Esercitazione Java e Database: Gestione delle ordinazioni in un ristorante

Obiettivo: Implementare un programma Java per gestire le ordinazioni ai tavoli di un ristorante. Le ordinazioni consistono nella selezione di un primo piatto, un secondo piatto, un dessert e una bevanda da parte del cliente.

esempio menu del giorno

Ecco un esempio di tabella in formato Markdown per mostrare i dati dei piatti divisi per categorie (primi, secondi, dessert):

Primi	**Secondi**	**Dessert**	**Bevande**
Ravioli al ragù	Stoccafisso alla Genovese	Torta di pinoli	Vino rosso
Pansotti in salsa di noci	Polpo in aglio e prezzemolo	Torta di mele	Vino bianco
Trenette al pesto	Cima	Tiramisù	Birra
Minestrone	Torta pasqualina	Torta di rose	Acqua

Questa tabella mostra i piatti divisi per categorie: primi, secondi e dessert. Puoi inserire i dati dei piatti nelle colonne appropriate. Se un piatto non appartiene a una certa categoria, lascia la cella vuota.

Requisiti:

1. Implementare una classe `Ordinazione` che rappresenti un singolo ordine, contenente i seguenti attributi:
    - `primoPiatto`: String
    - `secondoPiatto`: String
    - `dessert`: String
    - `bevanda`: String
    - `prezzoTotale`: double (calcolato come somma dei prezzi dei singoli elementi dell'ordine)
2. Implementare una classe `Menu` che contenga le opzioni disponibili per i primi piatti, i secondi piatti, i dessert e le bevande, insieme ai relativi prezzi. Utilizzare una struttura dati adeguata per memorizzare queste informazioni (ad esempio, una mappa).
3. Implementare una classe `Ristorante` che gestisca le ordinazioni dei clienti. Deve essere possibile aggiungere un'ordinazione, calcolare il totale delle ordinazioni e ottenere un elenco delle ordinazioni effettuate.
4. Utilizzare un database per memorizzare le ordinazioni. Creare una tabella `ordinazioni` che contenga i campi necessari per memorizzare le informazioni di ciascuna ordinazione.
5. Implementare le operazioni CRUD (Create, Read, Update, Delete) per le ordinazioni nel database utilizzando JDBC (Java Database Connectivity).

6. Implementare un'interfaccia utente (console o GUI) che consenta all'utente di:
    - Effettuare una nuova ordinazione
    - Visualizzare l'elenco delle ordinazioni effettuate
    - Visualizzare il totale delle ordinazioni effettuate

Nota: Assumere che ogni ordine includa esattamente un primo piatto, un secondo piatto, un dessert e una bevanda.

Primo esempio di progettazione

Esempio di implementazione della classe `Ordinazione`:

```
public class Ordinazione {
    private String primoPiatto;
    private String secondoPiatto;
    private String dessert;
    private String bevanda;
    private double prezzoTotale;

    // Costruttore
    public Ordinazione(String primoPiatto, String secondoPiatto, String
dessert, String bevanda) {
        this.primoPiatto = primoPiatto;
        this.secondoPiatto = secondoPiatto;
        this.dessert = dessert;
        this.bevanda = bevanda;
        this.prezzoTotale = calcolaPrezzoTotale();
    }

    // Metodi getter
    public String getPrimoPiatto() {
        return primoPiatto;
    }

    public String getSecondoPiatto() {
        return secondoPiatto;
    }

    public String getDessert() {
        return dessert;
    }

    public String getBevanda() {
        return bevanda;
    }

    public double getPrezzoTotale() {
        return prezzoTotale;
    }

    // Metodo privato per calcolare il prezzo totale
```

```
    private double calcolaPrezzoTotale() {
        // Implementazione per calcolare il prezzo totale in base ai prezzi
dei singoli elementi
        // Utilizzare il menu per ottenere i prezzi dei piatti e delle
bevande
    }
}
```

