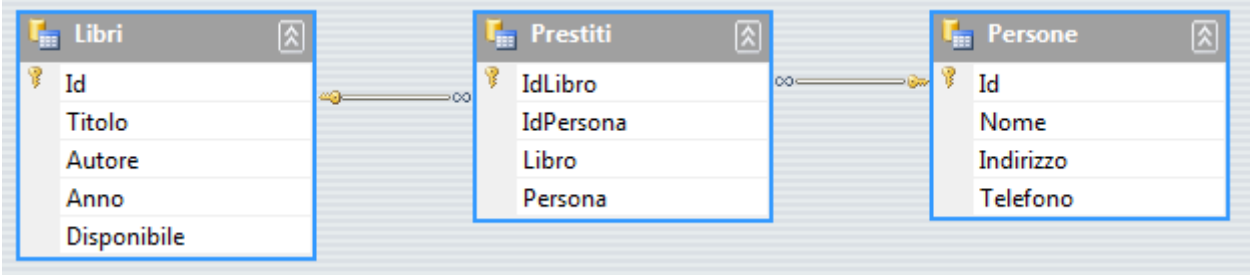
Laboratorio 6

Passo 0 – Creare un nuovo progetto di tipo *Windows Forms Application* e di nome **`Lab6`** ed eliminare i file **`Program`** e **`Form1`**. Scaricare dalla pagina web del corso **`MiniLib.zip`**, **`Lab5bis.zip`** e **`Lab6Start.zip`**. il progetto **`MiniLib`** deve essere aggiunto alla *solution* e non deve essere modificato. Tutti i file (e folder) del progetto **`Lab5bis`** devono essere copiati nel direttorio del progetto **`Lab6`** e inseriti nel progetto stesso. Rinominare il *namespace* **`Lab5`** in **`Lab6`** in tutti i file del progetto e fare il *build*. L'applicazione deve funzionare come in precedenza. **`Lab6Start.zip`** contiene il file "BibliotecaDS.xml" da copiare nel direttorio del progetto **`Lab6`** e da utilizzare per il caricamento dei dati una volta effettuato il *refactoring* dell'applicazione.

Passo 1 – **Creazione del `DataSet`**.
Aggiungere al progetto **`Lab6`** un nuovo **`DataSet`** *typed* di nome **`BiblotecaDataSet`** e aggiungere le tabelle e le colonne di figura.



`Disponibile`, **`Libro`** e **`Persona`** sono colonne calcolate e il valore da assegnare alla loro proprietà **`Expression`** sarà definito in seguito. I tipi delle varie colonne sono deducibili dal modello dei dati. Tutte le colonne non devono accettare valori nulli (proprietà **`AllowDBNull`** = **`False`**) e devono avere come valore di *default* (proprietà **`DefaultValue`**) un valore congruente con il loro tipo: stringa vuota per le colonne di tipo **`string`**, 0 per le colonne di tipo **`int`**, **`true`** per la colonna **`Disponibile`**. I campi chiave **`Id`** delle tabelle **`Libri`** e **`Persone`**, devono essere impostati in modo che sia generato automaticamente un nuovo valore ogni volta che viene creata una nuova tupla (proprietà **`AutoIncrement = true`**).
Aggiungere le relazioni **`LibriPrestiti`** e **`PersonePrestiti`** che consentono di specificare i vincoli di integrità referenziale fra le *foreign keys* **`IdPersona`** e **`IdLibro`** di **`Prestiti`** e le chiavi di **`Libri`** e **`Persone`**.
La colonna **`Disponibile`** della tabella **`Libri`** deve restituire **`true`** se il libro NON è presente nella tabella **`Prestiti`**; le colonne **`Libro`** e **`Persona`** della tabella **`Prestiti`** devono restituire rispettivamente il titolo del libro e il nome della persona. In tutti e tre i casi, in *designer* assegnare alla proprietà **`Expression`** l'espressione che restituisce il risultato desiderato (si devono utilizzare le *keyword* **`Parent`** e **`Child`** – vedere la sezione "PARENT/CHILD RELATION REFERENCING" nella documentazione della proprietà **`Expression`**).

