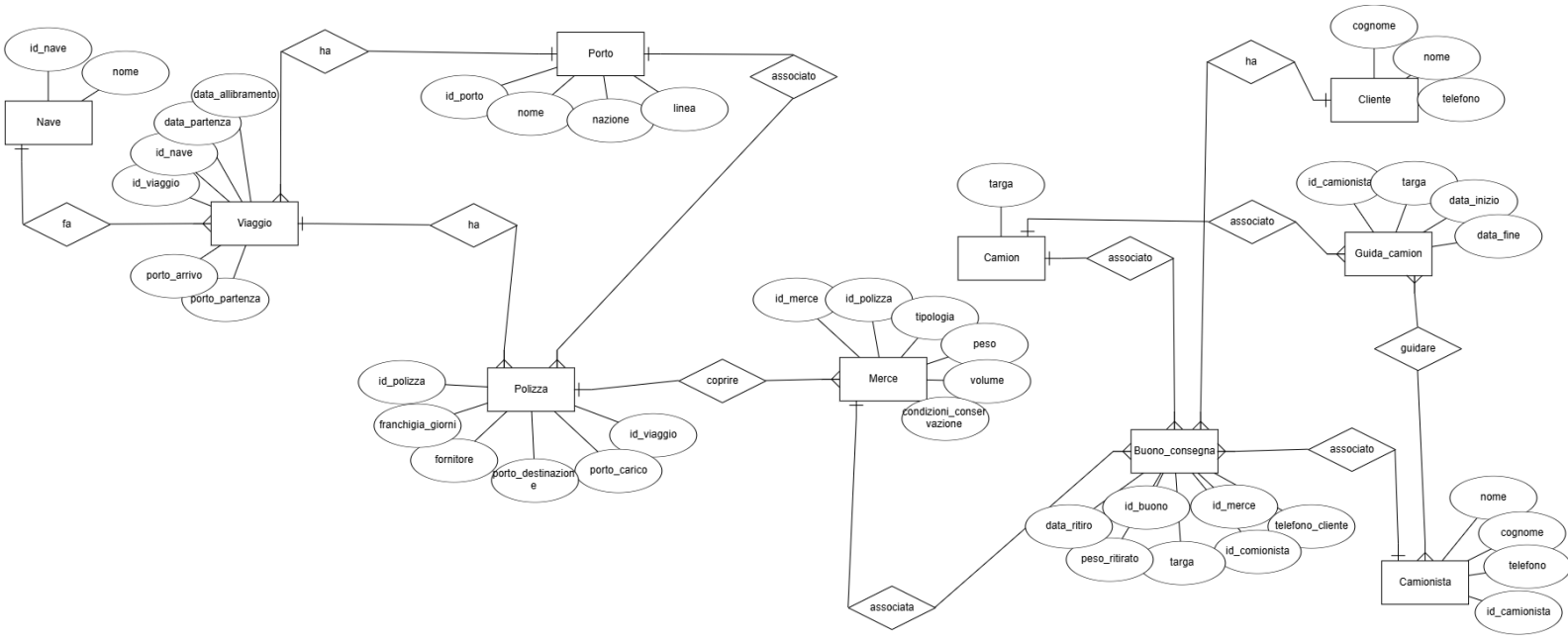
Simulazione della seconda prova d'esame

Un terminal marittimo deve realizzare un sistema informatico per la gestione del traffico delle merci che arrivano ad esso via nave e che da qui vengono trasportate a destinazione via camion. Il terminal gestisce varie tipologie di merci. Ogni nave, di cui si conosce il nome, effettua nel corso del tempo dei viaggi ad ognuno dei quali è caratterizzato da un porto e una data di partenza, un porto di arrivo e la data di allibramento corrispondente all'arrivo della merce in porto con conseguente scarico e immagazzinaggio delle merci nel terminal. Il porto di partenza determina la linea relativa alla direttrice del viaggio (Sudest Asiatico, Nord America, Sud America, Brasile, ecc) gestita principalmente a scopo statistico. Di ogni porto interessa osservare anche la nazionalità. Per ogni viaggio di una nave, la merce trasportata afferisce ad un insieme di documenti dette polizze di carico. Per ogni polizza di carico si conoscono il porto di carico e quello di destinazione, la tipologia della merce trasportata, il peso (in kg) della medesima e il suo fornitore. Il personale del terminal emette, su richiesta dei clienti che acquistano la merce, dei buoni di consegna necessari per il ritiro della medesima. Tali buoni vengono presentati al terminal da degli autotrasportatori incaricati dal cliente per il ritiro fisico della merce. Un buono di consegna fa riferimento, per un certo peso, alla merce di una specifica polizza di carico e la merce di una polizza può essere suddivisa in parti non necessariamente uguali per clienti diversi che la possono ritirare in più volte. Il fornitore offre per ogni polizza un certo numero di giorni di magazzinaggio in franchigia, nell'arco dei quali il esso è gratuito, dopo tale periodo i clienti debbono pagare un tariffa giornaliera prefissata per tonnellata di merce ancora immagazzinata. Si vuole mantenere un registro cronologico dei ritiri merce che tenga conto dei clienti, del peso ritirato, dei camion utilizzati (targa) e del nominativi dei conducenti. Un camion può essere guidato nel tempo da diversi autisti e viceversa.

