

**Anno 2017**

**Indirizzo: Informatica e telecomunicazioni**

**Articolazione: Informatica**

**Tipologia: C**

**Testo**

*Il candidato (che potrà eventualmente avvalersi delle conoscenze e competenze maturate attraverso esperienze di alternanza scuola-lavoro, stage o formazione in azienda) svolga la prima parte della prova e due tra i quesiti proposti nella seconda parte.*

**PRIMA PARTE**

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.

Il candidato, fatte le opportune ipotesi aggiuntive, sviluppi:

1. un'analisi della realtà di riferimento, giungendo alla definizione di uno schema concettuale della base di dati che, a suo motivato giudizio, sia idoneo a gestire la realtà presentata;
2. il relativo schema logico;
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.

### **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.
- II. In relazione al tema proposto nella prima parte, il candidato immagini di volere documentare al committente l'operatività della piattaforma proposta. A tal fine, imposti una relazione tecnica che presenti le principali caratteristiche dell'applicazione Web in termini di organizzazione e funzionalità. In particolare, imposti la struttura di tale relazione, motivando le scelte e scrivendo un esempio significativo dei relativi contenuti.
- III. Dato il seguente schema relazionale:  
film (id, titolo, durata, anno di produzione, genere);  
attore (id, nome, cognome, data\_nascita, fotografia);  
recita (id\_film, id\_attore, ruolo);  
il candidato:
  - determini la modalità di gestione del campo 'fotografia' che prevede la memorizzazione di una immagine dell'attore in un formato grafico (es. JPG);
  - formalizzi in linguaggio SQL lo schema fisico corrispondente allo schema relazionale, sapendo che:
    - a. il campo 'genere' ammette solo i seguenti valori: fantasy, giallo, commedia, horror, drammatico, fantascienza, azione;
    - b. per la relazione 'recita', i campi 'id\_film' e 'id\_attore' referenziano rispettivamente la chiave primaria delle relazioni 'film' e 'attore';
  - discuta l'uso degli indici nel modello fisico di una base di dati e suggerisca con motivato giudizio indici appropriati per questo schema relazionale, definendoli in linguaggio SQL.
- IV. Un'azienda desidera sviluppare un'applicazione Web per la prenotazione on-line di eventi culturali, fruibile sia da computer desktop che da dispositivi mobili come tablet e smartphone. Il candidato esponga i punti critici da affrontare relativamente alle differenti proprietà di visualizzazione delle varie tipologie di dispositivi e alla rispettiva fruizione dei contenuti. Illustri possibili misure risolutive, con esempi relativi all'applicazione in questione.

## PRIMA PARTE

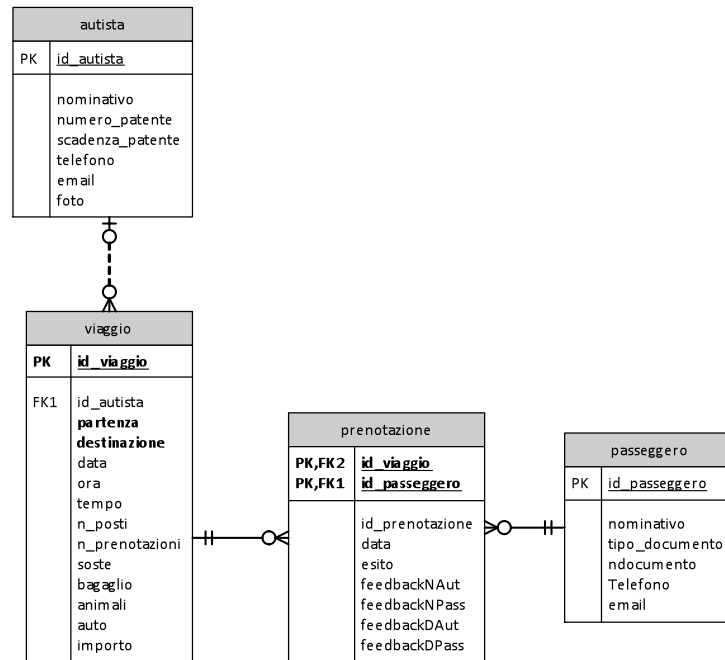
### 1. Analisi scenario e ipotesi aggiuntive

