PRIMA PARTE

1 Analisi e ipotesi aggiuntive

- la web community opera su più province in funzione della residenza dei propri iscritti: ogni iscritto può organizzare eventi esclusivamente nella provincia di residenza

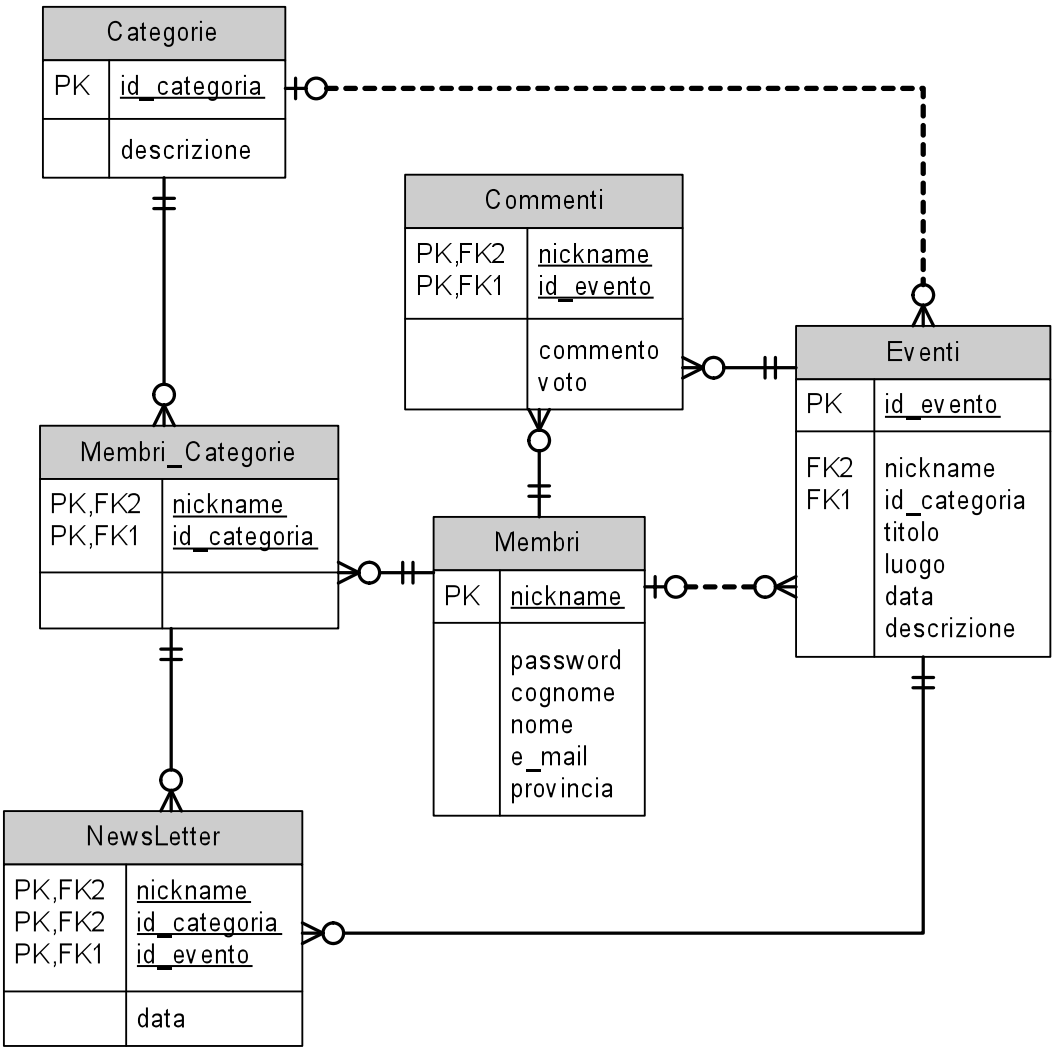
- i dati relativi agli artisti che si esibiscono sono inseriti nel campo descrittivo di ogni singolo evento; questa scelta è giustificata dal fatto che la possibile frammentazione dell'universo di artisti che si esibiscono, magari una sola volta, non giustifica la creazione di un'anagrafica dedicata

- la banca dati non prevede alcuna gestione relativamente agli utenti anonimi: essi possono operare solo per funzionalità limitate di interrogazione senza modificare in alcun modo i dati contenuti nel database

