



UNIVERSITÀ
DEGLI STUDI
DI UDINE



Centro Internazionale
di Scienze Meccaniche
*International Centre
for Mechanical Sciences*

ISD
Introduction
to Spatial Database

CISM-UniUD Summer School
coordinated by

Anna Frangipane
University of Udine
Italy



Udine August 27 - September 1 2018

SQL: Data Definition and Manipulation

Marco BASALDELLA, Ph.D.

Department of Mathematics, Computer Science and Physics
University of Udine
marco.basaldella@uniud.it | www.basaldella.it



Outline

1. Data Definition Language
2. Data Manipulation Language
3. Exercises

All the material is available online:

<https://github.com/basaldella/isd2018>



MySQL Workbench/Recap

We will use MySQL Workbench as our tool of choice for working with MySQL.

You will work with a database installed in the Artificial Intelligence Laboratory servers. The credentials are:

- hostname: `db.ailab.uniud.it`
- username: `isdXX`, where XX is the number of your PC
- password: `ISD2018`



Data Definition Language



Data Definition Language

Instruction set used to

- Create Tables
- Modify Tables
- Delete Tables
- Enforcing constraints



Table creation

```
CREATE TABLE Table_Name (  
    Field_1 DATATYPE FIELD_ATTRIBUTES,  
    Field_2 DATATYPE FIELD_ATTRIBUTES,  
    ...  
    TABLE_ATTRIBUTES  
);
```



Table creation/2

```
CREATE TABLE Students (  
    ID int NOT NULL  
    AUTO_INCREMENT,  
    LastName varchar(255),  
    FirstName varchar(255),  
    Address varchar(255),  
    Country varchar(255),  
    PRIMARY KEY (ID)  
);
```

We create a table...

- called “Students”,
- with the columns “ID”, “LastName”, “FirstName”, “Address”, “Country”,
- all the columns but “ID” will contain text
- “ID” will be numeric, it **must** be always assigned, and it’s automatically increased each time we add a row
- “ID” will be the primary key

Foreign Keys

Suppose that we need a table to keep track the receipts of the students attending a summer school. We could build something like that:

ID	student	amount	notes
1	3	300.00	IBAN: 123456
2	5	300.00	Paid in cash
3	1	0	Sponsored by uni
...	

The column “**student**” is a **foreign key** that references the previous table



Foreign keys/2

```
CREATE TABLE Receipts(  
    ID int NOT NULL AUTO_INCREMENT,  
    Student int,  
    Amount float,  
    Notes varchar(255),  
    PRIMARY KEY (ID),  
    FOREIGN KEY (Student) REFERENCES Students(ID)  
);
```



Table creation: key syntax

Primary key on column “Column_A”:

```
PRIMARY KEY (Column_A)
```

Multiple keys: `PRIMARY KEY (Column_A, Column_B, ...)`

Foreign key where “Column_A” references “Column_B” on “Table_B”:

```
FOREIGN KEY (Column_A) REFERENCES  
Table_B(Column_B)
```



SQL Data types

You may have noticed that each column is assigned a **type** (the green part after the column name).

Example:

- Amount float
- Notes varchar(255)

The **type** of a column determines what kind of data can we put inside it.



SQL Data types

Example:

- 255 is a number.
- 1.23 is a number.
- 'Hello' is a string.
- True is a Boolean value.
- '255' is a...?



SQL Data types

We usually put strings between quotes to avoid confusion!

- **255** is **different** from **'255'**
- **True** is **different** from **'True'**

MySQL has many, many data types. We will consider only a small subset of them, which should suffice for our purposes.

