Relational Databases

Petar Ivanov
September, 2019

PUBLIC

THE BEST RUN SAP

MySQL
Installation (*Windows*)

Install *MySQL Community Server*

§ https://dev.mysql.com/downloads/windows/installer/8.0.html
- *installer-web* – smaller file is downloaded, installation requires Internet connection
- *installer* – larger file is downloaded, installation does not require Internet connection
- After choosing between the two, scroll down for downloading without registration (‘*No thanks, just start my download*’ link)
- ‘*Server Only*’ as ‘*Setup Type*’ is sufficient

Install *Microsoft Visual C++ 2015 Redistributable Package* if haven’t already (MySQL installer will notify you if you need to install that)

§ https://www.microsoft.com/en-us/download/details.aspx?id=52685

Select ‘*Configure MySQL Server as a Windows Service*’. Take note of the name of the Service (e.g. ‘*MySQL80*’)

Deselect ‘*Start the MySQL Server at System Startup*’ (instructions on how to start it manually will follow)

Add MySQL’s *bin* directory to *PATH* environment variable

MySQL

Windows Service

Run a command prompt as *Administrator*

- Stop: *net stop MySQL80*
- Start: *net start MySQL80*

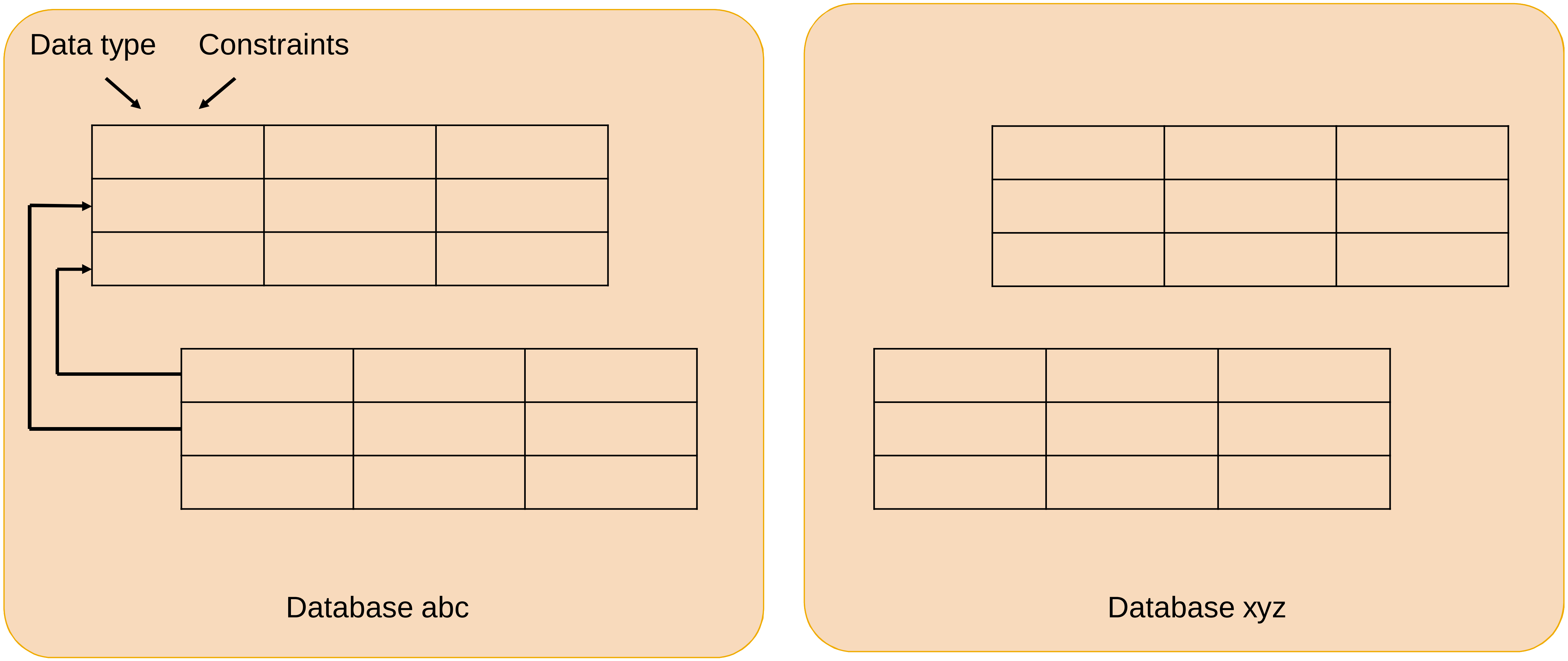
Connect to MySQL Server

- *mysql -u <user> -p*

Try it yourself

- Assuming the service is running
- Connect with *root* user (expecting to be successful)
- Execute *exit*
- *net stop MySQL80*
- Try connecting with *root* user (expecting a *Can't connect* error)
- *net start MySQL80*

Relational Databases



Relational Databases

Retrieving records

employee

id	first_name	last_name
1	John	Doe
2	Jack	Smith

*select * from employee;*

select first_name, last_name from employee;

MySQL
Accounts

List users

§ *select user from mysql.user;*

Create a user

§ *create user student identified by 'some-password';*

Remove a user

§ *drop user student;*

Try it yourself

- Try connecting to MySQL server with the *student* user prior to creating it (expecting *access denied* error)
- Create user *student*
- Connect to the MySQL server with it
- Execute *select current_user();*
- Execute *exit*

MySQL
Accounts

Taking hosts into account. ‘*%*’ is a wildcard.

List users

§ *select user, host from mysql.user;*

Create user *remote*

§ *create user remote@example.com identified by 'some-password';*

Remove user *remote*

§ *drop user remote@example.com;*

Create user *local*

§ *create user local@localhost identified by 'some-password';*

Remove user *local*

§ *drop user local@localhost;*

Try it yourself

- Create user *remote* from the left
- Try connecting to the MySQL server with user *remote* (expecting *access denied* error)
- Remove user *remote*
- Create user *local* from the left
- Connect with user *local* to MySQL
- Execute *select current_user();*
- Execute *exit*
- Remove user *local*

MySQL
Permissions

As *student* user

- *create database geeky;*
  (expecting *access denied* error)

As *root* user

- *grant all on geeky.* to 'student';*

As *student* user

- *create database geeky;*
- *show databases;*
- *use geeky;*
- *show tables;*

MySQL
Data types

Integer data types (can be signed and unsigned)

Data type	Size (bytes)
tinyint	1
smallint	2
mediumint	3
int	4
bigint	8

serial – synonym for *bigint unsigned not null auto_increment unique*

Floating point data types – *float*, *double*

Fixed point data types – *decimal(precision, scale)*

Date-time data types – *date* (‘yyyy-mm-dd’), *time* (‘hh:mm:ss’), *datetime* (‘yyyy-mm-dd hh:mm:ss.ffffff’), *timestamp* (deals with time-zone offsets), *year* (‘yyyy’)

MySQL

Data types

char (const size – as specified but up to 255, trailing spaces)

varchar

binary

varbinary

LOBs (large, stored out of the table and just referenced)

- blob – tinyblob, blob, mediumblob, longblob
- text – tinytext, text, mediumtext, longtext

enum – one of predefined values

set – zero or more of predefined values without repetitions

MySQL

Data types

- *create table tbl (data tinyint);*
- *show tables;*
- *describe tbl;*

Try it yourself

- *insert into tbl values(15);*
- *insert into tbl values(255);*
  – (expecting an *out of range* error here – max value for signed tinyint is 127)
- *select * from tbl;*
- *drop table tbl;*

- *create table tbl (data tinyint unsigned);*
- *insert into tbl values(255);*
- *drop table tbl;*

MySQL

Data types

- *create table tbl (num int, str varchar(50));*
- *insert into tbl values (1, 'one'), (2, 'two'), (3, 'three');*
- *select num, str from tbl;*
- *drop table tbl;*