- relativamente alle newsletter è stato preso in considerazione solo un registro che tenga traccia dei messaggi inviati nel tempo ai singoli utenti: ogni membro riceve una mail per ogni singolo evento a cui può essere interessato in funzione delle categorie a cui è registrato

- le password dei membri registrati della web community sono memorizzate nel database in forma cifrata mediante un algoritmo hash (ad esempio MD5)

2 Diagramma database



3 Tabelle database

Categorie
È la tabella in cui sono censite le categorie in cui possono essere classificati i vari eventi:

Campo	R	Tipo	Dim.	Descrizione
id_categoria	PK	Carattere	5	Identificativo categoria eventi
descrizione		Carattere	50	Descrizione categoria

Chiave primaria: id_categoria

Commenti
È la tabella in cui i membri inseriscono i commenti e le valutazioni degli eventi:

Campo	R	Tipo	Dim.	Descrizione
nickname	PK/FK	Carattere	15	Identificativo membro
id_evento	PK/FK	Numerico		Identificativo evento
commento		Carattere	250	Commento evento
voto		Numerico		Valutazione numerica (1-10)

Chiave primaria: nickname, id_evento
Chiave esterna: id_evento → Eventi.id_evento
Chiave esterna: nickname → Membri.nickname

Eventi
È la tabella in cui i membri inseriscono gli eventi:

Campo	R	Tipo	Dim.	Descrizione
id_evento	PK	Numerico		Identificativo evento
nickname	FK	Carattere	15	Identificativo membro che inserisce l'evento
id_categoria	FK	Carattere	5	Identificativo categoria eventi
titolo		Carattere	50	Titolo evento
luogo		Carattere	20	Località svolgimento evento
data		Data/ora		Data e orario evento
descrizione		Carattere	255	Descrizione evento (artisti coinvolti, tipo performance, ...)

Chiave primaria: id_evento
Chiave esterna.: id_categoria → Categorie.id_categoria
Chiave esterna.: nickname → Membri.nickname

Membri
È la tabella che contiene l'anagrafica dei membri iscritti:

Campo	R	Tipo	Dim.	Descrizione
nickname	PK	Carattere	15	Identificativo membro
password		Carattere	32	Rappresentazione esadecimale della codifica MD5 della password
cognome		Carattere	20	Cognome
nome		Carattere	20	Nome

e_mail		Carattere	25	Indirizzo e-mail
provincia		Carattere	2	Sigla provincia di residenza

Chiave primaria: nickname

Membri_Categorie
È la tabella che permette di spezzare l'associazione M:N tra le due entità Membri e Categorie in due associazioni 1:N:

Campo	R	Tipo	Dim.	Descrizione
nickname	PK/FK	Carattere	15	Identificativo membro
id_categoria	PK/FK	Carattere	5	Identificativo categoria eventi

Chiave primaria: nickname, id_categoria
Chiave esterna: id_categoria → Categorie.id_categoria
Chiave esterna: nickname → Membri.nickname

NewsLetter
È la tabella che tiene traccia delle newsletter inviate nel tempo ai membri iscritti:

Campo	R	Tipo	Dim.	Descrizione
nickname	PK/FK	Carattere	15	Identificativo membro
id_categoria	PK/FK	Carattere	5	Identificativo categoria eventi
id_evento	PK/FK	Numerico		Identificativo evento
data		Data/ora		Data newsletter

Chiave primaria: nickname, id_categoria, id_evento
Chiave esterna: id_evento → Eventi.id_evento
Chiave esterna: nickname, id_categoria → Membri_Categorie.nickname, id_categoria