Schema E/R



Relazioni

NAVE — VIAGGIO:

- Cardinalità 1:N: Ogni nave può fare più viaggi, ma ogni viaggio è legato a una sola nave.
- Totalità su Viaggio, parziale su Nave: Ogni viaggio deve essere associato a una nave (totalità su Viaggio), mentre una nave non è obbligata a fare un viaggio (parziale su Nave).

PORTO — VIAGGIO:

- Cardinalità 1:N: Un porto può essere il punto di partenza o arrivo di molti viaggi, ma ogni viaggio ha un solo porto di partenza e uno di arrivo.
- Totalità su Viaggio, parziale su Porto: Ogni viaggio deve avere un porto di partenza e uno di arrivo (totalità su Viaggio), mentre un porto non è obbligato a essere coinvolto in ogni viaggio (parziale su Porto).

VIAGGIO — POLIZZA:

- Cardinalità 1:N: Un viaggio può essere associato a molte polizze, ma ogni polizza è legata a un solo viaggio.
- Totalità su Polizza: Ogni polizza deve essere associata a un viaggio (totalità su Polizza), ma un viaggio non è obbligato a essere associato a una polizza (parziale su Viaggio).

PORTO — POLIZZA:

- Cardinalità 1:N: Un porto può essere associato a molte polizze, ma ogni polizza ha un solo porto di carico e uno di destinazione.
- Totalità su Polizza: Ogni polizza deve avere un porto di carico e uno di destinazione (totalità su Polizza), ma un porto non è obbligato a essere il porto di carico o di destinazione in ogni polizza (parziale su Porto).

POLIZZA — MERCE:

- Cardinalità 1:N: Una polizza può coprire molte merci, ma ogni merce è associata a una sola polizza.
- Totalità su Merce: Ogni merce deve essere associata a una polizza (totalità su Merce), ma una polizza non è obbligata a coprire tutte le merci (parziale su Polizza).

MERCE — BUONO_CONSEGNA:

- Cardinalità 1:N: Una merce può essere associata a molti buoni di consegna, ma ogni buono di consegna è relativo a una sola merce.
- Totalità su Buono_consegna: Ogni buono di consegna deve essere associato a una merce (totalità su Buono_consegna), ma una merce non è obbligata a essere associata a un buono di consegna (parziale su Merce).

CLIENTE — BUONO_CONSEGNA:

- Cardinalità 1:N: Un cliente può avere molti buoni di consegna, ma ogni buono di consegna è associato a un solo cliente (identificato dal numero di telefono come PK).

- Totalità su Buono_consegna, parziale su Cliente: Ogni buono di consegna deve essere associato a un cliente (totalità su Buono_consegna), ma non tutti i clienti devono avere un buono di consegna (parziale su Cliente).

CAMIONISTA — BUONO_CONSEGNA:

- Cardinalità 1:N: Un camionista può essere associato a molti buoni di consegna, ma ogni buono di consegna è associato a un solo camionista.
- Totalità su Buono_consegna: Ogni buono di consegna deve essere associato a un camionista (totalità su Buono_consegna), ma un camionista non è obbligato a essere associato a un buono di consegna (parziale su Camionista).

CAMION — BUONO_CONSEGNA:

- Cardinalità 1:N: Un camion può essere associato a molti buoni di consegna, ma ogni buono di consegna è relativo a un solo camion.
- Totalità su Buono_consegna: Ogni buono di consegna deve essere associato a un camion (totalità su Buono_consegna), ma un camion non è obbligato a essere associato a un buono di consegna (parziale su Camion).

