JDBC: UNIVERSITÀ

Disheng Qiu
disheng.qiu@gmail.com

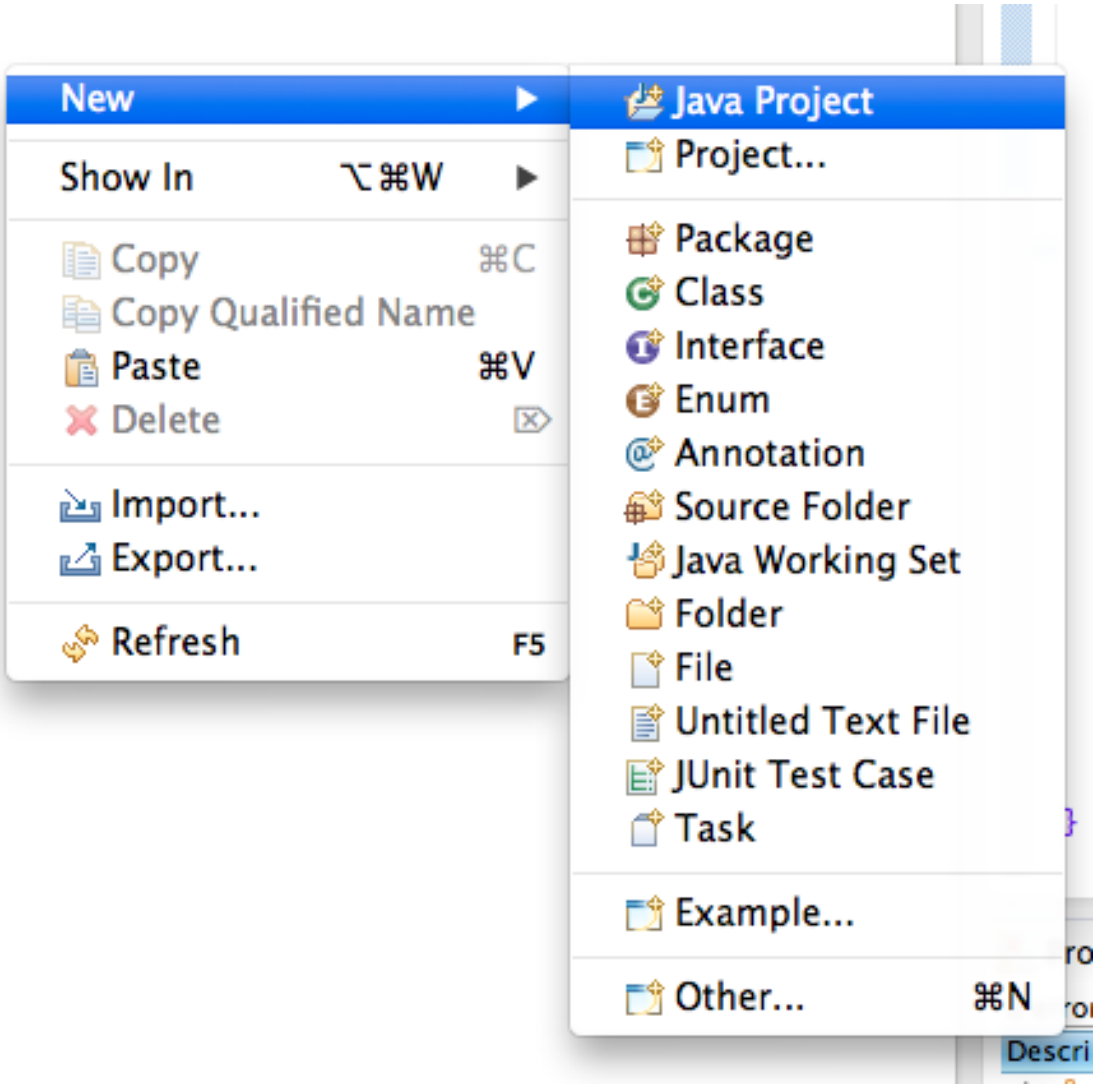
Emanuel Weitschek
emanuel@dia.uniroma3.it