Per creare la tabella e inserire i dati nel database, possiamo utilizzare il linguaggio SQL. Ecco la query per creare la tabella `ordinazioni` e gli `INSERT` per i dati forniti:

```sql
-- Creazione della tabella ordinazioni
CREATE TABLE ordinazioni (
    id INT AUTO_INCREMENT PRIMARY KEY,
    primo_piatto VARCHAR(100),
    prezzo_primo DECIMAL(5, 2),
    secondo_piatto VARCHAR(100),
    prezzo_secondo DECIMAL(5, 2),
    dessert VARCHAR(100),
    prezzo_dessert DECIMAL(5, 2),
    bevanda VARCHAR(100),
    prezzo_bevanda DECIMAL(5, 2)
);

-- Inserimento dei dati
INSERT INTO ordinazioni (primo_piatto, prezzo_primo, secondo_piatto,
prezzo_secondo, dessert, prezzo_dessert, bevanda, prezzo_bevanda)
VALUES
    ('Ravioli al ragù', 5.00, 'Torta pasqualina', 6.00, 'Torta di pinoli',
3.00, 'Birra', 1.50),
    ('Trenette al pesto', 4.00, 'Stoccafisso alla Genovese', 5.00, 'Torta
di mele', 2.50, 'Vino rosso', 2.00),
    ('Minestrone', 3.50, 'Polpo in aglio e prezzemolo', 4.50, 'Tiramisù',
3.00, 'Acqua', 1.00),
    ('Pansotti in salsa di noci', 5.50, 'Cima', 4.50, 'Torta di rose',
2.50, 'Vino bianco', 2.00);
```

Questa query crea la tabella `ordinazioni` con i campi per i vari piatti (primo, secondo, dessert, bevanda) e i relativi prezzi. Successivamente, inserisce i dati forniti nelle righe della tabella.

Riprogettazione/evoluzione: mi accorgo che la soluzione precedente non è efficiente

Esercitazione: Gestione delle Ordinazioni in un Ristorante con Java e MySQL

Obiettivo: L'obiettivo di questa esercitazione è quello di applicare le conoscenze acquisite su Java e l'interazione con un database MySQL per creare un'applicazione di gestione delle ordinazioni in un ristorante.

Descrizione: Immagina di essere un programmatore che lavora per un ristorante e che deve sviluppare un sistema per gestire le ordinazioni dei clienti. Il ristorante tiene traccia di ogni ordine effettuato dai clienti, registrando i piatti scelti e i relativi prezzi, così come le bevande e i dessert. Ogni ordine è associato a un tavolo specifico.

Ti è stato fornito uno schema del database con due tabelle:

1. `ordinazioni`: contiene le informazioni sull'ordine, come il primo piatto, il secondo piatto, il dessert, la bevanda e i relativi prezzi.
2. `ordinazioni_tavolo`: associa ogni ordine a un tavolo specifico.

Compito: Utilizzando Java e JDBC (Java Database Connectivity), completa il seguente compito:

1. Scrivi un'applicazione Java che consenta di inserire una nuova ordinazione nel database. L'applicazione dovrebbe prendere in input i dettagli dell'ordinazione, come i piatti scelti e i relativi prezzi, la bevanda, il dessert e il tavolo a cui è associato l'ordine.

2. Assicurati che l'applicazione gestisca correttamente l'inserimento dell'ordine nelle tabelle `ordinazioni` e `ordinazioni_tavolo`, garantendo la coerenza dei dati.

3. Testa l'applicazione inserendo almeno un'ordinazione di prova e verifica che i dati siano correttamente registrati nel database.

Suggerimenti:

- Utilizza la classe `Connection` di JDBC per connetterti al database MySQL.
- Usa `PreparedStatement` per inserire in modo sicuro i dati nelle query SQL, prevenendo così le vulnerabilità legate all'iniezione di SQL.
- Ricorda di gestire le eccezioni durante l'interazione con il database.
- Verifica che il tuo ambiente di sviluppo sia configurato correttamente con il driver JDBC per MySQL.

Risorse:

- Documentazione Java JDBC: https://docs.oracle.com/javase/tutorial/jdbc/index.html
- Documentazione MySQL JDBC: https://dev.mysql.com/doc/connector-j/8.0/en/
- Esempi di codice e tutorial online su Java e JDBC.

Riprogettare la base dati

Per normalizzare lo schema delle tabelle e rendere il database più efficiente, possiamo separare i piatti e i relativi prezzi in una singola tabella e creare una tabella separata per i tipi di piatti. Ecco come potrebbe apparire lo schema normalizzato:

```
-- Tabella dei piatti
CREATE TABLE piatti (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100),
    prezzo DECIMAL(5, 2),
    tipo_piatto_id INT,
    FOREIGN KEY (tipo_piatto_id) REFERENCES tipi_piatto(id)
```

```
);

-- Tabella dei tipi di piatto
CREATE TABLE tipi_piatto (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(20)
);

-- Inserimento dei dati dei tipi di piatto
INSERT INTO tipi_piatto (nome) VALUES
    ('Primo'),
    ('Secondo'),
    ('Dessert'),
    ('Bevanda');

-- Inserimento dei dati dei piatti
INSERT INTO piatti (nome, prezzo, tipo_piatto_id) VALUES
    ('Ravioli al ragù', 5.00, 1),  -- 1 corrisponde a "Primo"
    ('Torta pasqualina', 6.00, 1),
    ('Torta di pinoli', 3.00, 3),   -- 3 corrisponde a "Dessert"
    ('Birra', 1.50, 4),             -- 4 corrisponde a "Bevanda"
    ('Trenette al pesto', 4.00, 1),
    ('Stoccafisso alla Genovese', 5.00, 2),
    ('Torta di mele', 2.50, 3),
    ('Vino rosso', 2.00, 4),
    ('Minestrone', 3.50, 1),
    ('Polpo in aglio e prezzemolo', 4.50, 2),
    ('Tiramisù', 3.00, 3),
    ('Acqua', 1.00, 4),
    ('Pansotti in salsa di noci', 5.50, 1),
    ('Cima', 4.50, 2),
    ('Torta di rose', 2.50, 3),
    ('Vino bianco', 2.00, 4);
```