SQL Data types: reference

Type	Name	Description	Example
String	VARCHAR(length) VARCHAR(255)	Stores a string with length up to the value specified. Usually we set length=255.	john.doe New York
Numeric	INT, INTEGER FLOAT, DOUBLE, REAL	Store integers or floating point numbers	12 3.1415
Date	DATE, TIME DATETIME, TIMESTAMP	Store dates, times, or date <i>and</i> time (with precision up to a second).	01-01-2018 01-01-2018 00:00:00
Boolean	BOOLEAN	Boolean values	TRUE, FALSE

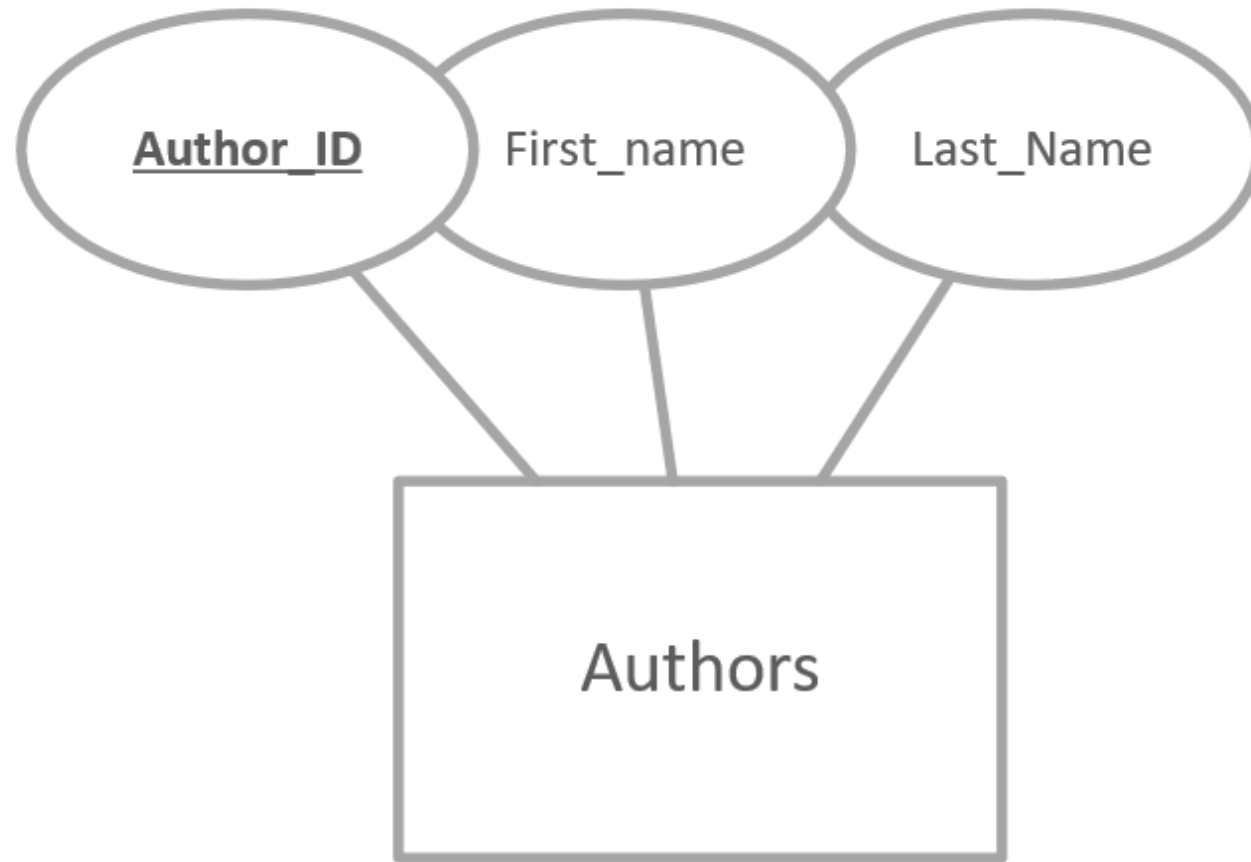


Field Properties

- **NOT NULL**: the field must always have a value
Example: first name must be not null, middle name may be optional
- **UNIQUE**: the column won't contain duplicate values
- **DEFAULT value**: set the default value for a column, if it's not set by the user
Example: `item_price float DEFAULT 0.0`
- **AUTO_INCREMENT**: usually used with keys, ensures that the value in the column is unique, and generates automatically new values