Try it yourself

- *create table tbl (product varchar(100), price decimal(10, 2), purchased_on date);*
- *insert into tbl values('hair dryer', 76.98, '2019-05-27');*
- *insert into tbl values ('dishwasher', 204.32, '2019-08-14');*
- *select * from tbl;*
- *drop table tbl;*

MySQL
Constraints

Not null

- *create table tbl (data tinyint not null);*
- *insert into tbl values(null);*
  – (expecting *value cannot be null* error)
- *drop table tbl;*

Try it yourself

Default

- *create table tbl(data tinyint default 16);*
- *insert into tbl values ();*
- *select * from tbl;*
- *drop table tbl;*

MySQL
Constraints

Check

- *create table tbl (data tinyint check (data > 30));*
- *insert into tbl values(30);*
  - (expecting a *check constraint violation* error)
- *drop table tbl;*

Try it yourself

Unique

- create table tbl(data tinyint unique);
- insert into tbl values(5);
- insert into tbl values(5);
  - (expecting a *unique constraint violation* error)
- drop table tbl;

Primary key and foreign key will be illustrated when discussing relations between tables;

MySQL
Altering table

Try it yourself

- *create table tbl(num int not null);*
- *insert into tbl values (1), (2);*
- *describe tbl;*
- *select * from tbl;*

- *alter table tbl add added_later varchar(500) default 'goofy';*
- *describe tbl;*
- *select * from tbl;*

- *alter table tbl modify num tinyint not null;*
- *describe tbl;*
- *select * from tbl;*

- *alter table tbl drop num;*
- *describe tbl;*
- *select * from tbl;*

- *alter table tbl modify added_later int;*
  - (expecting an error – cannot convert value to int)
- *drop table tbl;*

MySQL

Single table queries

Try it yourself

John Doe
Jack Smith
Emily Roberts
Sophie Davis

- *create table exam (name varchar(50), exam_number tinyint unsigned check (exam_number <= 3), score tinyint unsigned check (score <= 100));*

- *insert into exam values('John Doe', 1, 78),*
  ('Jack Smith', 1, 89),
  ('Emily Roberts', 1, 93),
  ('Sophie Davis', 1, 55);

- *insert into exam values('John Doe', 2, 64),*
  ('Jack Smith', 2, 58),
  ('Emily Roberts', 2, 88),
  ('Sophie Davis', 2, 92);

- *insert into exam values('John Doe', 3, 85),*
  ('Jack Smith', 3, 100),
  ('Emily Roberts', 3, 94),
  ('Sophie Davis', 3, 98);

MySQL

Single table queries, retrieving data

Try it yourself

Retrieve exam number, name and score for all students for exam #1

```
select exam_number, name, score from exam where exam_number = 1;
```

Retrieve all data for all students for exam #1 ordered by score in ascending order

```
select * from exam where exam_number = 1 order by score;
```

Retrieve the names and the scores of the 2 students with highest score on exam #3

```
select name, score from exam where exam_number = 3 order by score desc limit 2;
```

Retrieve all exam results of Emily Roberts and John Doe

```
select * from exam where name = 'Emily Roberts' or name = 'John Doe' order by name;
select * from exam where name in ('Emily Roberts', 'John Doe') order by name;
```

MySQL

Single table queries, retrieving data

Try it yourself

Retrieve the exam results of all students whose name starts with ‘J’

*select * from exam where name like 'J%' order by name;*

Retrieve name and score of all students who have 70 or more points on exam #2

select exam_number, name, score from exam where score >= 70 and exam_number = 2;

Retrieve the content of the whole table ordered by the exam number in descending order and score in ascending order

*select * from exam order by exam_number desc, score asc;*

Retrieve all the different student names

select distinct name from exam;

MySQL

Single table queries, aggregation functions

Try it yourself

Retrieve the number of records in the *exam* table

select count(*) from exam;

Retrieve the number of students who have participated in exam #1

select count(*) from exam where exam_number = 1;

Retrieve the sum of the scores of Sophie Davis for all tests

select sum(score) from exam where name = 'Sophie Davis';

Retrieve the average score in exam #3

select avg(score) from exam where exam_number=3;

Retrieve the record for the highest score in exam #2

select score from exam where exam_number=2 order by score desc limit 1;

select max(score) from exam where exam_number = 2;

MySQL

Single table queries, group by, aggregation functions and having

Try it yourself

select exam_number from exam;

select exam_number from exam group by exam_number;

Retrieve the average score in exam #3

select avg(score) from exam where exam_number=3;

Retrieve the average score for each exam

select exam_number, avg(score) from exam group by exam_number;

Retrieve the average scores of exams that are 80 or above

select exam_number, avg(score) from exam group by exam_number having avg(score) >= 80;

select exam_number, avg(score) as av from exam group by exam_number having av >= 80;

MySQL

Single table queries, update and delete

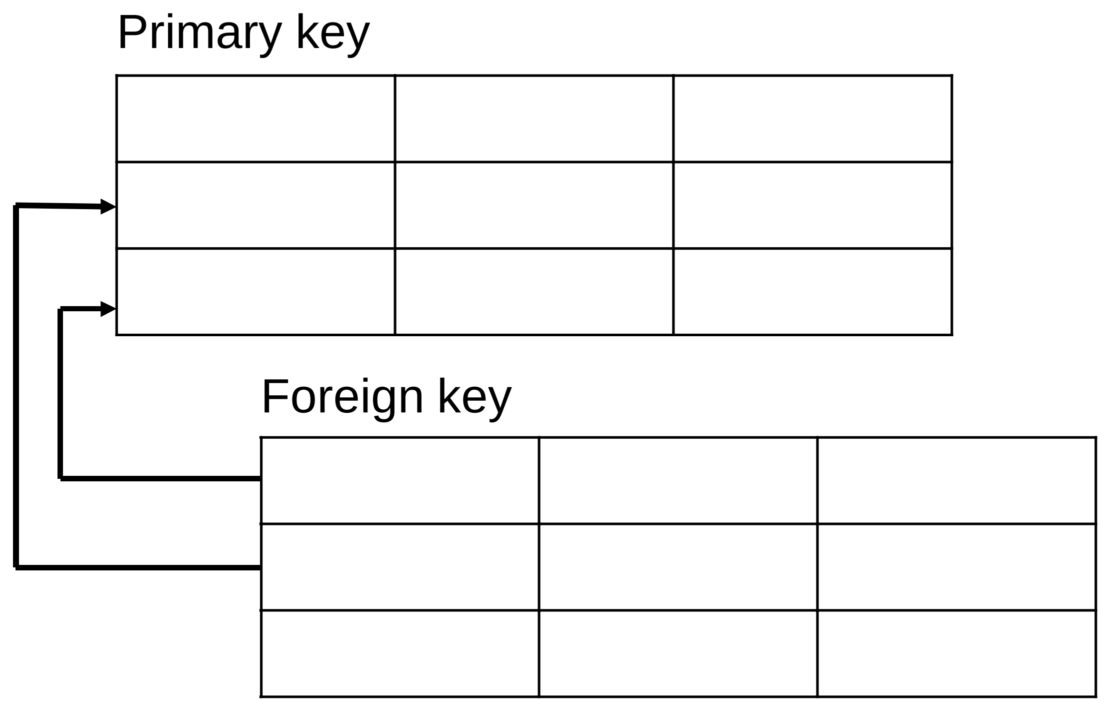
Try it yourself

```
create table upper_case (num int, str varchar(50));
insert into upper_case values (0, 'zero'), (1, 'one'), (2, 'two'), (-5, 'nonsense');


update upper_case set str='ZERO' where num=0;
update upper_case set str=upper(str) where num=1 or num=2;


delete from upper_case where num=-5;
delete from upper_case;
```