Passo 2 – **Refactoring**.
Nella classe **`Biblioteca`**, aggiungere un campo *readonly* (**`_dataSet`**) di tipo **`BibliotecaDataSet`** e la corrispondente proprietà. Esporre le **`DataTable`** del **`DataSet`** mediante le proprietà: **`LibriTable`**, **`PersoneTable`** e **`PrestitiTable`**. Ridefinire il metodo **`Invalidate`** in modo che, oltre a richiamare il metodo base, invochi **`AcceptChanges`** sul **`DataSet`**. Modificare i metodi di caricamente e salvataggio su XML,

in modo che sia il **DataSet** ad occuparsi direttamente del caricamento (metodo **ReadXml**) e del salvataggio (metodo **WriteXml**).
Le classi **Libro**, **Persona** e **Prestito** devono diventare dei *wrapper* "usa e getta" delle corrispondenti **DataRow**. A tal fine, aggiungere alla classe **Libro** un campo *readonly* **_libroRow** di tipo **BibliotecaDataSet.LibriRow**, alla classe **Persona** un campo *readonly* **_personaRow** di tipo **BibliotecaDataSet.PersoneRow** e alla classe **Prestito** un campo *readonly* **_prestitoRow** di tipo **BibliotecaDataSet.PrestitiRow** (si noti che **LibriRow**, `PersoneRow` e `PrestitiRow` sono classi annidate di `BibliotecaDataSet` create in automatico dal *designer*). Aggiungere alle tre classi anche una proprietà che restituisca il valore dei campi di cui sopra. Eliminare tutti i vecchi campi (**_id**, **_titolo**, ecc.) e, nel codice delle corrispondenti proprietà, accedere direttamente alla proprietà equivalente della **DataRow** *wrappata*. In ognuna delle classi di cui sopra, eliminare tutti i costruttori che inizializzano un'istanza da XML (dopo le modifiche alla classe **Biblioteca** non servono più); nei costruttori rimasti inserire la creazione di una nuova **DataRow**; infine, aggiungere un costruttore che accetti un riferimento a un oggetto **Biblioteca** e un riferimento alla **DataRow** "tipata" relativa.
Nella classe **Biblioteca**, al posto dei **Dictionary** utilizzare le corrispondenti **DataTable** e modificare di conseguenza il codice di tutti i metodi coinvolti, utilizzando in modo adeguato le classi *wrapper* delle **DataRow**; ad esempio, nel caso dei servizi sui libri: **GetLibro(int id)** deve restituire **null** o una nuova istanza di **Libro** che *wrappa* la **DataRow** di **LibriDataTable** con l'id specificato; **GetLibri()** deve restituire una lista di *wrapper* su tutte le **DataRow** di **LibriDataTable**; **Aggiungi(Libro libro)** deve inserire in **LibriDataTable** la **DataRow** *wrappata* da **libro**.
Nella classe **BibliotecaController**, nei metodi **NewDocument** e **OpenDocument** effettuare il *databinding* sulle **DataTable** – poiché utilizzando le **DataTable** il *refresh* dei dati nelle *grid* è completamente automatico, è indispensabile eliminare tutte le invocazioni ai metodi **RefreshItem**, **AddItem** e **RemoveItem** delle *grid*. Infine, modificare il codice delle proprietà **LibroCorrente**, **PersonaCorrente** e **PrestitoCorrente** – attenzione a ciò che viene restituito dalla proprietà **Current** del controllo **Grid**.
Perché l'applicazione funzioni correttamente, nella classe **BibliotecaManager** è indispensabile che non venga utilizzato il **Logger**: a tal fine, commentare tutte le invocazioni ai metodi **OpenLog** e **CloseLog**.

Passo 3 – **Validazione dei dati**.

Aprire in *designer* **BibliotecaDataSet** e fare un doppio click sulla tabella **Libro** e sulla tabella **Persona**. Il *designer* crea in automatico il file "BibliotecaDataSet.cs" con una parte delle classi **LibriDataTable** e **PersoneDataTable**. Ridefinire in entrambe le classi il metodo **OnColumnChanged** e fare in modo che quando viene modificato il valore di una colonna di una riga, in caso di valore non accettabile, venga associato a quella colonna un messaggio di errore mediante il metodo **SetColumnError**. In particolare, nel caso di un libro, il titolo e l'autore non devono essere stringhe vuote; nel caso di una persona, il nome non deve essere una stringa vuota.