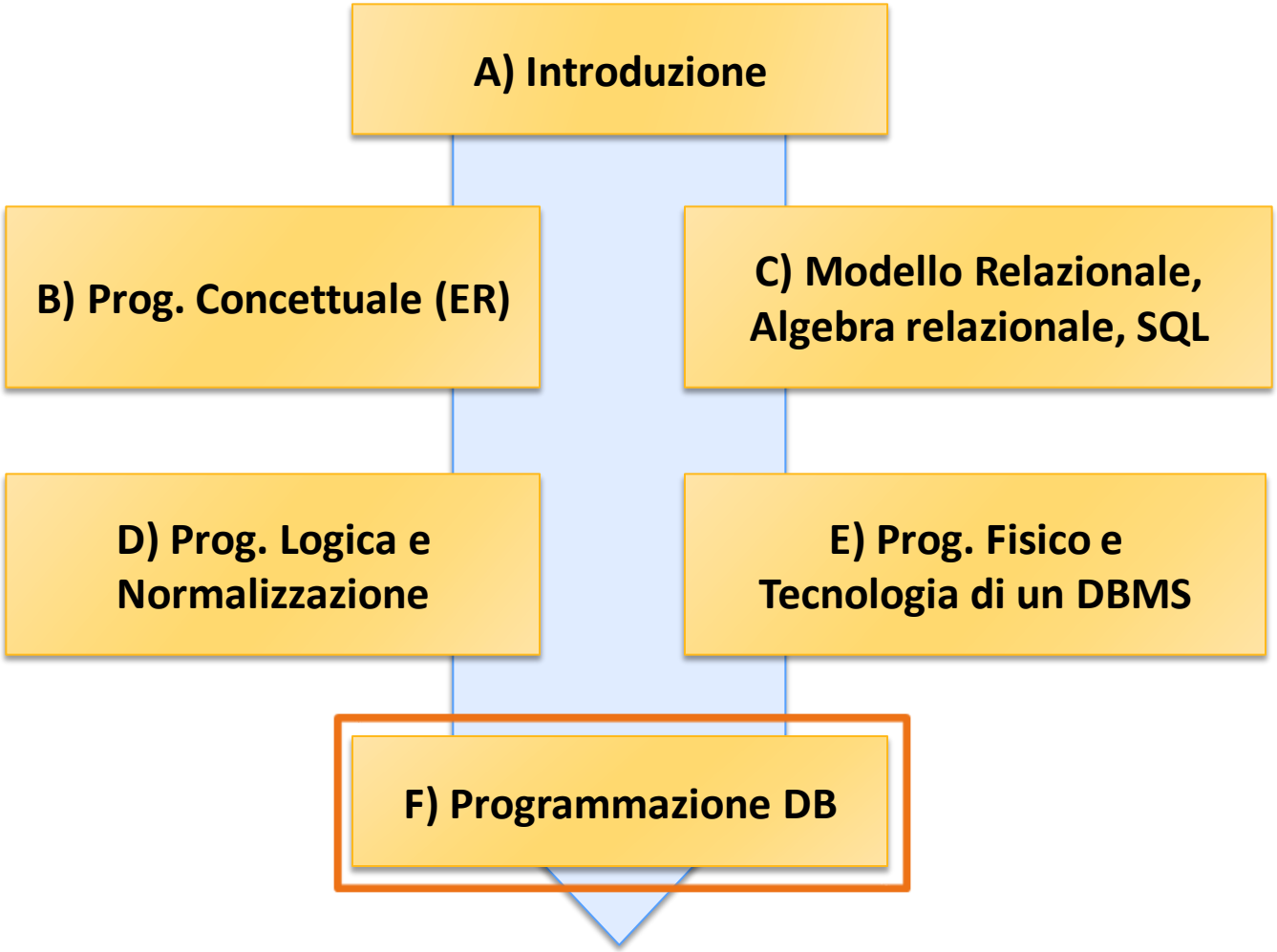
Basi di Dati

Esercizio completo JDBC

Basi di Dati – Dove ci troviamo?



