Tomas Sandnes
tomas.sandnes@kristiania.no

Week 1

DB1102 / PGR 111 – DATABASES

Today's topics

(Today's chapters: 1 in Norwegian Book, 1-2 in English book)

- Introduction
  - Lecturer: Tomas Sandnes
  - This course

- Databases in general
  - Installing MySQL
  - SQL statement basics

Introduction

About me: Tomas Sandnes

- Contractor at Kristiania University College (School of Economics, Innovation and Technology – SEIT). The stuff that I currently teach:
  - Databases, Software design with C#, Content Management Systems, Game AI & Unity development.

- Qualifications & background:
  - Holds an engineering degree in Computer Science.
  - Have worked with all parts related software development. (And has made software for several business types: stock trading, online marketing solutions, computer games, car park solutions, …)
  - Have been teaching part-time at Kristiania's department of technology (formerly NITH) since 2007. Some topics I have taught previously: Artificial Intelligence / Machine Learning, C++ programming, PHP programming, JavaScript programming, Game engine programming, Computer Game design, ++
  - *I love computer gaming, btw! ;-)*

- I also run my own (very small) IT consultant company: Commodore Consulting AS.

But hold on: WELCOME to Kristiania! :-)

- This is your first lesson with me!
- WELCOME to us! :-D

About this course: structure

- We have lessons over 12 weeks, once a week.

- Mondays: Shared course for 5 Bachelors:
  - Lectures are at 11:15 to 13:00. Exercises are at 13:15 to 15:00.
  - Data Science, E-business, Frontend- and Mobile Development, Interactive Design, Artificial Intelligence.

- Wednesdays: Shared course for 3 Bachelors:
  - Lectures are at 12:15 to 14:00. Exercises are at 14:15 to 16:00.
  - Cyber Security, Programming, Game Technology.

Learning Databases

- It takes more than 4 hours a week (the time you have with me) to learn Databases!
  - As a rule of thumb, match every hour with a teacher with the same amount of hours on your own.
  - In other words, aim for 8 hours total per week for Databases. (4 with me, 4 on your own.)

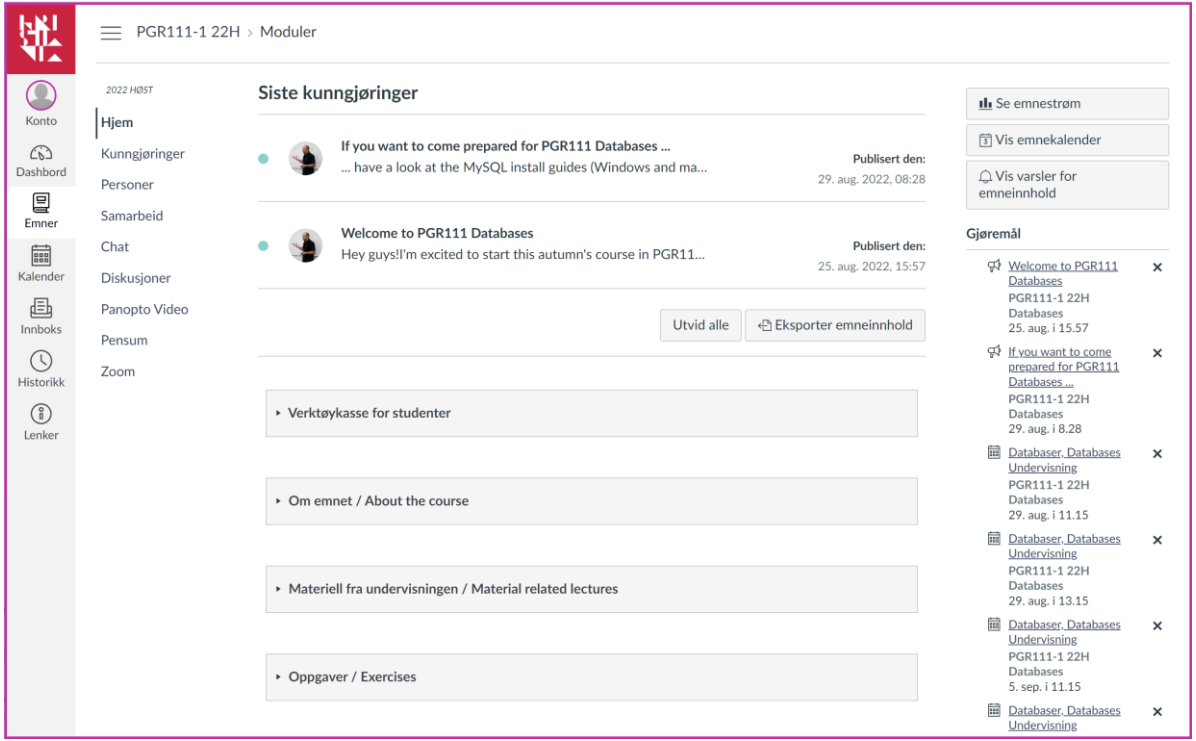
- My suggestion:
  - Start a study-group with some of your fellow students! :-)
  - Agree on some weekdays and timeslots to meet and work together.
  - *For example:* Work 2 hours before the lecture + 2 hours on Fridays.