MySQL
Linking tables



MySQL

Linking tables, primary key

Primary key

- Unambiguously identifies a row in a table
- Cannot be null
- Should be unique

Try it yourself

- *create table pk_example (*$\overbrace{\textit{id bigint unsigned not null unique auto\_increment}}^{\textit{serial}}$*, primary key(id), value varchar(50));*
- *insert into pk_example values(null, 'something');*
- *insert into pk_example (value) values('something-else');*

MySQL

Linking tables, primary key

Composite primary key

Try it yourself

- *insert into accounts_example values (0, 0, 'John Doe'),*
  (0, 1, 'Jack Smith'), (1, 0, 'Emily Roberts'), (1, 1, 'Sophie Davis');

- *insert into accounts_example values(0, 1, 'Someone Else');*
  – (expecting a *duplicate key* error)
- *insert into accounts_example values(0, 2, 'Julie Roberts');*

MySQL

Linking tables, 1:1 relationship

upper_case

value	id (PK)
ONE	1
TWO	2
THREE	3

lower_case

upper_id (FK, unique)	value
1	one
2	two
3	three

MySQL

Linking tables, 1:1 relationship

Try it yourself

- *create table upper_case (id serial, primary key(id), value varchar(50) not null);*
- *insert into upper_case (value) values ('ONE'), ('TWO'), ('THREE');*

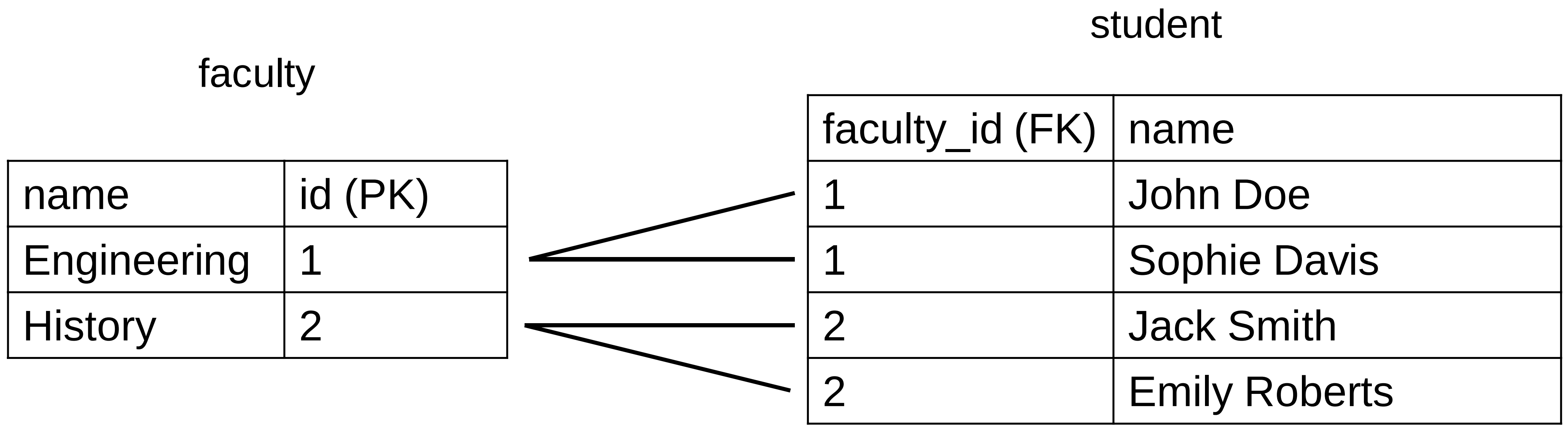
- *create table lower_case(value varchar(50) not null, upper_id bigint unsigned unique not null, foreign key(upper_id) references upper_case(id));*
- *insert into lower_case values ('one', 1), ('two', 2), ('three', 3);*

- *select lower_case.value, upper_case.value from lower_case join upper_case on lower_case.upper_id=upper_case.id;*

- *insert into lower_case values ('something', 3);*
  – (expecting to fail with *duplicate entry* error)

MySQL

Linking tables, 1:M relationship



faculty

name	id (PK)
Engineering	1
History	2

student

faculty_id (FK)	name
1	John Doe
1	Sophie Davis
2	Jack Smith
2	Emily Roberts

MySQL

Linking tables, 1:M relationship

Try it yourself

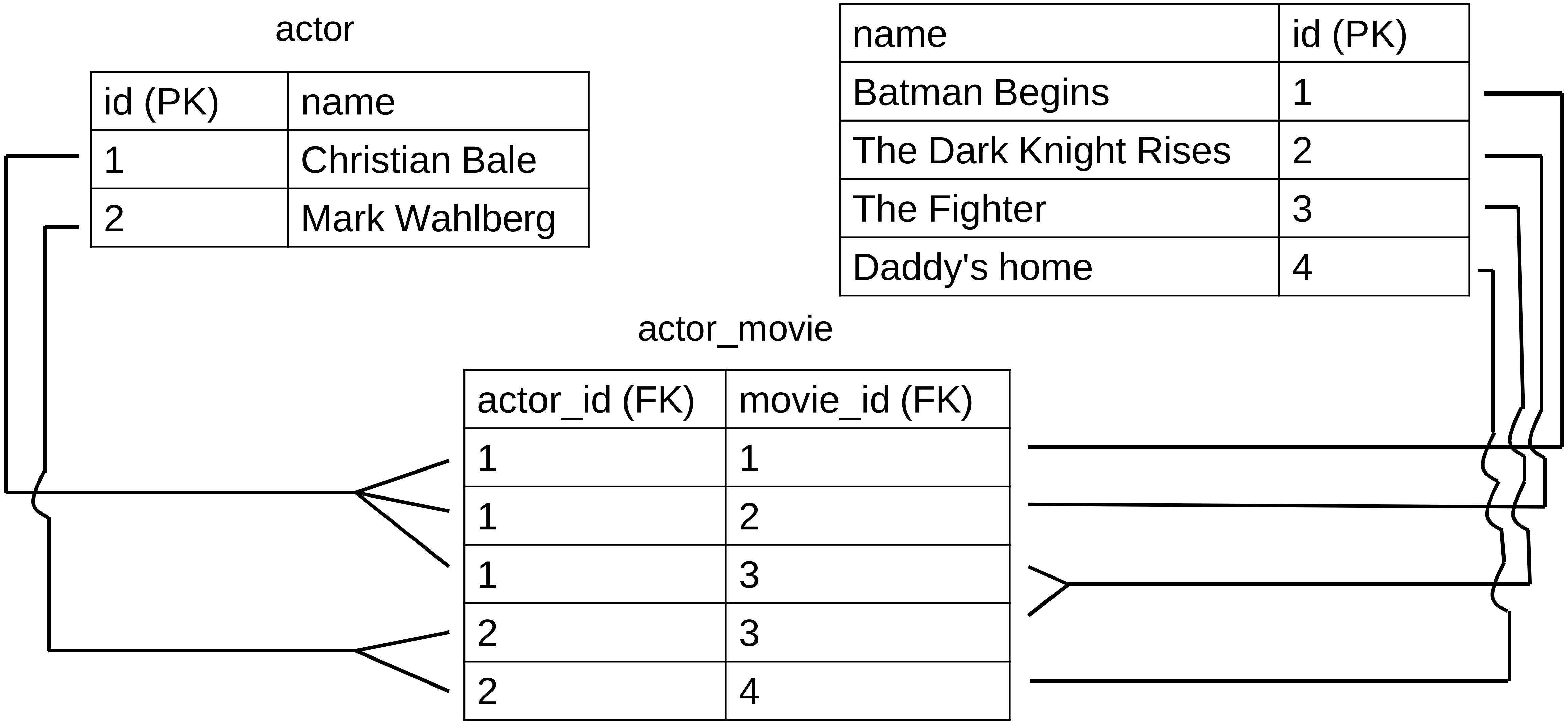
- *create table faculty (id serial, primary key(id), name varchar(50) not null);*
- *insert into faculty values (null, 'Engineering'), (null, 'History');*

