Un'azienda *start-up* vuole costruire una piattaforma Web che consenta il *car pooling* tra viaggiatori sul territorio nazionale, con l'obiettivo di diffondere l'uso di una mobilità flessibile e personalizzata in termini di percorsi e costi.
Gli utenti della piattaforma possono essere di due tipi: utenti-autisti (coloro che offrono un passaggio con la propria macchina) e utenti-passeggeri (coloro che usufruiscono del passaggio).
Gli autisti devono registrarsi sul sito ed inserire i propri dati: generalità, numero e scadenza patente di guida, dati dell'automobile utilizzata, recapito telefonico, email, fotografia.
Per ogni viaggio che intendono condividere, gli autisti devono indicare città di partenza, città di destinazione, data ed ora di partenza, contributo economico richiesto ad ogni passeggero, tempi di percorrenza stimati. È responsabilità dell'autista, mano a mano che accetterà passeggeri per un certo viaggio, dichiarare chiuse le prenotazioni per quel viaggio, utilizzando un'apposita funzione sul portale.
L'utente-passeggero si deve registrare sulla piattaforma, indicando cognome e nome, documento di identità, recapito telefonico ed email. La piattaforma fornisce ai passeggeri la possibilità di indicare città di partenza e di destinazione e data desiderata; presenta quindi un elenco di viaggi (per cui non siano ancora chiuse le prenotazioni), ciascuno con le caratteristiche dell'autista e le modalità del viaggio stesso inserite dall'autista (orario, eventuali soste previste alle stazioni di servizio, possibilità di caricare bagaglio o animali, …).
Il passeggero sceglie quindi il viaggio desiderato con il corrispondente autista, anche esaminando il voto medio e i giudizi dei *feedback* assegnati tramite la piattaforma dai precedenti passeggeri all'autista stesso, e si prenota. Le informazioni sul passeggero vengono inviate per email dalla piattaforma all'autista scelto, il quale può consultare sul portale il voto medio e i giudizi dei *feedback* ricevuti dal passeggero da parte di precedenti autisti e decidere se accettarlo o meno. Il passeggero di conseguenza riceverà una email di accettazione o di rifiuto della prenotazione effettuata, contenente, in caso di accettazione, un promemoria con città di partenza e destinazione, data e orario del viaggio, dati dell'autista e della sua automobile.
A viaggio effettuato, il passeggero può inserire un *feedback* sull'autista, espresso sia in forma di voto numerico che di giudizio discorsivo. A sua volta, l'autista può inserire un *feedback* sul passeggero, espresso sia in forma di voto numerico che di giudizio discorsivo. Sia i voti medi che i singoli giudizi dei *feedback* ricevuti da ciascun autista sono disponibili ai passeggeri; analogamente, sia i voti medi che i singoli giudizi dei *feedback* ricevuti da ciascun passeggero sono disponibili agli autisti.