Con questo schema normalizzato, abbiamo una tabella separata per i piatti e i relativi prezzi, e una tabella per i tipi di piatto. Ogni piatto è associato a un tipo di piatto tramite la chiave esterna `tipo_piatto_id`. Questo approccio migliora la manutenibilità e la flessibilità del database.

Per creare la tabella e inserire i dati nel database, possiamo utilizzare il linguaggio SQL. Ecco la query per creare la tabella `ordinazioni` e gli `INSERT` per i dati forniti:

Per registrare le ordinazioni di un generico tavolo del ristorante, possiamo creare una nuova tabella chiamata, ad esempio, `ordinazioni_tavolo`. Ecco una possibile struttura per questa tabella:

```
-- Creazione della tabella ordinazioni_tavolo
CREATE TABLE ordinazioni_tavolo (
    id INT AUTO_INCREMENT PRIMARY KEY,
    id_tavolo INT,
    id_ordine INT,
    FOREIGN KEY (id_ordine) REFERENCES ordinazioni(id)
```

```
);

-- Inserimento dei dati
INSERT INTO ordinazioni_tavolo (id_tavolo, id_ordine)
VALUES
    (1, 1), -- Ordine del tavolo 1 (id_ordine 1)
    (1, 3), -- Ordine del tavolo 1 (id_ordine 3)
    (2, 2), -- Ordine del tavolo 2 (id_ordine 2)
    (3, 4), -- Ordine del tavolo 3 (id_ordine 4)
    (3, 5), -- Ordine del tavolo 3 (id_ordine 5)
    (4, 6), -- Ordine del tavolo 4 (id_ordine 6)
    (4, 7), -- Ordine del tavolo 4 (id_ordine 7)
    (4, 8); -- Ordine del tavolo 4 (id_ordine 8)
```

In questa tabella, ogni record rappresenta una singola ordinazione effettuata da un tavolo specifico. Ogni record contiene l'ID univoco dell'ordinazione (`id`), l'ID del tavolo (`id_tavolo`) e l'ID dell'ordine preso dalla tabella `ordinazioni` (`id_ordine`). La chiave esterna `id_ordine` è collegata alla chiave primaria `id` della tabella `ordinazioni`.

Questa struttura consente di associare facilmente le ordinazioni a tavoli specifici, garantendo allo stesso tempo la coerenza dei dati con la tabella delle ordinazioni.

Form php

Ecco un esempio di pagina web HTML con un modulo per l'inserimento dei dati dei piatti:

```
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inserimento dati piatti</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        form {
            width: 400px;
            margin: 0 auto;
        }
        label, input {
            display: block;
            margin-bottom: 10px;
        }
        button {
            padding: 10px 20px;
            background-color: #4CAF50;
            color: white;
            border: none;
```

```
            cursor: pointer;
        }
        button:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <h2>Inserimento dati piatti</h2>
    <form id="piattoForm" action="inserisci_piatto.php" method="post">
        <label for="nome">Nome del piatto:</label>
        <input type="text" id="nome" name="nome" required>

        <label for="prezzo">Prezzo:</label>
        <input type="number" id="prezzo" name="prezzo" step="0.01" min="0"
required>

        <label for="tipo_piatto">Tipo di piatto:</label>
        <select id="tipo_piatto" name="tipo_piatto" required>
            <option value="">Seleziona il tipo di piatto</option>
            <option value="1">Primo</option>
            <option value="2">Secondo</option>
            <option value="3">Dessert</option>
            <option value="4">Bevanda</option>
        </select>

        <button type="submit">Inserisci Piatto</button>
    </form>
</body>
</html>
```

In questa pagina web, abbiamo un semplice modulo HTML con campi per il nome del piatto, il prezzo e il tipo di piatto. Quando viene inviato il modulo, i dati vengono inviati a un file PHP chiamato `inserisci_piatto.php` per l'elaborazione. Assicurati di creare il file `inserisci_piatto.php` per gestire l'inserimento dei dati nel database.