- *create table student (faculty_id bigint unsigned not null, foreign key(faculty_id) references faculty(id), name varchar(50) not null);*
- *insert into student values(1, 'John Doe'), (1, 'Sophie Davis');*
- *insert into student values(2, 'Jack Smith'), (2, 'Emily Roberts');*

- *select student.name, faculty.name from student join faculty on student.faculty_id=faculty.id;*

MySQL

Linking tables, M:M relationship



actor

id (PK)	name
1	Christian Bale
2	Mark Wahlberg

movie

name	id (PK)
Batman Begins	1
The Dark Knight Rises	2
The Fighter	3
Daddy's home	4

actor_movie

actor_id (FK)	movie_id (FK)
1	1
1	2
1	3
2	3
2	4

MySQL

Linking tables, M:M relationship

Try it yourself

- *create table actor (id serial, primary key(id), name varchar(50) not null);*
- *insert into actor (name) values ('Christian Bale'), ('Mark Wahlberg');*

- *create table movie (id serial, primary key(id), name varchar(50) not null);*
- *insert into movie (name) values ('Batman Begins'), ('The Dark Knight Rises'), ('The Fighter'), ('Daddy\'s home');*

- *create table actor_movie (actor_id bigint unsigned not null, foreign key(actor_id) references actor(id), movie_id bigint unsigned not null, foreign key(movie_id) references movie(id), unique(actor_id, movie_id));*
- *insert into actor_movie values (1, 1), (1, 2), (1, 3), (2, 3), (2, 4);*

- *select movie.name as Movie, actor.name as Actor from movie join actor_movie on actor_movie.movie_id=movie.id join actor on actor_movie.actor_id=actor.id;*

MySQL

Linking tables, types of join

Try it yourself

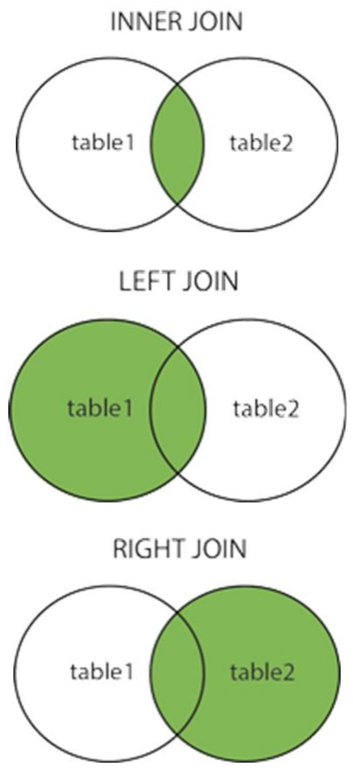
- *create table artist (id serial, primary key(id), name varchar(50) not null);*
- *insert into artist (name) values ('Gims'), ('David Guetta'), ('Beyonce');*

- *create table song (name varchar(50) not null, artist_id bigint unsigned, foreign key(artist_id) references artist(id));*
- *insert into song values ('Titanium', 2), ('Halo', 3), ('Havana', NULL);*

- *select artist.name as Artist, song.name as Song from artist join song on song.artist_id=artist.id;*

- *select artist.name as Artist, song.name as Song from artist left join song on song.artist_id=artist.id;*

- *select artist.name as Artist, song.name as Song from artist right join song on song.artist_id=artist.id;*



MySQL

Linking tables, on update and on delete

Try it yourself

- *create table faculty (id int, primary key(id), name varchar(50) not null);*
- *insert into faculty values(1, 'Engineering');*

- *create table student (faculty_id int, foreign key(faculty_id) references faculty(id) on update restrict, name varchar(50) not null);*
- *insert into student values(1, 'John Doe');*
- *update faculty set id=2 where id=1;*

restrict – does not allow the update (default)

cascade – updates both the PK and the FK

set null – updates the PK, sets FK to *null*

- *create table student (faculty_id int, foreign key(faculty_id) references faculty(id) on delete restrict, name varchar(50) not null);*
- *insert into student values(1, 'John Doe');*
- *delete from faculty where id=1;*

restrict – does not allow the delete (default)

cascade – deletes the row of the PK and the one of the FK

set null – deletes the row of the PK, sets the FK to *null*

MySQL

Exercise

Try it yourself

address

id	country	city	street	number

company

id	name	address_id

customer

id	first_name	last_name	address_id

product

id	name	category	company_id	price

purchase

id	date	customer_id

purchased_items

purchase_id	product_id

MySQL
Exercise

Try it yourself

As *root*

- *grant all on store.* to student;*

As *student*

- *create database store;*
- *use store;*

- *create table address (id serial, primary key(id), country varchar(50) not null, city varchar(50) not null, street varchar(50) not null, number int unsigned null);*

- *insert into address values (null, 'Belgium', 'Brussels', 'Rue Antoine Dansaert', 208);*
- *insert into address values (null, 'Belgium', 'Brussels', 'Boulevard Adolphe Max', 1);*
- *insert into address values (null, 'Bulgaria', 'Sofia', 'Baba Vida', null);*
- *insert into address values (null, 'Bulgaria', 'Sofia', 'Nesho Bonchev', 36);*
- *insert into address values (null, 'Bulgaria', 'Plovdiv', 'Pririn planina', 9);*
- *insert into address values (null, 'Bulgaria', 'Plovdiv', 'Rozova dolina', 83);*

MySQL
Exercise

Try it yourself

- *create table company (id serial, primary key(id), name varchar(50) not null, address_id bigint unsigned, foreign key(address_id) references address(id));*

- *insert into company values (null, 'Royal Bakery', 2);*
- *insert into company values (null, ‘Starite pekari', 5);*
- *insert into company values (null, 'Top Kalafche', 4);*

- *create table customer (id serial, primary key(id), first_name varchar(50) not null, last_name varchar(50) not null, address_id bigint unsigned, foreign key(address_id) references address(id));*

- *insert into customer values(null, 'Louis', 'Adeheim', 1);*
- *insert into customer values(null, 'Todor', 'Karakolev', 3);*
- *insert into customer values(null, 'Maria', 'Peneva', 6);*

MySQL
Exercise

Try it yourself

- *create table product (id serial, primary key(id), name varchar(50) not null, category varchar(50), company_id bigint unsigned, foreign key(company_id) references company(id), price decimal(15, 2) not null);*
  (assuming a common currency)

- *insert into product values(null, 'King Cookies', 'Bakery', 1, 10.99);*
- *insert into product values(null, 'King Cake', 'Bakery', 1, 18.98);*
- *insert into product values(null, 'Banitza', 'Bakery', 2, 4.00);*
- *insert into product values(null, 'Mini mekitza', 'Bakery', 2, 0.68);*
- *insert into product values(null, 'Vita banitza (XL)', 'Bakery', 2, 19.99);*
- *insert into product values(null, 'Vita banitza', 'Bakery', 2, 1.80);*
- *insert into product values(null, 'Huawei P30 lite, leather case, red', 'Smartphone accessories', 3, 29.99);*
- *insert into product values(null, 'Huawei P30 lite, leather case, blue', 'Smartphone accessories', 3, 29.99);*
- *insert into product values(null, 'Samsung Galaxy S10, silicone case, yellow', 'Smartphone accessories', 3, 20.00);*
- *insert into product values(null, 'Samsung Galaxy S10, silicone case, grey', 'Smartphone accessories', 3, 20.00);*

MySQL
Exercise

Try it yourself

- *create table purchase (id serial, primary key(id), date date, customer_id bigint unsigned, foreign key(customer_id) references customer(id));*

- *insert into purchase values(null, '2019-07-28', 1);*
- *insert into purchase values(null, '2019-08-10', (select id from customer where first_name='Louis' and last_name='Adeheim'));*
- *select * from purchase;*