ER → 1°

LOGICO → 2°

CREAZ. TABELLE → 3°

QUERY SQL → 6°

ARCHITETTURA → 5°

SECONDA PARTE

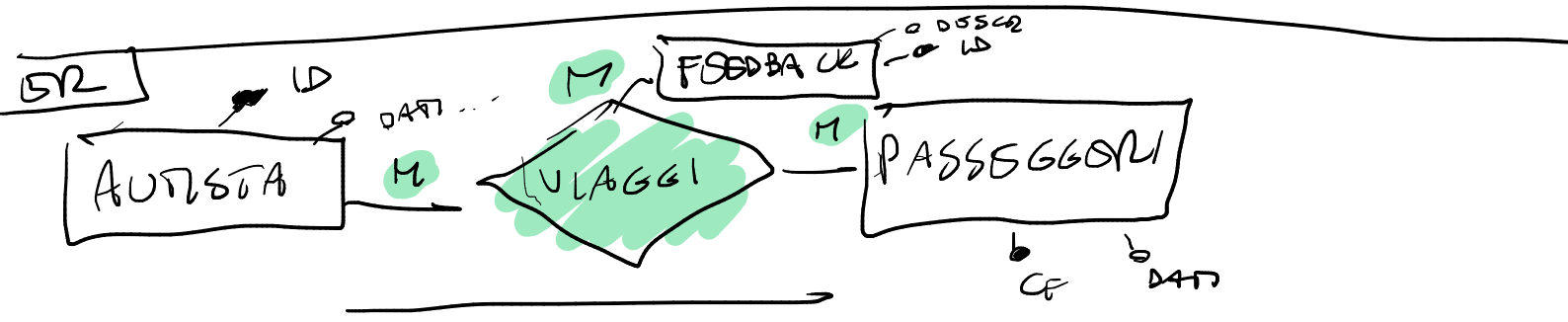
2 QUESITI

A SCELTA

TEORIA

PRATICA

PHP

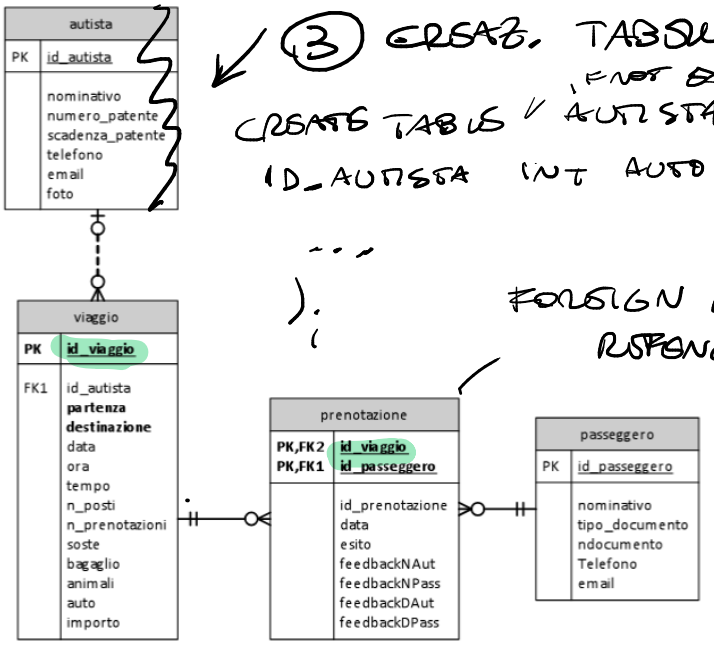


M - M = VIAGGI

TABELLA = LOGICO

VIAGGI:
FK (AUTISTA)
FK (PASSEGGERI)
FK (FEEDBACK)

2.1 Diagramma database



3. le seguenti interrogazioni espresse in linguaggio SQL:
   a) data una città di partenza, una di arrivo e una data, elencare gli autisti che propongono un viaggio corrispondente con prenotazioni non ancora chiuse, in ordine crescente di orario, riportando i dati dell'auto e il contributo economico richiesto;
   b) dato il codice di una prenotazione accettata, estrarre i dati necessari per predisporre l'email di promemoria da inviare all'utente passeggero;
   c) dato un certo viaggio, consentire all'autista di valutare le caratteristiche dei passeggeri visualizzando l'elenco di coloro che lo hanno prenotato, con il voto medio dei feedback ricevuti da ciascun passeggero, presentando solo i passeggeri che hanno voto medio superiore ad un valore indicato dall'autista;
4. il progetto di massima della struttura funzionale dell'applicazione Web, realizzando, con appropriati linguaggi a scelta sia lato client che lato server, un segmento significativo dell'applicazione che consente l'interazione con la base di dati.

4 Pagina web

Il sito web che gestisce i dati del sistema di *car-pooling* dovrà prevedere la possibilità per i potenziali passeggeri di verificare la disponibilità di offerta relativamente ai viaggi. Ogni passeggero, dopo essersi registrato, sarà dotato di username e password forniti dal sistema per l'accesso ai servizi previsti dall'interfaccia del sito. Una volta effettato l'accesso sarà possibile ricercare viaggi offerti su un itinerario desiderato (partenza/destinazione) in un periodo di tempo compreso tra due date. Il report fornito dal sistema fornirà il risultato in maniera cronologica: data e ora di inizio viaggio, tempo medio previsto in ore per il viaggio, nome ed e-mail dell'autista, tipo di auto, posti ancora disponibili.