- *And remember:* Invite others to join you. Take care of each other! :-)

Canvas – our learning platform

- See Canvas for:
  - Subject description
  - Lecturing material
  - Exercises
  - Exam Q&A
  - External resources

- Examination:
  - Graded Pass / Fail



Lesson structure

DB1102 / PGR111 lessons

Lesson	Monday lecture	Wednesday lecture	Topics	Norwegian textbook	English textbook
1	29.aug	31.aug	Introduction	Ch 1	Ch 1 - 2
2	05.sep	07.sep	Basic SQL	Ch 2	Ch 3
3	12.sep	14.sep	Create and use tables	Ch 3	Ch 3.2 & 4
4	19.sep	21.sep	JOIN	Ch 4	Ch 4.1
5	26.sep	28.sep	VIEW & subquery	Ch 5.1 - 5.3	Ch 4.2 & 3.8
6	03.okt	05.okt	Sumup, lesson 1-5	Sum-up, no new chapters	Sum-up, no ne
7	10.okt	12.okt	ER modelling, part 1	*For 2 lessons:* Ch 7 - 8.1	
8	17.okt	19.okt	ER modelling, part 2	*For 2 lessons:* Ch 7 -	
9	24.okt	26.okt	Normalization: UNF - 3NF	Ch 8.2.1 - 8.2.6	
10	31.okt	02.nov	Normalization: UNF - BCNF	The rest of Ch 8.2	*For 2 lessons:* Ch 7
11	07.nov	09.nov	Misc smaller topics	Ch 9.3, 10.2.1, 10.3.2, 11.2.2	Ch 4.6 (+ 14), 4.3, 17, 1.8
12	14.nov	16.nov	Course sum-up, focus on lesson 7-11	Sum-up, no new chapters	Sum-up, no new chapters

Mondays: **Data Science, E-business, Frontend- and Mobile Development, Interactive Design, Artificial Intelligence**

Wednesdays: **Cyber Security, Programming, Game Technology**

Note, this is also on Canvas:
About the course -> Lesson structure

How to participate in my lectures

- A few things I want to mention, so we all can have a good experience:

- When you meet up for a lecture, pay attention!
  - Don't check YouTube, Facebook or do other non-relevant PC-activity. (It's of course ok in the breaks.)
  - Do NOT sleep in class!