- *insert into purchase values (null, '2019-06-20', 2);*
- *insert into purchase values (null, '2019-06-22', 2);*
- *insert into purchase values (null, '2019-07-15', 2);*

- *insert into purchase values (null, '2019-08-16', 3);*
- *insert into purchase values (null, '2019-08-19', 3);*

MySQL
Exercise

Try it yourself

- *create table purchased_items (purchase_id bigint unsigned not null, foreign key(purchase_id) references purchase(id), product_id bigint unsigned not null, foreign key(product_id) references product(id));*

- *insert into purchased_items values(1, 1), (1, 2);*
- *insert into purchased_items values(2, 2);*

- *insert into purchased_items values(3, 3), (3, 4), (3, 6);*
- *insert into purchased_items values(4, 8), (4, 9);*

- *insert into purchased_items values(5, 4), (5, 5);*
- *insert into purchased_items values(6, 9);*

MySQL
Exercise

Try it yourself

Retrieve product names and categories

- *select name, category from product;*

Retrieve all products whose category is ‘Bakery’

- *select * from product where category='Bakery';*

Retrieve all customers whose first name contains an ‘o’ letter

- *select * from customer where first_name like '%o%';*

Retrieve all products that are more expensive than 10

- *select * from product where price > 10;*

Retrieve first names and last names of the customers order by the last name in descending order

- *select first_name, last_name from customer order by last_name desc;*

MySQL
Exercise

Try it yourself

Retrieve the customers’ first names, last names and their countries

- *select customer.first_name, customer.last_name, address.country from customer join address on customer.address_id=address.id;*

Retrieve the different countries and number of customer who are from these countries

- *select address.country, count(*) from address join customer on customer.address_id=address.id group by address.country;*

Retrieve first names and last names of customers who are from Bulgaria

- *select first_name, last_name from customer join address on customer.address_id=address.id where address.country='Bulgaria';*
- *select first_name, last_name from customer where address_id in (select id from address where country='Bulgaria');*

MySQL
Exercise

Try it yourself

Retrieve the names of the products produced by a company located in Belgium

- *select name from product where company_id in (select id from company where address_id in (select id from address where country='Belgium'));*

Retrieve all products and the company that produces them

- *select product.name, company.name from product join company on product.company_id=company.id;*

Retrieve the first names, last names of the customers, dates of their purchases and the purchased products

- *select customer.first_name, customer.last_name, purchase.date, product.name from purchase join purchased_items on purchased_items.purchase_id=purchase.id join customer on purchase.customer_id=customer.id join product on purchased_items.product_id=product.id;*

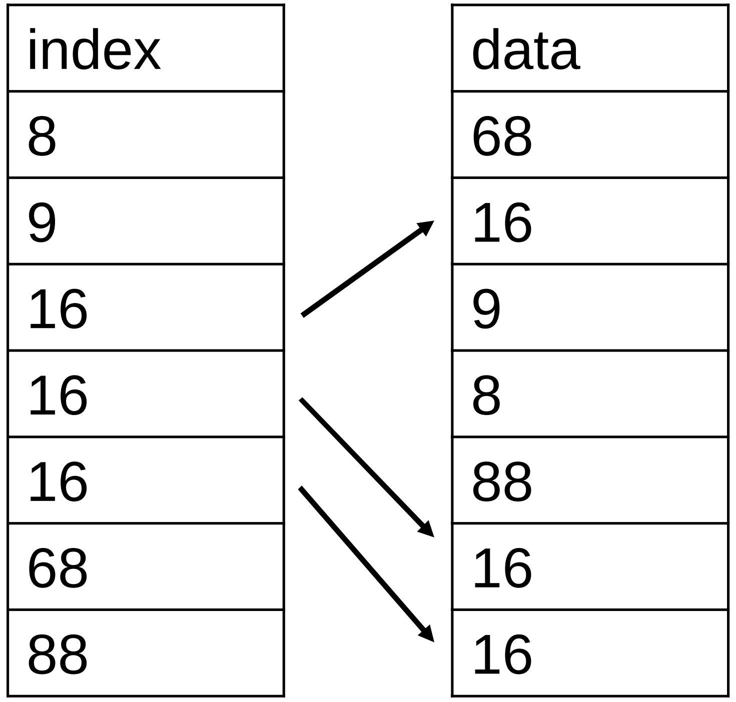
Based on the query above, create a query that retrieves the first names, last names of customers and the total sum of money they have spent in goods.

- *select customer.first_name, customer.last_name, sum(product.price) from purchased_items join purchase on purchased_items.purchase_id=purchase.id join product on purchased_items.product_id=product.id join customer on purchase.customer_id=customer.id group by customer.id;*

MySQL
Indexes

- Like a book index
- Used to make searches faster – the goal is not to scan the whole content
- Sorted
- Suitable when doing lots of data reads and far less inserts
- Better results when values are mostly unique
- Take storage

*select * from tbl where data = 16;*



MySQL

Transactions

- Sequence of SQL queries – the results can either be committed or rolled back.

- *use geeky;*
- *create table bank_account (customer_name varchar(50) not null, amount decimal(15, 2) not null check (amount>=0));*
- *insert into bank_account values('John Doe', 200), ('Jack Smith', 100);*

- *select * from bank_account;*
- *start transaction;*
- *update bank_account set amount = amount + 50 where customer_name = 'Jack Smith';* OK
- *select * from bank_account;* (in the transaction)
- *update bank_account set amount = amount - 50 where customer_name = 'John Doe';* OK
- *select * from bank_account;* (in the transaction)
- *commit;*
- *select * from bank_account;* (outside the transaction)

MySQL
Transactions

- *start transaction;*
- *update bank_account set amount = amount + 200 where customer_name = 'Jack Smith';* OK
- *select * from bank_account;* (in the transaction)
- *update bank_account set amount = amount - 200 where customer_name = 'John Doe';* Error
- *select * from bank_account;* (in the transaction)
- *rollback;*
- *select * from bank_account;* (outside the transaction)

MySQL

DDL & DML

DDL

- create database
- drop database
- create table
- alter table
- drop table

DML

- insert
- update
- delete
- select

MySQL

Procedures

- *delimiter ;;*
- *create procedure calc(in a int, inout b int, out c int)*

  -> begin

  *-> set c = a * b;*

  -> set b = a + b;

  -> select 'Hello World' as text;

