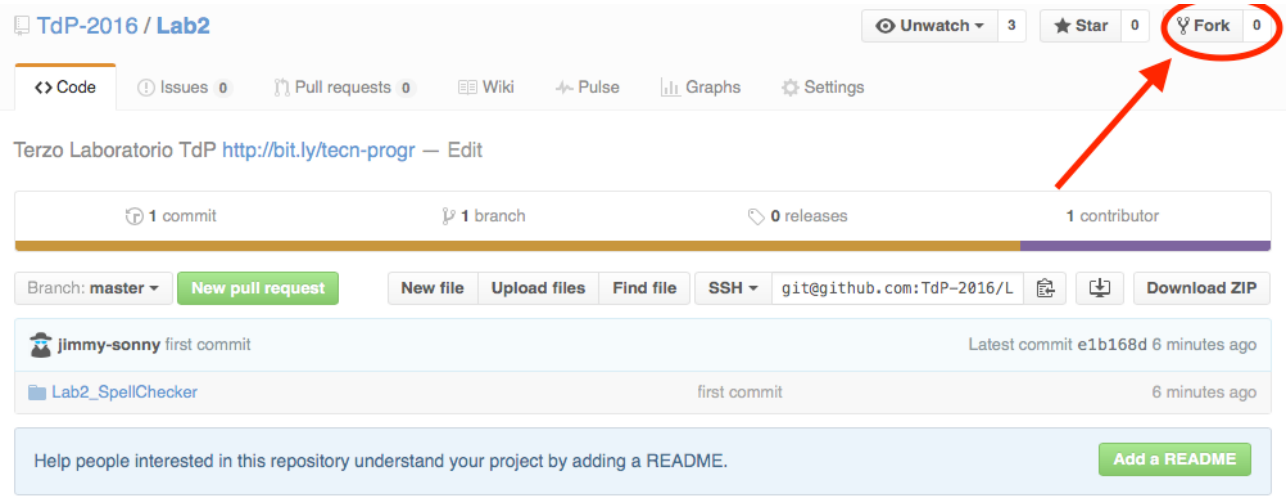
03FYZ TECNICHE DI PROGRAMMAZIONE
Istruzioni per effettuare il fork di un repository GitHub

- Effettuare il login su GitHub utilizzando il proprio username e password.
- Aprire il repository su GitHub relativo al decimo laboratorio:
- https://github.com/TdP-2018/Lab10
- Utilizzare il pulsante *Fork* in alto a destra per creare una propria copia del progetto.



L'azione di Fork crea un nuovo repository nel proprio accout GitHub con una copia dei file necessari per l'esecuzione del laboratorio.

- Aprire Eclipse, andare su *File -> Import*. Digitare *Git* e selezionare *Projects from Git -> Next -> Clone URI -> Next*.
- Utilizzare la URL del **proprio** repository che si vuole clonare (**non** quello in TdP-2018!), ad esempio:

  https://github.com/my-github-username/Lab10
- Fare click su *Next*. Selezionare il branch (*master* è quello di default) fare click su *Next*.
- Selezionare la cartella di destinazione (quella proposta va bene), fare click su *Next*.
- Selezionare *Import existing Eclipse projects*, fare click su *Next* e successivamente su *Finish*.
- Il nuovo progetto Eclipse è stato clonato ed è possibile iniziare a lavorare.
- A fine lavoro ricordarsi di effettuare Git commit e push, utilizzando il menù *Team in Eclipse*.

ATTENZIONE: solo se si effettua Git **commit** e successivamente Git **push** le modifiche locali saranno propagate sui server GitHub e saranno quindi accessibili da altri PC e dagli utenti che ne hanno visibilità.

03FYZ TECNICHE DI PROGRAMMAZIONE
Esercitazione di Laboratorio 10 – 23 maggio 2018