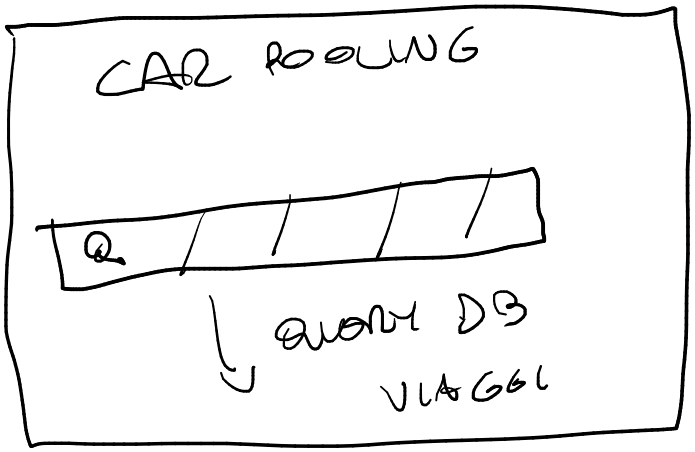
Supponendo di aver già effettuato il login e inserito l'intervallo temporale di riferimento e le città di origine e destinazione, uno *script* minimale in linguaggio PHP per la realizzazione del report è il seguente:

```
 <title>Viaggi disponibili</title>
</head>
<body>
 <?php
  $data1=$_GET['data_iniziale'];
  $data2=$_GET['data_finale'];
  $partenza=$_GET['partenza'];
  $destinazione=$_GET['destinazione'];

  $connection = mysqli_connect("localhost", "…", "…", "carpooling");
  if (mysqli_connect_errno($connection))
    {
     echo "Errore di connessione al DBMS My-SQL." ;
     die();
    }

  $query = "SELECT autista.nominativo, autista.email,
                   viaggio.data, viaggio.ora, viaggio.tempo,viaggio.aut
                   (viaggio.n_posti-viaggio.n_prenotazioni) AS posti_di
              FROM autista INNER JOIN viaggio
                   ON autista.id_autista = viaggio.id_autista
             WHERE viaggio.partenza='".$partenza."'
               AND viaggio.destinazione='".$destinazione."'
               AND viaggio.data BETWEEN '".$data1."' AND '".$data2."'
               AND viaggio.n_posti-viaggio.n_prenotazioni>0
             ORDER BY viaggio.data,viaggio.ora;";
  $result = mysqli_query($connection, $query);
  if (!$result)
    {
     echo "Errore esecuzione query SQL.";
     die();
    }
  if (mysqli_num_rows($result) == 0)
    {
     echo "Nessun viaggio trovato.";
     die();
```

```
 ?>
 <table border>
  <caption><b>Viaggi disponibili</b></caption>
  <thead>
   <tr>
    <th>Data</th>
    <th>Ora</th>
    <th>Tempo(ore)</th>
    <th>Autista</th>
    <th>E-mail</th>
    <th>Auto</th>
    <th>Posti disponibili</th>
   </tr>
  </thead>
  <tbody>
  <?php
   while ($row = mysqli_fetch_assoc($result))
         {
  ?>
          <tr>
           <td><?php echo ($row['data']); ?></td>
           <td><?php echo ($row['ora']); ?></td>
           <td><?php echo ($row['tempo']); ?></td>
           <td><?php echo ($row['nominativo']); ?></td>
           <td><?php echo ($row['email']); ?></td>
           <td><?php echo ($row['auto']); ?></td>
           <td><?php echo ($row['posti_disponibili']); ?></td>
          </tr>
  <?php
         }
   mysqli_free_result($result);
   echo "</tbody>\n";
   echo "</table>\n";
   mysqli_close($connection);
  ?>
 </body>
```



"PROGETTO" ⇒ CODICE RILEVANTE

1° → INPUT → $_GET

2° → CONNESSIONE → $conn = MYSQLI_CONNECT ("DB", "IND", "UT", "PSW")

3° → QUERY → $RIS = MYSQLI_QUERY ($conn, $query);

4° → VIS. QUERY → MYSQLI_NUM_ROWS
WHILE ($row = MYSQLI_FETCH_ASSOC ($RISULT);

5° → CHIUS. CONNESSIONE

SECONDA PARTE

→ [2] TEORICA

→ [2] PRATICA (DI CUI (1) È CODICE - PARTE 2)

