UNIVERSITÀ DEGLI STUDI DI SALERNO

Gruppo 10

Prof. Giancarlo Nota

Software Engineering

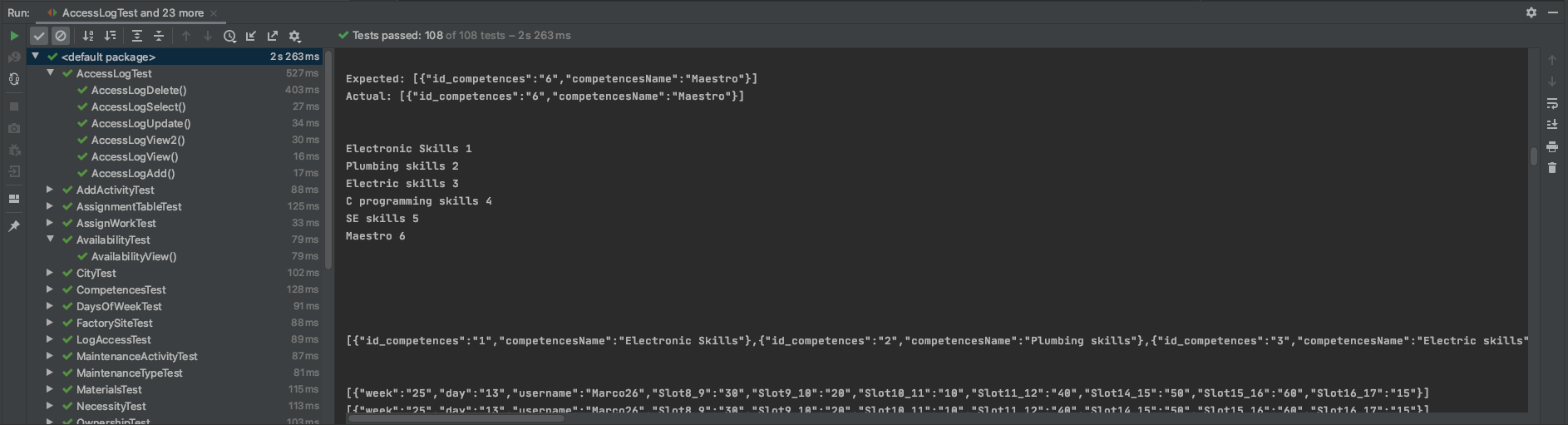
Relazione di Testing

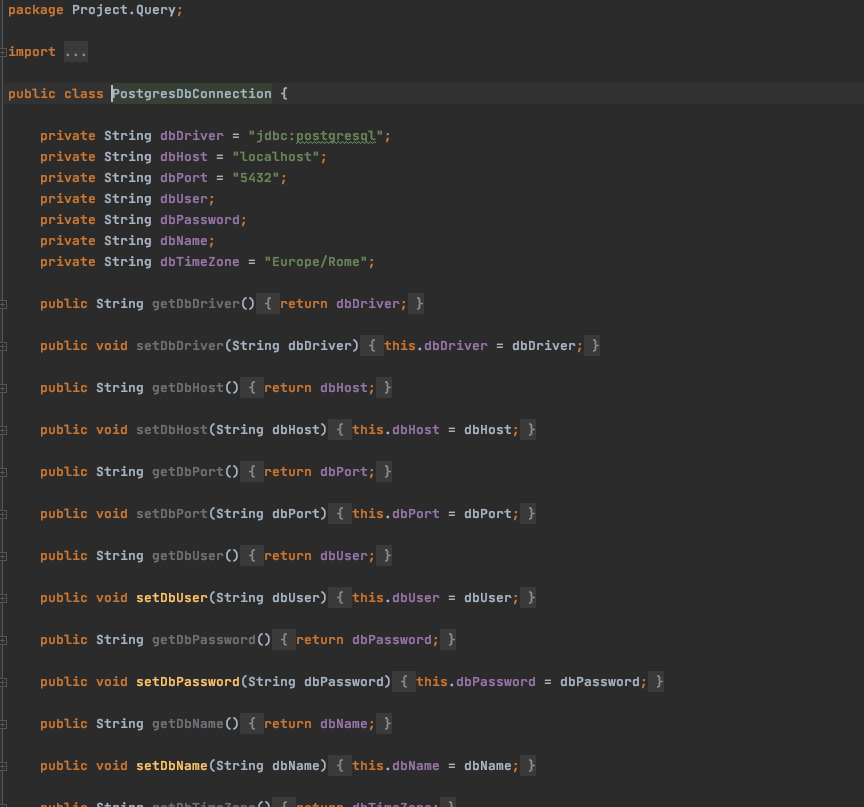
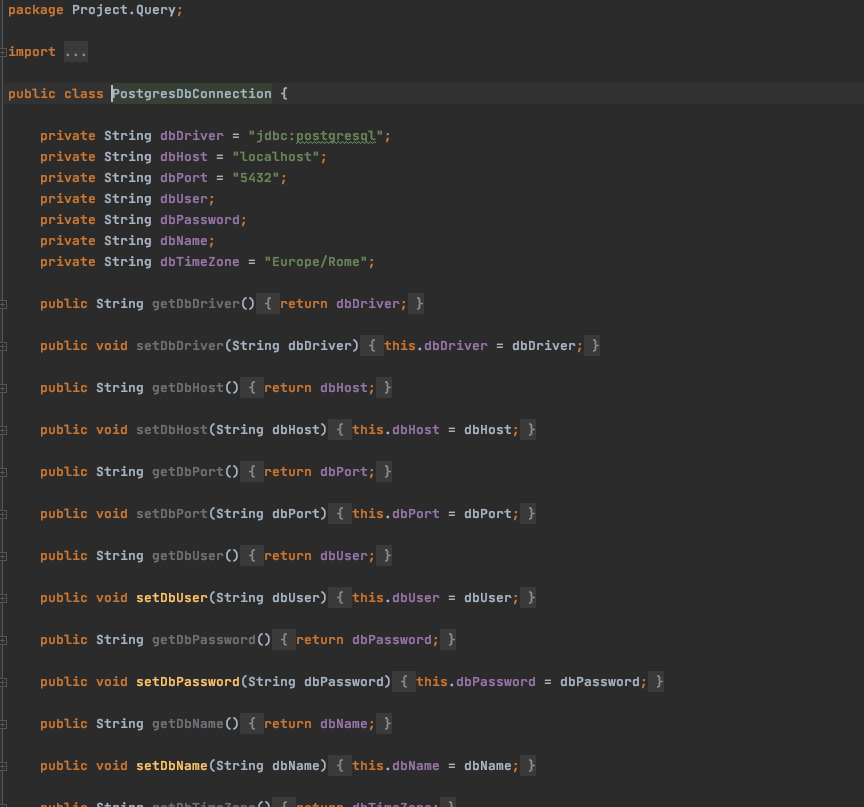
La relazione seguente ha come tema quello di riassumere l’attività di Testing relativa alla costruzione delle classi che implementano i metodi , in Java, per interagire con il Server Postgres.