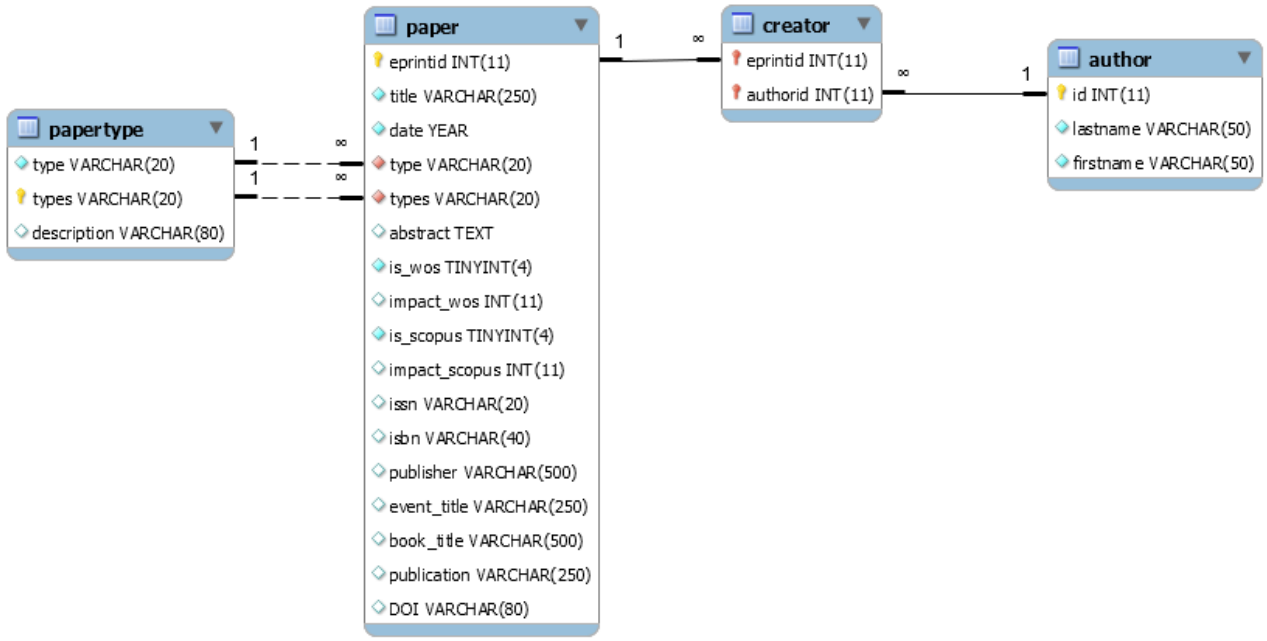
Obiettivi dell'esercitazione:

- Object Relational Mapping
- Identity Map
- Connection Pooling

ESERCIZIO 1:

Si consideri il data-set **porto.sql**, che contiene una lista di alcune[1] pubblicazioni scientifiche del Politecnico di Torino. La versione aggiornata del database è inclusa nel repository di questo laboratorio (cartella *database*). In *Fig 1* è mostrata la struttura delle tabelle.

Fig 1



La tabella *paper* contiene gli articoli pubblicati, indicandone il titolo, l'anno di pubblicazione (*date*) e gli altri dati bibliografici. Ogni articolo appartiene ad una delle tipologie definite da *papertype* (classificate sulla base di due codici, *type* e *types*, di cui il secondo è più specifico del primo ed è quindi adatto come identificatore univoco). I ricercatori del Politecnico coinvolti nelle pubblicazioni sono raccolti nella tabella *author*, mentre la tabella *creator* lega ciascuna pubblicazione ai propri autori e, viceversa, ciascun autore alle proprie pubblicazioni.

Dopo aver effettuato il fork del progetto relativo al decimo laboratorio (Lab10), sviluppare un'applicazione JavaFX che permetta di identificare tutti i co-autori di un determinato autore, selezionato dall'utente tramite un menu a tendina. Un co-autore è definito come un autore che ha scritto almeno un articolo in comune con l'autore dato.

---

[1] In particolare, contiene le pubblicazioni del dipartimento DAUIN tra l'anno 2012 e l'anno 2015, compresi, estratte dal sito ufficiale http://porto.polito.it/

Per risolvere l'esercizio, si costruisca un grafo (semplice, non orientato e non pesato), i cui vertici rappresentino gli autori, ed i cui archi rappresentino il fatto che i due autori siano tra loro co-autori. L'utente può selezionare dal menu a tendina un autore, e facendo click sul pulsante "Trova co-autori" nell'area di testo sottostante vengono visualizzati tutti gli autori con cui l'autore selezionato ha scritto almeno un articolo.

ESERCIZIO 2:

Si estenda il programma dell'esercizio precedente, permettendo all'utente di selezionare, da una seconda tendina, un altro autore, che non sia co-autore del primo.

Alla pressione del bottone "Sequenza articoli", il programma deve elencare una lista di articoli in grado di "collegare" il primo autore al secondo. Nel dettaglio, il primo articolo della lista deve avere, tra gli autori, il primo autore selezionato ed almeno un altro autore (a1); il secondo articolo deve avere l'autore a1 ed almeno un altro autore (a2); e così via fino all'ultimo articolo, che conterrà tra i propri autori anche il secondo autore selezionato.

Questo problema si può risolvere in diversi modi alternativi (sceglierne uno… oppure divertirsi a provarli tutti):

1. Calcolare il cammino minimo sul grafo dell'esercizio 1 (che contiene solo gli autori), e poi, per ogni coppia di vertici adiacenti, consultare il database per trovare almeno un articolo (che esisterà sicuramente) che abbia tali vertici tra gli autori.

2. Modificare la creazione del grafo, utilizzando un tipo di arco personalizzato (ereditato da DefaultEdge), nel quale memorizzare almeno un articolo (oppure la lista di tutti gli articoli).

3. Costruire, utilizzando le regole dell'Object Relational Mapping, l'elenco completo degli articoli e degli autori, e le relative relazioni molti-a-molti. Una volta trovato il cammino minimo sul grafo dell'esercizio 1, utilizzare le relazioni per trovare gli articoli richiesti (suggerimento: è sufficiente l'intersezione tra i due insiemi autore1.getArticoli() ed autore2.getArticoli()).