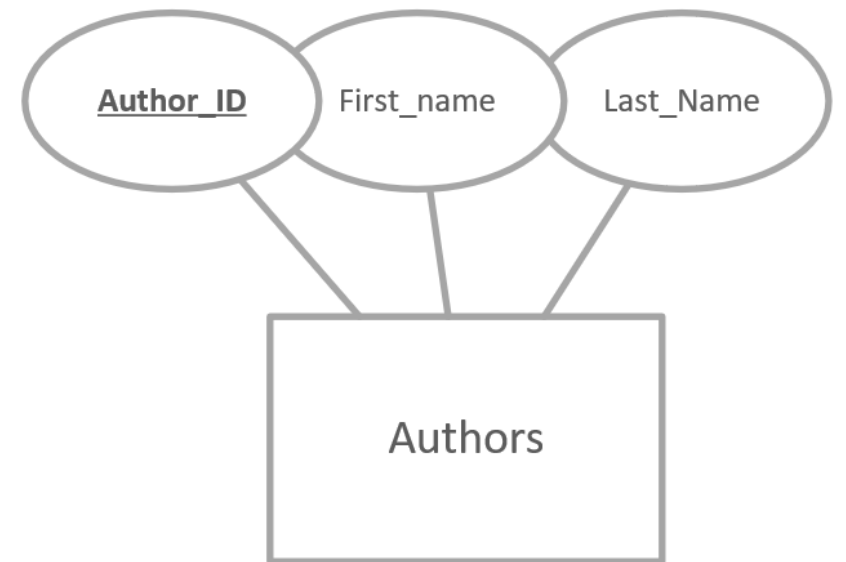
Example: the table **Authors**





Code

```
CREATE TABLE Authors(  
  
);
```



Code

```
CREATE TABLE Authors(  
    author_id INT AUTO_INCREMENT,  
  
);
```



Code

```
CREATE TABLE Authors(  
  author_id INT AUTO_INCREMENT,  
  first_name VARCHAR(255),  
  
  );
```



Code

```
CREATE TABLE Authors(  
  author_id INT AUTO_INCREMENT,  
  first_name VARCHAR(255),  
  last_name VARCHAR(255),  
  
  );
```



Code

```
CREATE TABLE Authors(  
  author_id INT AUTO_INCREMENT,  
  first_name VARCHAR(255),  
  last_name VARCHAR(255),  
  PRIMARY KEY (author_id)  
);
```





Modify Table

The instruction `ALTER TABLE` can be used to modify tables.

- `ALTER TABLE table_name
ADD column_name type properties;`
- `ALTER TABLE table_name
DROP COLUMN column_name;`
- `ALTER TABLE table_name
MODIFY column_name type properties;`
- `ALTER TABLE table_name
ADD CONSTRAINT table_constraint`



Delete a Table

The instruction `DROP TABLE` can be used to delete tables.

- Drop a table:
`DROP TABLE` `table_name`
- Drop the table, but first check if exists:
`DROP TABLE IF EXISTS` `table_name`

WARNING: dropping a table is NOT reversible



Data Manipulation Language



Data Manipulation Language

Instruction set used to

- Insert
- Update
- Remove

data from tables.

Note that we may want to update or remove only a subset of a table, so what instruction may come in handy..?



Insert data/1

We use the **INSERT INTO** instruction to insert data into a table. We can use it in two ways:

1. We can specify the columns we want to set; or
2. We can add data for all the columns.

Note that

- We must always set columns that are marked **NOT NULL**
- We don't have to set columns that are marked **AUTO_INCREMENT** when we use the first syntax.