Form ajax JSP

Ecco la versione del codice HTML senza i riferimenti a PHP, pronta per essere utilizzata con JSP:

```
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Inserimento dati piatti</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
```

```
        }
        form {
            width: 400px;
            margin: 0 auto;
        }
        label, input, select {
            display: block;
            margin-bottom: 10px;
        }
        button {
            padding: 10px 20px;
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
        }
        button:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <h2>Inserimento dati piatti</h2>
    <form id="piattoForm">
        <label for="nome">Nome del piatto:</label>
        <input type="text" id="nome" name="nome" required>

        <label for="prezzo">Prezzo:</label>
        <input type="number" id="prezzo" name="prezzo" step="0.01" min="0"
required>

        <label for="tipo_piatto">Tipo di piatto:</label>
        <select id="tipo_piatto" name="tipo_piatto" required>
            <option value="">Seleziona il tipo di piatto</option>
            <option value="1">Primo</option>
            <option value="2">Secondo</option>
            <option value="3">Dessert</option>
            <option value="4">Bevanda</option>
        </select>

        <button type="button" onclick="inserisciPiatto()">Inserisci
Piatto</button>
    </form>

    <script>
        function inserisciPiatto() {
            // Recupera i valori dai campi del modulo
            var nome = document.getElementById("nome").value;
            var prezzo = document.getElementById("prezzo").value;
            var tipoPiatto = document.getElementById("tipo_piatto").value;

            // Effettua una richiesta AJAX per inviare i dati al server
            var xhr = new XMLHttpRequest();
            xhr.open("POST", "inserisci_piatto.jsp", true);
```

```
            xhr.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded");
            xhr.onreadystatechange = function () {
                if (xhr.readyState === 4 && xhr.status === 200) {
                    // Gestisce la risposta del server (opzionale)
                    alert(xhr.responseText);
                }
            };
            // Invia i dati al server
            xhr.send("nome=" + nome + "&prezzo=" + prezzo + "&tipo_piatto="
+ tipoPiatto);
        }
    </script>
</body>
</html>
```

In questo codice, abbiamo rimosso l'attributo `action` dal tag `<form>` e abbiamo cambiato il tipo del pulsante da `"submit"` a `"button"`. Inoltre, abbiamo aggiunto una funzione JavaScript `inserisciPiatto()` che viene chiamata quando viene cliccato il pulsante. Questa funzione recupera i valori dai campi del modulo e invia una richiesta AJAX al server utilizzando JavaScript puro. Assicurati di creare il file JSP `inserisci_piatto.jsp` per gestire l'inserimento dei dati nel database.

Menu piatti javascript

Ecco un esempio di pagina web HTML con una tabella per visualizzare i piatti proposti:

```
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Visualizzazione Piatti</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        table {
            width: 100%;
            border-collapse: collapse;
            margin-bottom: 20px;
        }
        th, td {
            border: 1px solid #ddd;
            padding: 8px;
            text-align: left;
        }
        th {
            background-color: #f2f2f2;
        }
```

```
    </style>
</head>
<body>
    <h2>Visualizzazione Piatti</h2>
    <table>
        <thead>
            <tr>
                <th>Nome del piatto</th>
                <th>Prezzo</th>
                <th>Tipo di piatto</th>
            </tr>
        </thead>
        <tbody id="piattiTableBody">
            <!-- Qui verranno aggiunti dinamicamente i dati dei piatti -->
        </tbody>
    </table>

    <script>
        // Simulazione di dati dei piatti (puoi caricare dinamicamente i
dati dal server)
        var piatti = [
            { nome: "Ravioli al ragù", prezzo: 5.00, tipoPiatto: "Primo" },
            { nome: "Torta pasqualina", prezzo: 6.00, tipoPiatto: "Primo"
},
            { nome: "Torta di pinoli", prezzo: 3.00, tipoPiatto: "Dessert"
},
            { nome: "Birra", prezzo: 1.50, tipoPiatto: "Bevanda" }
            // Aggiungi altri piatti se necessario
        ];

        // Funzione per popolare la tabella con i dati dei piatti
        function popolaTabellaPiatti() {
            var tableBody = document.getElementById("piattiTableBody");
            tableBody.innerHTML = ""; // Pulisce la tabella prima di
popolarla di nuovo

            // Itera sui piatti e aggiungi righe alla tabella
            piatti.forEach(function (piatto) {
                var row = "<tr>" +
                          "<td>" + piatto.nome + "</td>" +
                          "<td>" + piatto.prezzo.toFixed(2) + " €</td>" +
                          "<td>" + piatto.tipoPiatto + "</td>" +
                        "</tr>";
                tableBody.innerHTML += row;
            });
        }

        // Chiamata alla funzione per popolare la tabella all'avvio della
pagina
        popolaTabellaPiatti();
    </script>
</body>
</html>
```

