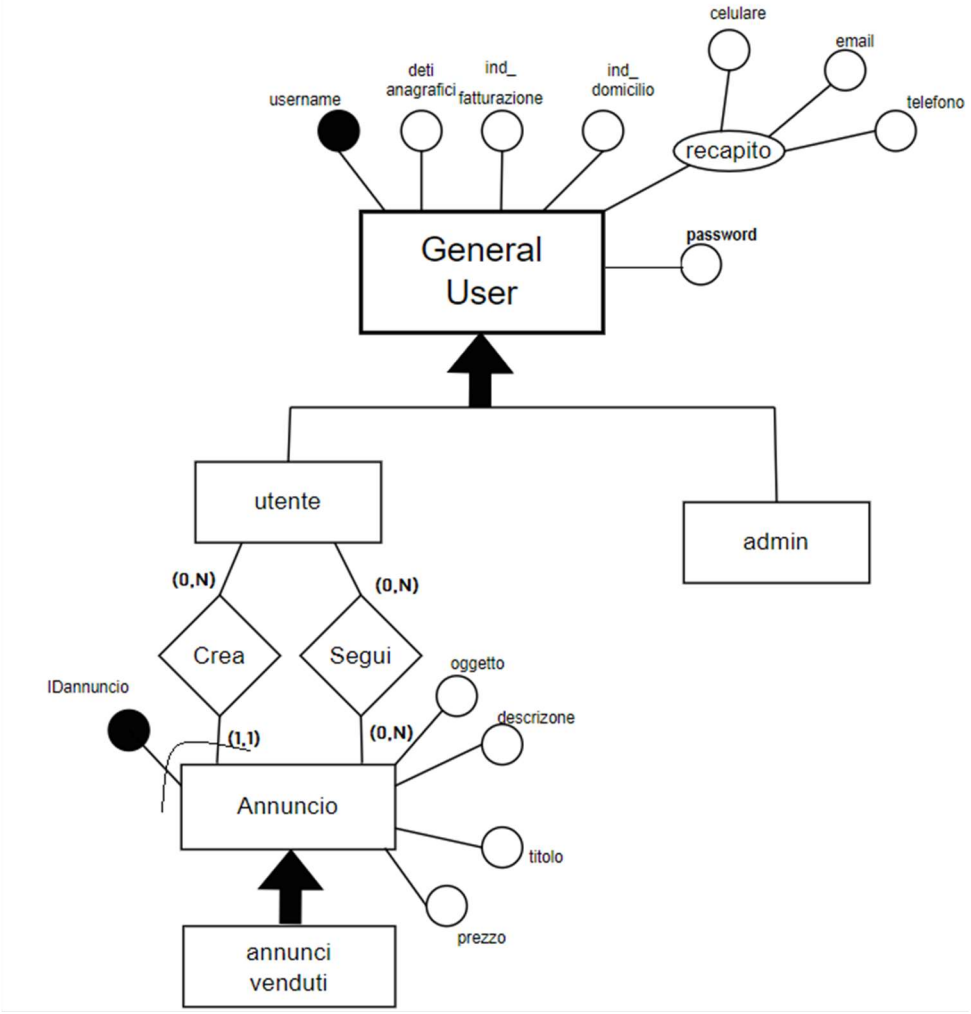
Per la costruzione dello schema ER ho adottato una strategia mista elaborando inizialmente uno scheletro del sistema con prime entità chiave ricavate dal testo:



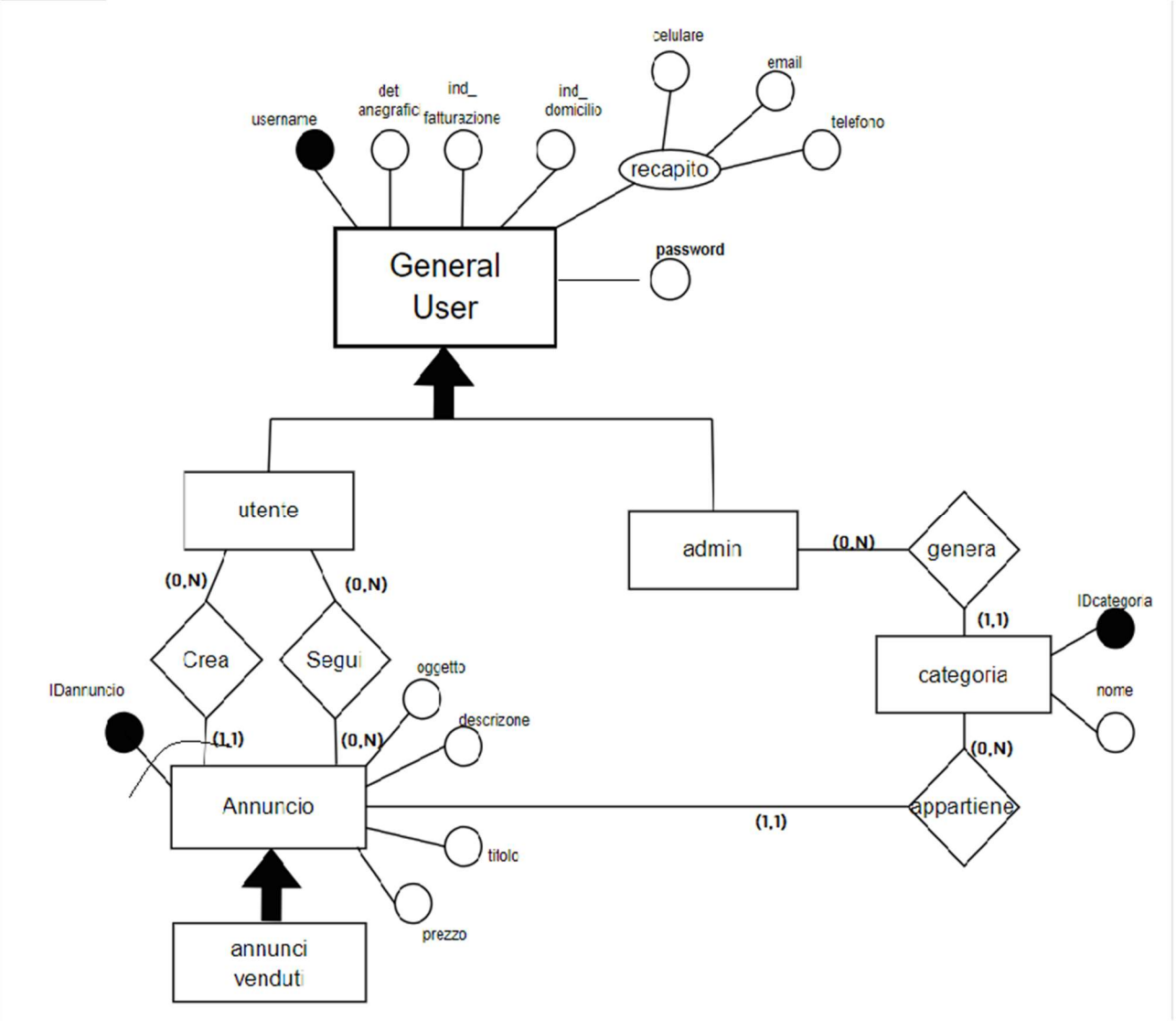
Dopo aver analizzato lo scheletro del sistema, ho individuato un'entità chiave da cui partire, l'entità 'general user', che rappresenta l'utente generico (compreso gestore dei servizi) che può registrarsi sulla piattaforma. Tramite una generalizzazione in utente e admin, vogliamo distinguere i due user per assegnare relazioni diverse tra loro.