CAMIONISTA — GUIDA_CAMION — CAMION:

- Cardinalità M:N con attributi (data_inizio, data_fine): Un camionista può guidare molti camion e un camion può essere guidato da più camionisti nel tempo. La relazione è di tipo molti a molti, con attributi che specificano il periodo di tempo (data di inizio e fine) in cui un camionista guida un determinato camion.
- Totalità su GUIDA_CAMION: Ogni camionista deve essere associato almeno a un record di guida (totalità su Camionista) e ogni camion deve essere associato almeno a un record di guida (totalità su Camion).

CAMION — GUIDA_CAMION:

- Cardinalità 1:N: Ogni camion può essere associato a molti record di guida (se più camionisti hanno guidato lo stesso camion nel tempo), mentre ogni record di guida associato a un camion è legato a un solo camionista (ogni camionista guida il camion in un periodo specifico).
- Totalità su Guida_camion per Camion: Ogni camion deve essere associato almeno a uno o più record di guida (totalità su Camion).
- Totalità su Guida_camion per Camionista: Ogni record di guida deve essere associato a un camionista (totalità su Guida_camion).

Le seguenti interrogazioni espresse in algebra relazionale e/o in linguaggio SQL:

a) produrre l'elenco dei viaggi con data di allibramento compresa tra i mesi di marzo e aprile del 2015 indicando nave, viaggio, data parenza, data allibramento e porto di partenza in ordine di arrivo;

```
SELECT V.id_viaggio, N.nome AS nave, V.data_partenza, V.data_allibramento, P.nome AS
porto_partenza
FROM VIAGGIO V
JOIN NAVE N ON V.id_nave = N.id_nave
JOIN PORTO P ON V.porto_partenza = P.id_porto
WHERE V.data_allibramento BETWEEN '2015-03-01' AND '2015-04-30'
ORDER BY V.data_allibramento
```

b) produrre il traffico in termini di peso totale per ogni linea e tipologia di merce nel corso del 2014;

```
SELECT PR.linea, M.tipologia, SUM(M.peso) AS peso_totale
FROM MERCE M
JOIN POLIZZA P ON M.id_polizza = P.id_polizza
JOIN VIAGGIO V ON P.id_viaggio = V.id_viaggio
JOIN PORTO PR ON V.porto_partenza = PR.id_porto
WHERE YEAR(V.data_allibramento) = 2014
GROUP BY PR.linea, M.tipologia
```

c) visualizzare il nome delle navi che non hanno mai trasportato un certo tipo di merce;

```
SELECT N.nome
FROM NAVE N
WHERE N.id_nave NOT IN (
  SELECT DISTINCT V.id_nave
  FROM VIAGGIO V
  JOIN POLIZZA P ON V.id_viaggio = P.id_viaggio
  JOIN MERCE M ON P.id_polizza = M.id_polizza
  WHERE M.tipologia = 'Frutta'
);
```

d) il nome del camionista che nel tempo ha ritirato la minore quantità (peso) di merce.

```
SELECT C.nome
FROM CAMIONISTA C
JOIN BUONO_CONSEGNA B ON C.id_camionista = B.id_camionista
GROUP BY C.id_camionista, C.nome
ORDER BY SUM(B.peso_ritirato)
```