Insert data/2

```
INSERT INTO table_name (field_1,field_2,...)
VALUES (val_1,val_2,...);
```

Example:

```
INSERT INTO Students (LastName,FirstName) VALUES
("Doe", "John");
```

Result:

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL

Insert data/2

```
INSERT INTO Students (LastName, FirstName) VALUES  
("Doe", "John");
```

Result:

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL

Note the ID, automatically set at 1, and the Address and Country automatically set as **NULL**

Insert data/3

```
INSERT INTO table_name  
VALUES (val_1, val_2, ...);
```

The order of the values must match the order of the columns.
Example:

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL
	100	Other	Anne	New York	USA

Insert data/3

```
INSERT INTO table_name  
VALUES (val_1, val_2, ...);
```

Example:

```
INSERT INTO Students  
VALUES (100, "Other", "Anne", "New York", "USA");
```

Result:

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL
	100	Other	Anne	New York	USA



Insert Data/4

Multiple insertions: just concatenate the values with a comma.

Examples:

```
INSERT INTO Students VALUES
```

```
(200, 'Rossi', 'Mario', 'Milano', 'Italia'),  
(201, 'Bianchi', 'Rosa', 'Udine', 'Italia')
```

```
INSERT INTO Students (LastName, FirstName) VALUES  
( 'Toulemonde', 'Pierre'),  
( 'Mustermann ', 'Max')
```

Insert Data/4

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL
	100	Other	Anne	New York	USA
	200	Rossi	Mario	Milano	Italia
	201	Bianchi	Rosa	Udine	Italia
	202	Toulemonde	Pierre	NULL	NULL
	203	Mustermann	Max	NULL	NULL

Insert Data/4

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL
	100	Other	Anne	New York	USA
	200	Rossi	Mario	Milano	Italia
	201	Bianchi	Rosa	Udine	Italia
	202	Toulemonde	Pierre	NULL	NULL
	203	Mustermann	Max	NULL	NULL



Update data/1

We use the **UPDATE** instruction to modify the rows of a table.

- To select the rows we want to update, we use the **WHERE** clause – much like in a **SELECT**

- Syntax:

```
UPDATE table_name  
SET field_1=val_1, field_2=val_2, ...  
WHERE field=val;
```

Update Data/Example

	ID	LastName	FirstName	Address	Country
▶	1	Doe	John	NULL	NULL
	100	Other	Anne	New York	USA
	200	Rossi	Mario	Milano	Italia
	201	Bianchi	Rosa	Udine	Italia
	202	Toulemonde	Pierre	NULL	NULL
	203	Mustermann	Max	NULL	NULL

Update Data/Example

Example:

- **UPDATE** Students
SET Address='Roma'
WHERE Country='Italia';

Result:

ID	LastName	FirstName	Address	Country
1	Doe	John	NULL	NULL
100	Other	Anne	New York	USA
200	Rossi	Mario	Roma	Italia
201	Bianchi	Rosa	Roma	Italia
202	Toulemonde	Pierre	NULL	NULL
203	Mustermann	Max	NULL	NULL



Delete Data/1

We use the **DELETE** instruction to delete the rows of a table.

- To select the rows we want to delete, we use the **WHERE** clause – much like in a **SELECT** and exactly as in a **UPDATE**
- Syntax:
DELETE FROM `table_name`
WHERE `field=val`;

Delete Data/Example

ID	LastName	FirstName	Address	Country
1	Doe	John	NULL	NULL
100	Other	Anne	New York	USA
200	Rossi	Mario	Roma	Italia
201	Bianchi	Rosa	Roma	Italia
202	Toulemonde	Pierre	NULL	NULL
203	Mustermann	Max	NULL	NULL

Delete Data/Example

Example:

- **DELETE FROM** Students
WHERE ?=?;

Result:

ID	LastName	FirstName	Address	Country
1	Doe	John	NULL	NULL
100	Other	Anne	New York	USA
202	Toulemonde	Pierre	NULL	NULL
203	Mustermann	Max	NULL	NULL

Delete Data/Example

Example:

- **DELETE FROM** Students
WHERE Country='Italia';

Result:

ID	LastName	FirstName	Address	Country
1	Doe	John	NULL	NULL
100	Other	Anne	New York	USA
202	Toulemonde	Pierre	NULL	NULL
203	Mustermann	Max	NULL	NULL



Recap

Now we have all the tools for

- Creating tables, setting primary and foreign keys;
- Inserting rows into tables;
- Updating and deleting rows;
- Searching into tables (from yesterday's lecture!)