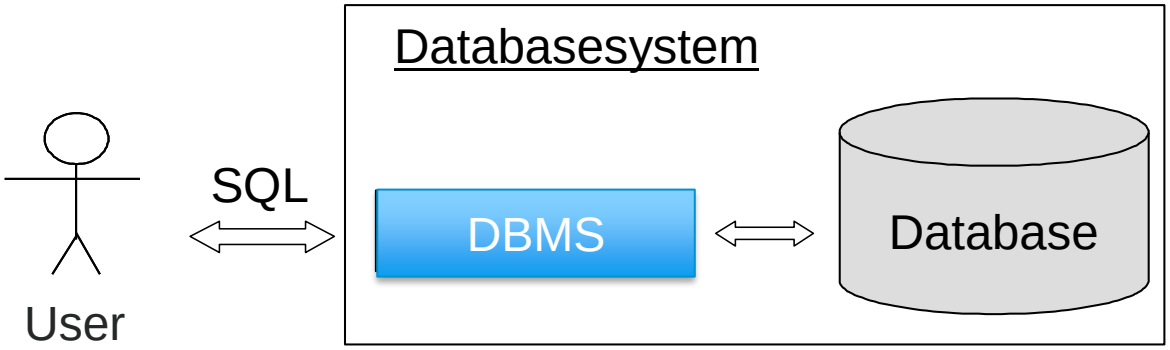
- *Why these rules?*
  - Out of respect for your fellow students!
  - (And of respect for me, but most importantly: for your fellow students!)

Databases in general

Database wording

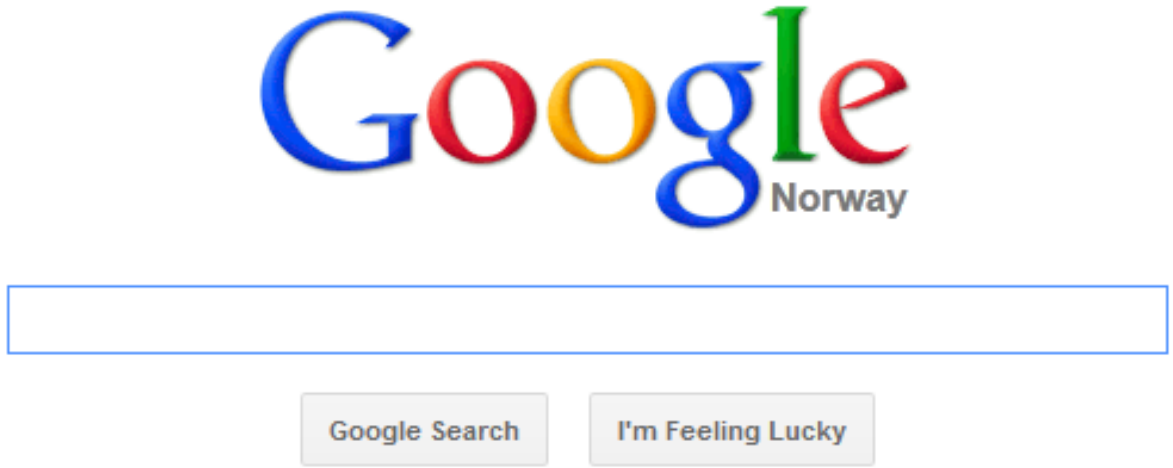
- Database system = Database + DBMS
- A Database is a structured collection of data.
- A DBMS – DataBase Management System – is a tool to store and retrieve large amounts of data over a long period of time, in a secure and efficient manner, possibly accessed by several users simultaneously.

Database wording – continued



- SQL (Structured Query Language) is a language to communicate with relational databases. (A common type of database, more on that later.)

"Everybody" uses databases!



Some facts regarding databases

- First databases appeared in the mid-1960s.
- Relational databases, a widely used DB of our time, are based on theories developed by Codd, around 1969/1970.
  - Codd, E. F.: "A Relational Model of Data for Large Shared Data Banks", in Communications of the ACM, 1970.
- Lately some new types of databases have appeared:
  - object databases, graph databases, and document databases.

Some facts regarding databases – cont.

- Databases that are not relational databases are often referred to by the general term NoSQL databases.
  - NoSQL => "Non SQL" or "Not Only SQL".

- We will primarily look at relational databases in this course:
  - MySQL, PostgreSQL, SQL Server, Oracle.

- MySQL is, according to themselves, the most used DB in the world.

Description of a relational database

- A relational database ("RDBMS") is a type of database that stores and provides access to data points that are related to one another.
- Relational databases are based on the relational model.
  - An intuitive, straightforward way of representing data in tables.
- In a relational database, each row in the table is a record with a unique ID called the key.
  - The columns of the table hold attributes of the data.
  - Each record usually has a value for each attribute, making it easy to establish the relationships among data points.