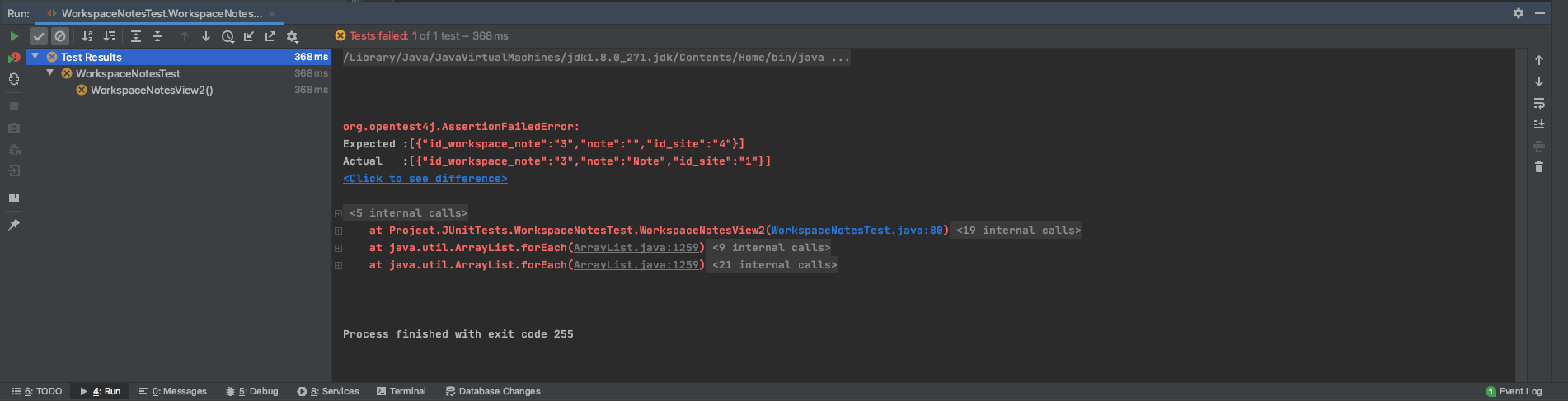
Per fare ciò abbiamo utilizzato l’IDE di programmazione IntelliJ e abbiamo creato due package, chiamati rispettivamente Query e JUnitTests. Il primo contiene le classi che implementano i metodi di aggiunta, rimozione, aggiornamento e visualizzazione di un’insieme di tuple all’interno del database, mentre il secondo contiene le classi, costruite importando la libreria JUnit, che effettuano delle verifiche su questi metodi.

Come detto precedentemente, per effettuare questi test abbiamo bisogno di una classe che ci permetta di effettuare un collegamento con il database e di un driver (nel nostro caso il driver Postgres) che importiamo nella libreria del nostro progetto.

Una volta costruita la classe relativa a ciascuna entità presente nel nostro database e dopo aver costruito i metodi, creiamo una classe Tester per ogni entità che richiama i metodi presenti nella rispettiva classe.

Selezionando accuratamente tutte le classi di Test all’interno del package, arriviamo al risultato presente nella seguente figura.

Esplicitamente, creiamo anche un metodo che induce il nostro test a fallire (in particolare, usando la funzione assert e confrontando i risultati che arrivano in input dal metodo con una stringa in formato JASON) e da come possiamo vedere nella prossima figura, la stringa che ci viene restituita risulta diversa da quella che ci aspettiamo di avere come output.

Ripetendo questo controllo per ogni classe, ci assicuriamo che ogni entità presente all’interno del database costruito funziona in maniera corretta.