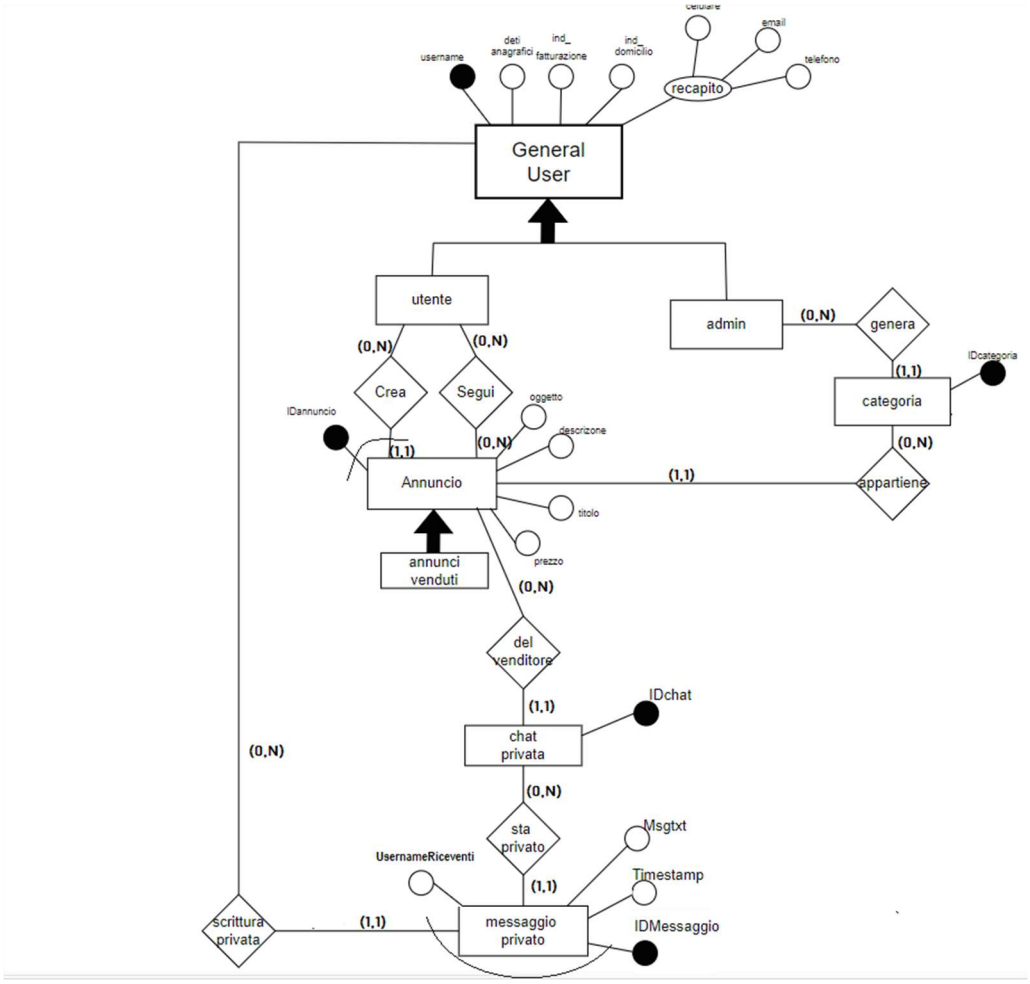
Successivamente passo ad un'altra entità fondamentale, ovvero ‘annuncio’ tale entità con determinati attributi si generalizza in ‘annunci venduti’ che sono gli annunci non più online poiché venduti. Tale entità annuncio è in relazione con ‘utente’ con la quale detiene una doppia relazione ‘crea’ e ‘segui’



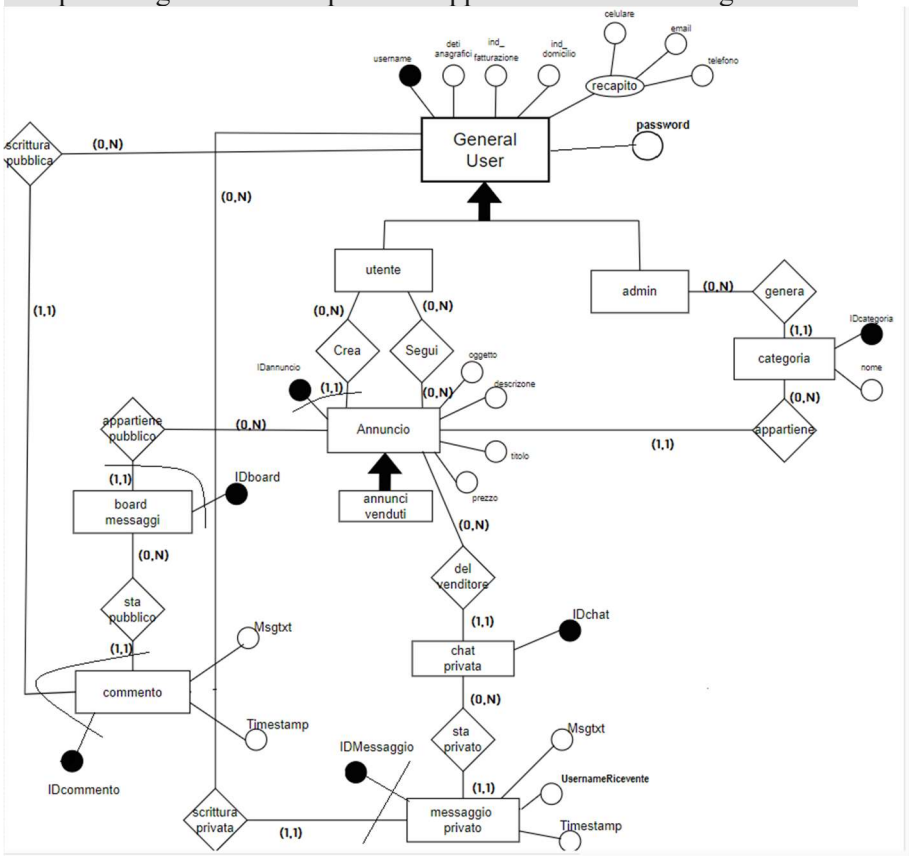
Ci rendiamo conto che serve l'uso di un'altra entità, ovvero 'categoria' tale entità è in relazione con admin e annuncio.