Exercises



Today's Database: Pizzerias

Today **you** will create the **Pizzerias** database, slightly modified from yesterday's afternoon lecture.

You will have to:

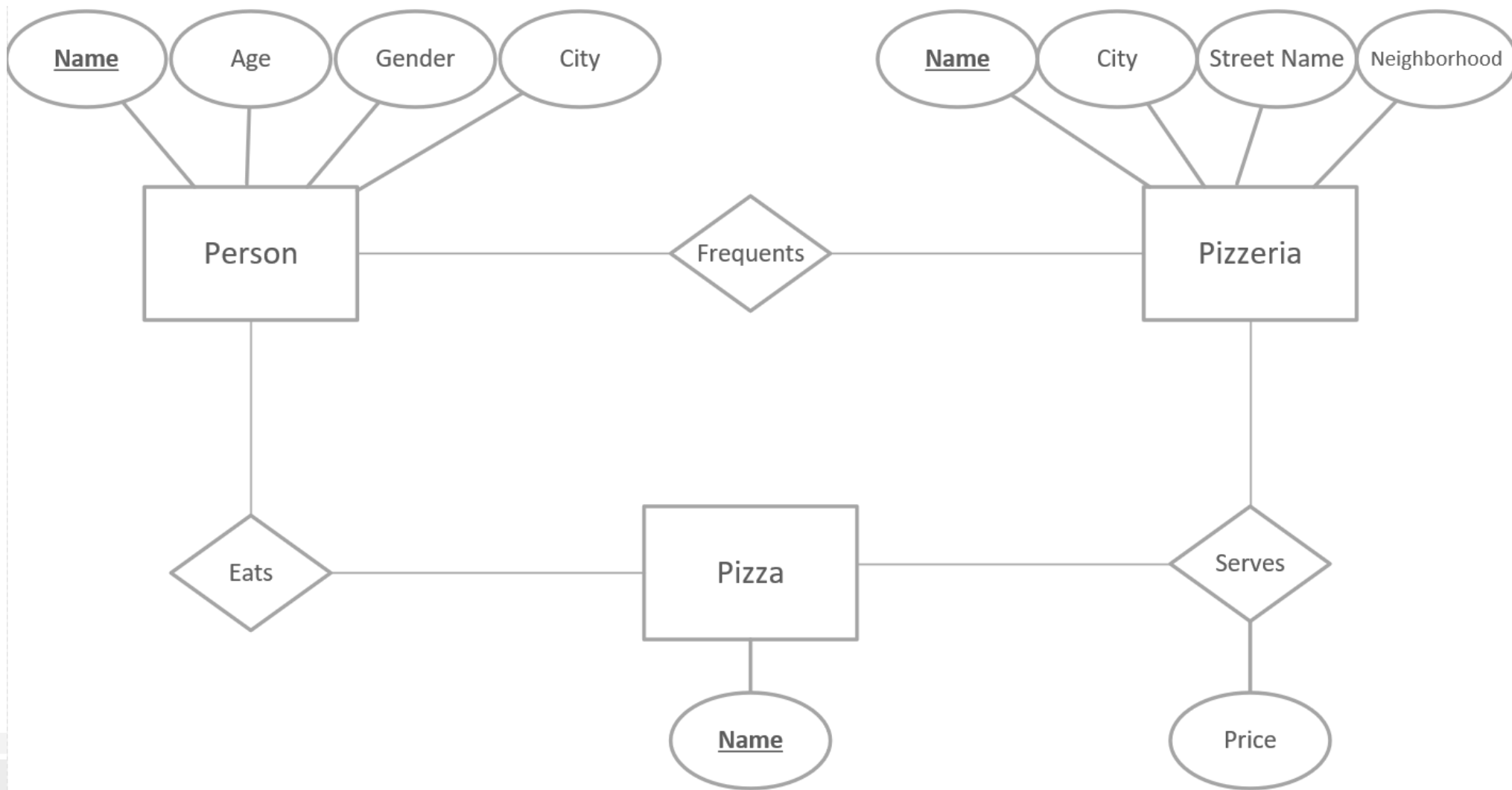
- Draw the ER diagram
- CREATE the tables
- I will provide you the data 😊
- UPDATE the data!