La soluzione esposta di seguito è essenziale, ma completa sotto tutti i punti di vista. Partendo dallo scenario descritto nel testo sono state fatte le assunzioni che seguono:

- ⊙ si ipotizza che un autista possa anche usare un'auto diversa in funzione del viaggio, pertanto l'auto viene vista come attributo del viaggio;
- ⊙ non si è ritenuto essenziale, per la soluzione del problema, prevedere una tabella specifica per la memorizzazione dei nomi delle città: la partenza e la destinazione saranno pertanto memorizzate come attributi del viaggio; sarà cura dei proponenti dei viaggi digitare in maniera corretta tali nomi;
- ⊙ ogni prenotazione verrà memorizzata anche se per una qualsiasi ragione non andrà a buon fine;
- ⊙ per le singole prenotazioni verrà memorizzata anche la data/ora in cui sono state effettuate;
- ⊙ le soste previste per un viaggio saranno memorizzate in un campo descrittivo che riporta un sommario dell'itinerario previsto; la stessa soluzione sarà adottata anche per quanto riguarda le prescrizioni relative al bagaglio e agli animali eventualmente da trasportare;
- ⊙ i feedback, sia per gli autisti che per i passeggeri, sono realizzati con due diversi valori: il primo numerico per esprimere un indice di gradimento con valori da 1 a 5 ed il secondo descrittivo;
- ⊙ la valutazione media sia per i passeggeri che per gli autisti non è un valore che viene memorizzato ma calcolato ogni volta che si rende necessario mediante una query SQL;
- ⊙ per quanto riguarda le foto degli autisti si ipotizza una soluzione in cui le varie immagini sono memorizzate in una specifica directory, mentre nella tabella relativa agli autisti viene registrato il *pathname* dei file relativi (ad esempio "carpooling\img\foto00075.jpg", dove il numero presente nel nome dell'immagine corrisponde al codice numerico identificativo di un autista);
- ⊙ per le prenotazioni, per questioni di integrità referenziale, si è scelto di usare come chiave primaria la coppia id\_viaggio/id\_passeggero pertanto il campo id\_prenotazione che dovrà contenere valori univoci sarà gestito a livello procedurale come combinazione di caratteri concatenando id\_viaggio

e `id_passeggero` 8+8 caratteri con *padding* di 0 non significativi (es.: “0000125400000525”) identificherà la prenotazione del passeggero 525 sul viaggio 1254).

## 2.1 Diagramma database



## 2.2 Tabelle database

### Autista

È la tabella in cui sono censiti i vari autisti che offrono viaggi:

Campo	R	Tipo	Dim.	Descrizione
id_autista	PK	Numerico		Identificativo autista
nominativo		Carattere	50	Cognome e nome
numero_patente		Carattere	20	Numero patente
scadenza_patente		Data/ora		Data scadenza patente
telefono		Carattere	15	Numero telefono
email		Carattere	30	Indirizzo e-mail
foto		Carattere	255	Pathname cartella foto autisti

**Chiave Primaria:** id\_autista

### Passeggero

È la tabella in cui sono censiti i vari passeggeri che fruiscono di viaggi:

Campo	R	Tipo	Dim.	Descrizione
id_passeggero	PK	Numerico		Codice passeggero
nominativo		Carattere	30	Cognome e nome
tipo_documento		Carattere	4	Tipo documento
ndocumento		Carattere	20	Numero documento
telefono		Carattere	20	Telefono passeggero
email		Carattere	30	Indirizzo e-mail

**Chiave Primaria:** id\_passeggero

### Prenotazione

È la tabella in cui sono registrate le prenotazioni di viaggi effettuate nel tempo:

Campo	R	Tipo	Dim.	Descrizione
id_viaggio	PK	Numerico		Identificativo viaggio
id_passeggero	PK	Numerico		Identificativo passeggero
id_prenotazione		Carattere	16	Identificativo univoco prenotazione (costruito come combinazione di caratteri concatenando id_viaggio e id_passeggero 8+8 caratteri con <i>padding</i> di 0 non significativi)
data_prenotazione		Data/ora		Data in cui è stata effettuata la prenotazione
esito		Carattere	1	Esito prenotazione (P=Positivo, N=Negativo)
feedbackNAut		Numerico		Feedback numerico autista (valutazione passeggero)
feedbackNPass		Numerico		Feedback numerico passeggero (valutazione autista)
feedbackDAut		Carattere	255	Feedback commento autista (valutazione passeggero)
feedbackDPass		Carattere	255	Feedback commento passeggero (valutazione autista)

**Chiave Primaria:** id\_viaggio, id\_passeggero

**Chiave Esterna:** id\_passeggero → Passeggero.id\_passeggero

**Chiave Esterna:** id\_viaggio → Viaggio.id\_viaggio

## Viaggio

È la tabella in cui sono registrati i dati dei viaggi offerti nel tempo dagli autisti registrati nel sistema:

Campo	R	Tipo	Dim.	Descrizione
id_viaggio	PK	Numerico		Identificativo viaggio
id_autista	FK	Numerico		Identificativo autista
partenza		Carattere	25	Città di partenza
destinazione		Carattere	25	Città di destinazione
data		Data/ora		Data partenza
ora		Data/ora		Ora partenza
tempo		Numerico		Tempo percorrenza stimato in ore (con decimali)
n_posti		Numerico		Numero posti complessivo
n_prenotazioni		Numerico		Numero prenotazioni già effettuate
soste		Carattere	255	Descrizione soste previste
bagaglio		Carattere	40	Prescrizioni bagaglio consentito
animali		Carattere	40	Prescrizioni trasporto animali
auto		Carattere	30	Tipo di auto previsto per il viaggio
importo		Numerico		Importo in euro richiesto per il viaggio

**Chiave Primaria:** id\_viaggio

**Chiave Esterna:** id\_autista → Autista.id\_autista

## 2.3 DB-schema

```
CREATE TABLE Autista(
    id_autista INT(8) NOT NULL,
    nominativo VARCHAR(50) NOT NULL,
    numero_patente VARCHAR(20) NOT NULL,
    scadenza_patente DATETIME NOT NULL,
    telefono VARCHAR(15),
    email VARCHAR(30),
    foto VARCHAR(100),
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id_autista)
);

CREATE TABLE Passeggero(
    id_passeggero INT(8) NOT NULL,
    nominativo VARCHAR(50) NOT NULL,
    tipo_documento VARCHAR(4) NOT NULL,
    ndocumento VARCHAR(20) NOT NULL,
    telefono VARCHAR(20),
    email VARCHAR(30) NOT NULL,
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id_passeggero)
);

CREATE TABLE Viaggio(
    id_viaggio INT(8) NOT NULL,
    id_autista INT(8) NOT NULL,
    partenza VARCHAR(25) NOT NULL,
    destinazione VARCHAR(25) NOT NULL,
    data DATE NOT NULL,
    ora TIME NOT NULL,
    tempo DOUBLE,
```

```

        n_posti INT(2) NOT NULL,
        n_prenotazioni INT(2) NOT NULL,
        soste VARCHAR(255),
        bagaglio VARCHAR(50),
        animali VARCHAR(50),
        auto VARCHAR(30),
        importo DOUBLE,
        CONSTRAINT ChiavePrimaria PRIMARY KEY(id_viaggio),
        CONSTRAINT AutistaViaggio FOREIGN KEY(id_autista)
        REFERENCES Autista(id_autista)
    );
CREATE TABLE Prenotazione(
    id_viaggio INT(8) NOT NULL,
    id_passeggero INT(8) NOT NULL,
    id_prenotazione VARCHAR(16) NOT NULL,
    data DATETIME NOT NULL,
    esito VARCHAR(1),
    feedbackNAut INT(1),
    feedbackNPass INT(1),
    feedbackDAut VARCHAR(255),
    feedbackDPass VARCHAR(255),
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id_viaggio,
                                           id_passeggero),
    CONSTRAINT PasseggeroPrenotazioni
    FOREIGN KEY(id_passeggero)
    REFERENCES Passeggero(id_passeggero),
    CONSTRAINT ViaggioPrenotazioni
    FOREIGN KEY(id_viaggio)
    REFERENCES Viaggio(id_viaggio)
);