Source: www.oracle.com/database/what-is-a-relational-database/

Description of a relational database – cont.

- A relation is a table, consisting of rows and columns.
  - (Usually it's referred to as a table rather than a relation.)

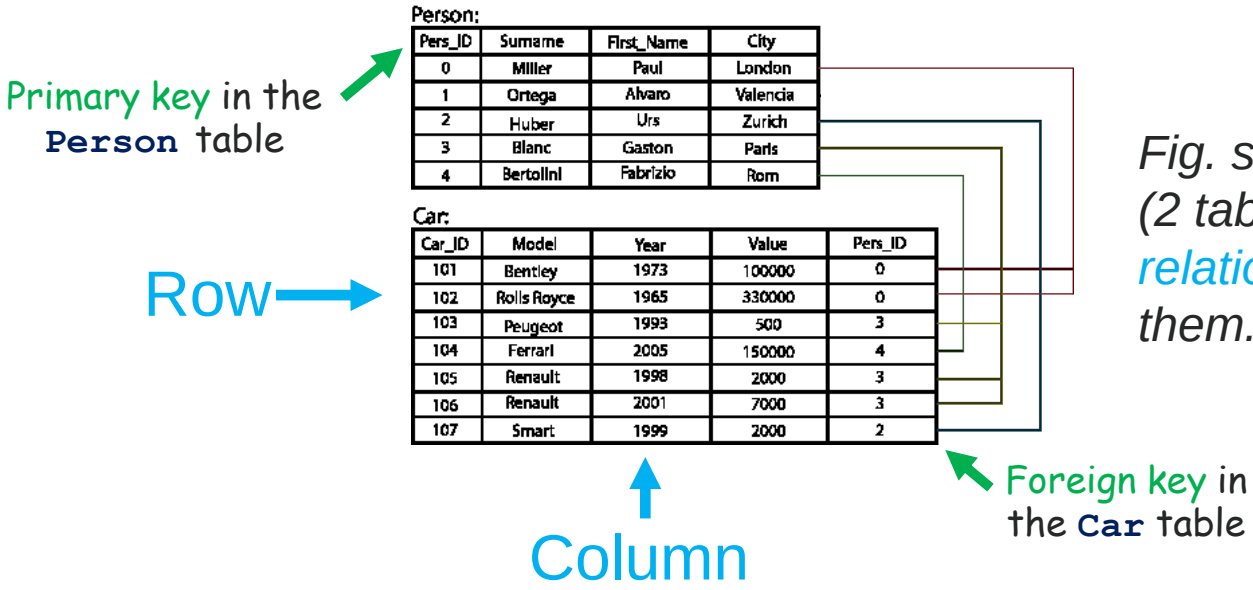


Fig. showing 2 relations (2 tables), and the relationship between them.

About MySQL

- MySQL is a relational database.
  - Open-source, under the GNU license.

- Owned by Oracle.
  - Originally owned by MySQL AB in Sweden.
  - Created by Michael Widenius.
  - Named after the daughter of its creator: My

- MySQL runs on Windows, macOS, and Linux.

Source: Wikipedia

About MySQL – cont.