Pizzerias

- A **pizzeria** has a name, and is located in a street, in a neighborhood, of a city.
- A **pizzeria** serves many **pizzas** at different prices.
- A **person** has a name, a gender, an age, and lives in a city.
- A person **frequents** one or more pizzerias.
- A person **likes** one more pizzas.

ER





Relational Schema

- **Person** (name, age, gender, city)
- **Pizzeria** (name, city, street_name, neighborhood)
- **Pizza** (name)
- **Frequents** (person, pizzeria)
- **Eats** (person, pizza)
- **Serves** (pizzeria, pizza, price)



Create the database

Use **CREATE TABLE** to create the tables.

- Set the data types.
Example: a pizza price is probably not an integer, so you should use which data type?...
- Set the foreign keys.
- Set the keys.
Note: some tables have **multiple keys**.
- Hint: begin your script with
DROP TABLE IF EXISTS `table_name`



Populate the database

- Go on the repository, find and run the code
 - Repository: <https://github.com/basaldella/isd2018>
 - File: **Day3/Pizzerias.sql**
- This file will create the tables and populate them with some data.



Exercises

- Move all the pizzerias in Milan to Turin.
- Convert the price from Euro to Iraqi Dinars
1 EUR = 1.395,32 IQD
- Try to **INSERT** a pizzeria who serves 'Pepperoni Pizza'.
What happens? Why?
(hint: MySQL knows that Pepperoni is not a thing 😊)
- Convert the gender of the persons from 'male'/'female' to
'M'/'F'

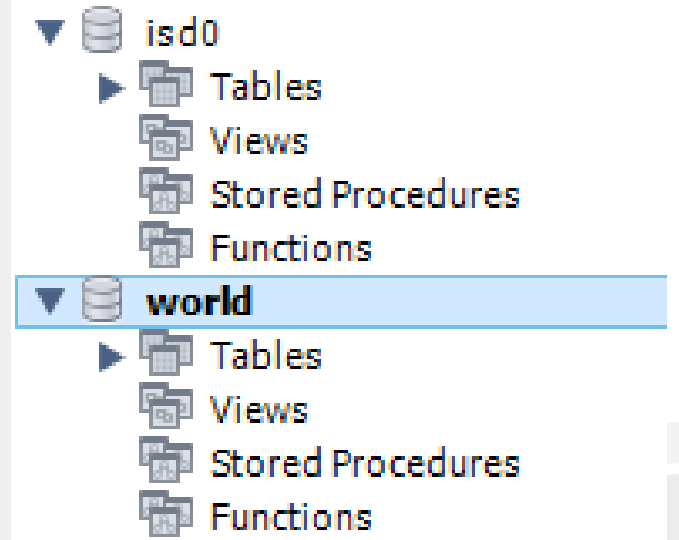


More Queries!