Ora lavoriamo sulla parte di messaggistica, per la chat privata ci rendiamo conto che serve un'entità 'chat privata' in relazione con annuncio, da questa chat privata ci rendiamo conto che serve l'utilizzo di un'entità 'messaggio privato' in relazione con chat privata e utente, così da poter permettere ad un utente di scrivere un messaggio privato nella chat privata del creatore dell'annuncio.

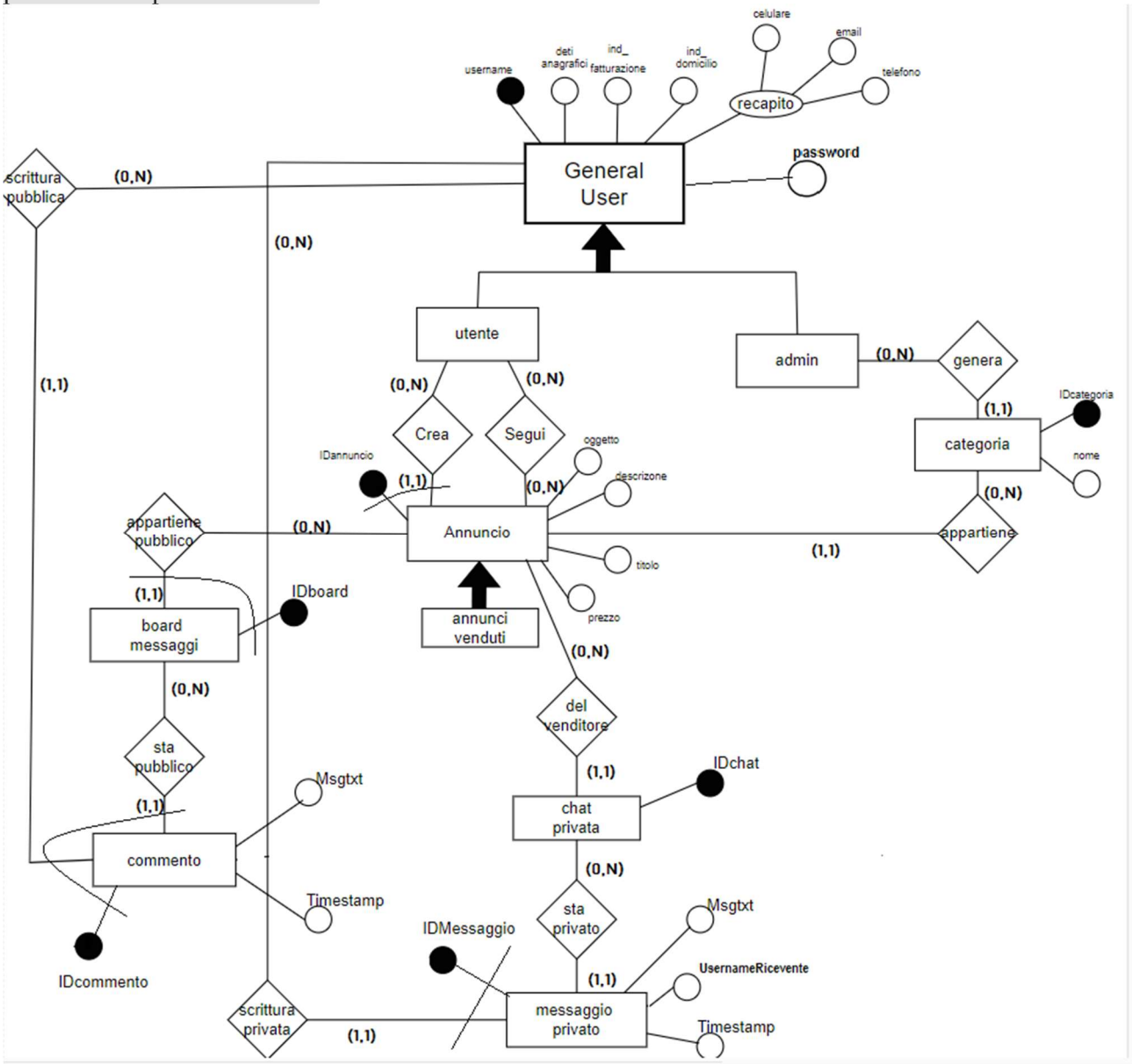


Per quanto riguarda la chat pubblica applichiamo lo stesso ragionamento



Integrazione finale

Nell'integrazione finale delle varie parti dello schema E-R è possibile che si evidenzino dei conflitti sui nomi utilizzati e dei conflitti strutturali. Prima di riportare lo schema E-R finale, descrivere quali passi sono stati adottati per risolvere tali conflitti per
Per differenziare le relazioni pubbliche da quelle private ho modificato il nome aggiungendo pubblico e privato alle rispettive relazioni



- # Regole aziendali

Laddove la specifica non sia catturata in maniera completa dallo schema E-R, corredare lo stesso in questo paragrafo con l'insieme delle regole aziendali necessarie a completare la progettazione concettuale.

- Un messaggio privato non può superare i 200 caratteri.
- Un messaggio pubblico non può superare i 350 caratteri.

- **Dizionario dei dati**

Completare la progettazione concettuale riportando nella tabella seguente il dizionario dei dati

Entità	Descrizione	Attributi	Identificatori
General User	Utente che si registra sulla piattaforma	Username, dati anagrafici, telefono, cellulare, e-mail, ind_fatturazione, ind_domicilio, password	username
utente	È l'utente specifico che utilizza la piattaforma per vendere e comprare		username
admin	È il gestore dei servizi che si occupa solo di amministrare		username
annuncio	È il mezzo per pubblicizzare e vendere oggetti	IDAnnuncio, titolo, descrizione, prezzo, oggetto	IDAnnuncio
Annunci venduti	Annunci non più online, poiché venduti		IDannuncio
categoria	È la tipologia di suddivisioni e ordinamenti dati per la tipologia dell'oggetto	IDcategoria, nome	IDcategoria
Board messaggi	Box dei messaggi pubblici per annuncio	IDboard	IDboard
commento	Il messaggio pubblico scritto dagli utenti sotto un annuncio	IDCommento, Timestamp, Msgtxt	IDCommento
Chat privata	È il raggruppamento dei messaggi privati scambiati tra due utenti	IDchat	IDchat
Messaggio privato	È il messaggio che si può scambiare tra due utenti	IDMessaggio, Timestamp, Msgtxt, UsernameRicevente	IDMessaggio

• Progettazione logica

• Volume dei dati

Questa sezione serve ad illustrare qual è il carico che la base di dati dovrà sopportare. A tal fine, è necessario prevedere un volume di dati attesi. Compilare la tabella sottostante, per ciascun concetto identificato nello schema E-R. I volumi devono essere stimati dallo studente in maniera ragionevole rispetto all'operatività presunta dell'applicativo.