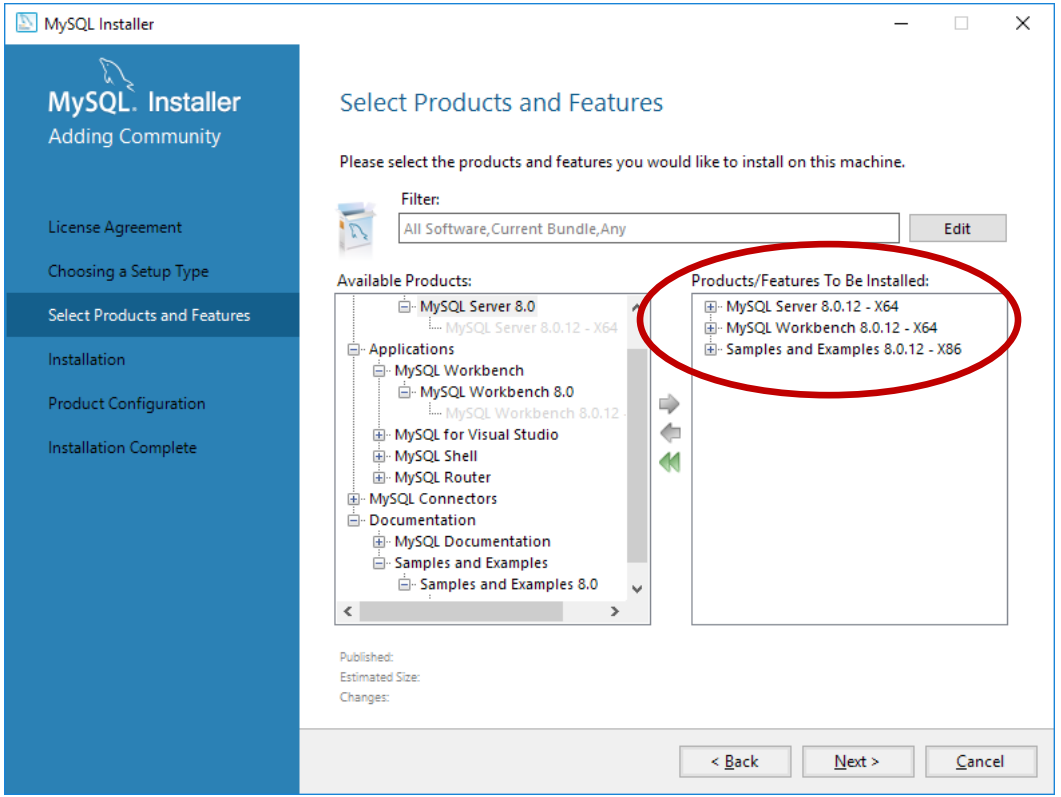
- MySQL is used by:

WIKIPEDIA
The Free Encyclopedia

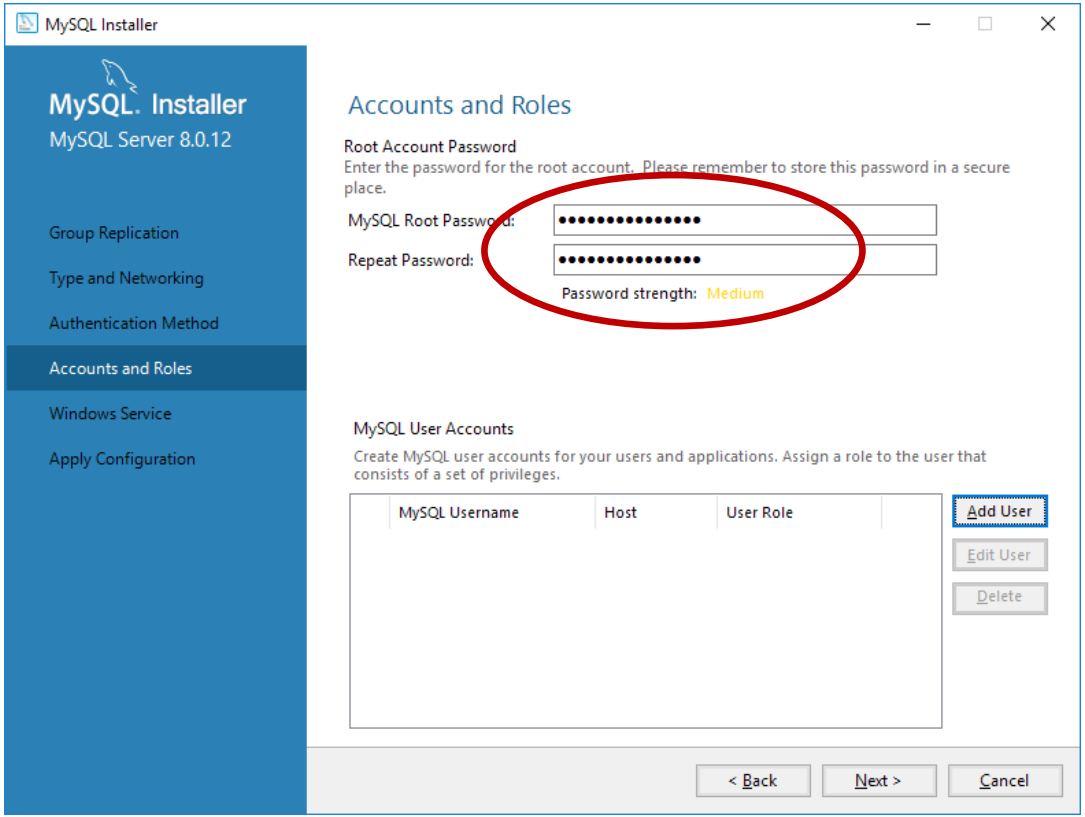
Windows installer, select these 3 elements:

- For Windows there are links to install guide + installation software on Canvas.
  - In "Eksterne ressurser / External resources".

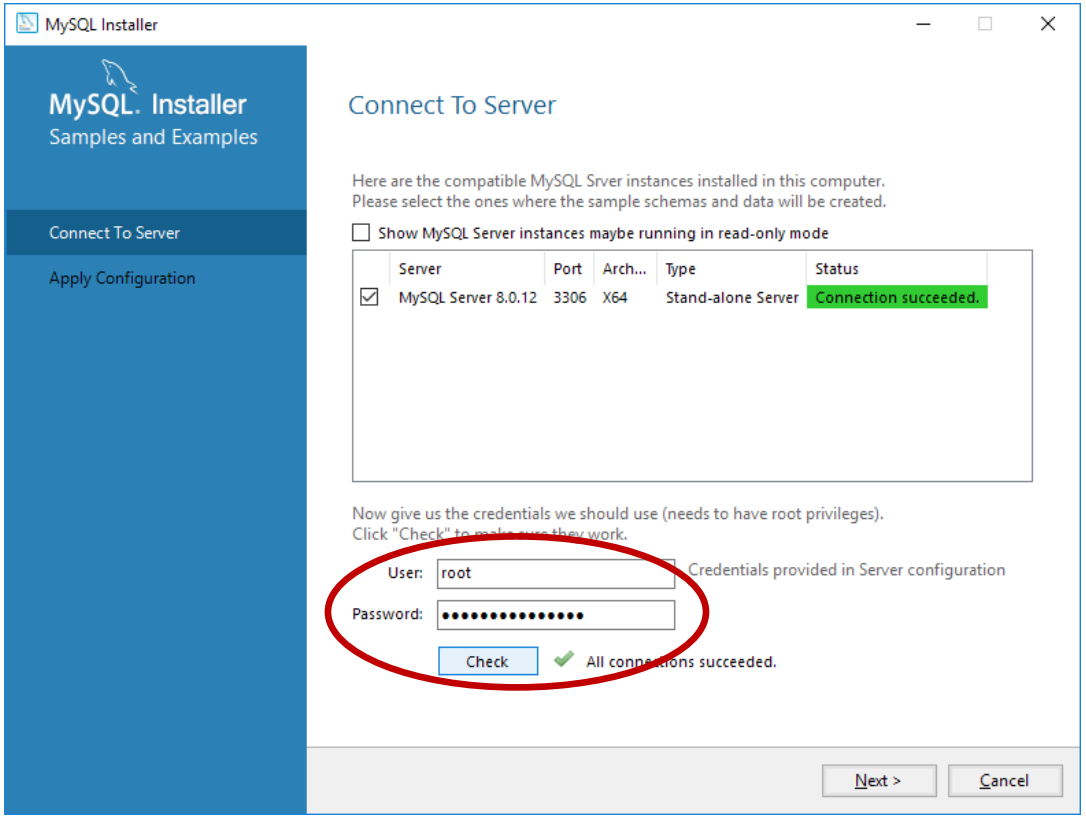
- *Regarding picture on the right:* The version number will be newer, that's ok. :-)



Remember your password!