Modello

- Exam — exams(student_code, score, course, date)
  - score
  - course
  - date

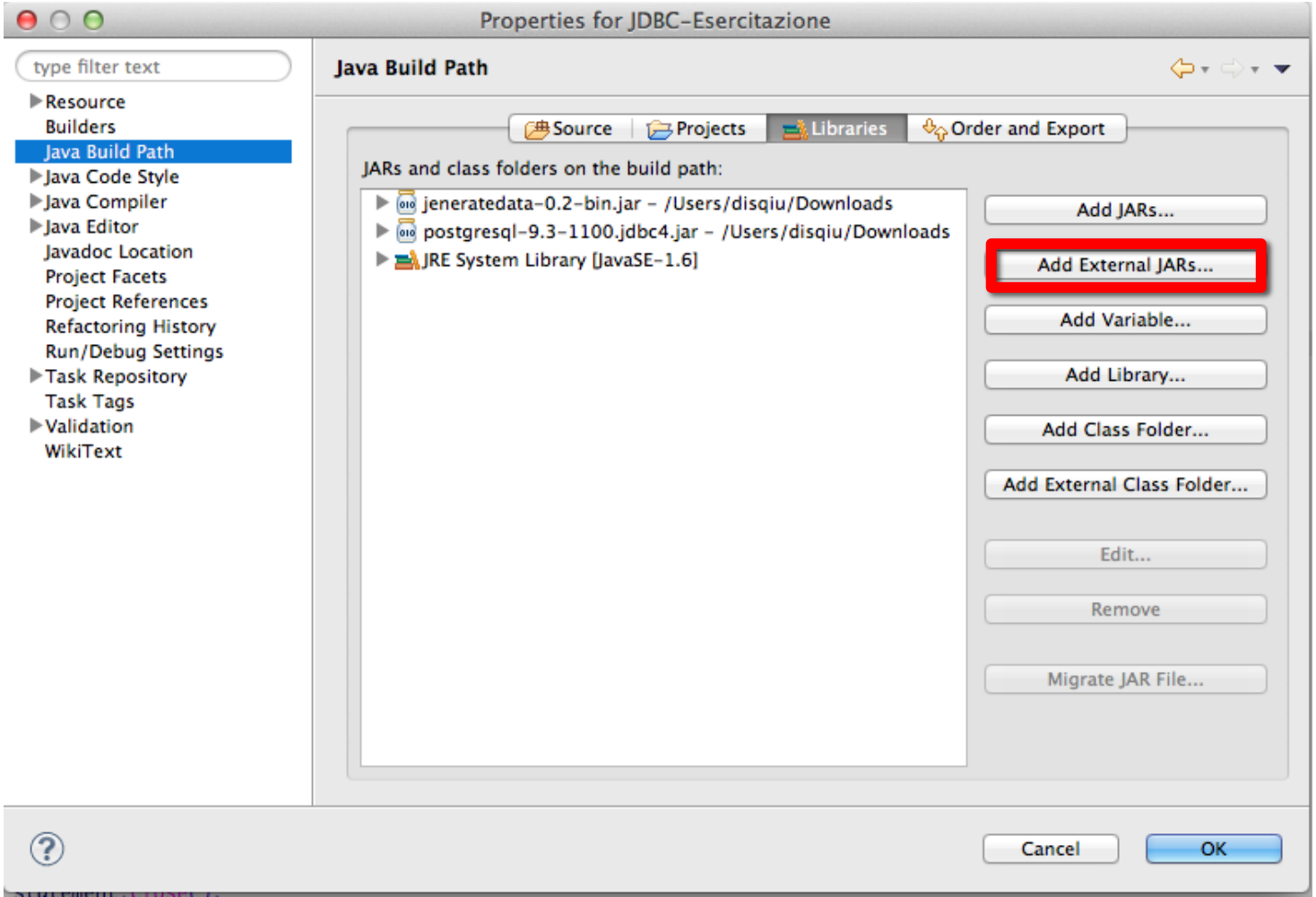
- Student — students(code, first_name, last_name, birth_date)
  - firstName
  - lastName
  - code
  - birthDate
  - exams

Creazione nuovo progetto



Import jar esterni

Progetto > Properties > Java Build Path > Add External Jars



Cosa c’è da importare?

- postgresql-9.3-1100.*
  - Driver JDBC per postgres
  - postgresql-9.3-1100.jdbc3 => versione java 1.5 o maggiore*
  - postgresql-9.3-1100.jdbc4 => versione java 1.6 o maggiore*
  - postgresql-9.3-1100.jdbc41 => versione java 1.7 o maggiore*

- jeneratedata-0.2-bin.jar
  - Generazione pseudo randomnica di valori
  - Stringhe, Nomi, Date etc

* Versione java: java –version

Packages

- It.uniroma3.dia.db1.jdbc
  - UniversityShell: con due metodi per ripulire il database e popolare il database con 1000 studenti e i relativi esami
- It.uniroma3.dia.db1.factory
  - ExamFactory: Creazione di istanze di Exam pseudo randomici
  - StudentFactory: Creazione di istanze di Student pseudo randomici (genera anche gli esami relativi)
- It.uniroma3.dia.db1.model
  - Student: modello dello studente
  - Exam: modello dell'esame
- It.uniroma3.dia.db1.persistence
  - Datasource: classe per la creazione di connessioni al dbms
  - StudentRepository: classe per il retrieve e per la persistenza di studenti (e gli esami associati)
  - ExsamRespository: classe per il persist e l'eliminazione degli esami.

Creazione del db

- Andare su pgAdmin e creare un nuovo database chiamato “universita”
- Eseguire i seguenti script per la generazione delle tabelle:

```
CREATE TABLE students(
        code integer NOT NULL,
        firstname character varying(64) NOT NULL,
        lastname character varying(64) NOT NULL,
        birthdate date NOT NULL,  CONSTRAINT pk_students PRIMARY
        KEY (code )
)
```

Creazione del db

- Andare su pgAdmin e creare un nuovo database chiamato “universita”
- Eseguire i seguenti script per la generazione delle tabelle:

```
CREATE TABLE exams(
        student_code integer,
        score integer,
        course character varying(64),
        date date,
        CONSTRAINT exams_student_code_fkey FOREIGN KEY
        (student_code)      REFERENCES students (code) MATCH SIMPLE
        ON UPDATE NO ACTION ON DELETE NO ACTION
)
```

Main

- Provate a lanciare il main:
  - deleteAllStudents() elimina i studenti dal dbms
  - generateAndPrint(int n) genera, persiste n studenti e stampa tutti gli studenti nel dbms

- Qual è il principale collo di bottiglia?
- Come si può fare per superare il collo di bottiglia?
- Di quanto migliora le performance?

StudentRepository

- findAll()
  - Ritorna la lista di tutti gli studenti
  - Perché left join?
  - Che differenza c'è tra questa soluzione e una soluzione che utilizza un possibile metodo List<Exam> getExams(String studentCode) di ExamRepository?
- Delete(Student student)
  - Che succede se si sposta l'eliminazione degli esami dopo l'eliminazione dello studente?
  - Cosa si può fare per risolvere il problema?