Concetto nello schema	Tipo	Volume atteso
utenti	E	3.000.000
Annunci	E	1.500.000
Admin	E	5
categorie	E	20
Board messaggi	E	1.500.000
Chat privata	E	30.000.000
Messaggio privato	E	120.000.000
commento	E	1.500.000
Annunci venduti	E	750.000
segui	R	450.000
crea	R	1.500.000
genera	R	20
Appartiene	R	20
Del venditore	R	30.000.000
Sta privato	R	120.000.000
Sta pubblico	R	1.500.000
Appartiene pubblico	R	1.500.000
Scrittura privata	R	120.000.000
Scrittura pubblica	R	1.500.000

• Tavola delle operazioni

Rappresentare nella tabella sottostante tutte le operazioni *non banali* sulla base di dati che devono essere supportate dall'applicazione, con la frequenza attesa. Le operazioni da supportare devono essere desunte dalle specifiche raccolte.

Cod.	Descrizione	Frequenza attesa
Op 1	Inserire un utente	1000/giorno
Op 2	Creare annuncio	2000/giorno
Op 3	Segui annuncio	1200/giorno
Op 4	Creare categoria	2/anno
Op 5	Scrivere commento	1500/giorno
Op 6	Scrivere messaggio privato	30.000/giorno
Op7	Indicare come venduto	900/giorno
Op8	Modificare annuncio	1000/giorno

• Costo delle operazioni

In riferimento a tutte le operazioni precedentemente indicate, calcolarne il costo supponendo, per questa fase del progetto, che il costo in scrittura di un dato sia doppio rispetto a quello in lettura.

1) Op1 costo: 1 operazione in lettura/scrittura in utente -> 2*Fa=2000/giorno
2) Op2 costo: 1 operazione in lettura in utente, ed 1 operazione in scrittura in annuncio -> (1+2) *Fa=6000/giorno
3) Op3 costo: 1 operazione in scrittura, una lettera sia in utenti che in annunci (1+1+2) *Fa=4800/giorno
4) Op4 costo: 1 operazione in scrittura in categoria 1*Fa-> 4/anno
5) Op5 costo: ho 3 letture in utente, annuncio, board messaggi e una scrittura in commento (1+1+1+2) *Fa=7500/giorno
6) Op6 costo: ho 3 letture in utente, annuncio, chat privata e una scrittura in messaggio privato (1+1+1+2) *Fa=150.000/giorno
7) Op7 costo: una lettura/scrittura in annuncio 2*Fa= 1800/giorno
8) Op8 costo: una lettura/scrittura in annuncio 2*Fa=2000/giorno

• Ristrutturazione dello schema E-R

Descrivere (laddove necessario fornendo anche degli schemi) quali passi vengono adottati per ristrutturare lo schema E-R, ad esempio in termini di:

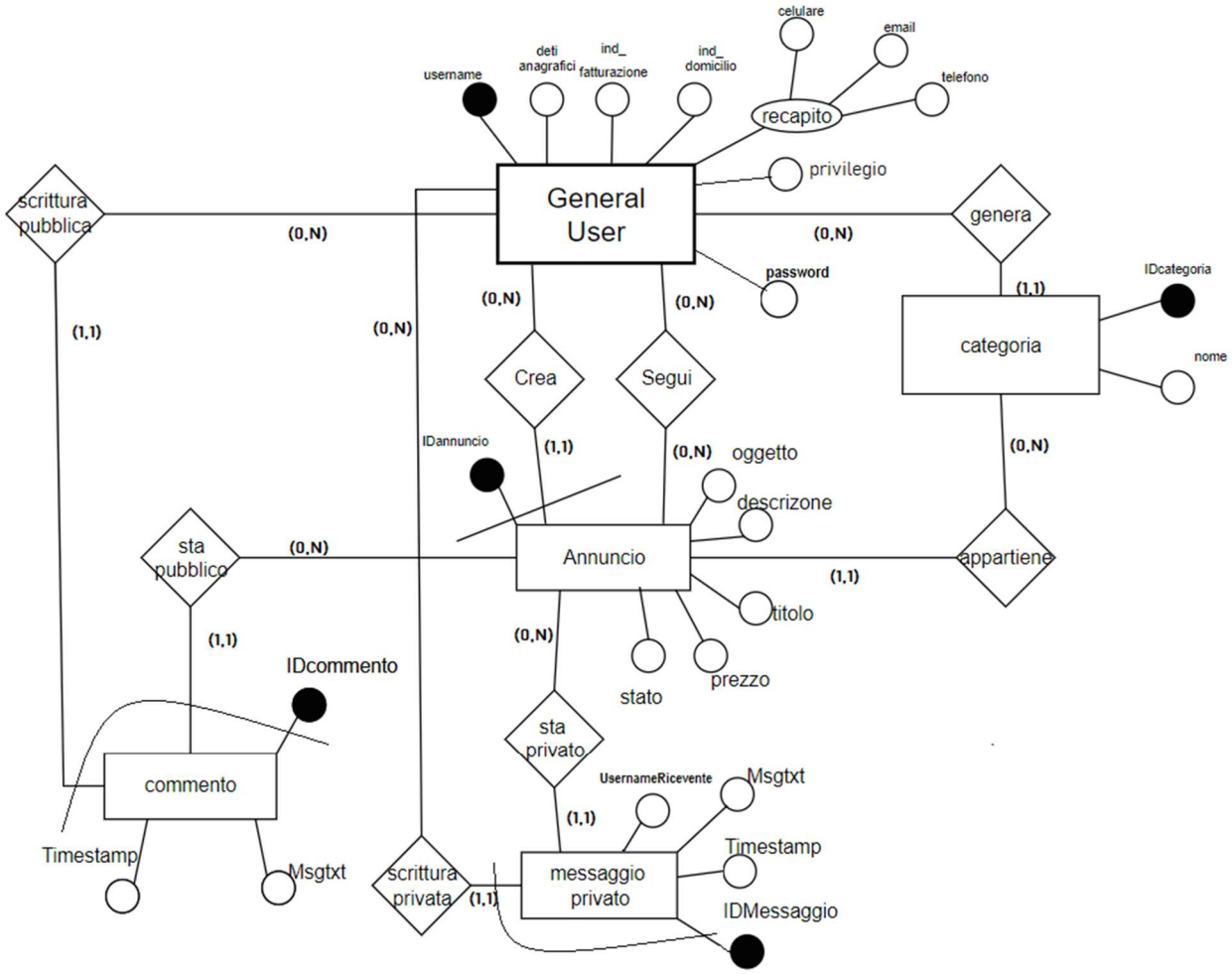
- Analisi delle ridondanze
- Eliminazione delle generalizzazioni
- Scelta degli identificatori primari