Esercizio JDBC

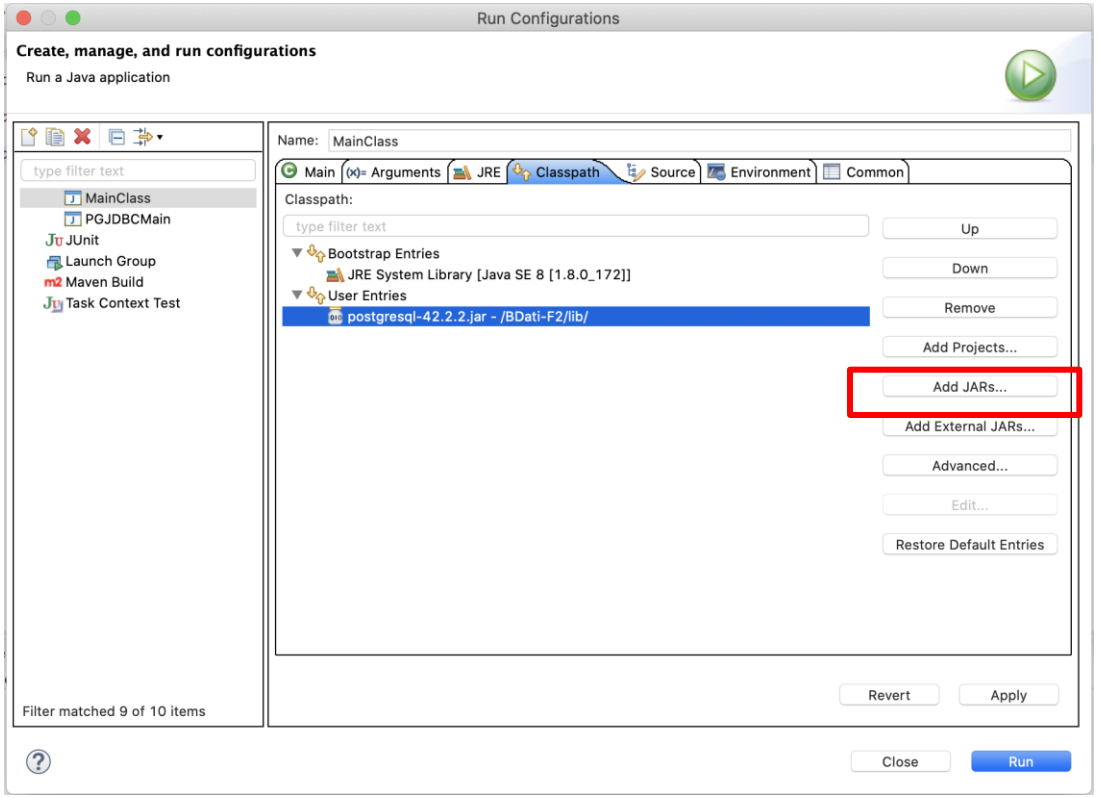
- In questa esercitazione, creeremo un programma Java-JDBC con il quale:
  - Creeremo una tabella nel database
  - Popoleremo la tabella con dati di esempio
  - Effettueremo interrogazioni sui dati

Esercizio JDBC

- Per cominciare:
  - Creare un nuovo database ‘Studenti’
  - Effettuare il download dello scheletro di codice dell’esercitazione dal sito di Basi di Dati
  - Effettuare il download del driver JDBC corretto (file JAR) in base alla propria versione di Java dal sito di PostgreSQL:
    - http://jdbc.postgresql.org/download.html
  - Sostituire il file JAR di esempio presente nella sottocartella “lib” del progetto con quello sopra scaricato

Inclusione del JAR: esempio con Eclipse

- Dopo aver posizionato il JAR nella sottocartella "lib" è sufficiente aggiungerlo al classpath della vostra run configuration (Run→Run Configurations→Classpath)



Esercizio JDBC

```
package esempio;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class MainClass {

    public static void main(String[] args){
        try{
            Class.forName ("org.postgresql.Driver");  // Load the Driver
            Connection conn = DriverManager.getConnection( "jdbc:postgresql://localhost:5432/studenti", "user", "pw" );
            Statement stmt = conn.createStatement();

            ...

            conn.close();
        }
        catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
        catch (SQLException e) {
            e.printStackTrace();
        }

    }
}
```

Esercizio JDBC – Parte 1.a

- Struttura del DATABASE:
  - STUDENTI (matr, cognome, nome)

- Completare il codice utilizzando uno Statement JDBC per:
  - creare una tabella STUDENTI avente lo schema sopra riportato
  - inserire i seguenti dati di esempio:
    - (1, 'rossi', 'mario')
    - (2, 'bianchi', 'sergio')

Esercizio JDBC – Parte 1.a: soluzione

```
String sql = "CREATE TABLE STUDENTI(matr integer primary key, cognome varchar, nome varchar)";
stmt.executeUpdate(sql);
sql = "INSERT INTO STUDENTI VALUES(1, 'rossi', 'mario'), (2, 'bianchi', 'sergio')";
stmt.executeUpdate(sql);
stmt.close();
```

Esercizio JDBC – Parte 1.b

- Completare il codice utilizzando lo stesso Statement di cui sopra e l'oggetto ResultSet per:
  - Effettuare l'interrogazione "SELECT * FROM STUDENTI"
  - Scorrere e stampare i risultati ottenuti

Esercizio JDBC – Parte 1.b: soluzione

```
sql = "SELECT * FROM STUDENTI";
ResultSet rs = stmt.executeQuery(sql);
while (rs.next()) {
    int matr = rs.getInt("matr");
    String cognome = rs.getString("cognome");
    String nome = rs.getString("nome");
    System.out.println(matr+" "+cognome+" "+nome);
}
rs.close();
stmt.close();
```

Esercizio JDBC – Parte 2.a

- Completare il codice utilizzando un PreparedStatement e il relativo ResultSet per:
  - Preparare l'interrogazione parametrica "SELECT * FROM STUDENTI WHERE nome = ? and matr > ?"
  - Eseguire l'interrogazione per la coppia di valori ("sergio", "0")

Esercizio JDBC – Parte 2.a: soluzione

```
sql = "SELECT * FROM STUDENTI WHERE nome = ? and matr > ?";
PreparedStatement preparedStatement = conn.prepareStatement(sql);
preparedStatement.setString(1, "sergio");
preparedStatement.setInt(2, 0);
rs = preparedStatement.executeQuery();
while (rs.next()) {
    int matr = rs.getInt(1);
    String cognome = rs.getString(2);
    String nome = rs.getString(3);
    System.out.println(matr+" "+cognome+" "+nome);
}
rs.close();
preparedStatement.close();
```

Esercizio JDBC

- Soluzione completa
  - Disponibile nell'archivio zip sul sito dell'insegnamento

JDBC – Ulteriori esercizi

- Provare il collegamento JDBC agli altri database utilizzati per il corso, incluso quello sviluppato per il proprio progetto