SECONDA PARTE

I. In relazione al tema proposto nella prima parte, il candidato integri il modello già realizzato al fine di gestire in automatico il numero di posti disponibili nei vari viaggi, evitando che sia responsabilità dell'autista dichiarare chiuse le prenotazioni sul portale. Nel momento in cui inserisce un viaggio, l'autista dichiara il numero massimo di posti disponibili. Mano a mano che gli autisti accettano le prenotazioni, il sistema visualizzerà solo i viaggi con posti ancora disponibili: a tal fine, una prenotazione non ancora accettata dall'autista non comporta alcun impegno del posto, che resta così ancora disponibile per prenotazioni di altri passeggeri. Per ciascun viaggio, la piattaforma mostrerà il numero dei posti disponibili e il numero delle prenotazioni non ancora accettate. Il candidato sviluppi poi la pagina web, sia lato client che lato server, per fornire ai passeggeri tali informazioni.


```php
<html>
 <head>
  <title>Viaggi disponibili</title>
 </head>
 <body>
  <?php
   $data1=$_GET['data_iniziale'];
   $data2=$_GET['data_finale'];
   $partenza=$_GET['partenza'];
   $destinazione=$_GET['destinazione'];

   $connection = mysqli_connect("localhost", "…", "…", "carpooling");
   if (mysqli_connect_errno($connection))
     {
      echo "Errore di connessione al DBMS My-SQL.";
      die();
     }

   $query = "SELECT *
             FROM
             (SELECT id_viaggio,autista.nominativo, autista.email,
                   viaggio.data, viaggio.ora, viaggio.tempo,viaggio.auto,
                  (viaggio.n_posti- viaggio.n_prenotazioni) AS posti_disponibili
               FROM autista INNER JOIN viaggio
                   ON autista.id_autista = viaggio.id_autista
               WHERE viaggio.partenza='".$partenza."'
                 AND viaggio.destinazione='".$destinazione."'
                 AND viaggio.data BETWEEN '".$data1."' AND '".$data2."'
                 AND viaggio.n_posti-viaggio.n_prenotazioni>0
                 AND viaggio.n_posti-viaggio.n_prenotazioni>0) AS t1,
             (SELECT viaggio.id_viaggio,
                   COUNT(*) AS prenotazioni_non_ancora_confermate
               FROM viaggio INNER JOIN prenotazione
                   ON viaggio.id_viaggio = prenotazione.id_viaggio
               WHERE viaggio.partenza='".$partenza."'
                 AND viaggio.destinazione='".$destinazione."'
                 AND viaggio.data BETWEEN '".$data1."' AND '".$data2."'
                 AND esito IS NULL
               GROUP BY viaggio.id_viaggio) AS T2
             WHERE t1.id_viaggio = t2.id_viaggio
             ORDER BY data, ora;";
   $result = mysqli_query($connection, $query);
   if (!$result)
     {
      echo "Errore esecuzione query SQL.";
      die();
     }
   if (mysqli_num_rows($result) == 0)
     {
      echo "Nessun viaggio trovato.";
      die();
     }
   ?>
   <table border>
    <caption><b>Viaggi disponibili</b></caption>
    <thead>
     <tr>
      <th>Data</th>
      <th>Ora</th>
      <th>Tempo(ore)</th>
      <th>Autista</th>
      <th>E-mail</th>
      <th>Auto</th>
      <th>Posti disponibili</th>
      <th>Prenotazioni non ancora confermate</th>
     </tr>
    </thead>
    <tbody>
     <?php
      while ($row = mysqli_fetch_assoc($result))
           {
     ?>
           <tr>
            <td><?php echo ($row['data']); ?></td>
            <td><?php echo ($row['ora']); ?></td>
            <td><?php echo ($row['tempo']); ?></td>
            <td><?php echo ($row['nominativo']); ?></td>
            <td><?php echo ($row['email']); ?></td>
            <td><?php echo ($row['auto']); ?></td>
            <td><?php echo ($row['posti_disponibili']); ?>
            <td><?php echo ($row['prenotazioni_non_ancora_
           </tr>
     <?php
           }
      mysqli_free_result($result);
      echo "</tbody>\n";
      echo "</table>\n";
      mysqli_close($connection);
      ?>
 </body>
</html>
```

AGGREGAZIONI [COUNT (MAX/ MIN (AVG

GROUP BY ? HAVING ? → QUANDO AGGREG. → USO GROUP BY

DATA = WHERE (IF)