DB Schema

```
CREATE TABLE Nave (
   id_nave INT PRIMARY KEY,
   nome VARCHAR(100) NOT NULL
);

CREATE TABLE Porto (
   id_porto INT PRIMARY KEY,
   nome VARCHAR(100) NOT NULL,
   nazione VARCHAR(100) NOT NULL,
   linea VARCHAR(100)
);

CREATE TABLE Viaggio (
   id_viaggio INT PRIMARY KEY,
   id_nave INT NOT NULL,
   data_partenza DATE NOT NULL,
   data_arribamento DATE,
   porto_partenza INT NOT NULL,
   porto_arrivo INT NOT NULL,
   FOREIGN KEY (id_nave) REFERENCES Nave(id_nave),
   FOREIGN KEY (porto_partenza) REFERENCES Porto(id_porto),
   FOREIGN KEY (porto_arrivo) REFERENCES Porto(id_porto)
);

CREATE TABLE Polizza (
   id_polizza INT PRIMARY KEY,
   franchigia_giorni INT,
   fornitore VARCHAR(100),
   destinazione VARCHAR(100),
   porto_carico INT,
   id_viaggio INT,
```

```
    FOREIGN KEY (porto_carico) REFERENCES Porto(id_porto),
    FOREIGN KEY (id_viaggio) REFERENCES Viaggio(id_viaggio)
);

CREATE TABLE Merce (
    id_merce INT PRIMARY KEY,
    tipologia VARCHAR(100),
    peso DECIMAL(5,2),
    volume DECIMAL(5,2),
    condizioni_conservazione VARCHAR(255),
    id_polizza INT,
    FOREIGN KEY (id_polizza) REFERENCES Polizza(id_polizza)
);

CREATE TABLE Camion (
    targa VARCHAR(20) PRIMARY KEY
);

CREATE TABLE Cliente (
    id_cliente INT PRIMARY KEY AUTO_INCREMENT,
    nome VARCHAR(100),
    cognome VARCHAR(100),
    telefono VARCHAR(20)
);

CREATE TABLE Camionista (
    id_camionista INT PRIMARY KEY,
    nome VARCHAR(100),
    cognome VARCHAR(100),
    telefono VARCHAR(20)
);
```

```
CREATE TABLE Guida_camion (
    id_camionista INT,
    targa VARCHAR(20),
    data_inizio DATE,
    data_fine DATE,
    PRIMARY KEY (id_camionista, targa, data_inizio),
    FOREIGN KEY (id_camionista) REFERENCES Camionista(id_camionista),
    FOREIGN KEY (targa) REFERENCES Camion(targa)
);

CREATE TABLE Buono_consegna (
    id_buono INT PRIMARY KEY,
    id_merce INT NOT NULL,
    id_camionista INT NOT NULL,
    targa VARCHAR(20) NOT NULL,
    telefono_cliente VARCHAR(20),
    data_ritiro DATE,
    peso_ritirato DECIMAL(10,2),
    FOREIGN KEY (id_merce) REFERENCES Merce(id_merce),
    FOREIGN KEY (id_camionista) REFERENCES Camionista(id_camionista),
    FOREIGN KEY (targa) REFERENCES Camion(targa)
);
```

Argomentazione

Considerando le caratteristiche del terminal e il fatto che probabilmente non dispone di personale tecnico specializzato per la gestione di un server, la soluzione migliore per realizzare un sistema web-based per la gestione delle operazioni di ingresso e uscita merci e per la fatturazione dei costi di magazzinaggio è quella di affidarsi a un servizio di hosting esterno. Questa scelta è più economica, garantisce che il sistema sia sempre attivo 24 ore su 24 e 7 giorni su 7 e permette di non dover gestire direttamente tutta l'infrastruttura tecnica necessaria. Il servizio di hosting dovrà prevedere lo spazio web per il sito, il supporto a PHP e la presenza di un database per la gestione dei dati relativi a navi, viaggi, polizze di carico, merci, clienti, camionisti e camion. Per quanto riguarda la gestione degli accessi, sarà fondamentale implementare una "profilazione" degli utenti, cioè assegnare ad ognuno un username e una password e ognuno avrà il suo "ruolo" con funzionalità a lui dedicate. Per esempio, il personale potrà gestire tutte le operazioni di carico e scarico merci, i clienti potranno controllare lo stato delle proprie spedizioni e scaricare le fatture, i fornitori potranno vedere i movimenti delle merci che hanno inviato e gli autotrasportatori avranno accesso ai buoni di consegna e alla registrazione dei ritiri. La struttura del sito sarà quindi organizzata con menù dinamici che si adattano al tipo di utente che ha effettuato l'accesso.