  -> end;;
- *delimiter ;*

- *set @b_param=10;*
- *set @c_param=0;*

- *call calc(5, @b_param, @c_param);*
- *select @b_param;*
- *select @c_param;*

MySQL
Procedures

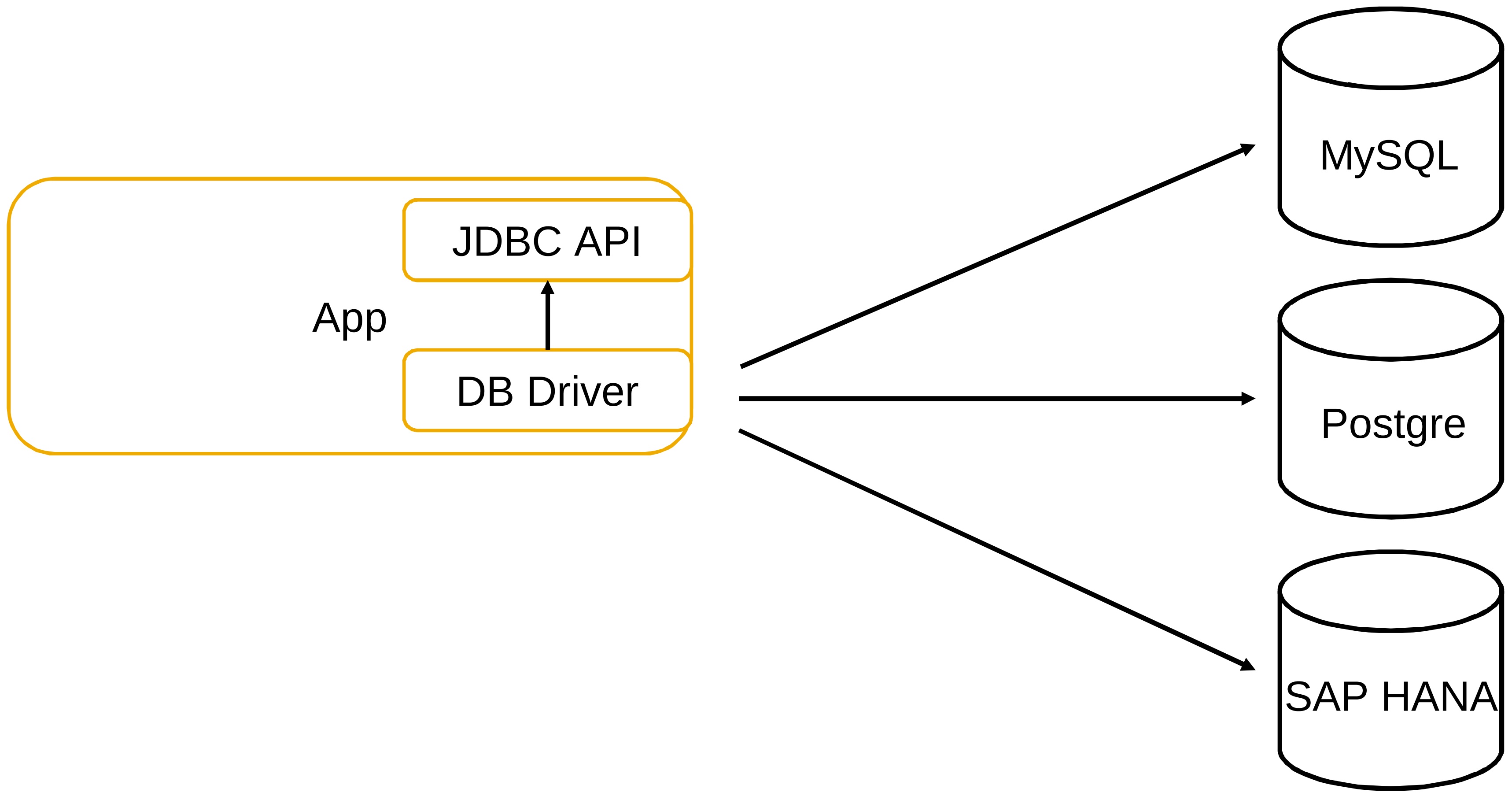
- *create table numbers (num int);*
- *insert into numbers values (1), (2), (3), (4);*
- *create table numbers_squared (num int);*
- *create table numbers_cubed (num int);*

- *delimiter ;;*
- *create procedure process_numbers ()*
  -> begin
  -> insert into numbers_squared select power(num, 2) from numbers;
  -> insert into numbers_cubed select power(num, 3) from numbers;
  -> end;;
- *delimiter ;*

- *call process_numbers();*
- *select * from numbers_squared;*
- *select * from numbers_cubed;*

- *drop procedure process_numbers;*

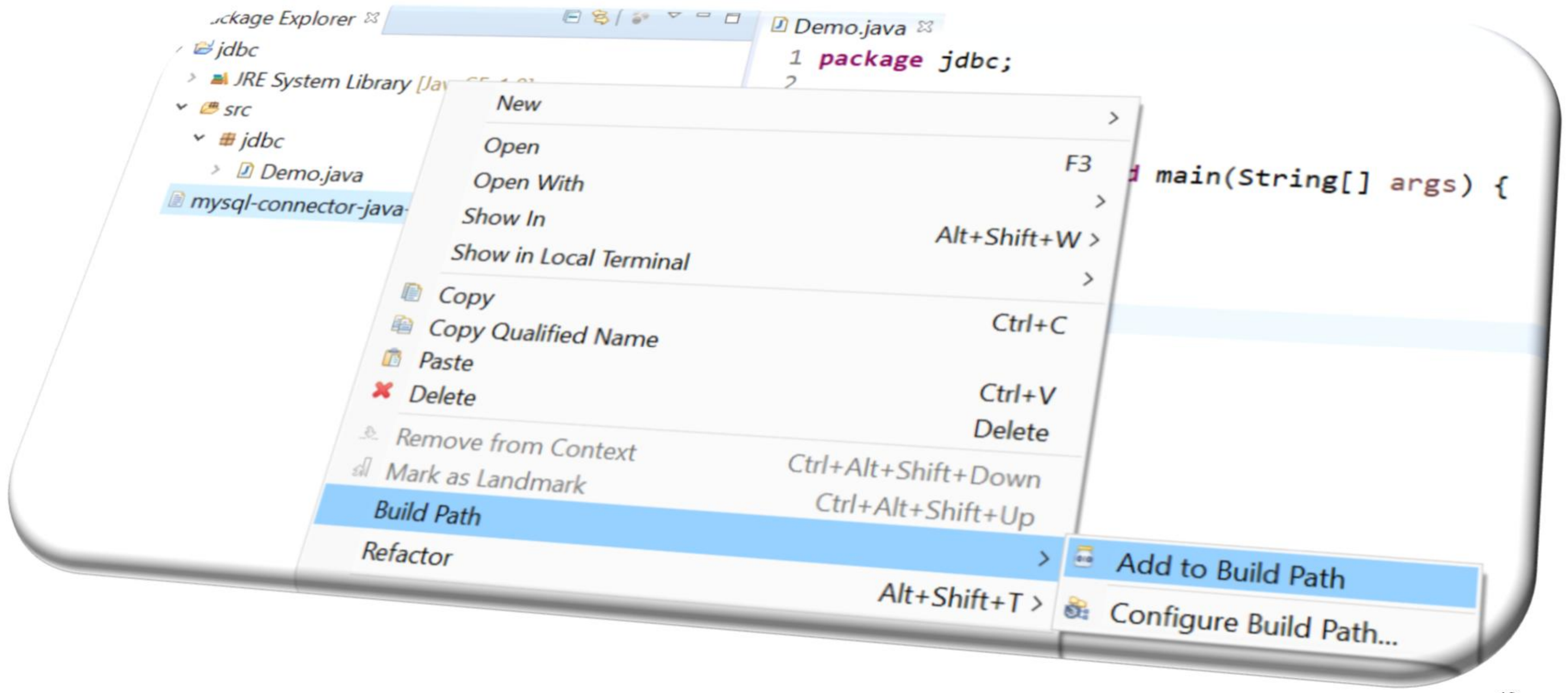
MySQL
JDBC



MySQL
JDBC

Download the MySQL driver

- https://dev.mysql.com/downloads/connector/j/
  (select Platform Independent,
  scroll down for downloading without registration (‘No thanks, just start my download’ link))

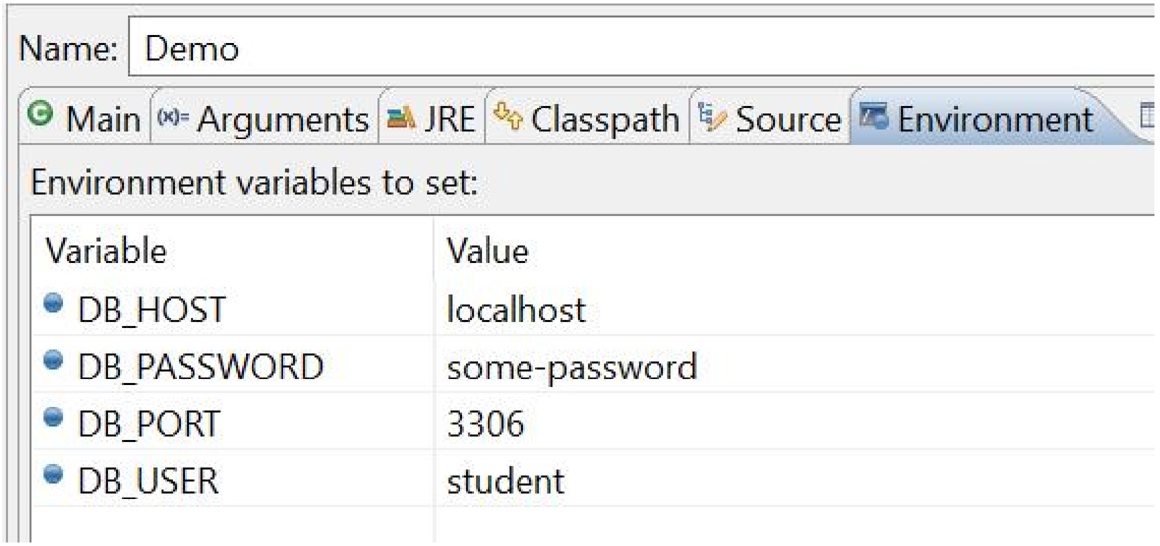


```
package jdbc;
...
main(String[] args) {
```