4 DB-schema

```
CREATE TABLE Categorie(
 id_categoria VARCHAR(5),
 descrizione VARCHAR(50),
 CONSTRAINT PrimaryKey PRIMARY KEY(id_categoria)
);

CREATE TABLE Eventi(
 id_evento INTEGER,
 nickname VARCHAR(15),
 id_categoria VARCHAR(5),
 titolo VARCHAR(50),
 luogo VARCHAR(20),
 data DATETIME,
 descrizione VARCHAR(255),
 CONSTRAINT CategorieEventi FOREIGN KEY(id_categoria)
            REFERENCES Categorie(id_categoria),
 CONSTRAINT PrimaryKey PRIMARY KEY(id_evento)
);

CREATE TABLE Membri(
 nickname VARCHAR(15),
 password VARCHAR(32),
```

```
 cognome VARCHAR(20),
 nome VARCHAR(20),
 e_mail VARCHAR(25),
 provincia VARCHAR(2),
 CONSTRAINT PrimaryKey PRIMARY KEY(nickname)
);

CREATE TABLE Membri_Categorie(
 nickname VARCHAR(15),
 id_categoria VARCHAR(5),
 CONSTRAINT CategorieMembri_Categorie FOREIGN KEY(id_categoria)
            REFERENCES Categorie(id_categoria),
 CONSTRAINT MembriMembri_Categorie FOREIGN KEY(nickname)
            REFERENCES Membri(nickname),
 CONSTRAINT PrimaryKey PRIMARY KEY(nickname,id_categoria)
);

CREATE TABLE NewsLetter(
 nickname VARCHAR(15),
 id_categoria VARCHAR(5),
 id_evento INTEGER,
 data DATETIME,
 CONSTRAINT EventiNewsLetter  FOREIGN KEY(id_evento)
            REFERENCES Eventi(id_evento),
 CONSTRAINT Membri_CategorieNewsLetter  FOREIGN KEY(nickname,id_categoria)
            REFERENCES Membri_Categorie(nickname,id_categoria),
 CONSTRAINT PrimaryKey PRIMARY KEY(nickname,id_categoria,id_evento)
);

CREATE TABLE Commenti(
 nickname VARCHAR(15),
 id_evento INTEGER,
 commento VARCHAR(250),
 voto INTEGER,
 CONSTRAINT EventiCommenti  FOREIGN KEY(id_evento)
            REFERENCES Eventi(id_evento),
 CONSTRAINT MembriCommenti  FOREIGN KEY(nickname)
            REFERENCES Membri(nickname),
 CONSTRAINT PrimaryKey PRIMARY KEY(nickname,id_evento)
);
```

5 Query

a)

```
SELECT *
FROM Eventi,Membri
WHERE data<CURDATE()
  AND Eventi.nickname = Membri.nickname
ORDER BY provincia;
```

b)

La richiesta è stata risolta tramite una query nidificata in cui vengono selezionati i nickname di coloro che hanno lasciato almeno un commento: la clausola **NOT IN** permette di escludere questi ultimi dalla selezione.

```
SELECT *
FROM Membri
WHERE nickname NOT IN (
                       SELECT DISTINCT nickname
                       FROM commenti
                      );
```

c)

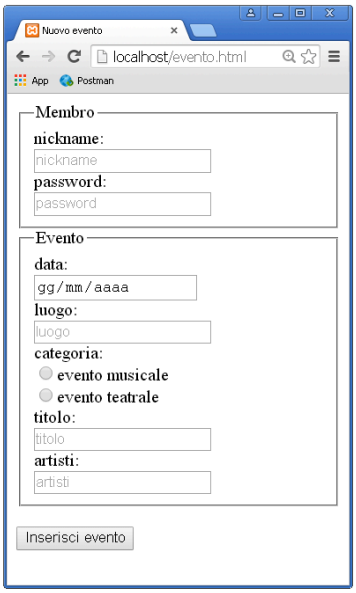
```
SELECT Eventi.id_evento, titolo, AVG(voto) AS voto_medio
FROM Commenti,Eventi
WHERE Commenti.id_evento = Eventi.id_evento
GROUP BY Eventi.id_evento, titolo
ORDER BY id_categoria,titolo;
```

d)

```
SELECT nickname
FROM (
      SELECT nickname, COUNT(*) AS n_eventi
      FROM Eventi
      GROUP BY nickname
     ) AS T1
WHERE n_eventi IN (
                   SELECT MAX(n_eventi)
                   FROM (
                         SELECT nickname, COUNT(*) AS n_eventi
                         FROM Eventi
                         GROUP BY nickname
                        ) AS T2
                   );
```