Sviluppo

//classe database in file apparte db.php

```
<?php
class DB {
    private $host = "localhost";
    private $user = "root";
    private $password = "";
    private $database = "terminal";
    private $conn;

    public function __construct() {
        $this->conn = new mysqli($this->host, $this->user, $this->password, $this->database);
    }

    function getViaggi() {
        $viaggi = [];
        $query = "SELECT V.id_viaggio, N.nome AS nave, V.data_partenza
                FROM VIAGGIO V
                JOIN NAVE N ON V.id_nave = N.id_nave
                ORDER BY N.nome, V.data_partenza";

        $result = $this->conn->query($query);

        while ($row = $result->fetch_assoc()) {
            $viaggi[] = new Viaggio($row['id_viaggio'], $row['id_nave'], $row['data_partenza']);
        }

        return $viaggi;
    }

    function getFornitori() {
        $fornitori = [];
        $query = "SELECT fornitore FROM POLIZZA";
        $result = $this->conn->query($query);
```

```
        while ($row = $result->fetch_assoc()) {
            $fornitori[] = $row['fornitore'];
        }

        return $fornitori;
    }

    function getGiacenze($id_viaggio, $fornitore) {
        $giacenze = [];
        $query = "
            SELECT P.id_polizza, SUM(M.peso) AS peso_totale
            FROM POLIZZA P
            JOIN MERCE M ON P.id_polizza = M.id_polizza
            WHERE P.id_viaggio = $id_viaggio
            AND P.fornitore = '$fornitore'
            GROUP BY P.id_polizza
        ";

        $result = $this->conn->query($query);

        while ($row = $result->fetch_assoc()) {
            $giacenze[] = $row;
        }

        return $giacenze;
    }
}
?>
```

//index.php

```
<?php
    require_once("db.php");
    $viaggi = getViaggi();
```

```
    $fornitori = getFornitori();
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
<h2>seleziona nave-viaggio e fornitore</h2>
<form action="risultatoMerce.php" method="post">
    <label>seleziona nave - viaggio:</label><br>
    <select name="viaggio" id="viaggio">
    <?php
    foreach ($viaggi as $v) {
        echo "<option value='".$v['id_viaggio']."'>".$v['nave']." - ".$v['data_partenza']."</option>";
    }
    ?>
    </select>
    <br><br>

    <label>seleziona fornitore:</label><br>
    <select name="fornitore" id="fornitore">
    <?php
    foreach ($fornitori as $f) {
        echo "<option value='".$f."'>".$f."</option>";
    }
    ?>
    </select>

    <br><br>
    <input type="submit" value="cerca polizze">
</form>
```

```
</body>
</html>
```

//risultatoMerce.php

```
<?php
    require_once("db.php");

    if (!isset($_POST["viaggio"]) || !isset($_POST["fornitore"])) {
        header("Location: index.php");
    }

    $id_viaggio = $_POST["viaggio"];
    $fornitore = $_POST["fornitore"];

    $giacenze = getGiacenze($id_viaggio, $fornitore);

    if (empty($giacenze)) {
        echo "<h2>nessuna merce trovata per il fornitore selezionato.</h2>";
    } else {
        echo "<h2>giacenza merce per polizza</h2>";
        foreach ($giacenze as $g) {
            echo "id:".$g['id_polizza']." - peso:".$g['peso_totale']."<br>";
        }
    }
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
```

</body>

</html>

Terza forma normale della tabella fornita

Prima forma normale (1NF): Una relazione è in 1NF quando tutti i domini sono indivisibili e non composti e la chiave primaria è ben definita. Nel nostro caso, i domini non sono ulteriormente scomponibili e la chiave primaria potrebbe essere la coppia (ora_volo, tratta). Le dipendenze funzionali che possiamo osservare sono:

- tratta - partenza: Ogni tratta ha una partenza unica.
- tratta - arrivo: Ogni tratta ha un arrivo unico.
- (ora_volo, tratta) - aereo: Ogni combinazione di ora e tratta è associata a un aereo specifico.
- aereo - posti: Ogni aereo ha un numero fisso di posti.

Seconda forma normale (2NF): Una relazione è in 2NF quando non ci sono dipendenze funzionali parziali, ovvero tutti gli attributi non chiave dipendono dalla chiave primaria in modo completo. La relazione voli non è in 2NF perché partenza e arrivo dipendono solo dalla tratta, quindi dobbiamo separare la relazione in due tabelle:

- voli2(ora_volo, tratta, aereo, posti)
- tratte(tratta, partenza, arrivo)

Anche se la relazione è ora in 2NF, potrebbero esserci ancora delle ridondanze e anomalie. In questo caso, abbiamo una dipendenza transitiva:

- aereo - posti (un aereo ha un numero fisso di posti)
- (ora_volo, tratta) - aereo - posti

Terza forma normale (3NF): Una relazione è in 3NF quando è in 2NF e non ci sono dipendenze transitiva tra attributi non chiave e la chiave primaria. Quindi, per eliminare la dipendenza transitiva tra posti e la chiave primaria (tramite aereo), dobbiamo scomporre ulteriormente la relazione voli2:

- voli3(ora_volo, tratta, aereo)
- aerei(aereo, posti)
- tratte(tratta, partenza, arrivo)