```

### 3 Query

Nel codice delle query sono state utilizzate le parentesi quadre per indicare i valori forniti dall'utente per la parametrizzazione delle stesse.

a)

```

SELECT Autista.nominativo, Viaggio.partenza, Viaggio.destinazione,
       Viaggio.data, Viaggio.ora, Viaggio.importo
FROM Autista INNER JOIN Viaggio ON Autista.id_autista = Viaggio.id_autista
WHERE Viaggio.partenza = [città_partenza]
      AND Viaggio.destinazione = [città_destinazione]
      AND Viaggio.data = [data_partenza]
ORDER BY ora;

```

b)

```

SELECT Viaggio.partenza, Viaggio.destinazione, Viaggio.data, Viaggio.ora,
       Autista.nominativo, Viaggio.auto
FROM (Autista INNER JOIN Viaggio ON Autista.id_autista = Viaggio.id_autista)
     INNER JOIN Prenotazione ON Viaggio.id_viaggio = Prenotazione.id_viaggio
WHERE Prenotazione.id_prenotazione = [prenotazione]
      AND Prenotazione.esito = 'P';

```

c)

```

SELECT *
FROM (SELECT Passeggero.id_passeggero, passeggero.Nominativo,
            AVG(feedbackNAut) AS voto_medio
      FROM Passeggero INNER JOIN Prenotazione

```

```

        ON Passeggero.id_passeggero = Prenotazione.id_passeggero
WHERE Prenotazione.id_viaggio = [viaggio]
GROUP BY Passeggero.id_passeggero, Passeggero.nominativo) AS T
WHERE voto_medio > [voto];

```

#### 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 effettuato 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:

```

<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();
    }

    $query = "SELECT autista.nominativo, autista.email,
                  viaggio.data, viaggio.ora, viaggio.tempo,viaggio.auto,
                  (viaggio.n_posti-viaggio.n_prenotazioni) AS posti_disponibili
    FROM autista INNER JOIN viaggio
    ON autista.id_autista = viaggio.id_autista
    WHERE viaggio.partenza='".$partenza.'"
      AND viaggio.destinazione='".$destinazione.'"
      AND viaggio.data BETWEEN '".$data1.'" AND '".$data2.'"
      AND viaggio.n_posti-viaggio.n_prenotazioni>0
    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>
    </html>
```

## SECONDA PARTE

### I)

Il database progettato nella prima parte comprende già tutti i dati necessari per gestire quanto richiesto dal quesito. Il seguente script PHP modifica quello presentato nella prima parte per visualizzare al passeggero l'informazione aggiuntiva richiesta e cioè il numero di prenotazioni non ancora accettate/rifiutate dall'autista per un determinato viaggio:

```
<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 ($row['posti_disponibili']); ?></td>
            <td><?php echo ($row['prenotazioni_non_ancora_confermate']); ?></td>
          </tr>
        <?php
          }
      mysqli_free_result($result);
      echo "</tbody>\n";
      echo "</table>\n";
      mysqli_close($connection);
    ?>
  </body>
</html>

```

## II)

La relazione tecnica per il committente è strutturata a partire dal seguente indice che copre tutti gli aspetti organizzativi e funzionali dell'applicazione web:

0. Servizi realizzati
  - sito web
  - web-service
1. Piattaforma di erogazione
  - virtualizzazione hardware
  - tecnologie software
    - DBMS
    - server web
    - server applicativo
    - client e APP
2. Struttura del sito web
  - registrazione e autenticazione degli utenti

- offerta di un viaggio
  - visualizzazione delle prenotazioni, valutazione di un passeggero e accettazione di una prenotazione
  - ricerca di un viaggio, valutazione di un autista, inoltro di una prenotazione
  - verifica dello stato di una prenotazione
  - inserimento valutazioni di feedback
3. Web API REST
- registrazione e autenticazione degli utenti
  - offerta di un viaggio
  - acquisizione delle valutazioni di un passeggero
  - accettazione di una prenotazione
  - ricerca di un viaggio
  - acquisizione delle valutazioni di un autista
  - inoltro di una prenotazione
  - verifica dello stato di una prenotazione
  - inserimento valutazioni di feedback
4. Sicurezza e privacy
- continuità di erogazione del servizio
  - integrità e ridondanza dei dati gestiti dal servizio
  - autenticazione e monitoraggio degli accessi da parte degli utenti del servizio
  - riservatezza dei dati gestiti in relazione ai profili degli utenti del servizio

Data la natura tecnica e lo scopo della relazione alcuni aspetti possono essere documentati con un diagramma UML (dei casi d'uso, delle attività, degli stati o di sequenza).