Si noti che in questa fase è possibile fare riferimento al costo delle operazioni precedentemente realizzato per guidare le scelte. Ad esempio, un leggero spreco di memoria legato alla non rimozione di ridondanze può essere facilmente giustificato da un guadagno in termini di prestazioni.

1)come prima azione di ristrutturazione ho eliminato la generalizzazione di ‘general user’, accorpando l’entità ‘utente’ e l’entità ‘admin’ in ‘general user’ aggiungendo un attributo ‘privilegio’ booleano, dove in caso di ‘truee’= utente in caso di ‘false’=admin.

2) elimino la generalizzazione di ‘annunci venduti’ accorpandola ad annunci, inserendo un’attributo che definisca lo stato dell’annuncio.

3)mi rendo conto che sia l’entità ‘chat privata’ che l’entità ‘board messaggi’ sono ridondanti ed eliminandole tramite una dipendenza delle entità commenti e messaggi privati verso annunci, posso ricavare le chiavi necessarie per le operazioni



- ## Trasformazione di attributi e identificatori

Qualora siano presenti, in questa fase della progettazione, attributi ripetuti o identificatori esterni, descrivere quali trasformazioni vengono realizzate sul modello per facilitare la traduzione nello schema relazionale.

Dato che gli attributi: Timestamp e Msgtxt sono utilizzati in più entità diventano nell'entità commento

Timestamp_commento

Msgtxt_commento

L'identificatore esterno di 'Annuncio' ovvero 'Username' proveniente da 'General User' verrà chiamato UsernameC

- ## Traduzione di entità e associazioni

Riportare in questa sezione la traduzione di entità ed associazioni nello schema relazionale.
Fornire una rappresentazione grafica del modello relazionale completo.

General User (Username, Dati anagrafici, telefono, cellulare, e-mail, ind_fatturazione, ind_domicilio, privilegio, password)

Annuncio (IDAnnuncio, titolo, prezzo, oggetto, descrizione, stato, IDA, UsernameA)

- **Annuncio (UsernameA)->General User**
- **Annuncio (IDA)->Categoria**

Categoria (IDCategoria, Nome, UsernameC)

- **Categoria (UsernameC)->General User**

Commento (IDcommento, Timestamp_commento, Msgtxt_commento, IDAnnuncioC, Username_Commento, UsernameCommento)

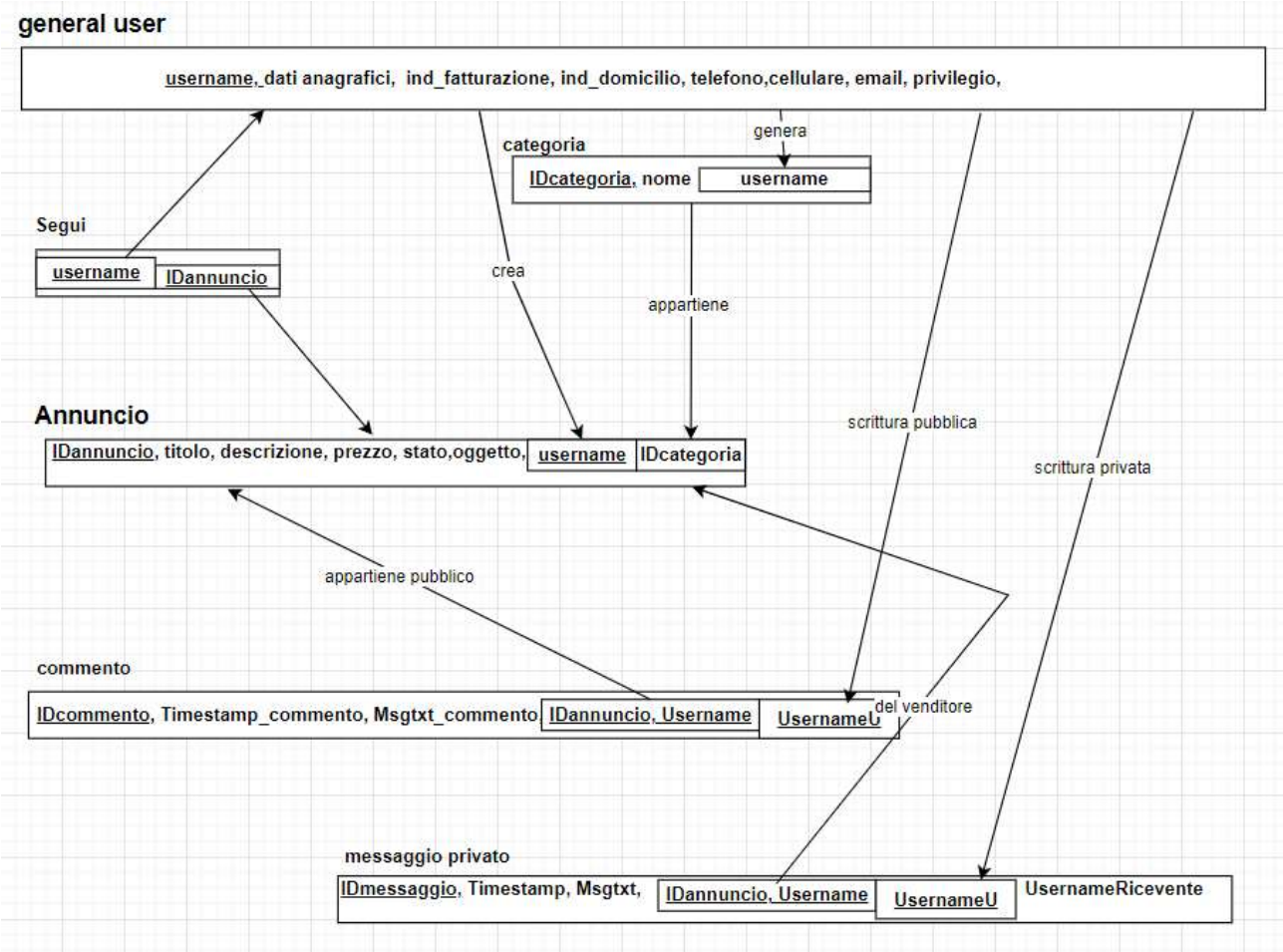
- **Commento (IDAnnuncioC)-> Annuncio**
- **Commento (UsernameCommento)->Annuncio**
- **Commento (Username_Commento) ->General User**