SELECT

WHERE

ORDER BY

AVG

3. le seguenti interrogazioni espresse in linguaggio SQL:
   a) data una città di partenza, una di arrivo e una data, elencare gli autisti che propongono un viaggio corrispondente con prenotazioni non ancora chiuse, in ordine crescente di orario, riportando i dati dell'auto e il contributo economico richiesto;
   b) dato il codice di una prenotazione accettata, estrarre i dati necessari per predisporre l'email di promemoria da inviare all'utente passeggero;
   c) dato un certo viaggio, consentire all'autista di valutare le caratteristiche dei passeggeri visualizzando l'elenco di coloro che lo hanno prenotato, con il voto medio dei feedback ricevuti da ciascun passeggero, presentando solo i passeggeri che hanno voto medio superiore ad un valore indicato dall'autista;

SELECT COUNT (*) AS NUMERO, NOME

... GROUP BY NOME ?

-> CONTROLLA CHE
IL NUMERO DI USER COSTUMI
SIA MASSIMO

1° QUERY

SELECT COUNT(*) AS NUMERO
FROM TABELLA

AS q1

2° QUERY

SELECT q1. NUMERO
FROM q1
WHERE NUMERO2
(SELECT MAX(NUMERO)
FROM q1))

PRIMA PARTE

Si vuole realizzare una web community per condividere dati e commenti relativi a eventi dal vivo di diverse categorie, ad esempio concerti, spettacoli teatrali, balletti, ecc. che si svolgono in Italia.

Gli eventi vengono inseriti sul sistema direttamente dai membri stessi della community, che si registrano sul sito fornendo un nickname, nome, cognome, indirizzo di e-mail e scegliendo una o più categorie di eventi a cui sono interessati.

Ogni membro iscritto riceve periodicamente per posta elettronica una newsletter, emessa automaticamente dal sistema, che riporta gli eventi delle categorie da lui scelte, che si svolgeranno nella settimana seguente nel territorio provinciale dell'utente.

I membri registrati possono interagire con la community sia inserendo i dati di un nuovo evento, per il quale occorre specificare categoria, luogo di svolgimento, data, titolo dell'evento e artisti coinvolti, sia scrivendo un post con un commento ed un voto (da 1 a 5) su un evento.

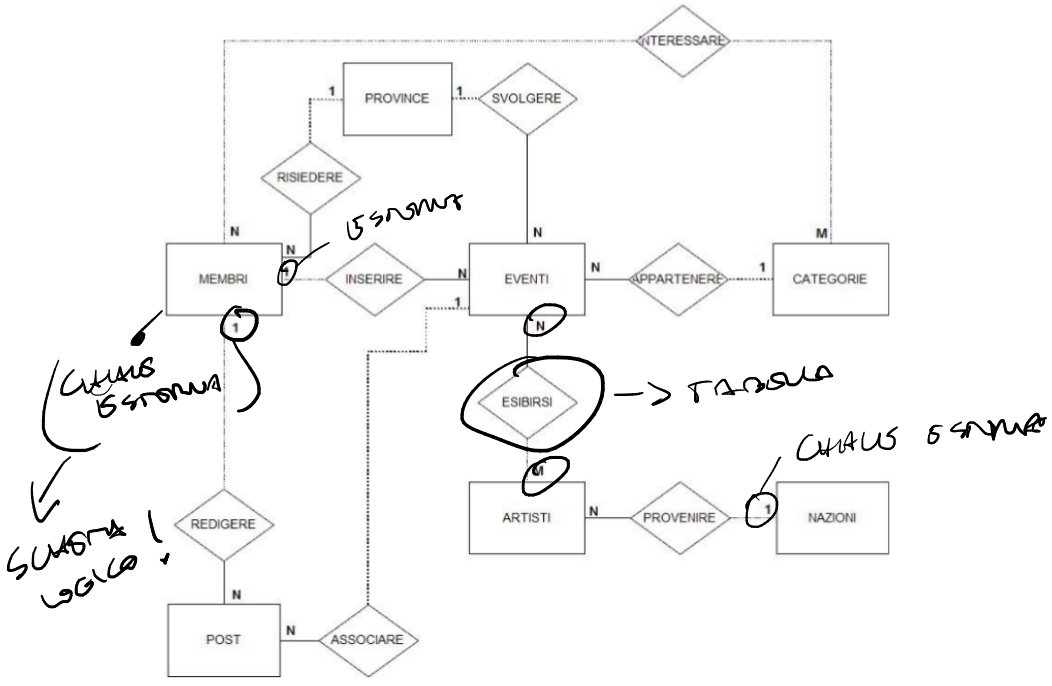
Il sito della community offre a tutti, sia membri registrati sia utenti anonimi, la consultazione dei dati on line, tra cui:

- visualizzazione degli eventi di un certo tipo in ordine cronologico, con possibilità di filtro per territorio di una specifica provincia
- visualizzazione di tutti i commenti e voti relativi ad un evento.

Il candidato, fatte le opportune ipotesi aggiuntive, sviluppi

1. un'analisi della realtà di riferimento individuando le possibili soluzioni e scelga quella che a suo motivato giudizio è la più idonea a rispondere alle specifiche indicate
2. uno schema concettuale della base di dati

Diagramma E/R in notazione di Chen estesa.



1 – Analisi

Leggendo attentamente la traccia si individuano agevolmente i fabbisogni richiesti dalla 'web community' e di conseguenza gli elementi importanti della realtà da modellare.

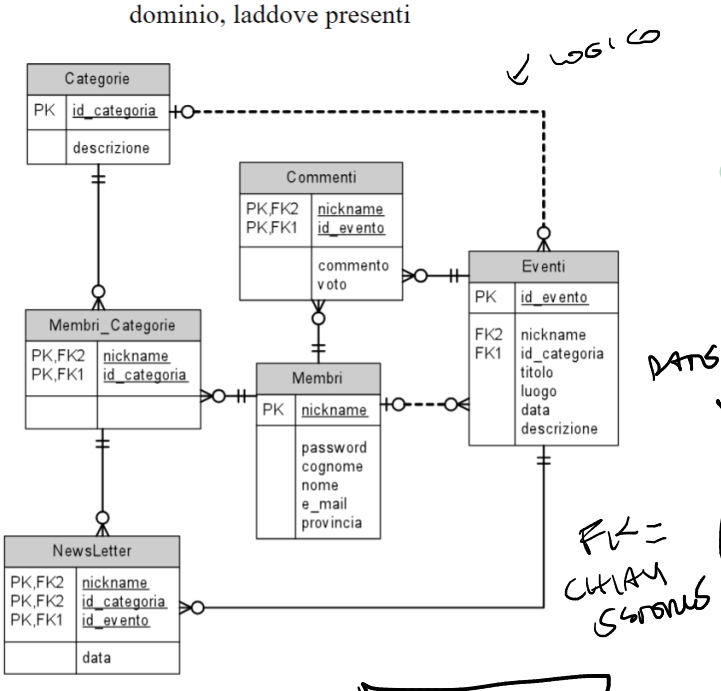
- Gestione dei membri della community che devono potersi iscrivere e naturalmente poter aggiornare i loro dati e le preferenze. Durante l'iscrizione, che comprende l'inserimento di vari dati anagrafici nonché di nickname e password, devono poter scegliere la provincia di appartenenza e le categorie di eventi di proprio interesse, nonché impostare un indicatore per ricevere o meno una newsletter.
Le categorie sono memorizzate da un amministratore una volta per tutte e così pure le province.

- Gestione degli eventi 'dal vivo' che si svolgono in Italia. I membri della community possono inserire eventi corredati da varie informazioni tra cui la categoria di appartenenza dell'evento e la provincia di svolgimento dell'evento. I membri possono inserire eventi che si svolgono anche in province diverse da quella della loro residenza.
Gli artisti che partecipano ad un evento sono inseriti con apposita funzione dal membro stesso, dopo una ricerca con esito negativo nell'archivio artisti. Gli artisti vengono identificati con dati anagrafici essenziali e con la nazione di residenza/provenienza scelta da un archivio nazioni precaricato da un amministratore.

- Gestione dei 'Commenti e Voti' identificati per brevità con 'post'. I membri possono inserire un post per un dato evento; un membro non può inserire più commenti e voti sul solito evento.
In particolare un membro, dopo aver fatto il 'login' con nickname e password, effettua una ricerca sugli eventi per provincia e/o per categoria e poi dall'elenco prodotto sceglie l'evento da commentare e votare.