Viene di seguito sviluppato il punto 1 dell'indice:

## 1. Piattaforma di erogazione

### Virtualizzazione hardware

Il servizio erogato deve risultare scalabile in funzione del numero complessivo degli utenti e del numero di utenti connessi contemporaneamente. A questo scopo esso è erogato mediante un servizio cloud di virtualizzazione hardware le cui caratteristiche prestazionali si adattano – anche dinamicamente – alle richieste ricevute.

### Tecnologie software

Il **DBMS** utilizzato è Oracle My-SQL che, oltre a garantire le caratteristiche funzionali, prestazionali, di gestione e di sicurezza richieste per l'erogazione del servizio, viene normalmente reso disponibile come preinstallato e configurato nei servizi cloud di virtualizzazione dei server. Oracle My-SQL fornisce un'implementazione a elevate prestazioni del linguaggio SQL adatta per la realizzazione funzionale dell'applicazione web.

Il **server web** utilizzato è Apache HTTP server: trattandosi del server web maggiormente diffuso ed utilizzato al per le sue caratteristiche di affidabilità e sicurezza anche in questo caso è comune che sia reso disponibile come preinstallato e configurato nei servizi cloud di virtualizzazione dei server. Apache HTTP server integra il linguaggio PHP impiegato per lo sviluppo lato server delle pagine web dinamiche che costituiscono il sito dell'applicazione.

Il **server applicativo** è Apache Tomcat che consente di esporre una web API implementata in linguaggio Java che realizza un web-service. Sia il server web che il server applicativo accedono tramite la rete virtuale del servizio cloud al DBMS: il sito web mediante la specifica interfaccia del linguaggio PHP, il web-service mediante il connettore JDBC di My-SQL.

Nel caso del sito web il **client** dell'applicazione web è realizzato dal codice JavaScript contenuto nelle pagine web dinamiche generate dal server, mentre l'esposizione della web API REST ha lo scopo di supportare lo sviluppo di **APP** per i dispositivi mobili.

### III)

Relativamente al campo fotografia due possibili tecniche di gestione sono:

1. utilizzo di un campo *blob* (o altro equivalente a seconda del DBMS) in cui inserire direttamente l'immagine binaria della fotografia;
2. utilizzo di una soluzione in cui le varie immagini sono memorizzate in una specifica directory del server su cui è eseguito il DBMS che ospita il database, mentre nella tabella relativa agli attori viene registrato, nel campo fotografia, solo il *pathname* dei file relativi a tali immagini (ad esempio "cineteca/img/foto00075.jpg" e dove il numero presente nel nome dell'immagine corrisponde al codice identificativo dell'attore, in questo caso 75).

Nel caso in cui i dati vengano acceduti tramite un sito web è conveniente usare formati standard come JPEG o PMG. Ipotizzando di usare la tecnica di cui al precedente punto 2 sarà possibile visualizzare su una pagina web una foto facendo ricorso al tag HTML `<img>` nel contesto di una ipotetica istruzione PHP del tipo `echo "<img src=\"\$foto\">";` dove la variabile `$foto` conterrà il *pathname* di un file relativo alla fotografia di un attore ottenuto dal campo *fotografia* con una banale query SQL sulla tabella *Attore*.