Messaggio privato (IDMessaggio, Timestamp, Msgtxt, UsernameM, IDAnnuncioM, UsernameC, UsernameRicevente)

- **Messaggio privato (UsernameCM)->General User**
- **Messaggio privato (UsernameM)->Annunci**
- **Messaggio privato (IDannuncioM)-> Annunci**

Segui (IDAnnuncioS, UsernameS)

- **Segui (IDAnnuncioS)->Annuncio**
- **Segui (UsernameS)->General User**



- **Normalizzazione del modello relazionale**

Mostrare se il modello relazionale precedentemente descritto è in forma normale. Se non lo è, descrivere quali trasformazioni si effettuano per normalizzarlo. Se si sceglie di non normalizzarlo, giustificare le motivazioni da un punto di vista prestazionale.

• Progettazione fisica

• Utenti e privilegi

Descrivere, all'interno dell'applicazione, quali utenti sono stati previsti con quali privilegi di accesso su quali tabelle, giustificando le scelte progettuali.

Nel mio sistema ho previsto 3 ruoli:

- Login: in esecuzione sull'operazione Op1
- Utente: in esecuzione sulle operazioni: Op2 Op3 Op5 Op6 Op7 Op8
- Amministratore: in esecuzione sulle operazioni: Op4 Op9

• Strutture di memorizzazione

Compilare la tabella seguente indicando quali tipi di dato vengono utilizzati per memorizzare le informazioni di interesse nelle tabelle, per ciascuna tabella.

Tabella <General User>		
Colonna	**Tipo di dato**	**Attributi**
Username	VARCHAR (45)	PK NN
Dati anagrafici	DATE	NN
ind_fatturazione	VARCHAR (45)	NN
Ind_domicilio	VARCHAR (45)	NN
privilegio	enum	NN
password	VARCHAR (10)	NN
telefono	Varchar (10)	
cellulare	Varchar (10)	
e-mail	VARCHAR (45)	
Tabella <Annuncio>		
Colonna	**Tipo di dato**	**Attributi**
IDannuncio	INT	PK NN AI
UsernameA	VARCHAR (45)	PK NN
IDA	INT	NN
titolo	VARCHAR (45)	NN
prezzo	INT	NN
oggetto	VARCHAR (45)	NN
descrizione	VARCHAR (100)	NN
stato	enum	NN
Tabella <Messaggio privato>		
Colonna	**Tipo di dato**	**Attributi**
IDMessaggio	INT	PK NN AI
UsernameM	VARCHAR (45)	PK NN
UsernameCM	VARCHAR (45)	PK NN
IDAnnuncioM	INT	PK NN
Timestamp	DATETIME	NN
Msgtxt	VARCHAR (200)	NN
UsernameRicevente	VARCHAR (45)	NN
Tabella <Categoria>		
Colonna	**Tipo di dato**	**Attributi**
IDCategoria	INT	PK NN AI

Nome	VARCHAR (45)	NN
UsernameC	VARCHAR (45)	NN
Tabella <Commento>		
Colonna	**Tipo di dato**	**Attributi**
IDCommento	INT	PK NN AI
Username_Commento	VARCHAR (45)	PK NN
UsernameCommento	VARCHAR (45)	PK NN
IDAnnuncioC	INT	PK NN
Timestamp_commento	DATETIME	NN
Msgtxt_commento	VARCHAR (350)	NN
Tabella <Segui>		
Colonna	**Tipo di dato**	**Attributi**
IDAnnuncioS	INT	PK NN
UsernameS	VARCHAR (45)	PK NN

- ## Indici

Compilare la seguente tabella, per ciascuna tabella del database in cui sono presenti degli indici. Descrivere le motivazioni che hanno portato alla creazione di un indice, facendo riferimento al costo delle operazioni individuate nella sezione precedente.
Attenzione: non è necessario riportare gli indici autogenerati in fase di definizione dello schema (ad esempio, per la gestione della chiave primaria), ma *soltanto* quelli introdotti per motivi prestazionali.

- ## Trigger

Descrivere quali trigger sono stati implementati, mostrando il codice SQL per la loro instanziazione. Si faccia riferimento al fatto che il DBMS di riferimento richiede di utilizzare trigger anche per realizzare vincoli di check ed asserzioni.

Trigger che permette la notifica automatica della notifica di un annuncio a chi lo segue.

```
CREATE DEFINER = CURRENT_USER TRIGGER `progetto`.`MexNotifica` BEFORE UPDATE ON `Annuncio` FOR EACH ROW
BEGIN
declare var int;
   declare var3 int;
   declare varscri varchar(45);
         select count(*)
   from segui
   where IDAnnuncioS = new.IDAnnuncio
   into var;
   set var3 = 0;

   while var3 != var do


                  select UsernameS into varscri
      from segui
      where IDAnnuncioS = new.IDAnnuncio
```

```
    LIMIT 1 OFFSET var3;

    Insert into Messaggio_privato(timestamp, msgtxt, IDAnnuncioM, UsernameM,
UsernameCM,UsernameRicevente)
    value((NOW()), 'ho modifacato il post! vallo a guardare', new.IDAnnuncio,
new.UsernameA,new.UsernameA, varscri);

    set var3 = var3 + 1;
        end while;

END
```

Trigger che permette la pulizia delle tabelle dei messaggi e dei follow dopo che un annuncio è stato venduto

```
CREATE DEFINER = CURRENT_USER TRIGGER `progetto`.`Pulizia` BEFORE UPDATE ON
`Annuncio` FOR EACH ROW
BEGIN

        if new.stato='venduto' and old.stato != 'venduto'
  then
                delete from Messaggio_privato where IDAnnuncioM= new.IDAnnuncio;
    delete from Commento where IDAnnuncioC=new.IDAnnuncio;
    delete from segui where IDAnnuncioS=new.IDAnnuncio;
        end if;
END
```

Trigger che impedisce l'inserimento di commenti che superano le grandezze consentite per i commenti

```
CREATE DEFINER = CURRENT_USER TRIGGER `progetto`.`Commento_BEFORE_INSERT` BEFORE
INSERT ON `Commento` FOR EACH ROW
BEGIN
        if length(new.msgtxt_commento)>349
  then
  signal sqlstate '45000'
  set MESSAGE_TEXT ='il Commento supera la lunghezza consentita';
  end if;

END
```

Trigger che impedisce l'inserimento dei messaggi che superano la grandezze consentite dei messaggi

```
CREATE DEFINER = CURRENT_USER TRIGGER `progetto`.`Messaggio_privato_BEFORE_INSERT`
BEFORE INSERT ON `Messaggio_privato` FOR EACH ROW
BEGIN
        if length(new.msgtxt)>199
  then
  signal sqlstate '45000'
  set MESSAGE_TEXT ='il messaggio supera la lunghezza consentita';
  end if;
END
```

- ## Eventi

Descrivere quali eventi sono stati implementati, mostrando il codice SQL per la loro instanziazione. Si descriva anche se gli eventi sono istanziati soltanto in fase di configurazione del sistema, o se alcuni eventi specifici vengono istanziati in maniera effimera durante l'esecuzione di alcune procedure.