6 Pagina web

Una delle operazioni specifiche di un utente registrato è quella relativa alla creazione di un nuovo evento che può essere effettuata con una pagina web contenente una *form* come la seguente (la pagina è volutamente priva di struttura grafica) generata dal seguente codice HTML5:




```
<!DOCTYPE HTML>
<html>
 <head>
  <title>Nuovo evento</title>
 </head>
 <body>
  <form action="www.webcommunity.org/new_event.html" method="get">
   <fieldset>
    <legend>Evento</legend>
    nickname:<br>
    <input type="text" placeholder="nickname" name="nickname" required><br>
    password:<br>
    <input type="password" placeholder="password" name="password" required>
   </fieldset>
   <fieldset>
    <legend>Evento</legend>
    data:<br>
    <input type="date" name="data" required><br>
    luogo:<br>
    <input type="text" placeholder="luogo" name="luogo" required><br>
    categoria:<br>
    <input type="radio" name="categoria" value="musica">evento musicale<br>
    <input type="radio" name="categoria" value="teatro">evento teatrale<br>
    titolo:<br>
    <input type="text" placeholder="titolo" name="titolo" required><br>
    artisti:<br>
    <input type="text" placeholder="artisti" name="artisti">
   </fieldset>
   <br><input type="submit" value="Inserisci evento">
  </form>
 </body>
</html>
```

7 Sito web

Date le caratteristiche e la natura dell'organizzazione che, probabilmente, non dispone di personale specializzato per la gestione di un server per ospitare il sito web, la soluzione più indicata è quella di acquistare un servizio di *hosting* del sito da un fornitore di servizi web (spesso è possibile acquistare il servizio di *hosting* del sito insieme alla registrazione del dominio che in questo caso potrebbe essere www.webcommunity.org): questa soluzione è senz'altro la migliore tenuto conto che il sito deve essere operativo 24 ore su 24 e 7 giorni su 7, ed è anche economica. È necessario acquistare, oltre allo spazio web per ospitare il sito, l'integrazione dell'interprete PHP nel server web e la disponibilità di un database My-SQL per la gestione del database: si tratta in ogni caso di un pacchetto molto comune nell'offerta dei fornitori di servizi web.

Come esempio di pagina web dinamica che costituisce il sito della web community presentiamo due script PHP (il primo per inserire la richiesta ed il secondo per visualizzare la relativa risposta) che realizzano il comportamento richiesto per la visualizzazione degli eventi in programmazione in una certa provincia nel tempo, compresi quelli già effettuati. Si ipotizza che il server My-SQL sia in esecuzione sullo stesso computer del web server e che l'accesso al database "Web_community" sia effettuato come utente "root" privo di password; inoltre le pagine web sono completamente prive di struttura grafica.

richiesta.php:

```
<html>
 <head>
  <title>Ricerca eventi</title>
 </head>
 <body>
  Selezionare provincia e categoria eventi:
  <?php
   $connection = mysqli_connect("localhost", "root", "", "Web_community");
   if (mysqli_connect_errno($connection))
     {
      echo "Errore di connessione al DBMS My-SQL.";
      die();
     }
  ?>
  <form action="risposta.php" method="GET">
   <select name="provincia">
    <?php
     $query = "SELECT DISTINCT provincia
               FROM Eventi,Membri
               WHERE Eventi.nickname=Membri.nickname;";
     $result = mysqli_query($connection, $query);
     if (!$result)
       {
        echo "Errore esecuzione query SQL.";
        die();
       }
    if (mysqli_num_rows($result) == 0)
      {
       echo "Nessuna provincia presente.";
       die();
      }

    while ($row = mysqli_fetch_array($result))
         {
          echo "<option value=\"$row[0]\">$row[0]</option>";
```

```
        }
    mysqli_free_result($result);
    ?>
   </select>
   <br>
   <select name="categoria">
    <?php
     $query = "SELECT id_categoria, descrizione FROM Categorie;";
     $result = mysqli_query($connection, $query);
     if (!$result)
       {
        echo "Errore esecuzione query SQL.";
        die();
       }
     if (mysqli_num_rows($result) == 0)
       {
        echo "Nessuna categoria registrata.";
        die();
       }

     while ($row = mysqli_fetch_array($result))
          {
           echo "<option value=\"$row[0]\">$row[1]</option>";
          }
     mysqli_free_result($result);
     mysqli_close($connection);
    ?>
   </select>
   <br>
   <input type="submit" value="Cerca">
   <br>
  </form>
 </body>
</html>
```

risposta.php:

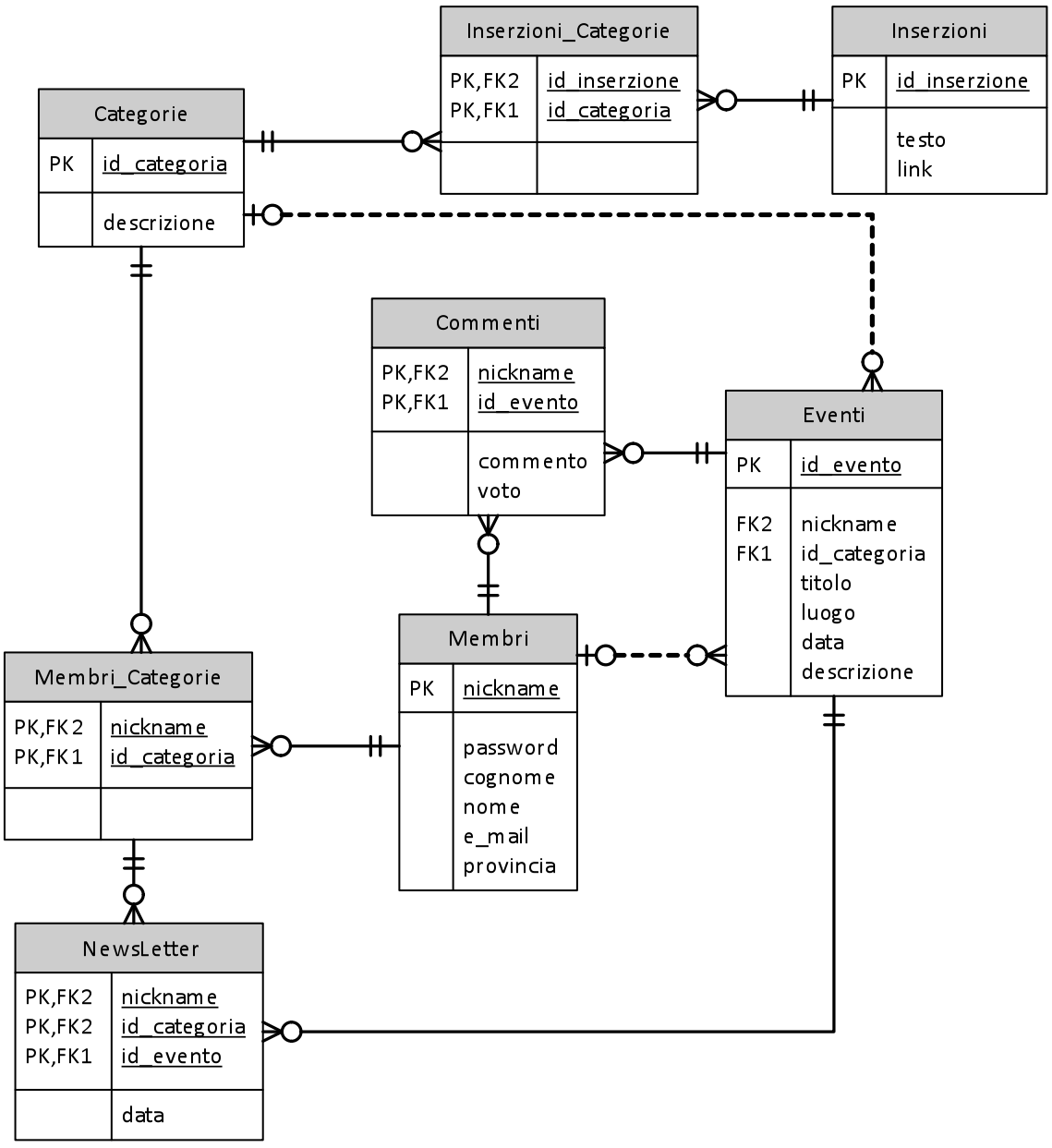
```
<html>
 <head>
  <title>Cronologia eventi</title>
 </head>
 <body>
  <?php
   $connection = mysqli_connect("localhost", "root", "", "Web_community");
   if (mysqli_connect_errno($connection))
     {
      echo "Errore di connessione al DBMS My-SQL.";
      die();
     }
   $query = "SELECT data, titolo, luogo, descrizione
             FROM Eventi, Membri
             WHERE Membri.nickname = Eventi.nickname
                   AND Eventi.id_Categoria = '$_GET[categoria]'
                   AND provincia = '$_GET[provincia]'
             ORDER BY data;";
   $result = mysqli_query($connection, $query);
   if (!$result)
     {
      echo "Errore esecuzione query SQL.";
      die();
```

```
    }
  if (mysqli_num_rows($result) == 0)
    {
     echo "Nessun evento trovato.";
     die();
    }
 ?>
 <table border>
  <caption><b>Eventi per provincia</b></caption>
  <thead>
   <tr>
    <th>Data</th>
    <th>Titolo</th>
    <th>Luogo</th>
    <th>Descrizione</th>
   </tr>
  </thead>
  <tbody>
   <?php
    while ($row = mysqli_fetch_assoc($result))
         {
   ?>
          <tr>
           <td><?php echo ($row['data']); ?></td>
           <td><?php echo ($row['titolo']); ?></td>
           <td><?php echo ($row['luogo']); ?></td>
           <td><?php echo ($row['descrizione']); ?></td>
          </tr>
   <?php
         }
    mysqli_free_result($result);
    echo "</tbody>\n";
    echo "</table>\n";
    echo "<br>\n";
    mysqli_close($connection);
   ?>
 </body>
</html>
```