- Gli utenti anonimi, di cui non si tiene ovviamente traccia, possono solo ricercare eventi visualizzandone i dati ed i relativi commenti e voti. Non possono inserire o variare dati degli eventi e dei post.
- Gestione della Newsletter. Questa viene generata in automatico dal sistema per essere inviata ai membri che hanno impostato il relativo indicatore per ricevere Newsletter, in base alle categorie di eventi da loro scelte e per gli eventi programmati nella provincia di residenza.
In questa soluzione si decide di non tenere traccia delle newsletter inviate ma potrebbe essere utile memorizzare tale invio con data, ora e categorie coinvolte.

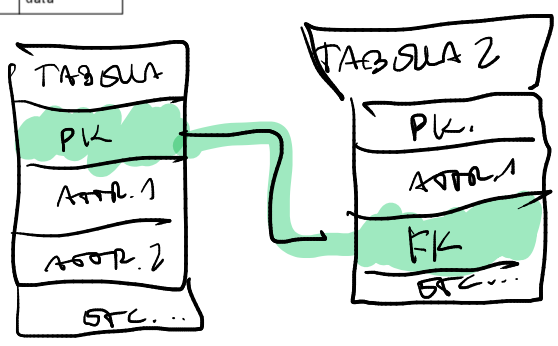
3. uno schema logico della base di dati
4. la definizione in linguaggio SQL di un sottoinsieme delle relazioni della base di dati in cui siano presenti alcune di quelle che contengono vincoli di integrità referenziale e/o vincoli di dominio, laddove presenti



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)
);
```



5. le seguenti interrogazioni espresse in linguaggio SQL:
   a. elenco degli eventi già svolti, in ordine alfabetico di provincia
   b. elenco dei membri che non hanno mai inserito un commento
   c. per ogni evento il voto medio ottenuto in ordine di categoria e titolo
   d. i dati dell'utente che ha registrato il maggior numero di eventi

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. il progetto della pagina dell'interfaccia WEB che permetta ad un utente registrato di svolgere le operazioni specificate
7. la codifica in un linguaggio a scelta di un segmento significativo dell'applicazione Web che consente l'interazione con la base di dati.

6 – Progetto della pagina dell'interfaccia WEB

Si può ipotizzare la pagina di 'Inserimento evento' :

La visualizzazione è senza alcuna struttura grafica. L'utente di accesso al database di nom
'webcommunity' si suppone 'root' con password 'root'.

```
<html>
<head>
<title>Elenco eventi per provincia e categoria</title>
</head>
<body>
<? php

        $connection = mysqli_connect("localhost", "root", "root", "webcommunity");

if (mysqli_connect_errno($connection))
{
echo "Errore di connessione al DBMS My-SQL.";
die();
}

$query = "SELECT data, ora, titolo, luogo_svolgimento, localita_svolgimento FROM Eventi
        WHERE rif_id_categpria = '$_GET[cod_categoria]'
        AND rif_sigla_provincia = '$_GET[sigla_provincia]'
ORDER BY data, titolo;";
$result = mysqli_query($connection, $query);
if (!$result)
{
        echo "Errore esecuzione query SQL.";
        die();
}

if (mysqli_num_rows($result) == 0)
{
        echo "Non è stato trovato alcun evento con il filtro applicato.";
        die();
}
?>
```

```
<center>
<b>Eventi per provincia</b>
<br><br>
<table>
<tr>
<th>Data</th>
<th>Ora</th>
<th>Titolo</th>
<th>Luogo</th>
<th>Località</th>
</tr>
</thead>
<tbody>
<?
while ($row = mysqli_fetch_assoc($result))
{
?>
<tr>
<td><? echo ($row['data']); ?></td>
<td><? echo ($row['ora']); ?></td>
<td><? echo ($row['titolo']); ?></td>
<td><? echo ($row['luogo_svolgimento']); ?></td>
<td><? echo ($row['localita_svolgimento']); ?></td>

</tr>
<?
}
mysqli_free_result($result);
echo "</tbody>\n";
echo "</table>\n";
echo "<br>\n";
mysqli_close($connection);
?>
</body>
</html>
```