- ## Viste

Mostrare e commentare il codice SQL necessario a creare tutte le viste necessarie per l'implementazione dell'applicazione.

- ## Stored Procedures e transazioni

Mostrare e commentare le stored procedure che sono state realizzate per implementare la logica applicativa delle operazioni sui dati, evidenziando quando (e perché) sono state realizzate operazioni transazionali complesse.

Procedura per Creare un Annuncio, inserendo titolo, prezzo, oggetto, stato, il proprio username e il nome della categoria, ritorna l'id dell'annuncio appena creato

```
CREATE PROCEDURE `creare_annuncio` (in var_titolo varchar(45), in var_prezzo int, in var_oggetto varchar(45), in var_descrizione varchar(100), in var_stato enum('venduto','online')  , in var_username varchar(45), in var_categoria varchar(45), out var_ritorno int)
BEGIN
        declare var INT;
   declare var2 INT;
        declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

        set transaction isolation level repeatable read;
   start transaction;
                SELECT COUNT(*)
   from `General_user`
   where Username=var_username into var;
   IF var > 0
   then
                SELECT COUNT(*)
      from Categoria
      where Nome =var_categoria into var2;
      if var2>0
      then
                INSERT INTO Annuncio (titolo, prezzo, oggetto, descrizione, stato, UsernameA, IDA)
                values (var_titolo, var_prezzo, var_oggetto, var_descrizione , var_stato, var_username ,(select IDCategoria from Categoria where (Nome=Var_categoria)));

                Set var_ritorno = last_insert_id();

      end if;
        end if;
```

```
    commit;
END
```

Procedura per creare una categoria, inserire il proprio username e il nome della categoria

```
CREATE DEFINER=`Manuel`@`localhost` PROCEDURE `creareCategoria`(in var_nome varchar(45), in var_username varchar(45))
BEGIN
        declare var INT;
        declare exit handler for sqlexception
    begin
        rollback;
        resignal;
    end;

        set transaction isolation level repeatable read;
    start transaction;


        SELECT count(*)
    from `General_user`
    where Username=var_username into var ;
    IF var > 0
    then

        INSERT INTO Categoria (Nome, UsernameC)
        VALUES (var_nome, var_username);


    end if;
    commit;
END
```

Procedura per inserire un utente che si registra nel sistema

```
CREATE PROCEDURE `inserireUtente` (in var_username varchar(45),in var_data DATE ,in var_ind_fatturazione varchar(45), in var_ind_domicilio varchar(45),
in var_password char(32), in var_privilegio enum('admin','utente'), in var_telefono varchar(10), in var_cellulare varchar(10), in var_email varchar(45))

BEGIN
insert into `General_user`(Username, Dati_anagrafici, ind_fatturazione, Ind_Domicilio, password, Privilegio, telefono, Cellulare, Email)
 values (var_username, var_data, var_ind_fatturazione, var_ind_domicilio, var_password, var_privilegio, var_telefono, var_cellulare, var_email);

END
```

Procedura per selezionare e vedere la lista degli annunci creati

```
CREATE PROCEDURE `listaAnnunci` ()
BEGIN
```

```
        declare var int;
   declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
   start transaction;
   select count(*)
   from Annuncio
   where  stato != 'venduto' into var;
   if var>0
   then
         select IDAnnuncio, titolo, prezzo, oggetto, descrizione, UsernameA, IDA
    from Annuncio;

    end if;
    commit;
end
```

---

procedura per effettuare il login, inserendo username e password ritorna il ruolo

```
CREATE PROCEDURE `login` (in var_username char(32), in var_pass varchar(45), out var_role
INT)
BEGIN
        declare var_user_role ENUM('admin','utente');
        select `privilegio` from `General_user`
                where `username` = var_username
      and `password` = var_pass
      into var_user_role;
   if var_user_role='admin' then

      set var_role=1;
        elseif var_user_role= 'utente' then

                set var_role=2;
        end if;

END
```

Procedura per modificare un annuncio inserendo ID titolo prezzo e descrizione

```
CREATE PROCEDURE `modifica_annuncio` (in var_IDannuncio int, in var_titolo varchar(45), in var_prezzo int, in var_descrizione varchar(100))
BEGIN
	declare var int;
	declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

	set transaction isolation level repeatable read;
   start transaction;
   select count(*)
   from Annuncio
   where IDAnnuncio=var_IDannuncio into var;
   if var>0
   then
		UPDATE `Annuncio`
		SET
			`titolo`= var_titolo,
			`prezzo`= var_prezzo,
			`descrizione`=var_descrizione
		WHERE
			`IDAnnuncio` = var_IDannuncio;
	end if;
   commit;
END
```

Questa procedura genera un report degli annunci venduti

```
CREATE PROCEDURE `Report` ()
BEGIN
   declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
   start transaction;
	select UsernameA,
		COUNT(CASE WHEN stato = 'venduto' THEN 1 END) AS annunci_venduti,
     COUNT(*) AS annunci_totali,
     ROUND((COUNT(CASE WHEN stato = 'venduto' THEN 1 END) * 100.0) / COUNT(*), 2) AS percentuale_venduti
   from Annuncio
   group by UsernameA;
END
```