MySQL

JDBC

```
private static final String DB_HOST = System.getenv("DB_HOST");
private static final String DB_PORT = System.getenv("DB_PORT");
private static final String DB_USER = System.getenv("DB_USER");
private static final String DB_PASSWORD = System.getenv("DB_PASSWORD");
```



Name: Demo

Environment variables to set:

Variable	Value
DB_HOST	localhost
DB_PASSWORD	some-password
DB_PORT	3306
DB_USER	student

Try it yourself

```
private static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";

public static void main(String[] args) throws ClassNotFoundException, SQLException {
  Class.forName(JDBC_DRIVER);
}
```

MySQL

JDBC

```
private static Connection createConnection() throws SQLException {
  String connectString = "jdbc:mysql://" + DB_HOST + ':' + DB_PORT +
       "?user=" + DB_USER + "&password=" + DB_PASSWORD;

  return DriverManager.getConnection(connectString);
}
```

Try it yourself

```
private static void readData() throws SQLException {
  String query = "select name, avg(score) from geeky.exam group by name";

  try (Connection conn = createConnection();
    Statement stmt = conn.createStatement();
    ResultSet rs = stmt.executeQuery(query)) {
    while(rs.next()) {
      System.out.println("- " + rs.getString(1) + ", " + rs.getBigDecimal(2));
    }
  }
}
```

MySQL

JDBC

Careful for SQL Injections!

Try it yourself

```
private static void insecureDelete(String untrustedUserInput) throws SQLException {
  String query = "delete from geeky.exam where exam_number=" + untrustedUserInput;

  try (Connection conn = createConnection();
    Statement stmt = conn.createStatement()) {
    int affectedRows = stmt.executeUpdate(query);
    System.out.println("- Affected Rows: " + affectedRows);
  }
}


insecureDelete("1");
insecureDelete("1 or 1=1"); // deletes the content of the whole table
```

MySQL

JDBC

```
private static void secureDelete(int examNumber) throws SQLException {
  String query = "delete from geeky.exam where exam_number=?";

  try (Connection conn = createConnection();
    PreparedStatement stmt = conn.prepareStatement(query)) {
      stmt.setInt(1, examNumber);
      int affectedRows = stmt.executeUpdate();
      System.out.println("- Affected Rows: " + affectedRows);
  }
}
```

Try it yourself

```
secureDelete(1);
```

MySQL

JDBC

```
private static void callProcedure() throws SQLException {
  String query = "call geeky.calc(?, ?, ?)";

  try(Connection conn = createConnection();
    CallableStatement stmt = conn.prepareCall(query)) {
      stmt.setInt(1, 5);
      stmt.setInt(2, 10);

      try (ResultSet rs = stmt.executeQuery()) {
        System.out.println("- b: " + stmt.getInt(2));
        System.out.println("- c: " + stmt.getInt(3));
        rs.next();
        System.out.println("- text: " + rs.getString(1));
      }

  }
}
```