SECONDA PARTE

A titolo di esempio viene fornita la risposta ai quesiti I e III.

I) Le inserzioni pubblicitarie sono identificate da un codice numerico e caratterizzate da un testo descrittivo e da un collegamento ipertestuale; la relazione tra inserzioni e categorie e generalmente di tipo N:M per cui, oltre alla tabella "Inserzioni" è necessario aggiungere al database la tabella "Inserzioni_Categorie" che spezza la relazione in due relazioni 1:N:



III)

Corsi

Cognome	Nome	Telefono	Livello	Tutor	Tel_tutor	Anticipo_versato
Verdi	Luisa	345698741	avanzato	Bianca	334563215	100
Neri	Enrico	348523698	avanzato	Carlo	369852147	150
Rosi	Rosa	347532159	base	Alessio	333214569	120
Bianchi	Paolo	341236547	base	Carlo	369852147	150
Rossi	Mario	349567890	base	Carlo	369852147	90
Neri	Enrico	348523698	complementi	Dina	373564987	100

Tutti i domini della tabella Corsi sono atomici, affinché essa sia in 1NF è necessario definire una chiave primaria che potrebbe essere il campo Telefono.
Si osservano le seguenti dipendenze funzionali che fanno in modo che la tabella non sia in 2NF:

Telefono → Cognome, Nome
(Telefono, Livello) → Anticipo_versato
(Telefono, Livello) → Tel_tutor, Tutor

Al fine di ottenere la seconda forma normale si possono operare le seguenti scomposizioni:

Corsi1

Telefono	Livello	Anticipo_versato
345698741	avanzato	100
348523698	avanzato	150
347532159	base	120
341236547	base	150
349567890	base	90
348523698	complementi	100

Utenti

Telefono	Cognome	Nome
345698741	Verdi	Luisa
348523698	Neri	Enrico
347532159	Rosi	Rosa
341236547	Bianchi	Paolo
349567890	Rossi	Mario
348523698	Neri	Enrico

Utenti_Tutor

Telefono	Livello	Tel_tutor	Tutor
345698741	avanzato	334563215	Bianca
348523698	avanzato	369852147	Carlo
347532159	base	333214569	Alessio
341236547	base	369852147	Carlo
349567890	base	369852147	Carlo
348523698	complementi	373564987	Dina

Nella tabella Utenti_Tutor c'è una dipendenza funzionale di tipo transitivo. Infatti il nome del tutor (Tutor) dipende transitivamente dalla chiave (Telefono, Livello) attraverso l'attributo Tel_tutor per cui è necessario scomporre Utenti_Tutor in Utenti_Tutor1 e Tutor per ottenere la 3FN:

Corsi1

Telefono	Livello	Anticipo_versato
345698741	avanzato	100
348523698	avanzato	150
347532159	base	120
341236547	base	150
349567890	base	90
348523698	complementi	100

Utenti

Telefono	**Cognome**	**Nome**
345698741	Verdi	Luisa
348523698	Neri	Enrico
347532159	Rosi	Rosa
341236547	Bianchi	Paolo
349567890	Rossi	Mario
348523698	Neri	Enrico

Utenti_Tutor1

Telefono	**Livello**	**Tel_tutor**
345698741	avanzato	334563215
348523698	avanzato	369852147
347532159	base	333214569
341236547	base	369852147
349567890	base	369852147
348523698	complementi	373564987

Tutor

Tel_tutor	**Tutor**
334563215	Bianca
369852147	Carlo
333214569	Alessio
373564987	Dina