Procedura per scrivere un commento inserendo il testo del messaggio, l'id dell'annuncio, l'username di chi ha creato l'annuncio , e il proprio di chi commenta

```
CREATE PROCEDURE `scrivi_commento` (in var_msgtctC varchar(350), in var_idannuncio int, in var_username varchar(45), in var_usernamec varchar(45))
BEGIN
```

```
        declare var INT;
    declare var2 INT;
        declare exit handler for sqlexception
    begin
        rollback;
        resignal;
    end;

        set transaction isolation level repeatable read;
    start transaction;
        SELECT count(*)
    from `Annuncio`
    where `IDAnnuncio` = var_idannuncio into var ;
    IF var > 0
    then
                    SELECT count(*)
                    from `General_user`
                    where `Username`=var_username into var2 ;
                    IF var2> 0
                    then
                            INSERT INTO Commento (timestamp_commento,
msgtxt_commento, IDAnnuncioC, Username_Commento, UsernameCommento)
                            VALUES ((now()), var_msgtctC, var_idannuncio, var_username,
var_usernamec);
                    end if;
        end if;
    commit;
END
```

Procedura per scrivere un messaggio privato ad un venditore

```
CREATE PROCEDURE `scrivi_messaggio` (in var_msgtct varchar(350), in var_idannuncio int, in
var_username varchar(45), in var_scrittore varchar(45), in var_ricevente varchar(45))
BEGIN
        declare var INT;
    declare var2 INT;
        declare exit handler for sqlexception
    begin
        rollback;
        resignal;
    end;

        set transaction isolation level repeatable read;
    start transaction;
        SELECT count(*)
    from `Annuncio`
    where `IDAnnuncio` = var_idannuncio into var ;
    IF var > 0
    then
                    SELECT count(*)
                    from `General_user`
                    where `Username`=var_username into var2 ;
                    IF var2> 0
                    then
                            INSERT INTO Messaggio_privato(timestamp, msgtxt, IDA
nnuncioM, UsernameM, UsernameCM,UsernameRicevente)
```

```
                                        VALUES ((now()), var_msgtct, var_idannuncio, var_username,
var_scrittore, var_ricevente);
                                end if;
            end if;
    commit;

END
```

Procedura per seguire un annuncio inserendo il proprio username e quello dell'annuncio

```
CREATE PROCEDURE `seguire_annuncio` (in var_idannuncio int, in var_username varchar(45))
BEGIN
            declare var int;
    declare var2 int;
            declare exit handler for sqlexception
    begin
        rollback;
        resignal;
    end;

            set transaction isolation level read committed;
    start transaction;
    select count(*)
    from General_user
    where Username=var_username into var;
    if var>0
    then
                        select count(*)
        from Annuncio
                        where IDAnnuncio=var_idannuncio into var2;
        if var2>0
        then
                                INSERT INTO segui ( UsernameS, IDAnnuncioS)
                                VALUES (var_username, var_idannuncio);
        end if;
            end if;
    commit;
END
```

Procedura per visualizzare una categoria

```
CREATE PROCEDURE `VisualizzaCategoria` ()
BEGIN
declare exit handler for sqlexception
    begin
        rollback;
        resignal;
    end;

    set transaction isolation level read uncommitted;
    set transaction read only;
    start transaction;
             select IDCategoria, nome
     from Categoria;
END
```

Procedura per visualizzare i commenti di un annuncio inserendo l'id di un annuncio

```
CREATE PROCEDURE `VisualizzaCommenti` (in var_id int)
BEGIN
declare var int;
        declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
    start transaction;
   select count(*)
   from Annuncio
   where IDAnnuncio= var_id into var;
   if var>0
   then
                select IDCommento, timestamp_commento, msgtxt_commento, UsernameCommento,
UsernameCommento
                from Commento
                where IDAnnuncioC= var_id;
        END IF;
END
```

Procedura per visualizzare tutti i messaggi mandati o inviati, inserendo il proprio username

```
CREATE PROCEDURE `VisualizzaLeChat` (in var_User varchar(45))
BEGIN
  declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
   start transaction;
        select  msgtxt,UsernameM, UsernameCM, UsernameRicevente, IDAnnuncioM, IDMessaggio
   from Messaggio_privato
   WHERE UsernameM IN (
      SELECT DISTINCT UsernameM
      FROM Messaggio_privato
      WHERE UsernameCM = var_User OR UsernameRicevente = var_User

   );

END
```

Procedura per visualizzare un messaggi di un annuncio con il venditore inserendo il proprio user e l'id

```
CREATE PROCEDURE `VisualizzaMessaggi` (in var_id int,in var_user varchar(45))
BEGIN
        declare var int;
```

```
   declare var2 int;
         declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
    start transaction;
      select count(*)
   from Annuncio
   where IDAnnuncio= var_id into var;
   if var>0
   then
                  select count(*)
      from General_user
      where var_user=Username into var2;
      if var2>0
      then
                        select IDMessaggio, timestamp,  msgtxt, usernameRicevente, UsernameCM,
UsernameM, IDAnnuncioM
                        from Messaggio_privato
                        where (UsernameM= var_user or usernameRicevente = var_user ) and
var_id=IDAnnuncioM;
                  end if;
         end if;
END
```

Procedura per mettere come venduto un annuncio inserendo l'id ti tale

```
CREATE PROCEDURE `annuncioVenduto` (in var_IDannuncio int, in var_stato ENUM('venduto','online'))
begin
         declare var int;
         declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

         set transaction isolation level Read committed;
   start transaction;
                  select count(*)
                  from Annuncio
                  where IDAnnuncio=var_IDannuncio into var;
                  if var>0
                  then
                        UPDATE `Annuncio`
                        SET
                              `stato` = var_stato
                        WHERE
                              `IDAnnuncio` = var_IDannuncio;
                        end if;
         commit;
END
```

Procedura per selezionare tutti gli annunci di un'utente inserendo l'username

```
CREATE PROCEDURE `prendi_annunciDiunVenditore` (in var_username varchar(45))
BEGIN
	declare var int;
	declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
    start transaction;

   select count(*)
   from General_user
   where Username=var_username into var;
   if var>0
   then
		select IDAnnuncio , titolo, prezzo, oggetto, descrizione, (select nome from Categoria where
(IDA=IDCategoria)) as IDA
		from Annuncio
		where UsernameA= var_username;
	end if;
END
```

Procedura per selezione un annuncio dato l'id dell'annuncio

```
CREATE PROCEDURE `prendiUno_Annuncio` (in var_idannuncio int)
BEGIN
	declare var int;
	declare exit handler for sqlexception
   begin
      rollback;
      resignal;
   end;

   set transaction isolation level read committed;
   set transaction read only;
    start transaction;
   select count(*)
   from Annuncio
   where IDAnnuncio=var_idannuncio and stato != 'venduto' into var;
   if var>0
   then

		select IDAnnuncio, titolo, prezzo, oggetto, descrizione, stato, UsernameA, IDA
		from `Annuncio`
		where IDAnnuncio=var_idannuncio;
   else
		signal sqlstate '45000'
		set MESSAGE_TEXT ='Lannuncio non è più in commercio ';
	end if;
	commit;
END
```