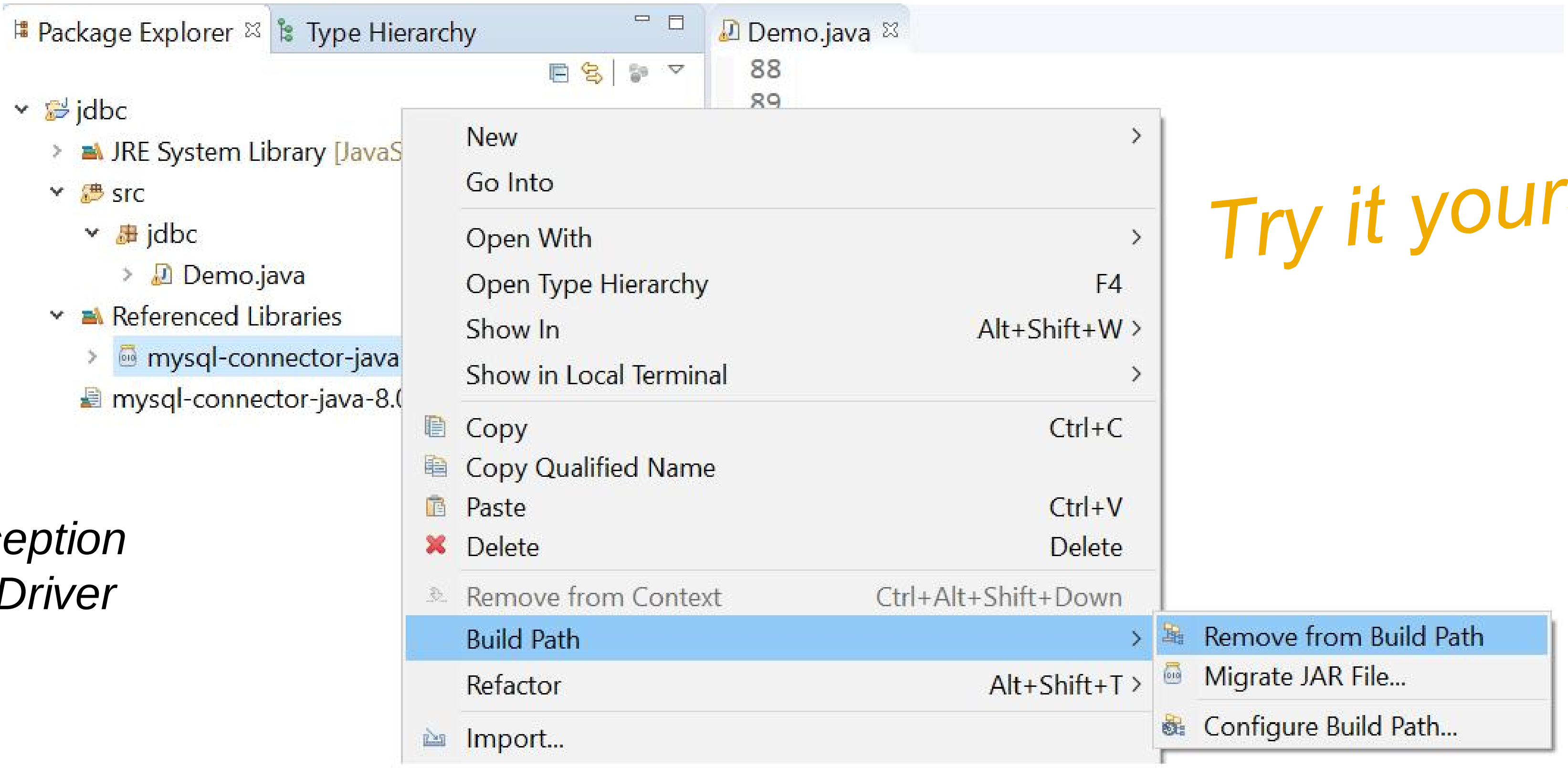
Try it yourself

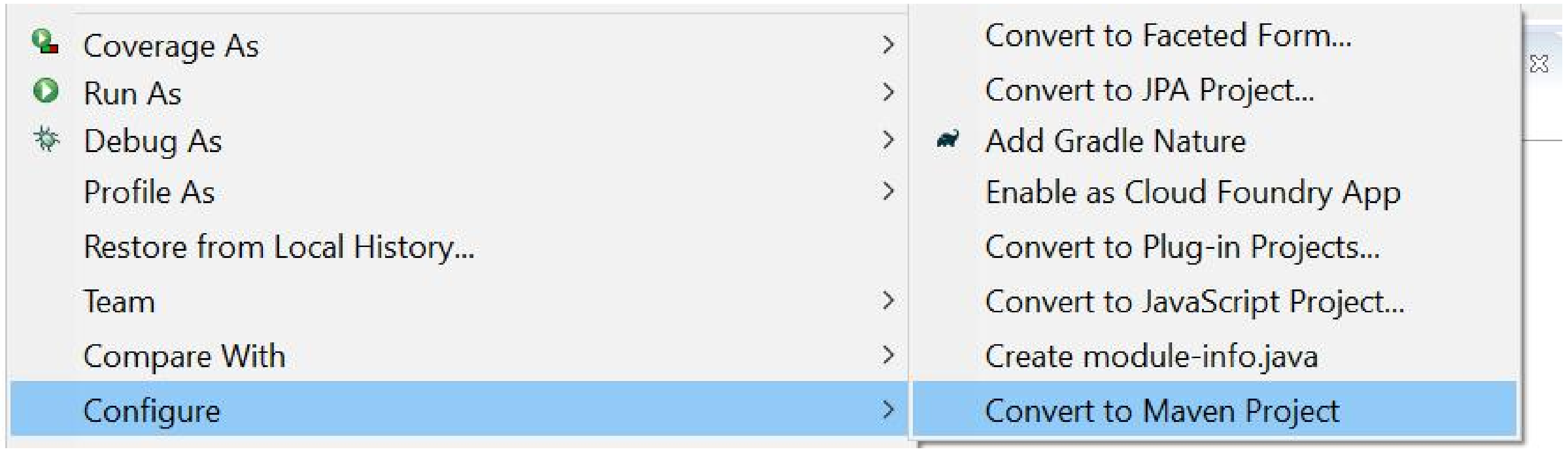
MySQL

JDBC & Maven

Try it yourself

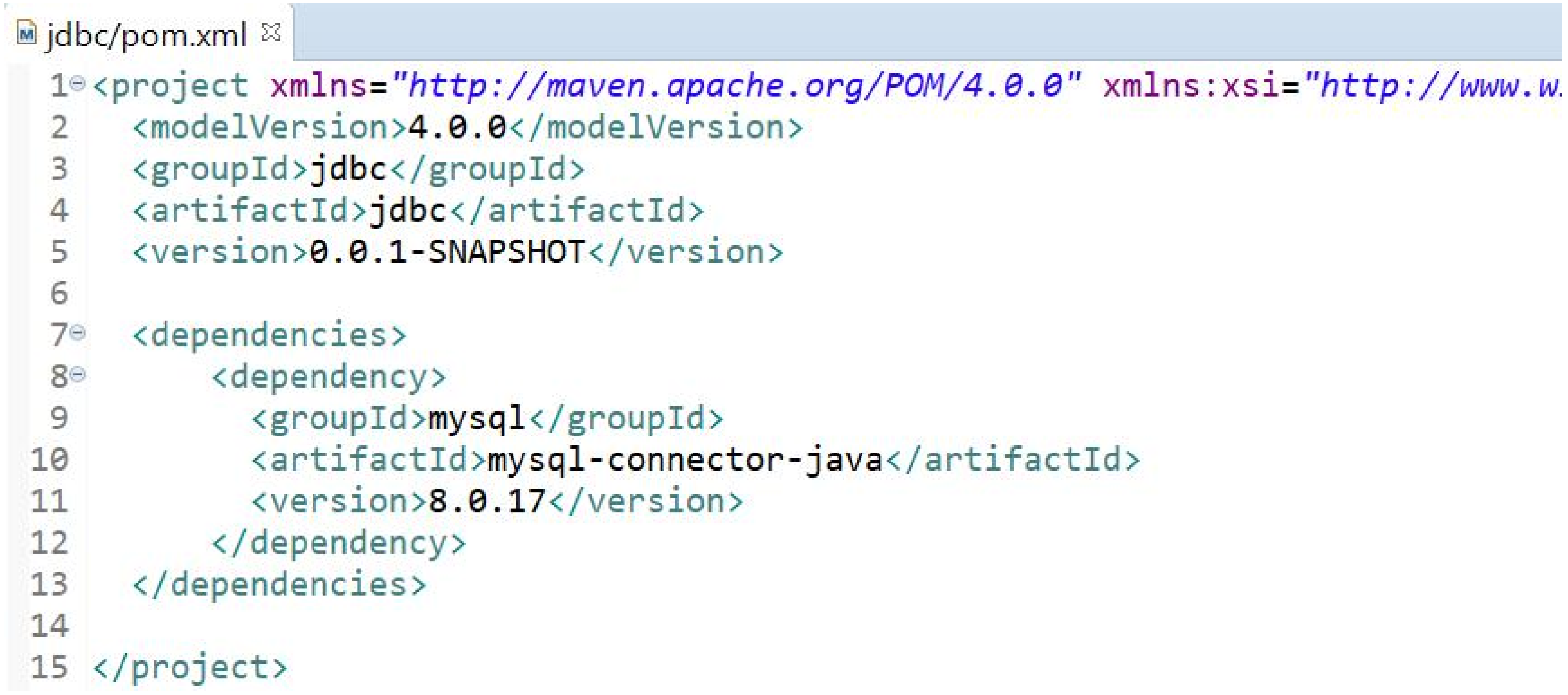
java.lang.ClassNotFoundException
com.mysql.cj.jdbc.Driver




Coverage As
Run As
Debug As
Profile As
Restore from Local History...
Team
Compare With
Configure
Convert to Faceted Form...
Convert to JPA Project...
Add Gradle Nature
Enable as Cloud Foundry App
Convert to Plug-in Projects...
Convert to JavaScript Project...
Create module-info.java
Convert to Maven Project

MySQL
JDBC & Maven

Try it yourself


jdbc/pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w
  <modelVersion>4.0.0</modelVersion>
  <groupId>jdbc</groupId>
  <artifactId>jdbc</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <dependencies>
      <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.17</version>
      </dependency>
  </dependencies>

</project>
```

Homework

Task #1 – Every GeekyCamp participant, lecturer, mentor gets a t-shirt (different sizes and colors).

- Create a table that contains the t-shirt choices (create table + inserts). **Note**: the table must not contain any names! Just type of the t-shirt (enum - male or female), color, size (enum).
- Create a query that aggregates the data in the table: retrieves type, color, size and count of t-shirts that are of the same (type, color, size). Order the results by type, color and size. Include the result set (as you see it in the MySQL CLI).

Task #2 – Create a database for high school students IT competitions. Each student has first name, last name, city. At least 6 students should be entered into the database. A student may be assigned to a single project, but a project can have multiple students assigned to it. A project has a name and a description. At least 4 projects should be available in the database. A competition has a name and a city in which is being held. At least 3 competition should be entered in the database. The database should be able to store a relation between a competition, project and the award that has received (‘gold’, ‘silver’, ‘bronze’ or null if a project has not been awarded in the competition). Every project has been part of each competition. Create the following queries (on the next slide):

Homework

- Retrieve students from a particular city
- Retrieve students whose last name end with letter ‘a’
- Retrieve all students (first name and last name) and the names of their projects
- Retrieve the first and last names of all students who work on a project with a given name
- Retrieve competition name, project name and the award that has been given (a project that has not been awarded should not be present in the result)