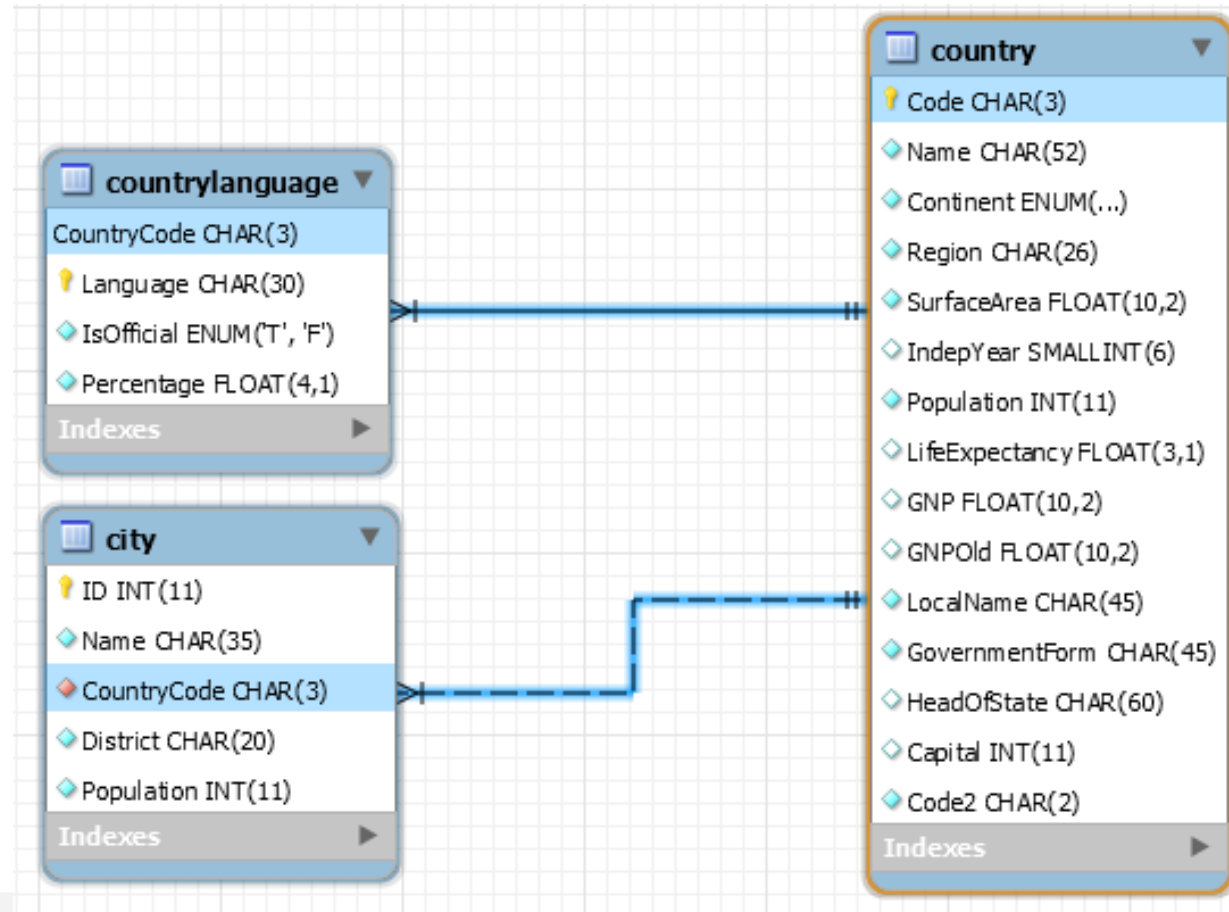
1. Find all the pizzerias in 'San Marco' neighborhood in Venice
2. Find the youngest person in the database
3. Find the least expensive pizzeria
4. Find the pizzerias which serve 'Marinara' in Milano

Even More Queries!

- **World database** provided by MySQL
- **Source:** <https://dev.mysql.com/doc/world-setup/en/>
- **Data by:** Statistics Finland, <http://www.stat.fi/worldinfigures>
- Data pre-loaded for you: double click on “world” under your own database 😊



Database diagram





Query the world

1. Find all the countries with no population, and sort them by surface area (biggest first!)
2. Find all the country codes of the Arabic-speaking countries
3. Find the most populated Arabic-speaking country
Hint: you can limit the result to 1 row using **LIMIT 1** as last instruction
4. Find the government form of the least populated French speaking country
5. Count the capital cities with at least 1 million inhabitants