In questo codice, abbiamo una tabella HTML con intestazioni per il nome del piatto, il prezzo e il tipo di piatto. Utilizzando JavaScript, popoliamo dinamicamente la tabella con i dati dei piatti. In questo esempio, i dati dei piatti sono simulati all'interno del codice JavaScript, ma potresti caricarli dinamicamente dal server.

Ordini al tavolo

Ecco un esempio di pagina web HTML per fare le ordinazioni al tavolo, utilizzando l'elenco dei piatti disponibili:

```
<!DOCTYPE html>
<html lang="it">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ordinazione al Tavolo</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 20px;
        }
        label, select, button {
            display: block;
            margin-bottom: 10px;
        }
        button {
            padding: 10px 20px;
            background-color: #4CAF50;
            color: white;
            border: none;
            cursor: pointer;
        }
        button:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <h2>Ordinazione al Tavolo</h2>
    <form id="ordinazioneForm">
        <label for="piatto">Seleziona un piatto:</label>
        <select id="piatto" name="piatto" required>
            <option value="">Seleziona un piatto</option>
            <!-- Qui verranno aggiunti dinamicamente i piatti disponibili -
->
        </select>

        <label for="quantita">Quantità:</label>
        <input type="number" id="quantita" name="quantita" value="1"
min="1" required>
```

```
        <button type="button" onclick="aggiungiPiatto()">Aggiungi
Piatto</button>
    </form>

    <h3>Ordini al tavolo:</h3>
    <ul id="ordiniTavolo">
        <!-- Qui verranno aggiunti dinamicamente gli ordini al tavolo -->
    </ul>

    <script>
        // Dati dei piatti disponibili (simulati)
        var piattiDisponibili = [
            { nome: "Ravioli al ragù", prezzo: 5.00 },
            { nome: "Torta pasqualina", prezzo: 6.00 },
            { nome: "Torta di pinoli", prezzo: 3.00 },
            { nome: "Birra", prezzo: 1.50 }
            // Aggiungi altri piatti se necessario
        ];

        // Funzione per popolare il menu a discesa con i piatti disponibili
        function popolaMenuPiatti() {
            var selectPiatto = document.getElementById("piatto");
            selectPiatto.innerHTML = ""; // Pulisce il menu a discesa prima
di popolarlo di nuovo

            // Itera sui piatti disponibili e aggiungi opzioni al menu a
discesa
            piattiDisponibili.forEach(function (piatto) {
                var option = document.createElement("option");
                option.text = piatto.nome + " - " +
piatto.prezzo.toFixed(2) + " €";
                option.value = JSON.stringify(piatto); // Salva l'oggetto
piatto come stringa JSON nel valore dell'opzione
                selectPiatto.add(option);
            });
        }

        // Chiamata alla funzione per popolare il menu a discesa all'avvio
della pagina
        popolaMenuPiatti();

        // Funzione per aggiungere un piatto all'ordine del tavolo
        function aggiungiPiatto() {
            var selectPiatto = document.getElementById("piatto");
            var selectedPiatto = JSON.parse(selectPiatto.value); // Parsa
la stringa JSON per ottenere l'oggetto piatto selezionato
            var quantita =
parseInt(document.getElementById("quantita").value);

            // Costruisce una stringa per rappresentare l'ordine da
visualizzare
            var ordine = selectedPiatto.nome + " - Quantità: " + quantita +
" - Totale: " + (selectedPiatto.prezzo * quantita).toFixed(2) + " €";
```

```
            // Aggiunge l'ordine alla lista degli ordini del tavolo
            var ordiniTavolo = document.getElementById("ordiniTavolo");
            var listItem = document.createElement("li");
            listItem.textContent = ordine;
            ordiniTavolo.appendChild(listItem);
        }
    </script>
</body>
</html>
```

In questa pagina, abbiamo un modulo che consente di selezionare un piatto disponibile e specificare la quantità desiderata. Premendo il pulsante "Aggiungi Piatto", l'ordine viene visualizzato nella lista degli ordini del tavolo. I dati dei piatti sono simulati all'interno del codice JavaScript, ma potresti caricarli dinamicamente dal server.