Still remembering your password? :-P

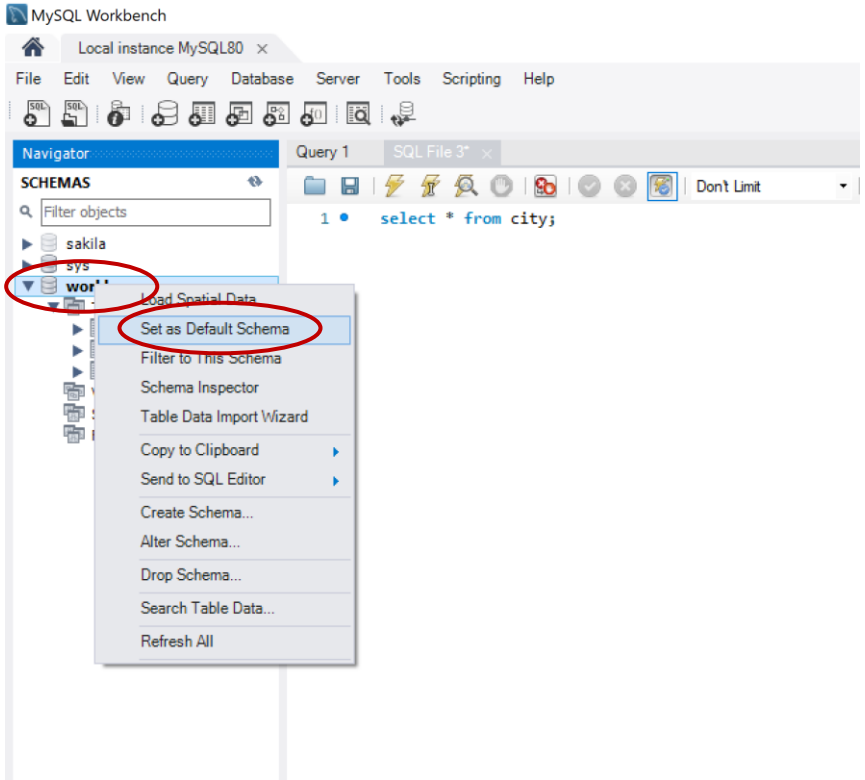


Are you using macOS? (Or Linux)

- There is no installer for macOS. :-\
- Canvas has a link to an installation guide (in "Eksterne ressurser / External resources").
  - Feel free to google a video tutorial or the like as well. :-)
- Follow the guide / tutorial, make sure you manually install all 3 elements from the previous slides.
  - *NOTE:* To download DB-schema on Canvas, use Chrome! *(Not Safari)*

Setting your schema as the "default" DB

- You need to activate the schema (DB) you want to work on.
- Right click that schema, choose:
  - Set as Default Schema.



SQL

- SQL follows normal computer language rules.
  - Meaning its quite strict on the syntax.

- In particular, note:
  - Each command should formally end with a semicolon. (However, MySQL Workbench doesn't care so long as you only run 1 command.)
  - NOT case sensitive.
  - A normal convention is to write language commands in uppercase and the rest as they are named. Example: `SELECT Name FROM city;`

SQL queries ("SELECT statements")

- Which columns to get: SELECT bla-bla-bla
- From which table: FROM bla-bla-bla
- Conditions for getting only some rows: WHERE bla-bla-bla
- Ordering the result: ORDER BY bla-bla-bla

Testing SQL, on the World schema

- NOTE: The examples on the following slides are executed on the World schema.

- After installing MySQL and the World schema, try it out yourself! :-)

The World schema ("database") contents

- `country`
  - Countries in the world.

Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExp	GNP	GI
NLD	Netherlands	Europe	Western Europe	41526.00	1581	15864000	78.3	371362.00	36
NOR	Norway	Europe	Nordic Countries	323877.00	1905	4478500	78.7	145895.00	15
NPL	Nepal	Asia	Southern and Central Asia	147181.00	1769	23930000	57.8	4768.00	48
NRU	Nauru	Oceania	Micronesia	21.00	1968	12000	60.8	197.00	NULL
NZL	New Zealand	Oceania	Australia and New Zealand	270534.00	1907	3862000	77.8	54669.00	64
OMN	Oman	Asia	Middle East	309500.00	1951	2542000	71.8	16904.00	16
PAK	Pakistan	Asia	Southern and Central Asia	796095.00	1947	156483000	61.1	61289.00	58

- `city`
  - Cities in the world.

ID	Name	CountryCode	District	Population
2807	Oslo	NOR	Oslo	508726
2808	Bergen	NOR	Hordaland	230948
2809	Trondheim	NOR	Sør-Trøndelag	150166
2810	Stavanger	NOR	Rogaland	108848
2811	Bærum	NOR	Akershus	101340
2812	Abidjan	CIV	Abidjan	2500000
2813	Bouaké	CIV	Bouaké	329850

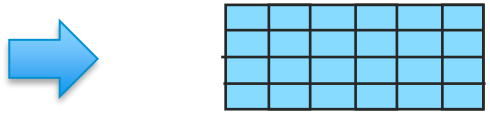
- `countryLanguage`
  - Languages per country.

CountryCode	Language	IsOfficial	Percentage
NOR	Danish	F	0.4
NOR	English	F	0.5
NOR	Norwegian	T	96.6
NOR	Saame	F	0.0
NOR	Swedish	F	0.3
NPL	Bhojpuri	F	7.5
NPL	Hindi	F	3.0

Some queries

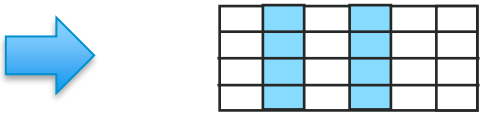
- When we want all the data in a table:

```
SELECT *
FROM city;
```



- When we only want certain columns from a table:

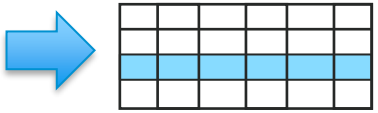
```
SELECT Name, Population
FROM city;
```



Some queries – cont.

- When we only want certain rows in a table:

```
SELECT *
FROM city
WHERE CountryCode = 'NOR';
```



- A condition (WHERE ...) is either true or false for each specific row.
  - NOTE: If we want more than one condition, we combine them with Boolean AND / OR.

Some queries – cont.

- Sorting the result: (here we combine with ',' *not* "and")

```
SELECT *
FROM city
ORDER BY CountryCode ASC, Population DESC;
```

  - Ascending order: ASC (or write nothing)
  - Descending order: DESC

- Note: ORDER BY can also use column number (instead of column name). Example: ORDER BY 2 DESC.
  - However: Use column name, easier to read / understand.

Some queries – cont.

- A SELECT statement ("query") uses one or more tables as in-data. The result is also presented in table format.
  - Meaning, we could chain queries, using the result of one query as input for the next.

```
SELECT Name, Population
FROM city
WHERE CountryCode = 'NOR'
ORDER BY Population DESC;
```

Name	Population
Oslo	508726
Bergen	230948
Trondheim	150166
Stavanger	108848
Bærum	101340

A selection of SQL functions

- SQL has some built in functions: ("premade logic")
  - COUNT(*) → gives number of rows
  - AVG(column_name) → the average column value of all selected rows
  - SUM(column_name) → the column sum of all selected rows
  - MIN(column_name) → the column minimum of all selected rows
  - MAX(column_name) → the column maximum of all selected rows.

- To get easier-to-read results, we can give out-columns names by using the keyword AS (or ALIAS).

```
SELECT COUNT(*) AS 'City count'
FROM City;
```

City count
4079

Today's exercises

- Now: 2 hours of exercises. Exercises are found on Canvas. Short summary:
  - Install MySQL.
  - Learn the basics of the SQL language.

- Remember: *You guys* are *not done* after 2 hours of exercises (4 hours total with me).
  - You should do approximately 4 additional hours of work (in a study group or alone) *before* our next session together. :-)

The