### DB-Schema

```
CREATE TABLE Attore(
    id INT(8) NOT NULL,
    nome VARCHAR(50) NOT NULL,
    cognome VARCHAR(50) NOT NULL,
    data_nascita DATE,
    fotografia VARCHAR(50),
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id)
);

CREATE TABLE Film(
    id INT(8) NOT NULL,
    titolo VARCHAR(200) NOT NULL,
    anno_di_produzione INT(4),
    genere ENUM('Fantasy', 'Giallo', 'Commedia', 'Horror',
                'Drammatico', 'Fantasy', 'Azione'),
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id);
);

CREATE TABLE Recita(
    id_film INT(8) NOT NULL,
    id_attore INT(8) NOT NULL,
    ruolo VARCHAR(50),
    CONSTRAINT ChiavePrimaria PRIMARY KEY(id_film, id_attore),
    CONSTRAINT RecitaFilm FOREIGN KEY(id_film)
    REFERENCES Film(id),
    CONSTRAINT RecitaAttore FOREIGN KEY(id_attore)
    REFERENCES Attore(id)
);
```

### Indici

L'indicizzazione di alcuni campi in una base di dati serve a velocizzare le ricerche che li coinvolgono. In genere i DBMS indicizzano chiavi primarie e chiavi esterne con lo scopo di controllare in modo efficiente l'eventuale tentativo di duplicare una chiave primaria, o il tentativo di inserire record orfani laddove sia stata definita una regola di integrità referenziale su una chiave esterna. L'ottimizzazione dei tempi di ricerca è infine fondamentale quando si effettuano operazioni di *join* tra tabelle.

Oltre ai casi discussi è possibile definire indici su campi non in chiave, ma che vengono spesso utilizzati nella clausola WHERE delle query. Nel modello fisico di uno schema relazionale questi ultimi possono essere definiti in un DB-Schema tramite la clausola:

```
INDEX nome_indice (nome_campo_indicizzato)
```

inserita direttamente all'interno di un comando CREATE TABLE.

In alternativa è possibile definire un indice con l'istruzione:

```
CREATE INDEX nome_indice ON nome_tabella (nome_campo_indicizzato);
```

Ad esempio, volendo indicizzare i titoli dei film, avremmo:

```
CREATE INDEX titoli ON Film ('titolo');
```

#### IV)

La risposta viene fornita in generale non rappresentando l'applicazione web di prenotazione on-line degli eventi culturali un caso particolare rispetto alle criticità implementative che sono essenzialmente le seguenti:

- 1) adattamento automatico alle caratteristiche dimensionali e di risoluzione dello schermo di visualizzazione;
- 2) necessità di sviluppare e mantenere il codice di una singola applicazione per piattaforme di esecuzione diverse (smartphone/tablet Android, IOS, ...; PC Windows, MacOS, Linux, ...);
- 3) accettazione di pagamenti da parte degli utenti;
- 4) registrazione e autenticazione degli utenti.

Il punto (1) – noto come *responsive web design* – si basa sulla possibilità offerta dai fogli di stile CSS e dal linguaggio JavaScript di effettuare *media queries* per conoscere le caratteristiche dello schermo di visualizzazione; le politiche usate per l'adattamento dei contenuti della pagina web alle caratteristiche dello schermo sono diverse: non visualizzazione di parte dei contenuti, redistribuzione spaziale (verticale/orizzontale) dei contenuti e cambiamento dell'ordinamento degli stessi.

Il punto (2) ha portato da tempo alla disponibilità di tecnologie software per lo sviluppo di APP multipiattaforma realizzate con tecnologie web (HTML/CSS/JavaScript) mediante il ricorso a framework specifici, molti dei quali integrano il framework Apache Cordova che incorpora una *webview* – cioè un browser *embedded* – per ogni specifica piattaforma di esecuzione.

Il punto (3) trova una soluzione universalmente accettata nel ricorso alla web API di *PayPal* – previa registrazione – che consente di validare varie forme di pagamento effettuate dall'utente.

Il punto (4) rappresenta una problematica tipica di una qualsiasi applicazione web che ha alcune soluzioni standard:

- registrazione – e successiva memorizzazione di un *fingerprint* calcolato mediante una funzione *hash* sicura – di una specifica password per ogni singolo utente utilizzata per la successiva autenticazione al momento del login;
- uso dell'autenticazione effettuata ad esempio per l'account *Google* per autenticare l'accesso all'applicazione web utilizzando il protocollo standard OpenID.