

Tomas Sandnes
tomas.sandnes@kristiania.no

Week 3

DB1102 / PGR 111 – DATABASES



Today's topics

(Today's chapters: 3 in Norwegian book, 3.2 & 4 in English book)

- Basics of keys:
 - Primary Keys (PKs) & Foreign Keys (FKs)
- Table changes:
 - Create table
 - Alter table
 - Drop table
- Changing the data:
 - Insert into
 - Update
 - Delete from



Basics of keys

Keys (repetition)

- **Primary key (PK)** – the unique identifier for each row in a table.
 - PKs can be composed of several columns.
 - PK columns *can not* contain NULL.
- **Foreign key (FK)** – one or more columns in a table that refers to (has the same value as) a primary key in a (possibly the same) table.
 - FK columns *can optionally* contain NULL.

Primary key in
the **Person** table



Person:

Pers_ID	Surname	First_Name	City
0	Miller	Paul	London
1	Ortega	Alvaro	Valencia
2	Huber	Urs	Zurich
3	Blanc	Gaston	Paris
4	Bertolini	Fabrizio	Rom

Car:

Car_ID	Model	Year	Value	Pers_ID
101	Bentley	1973	100000	0
102	Rolls Royce	1965	330000	0
103	Peugeot	1993	500	3
104	Ferrari	2005	150000	4
105	Renault	1998	2000	3
106	Renault	2001	7000	3
107	Smart	1999	2000	2

no relation

Primary key in
the **Car** table



Foreign key
in the **Car** table



Finding possible PKs and FKs

owner:

Id	Name	Address	Phone	Email
1	Ola Olsen	Liksomveien 2	22222222	ola@online.no
2	Ola Olsen	Januarveien 2	33333333	ola@gmail.com
3	Kari Jensen	Juliveien 3	44444444	kari@gmail.com

pet:

Id	Type	Name	Owner_Id
1	Cat	Mia	2
2	Dog	Passop	2
3	Parrot	Polly	1
4	Cat	Kitty	3

1. What columns here are suited as Primary Keys?
2. What columns here are suited as Foreign Keys?

Finding possible Keys, World db

1. What columns are suited as Primary Keys?
2. What columns are suited as Foreign Keys?

city

ID	Name	CountryCode	District	Population
1	Kabul	AFG	Kabul	1780000
2	Qandahar	AFG	Qandahar	237500
3	Herat	AFG	Herat	186800
4	Mazar-e...	AFG	Balkh	127800
5	Amster...	NLD	Noord-H...	731200
6	Rotterd...	NLD	Zuid-Holl...	593321
7	Haag	NLD	Zuid-Holl...	440900
8	Utrecht	NLD	Utrecht	234323

countrylanguage

CountryCode	Language	IsOfficial	Percentage
ABW	Dutch	T	5.3
ABW	English	F	9.5
ABW	Papiamentu	F	76.7
ABW	Spanish	F	7.4
AFG	Balochi	F	0.9
AFG	Dari	T	32.1
AFG	Pashto	T	52.4
AFG	Turkmenian	F	1.9

country

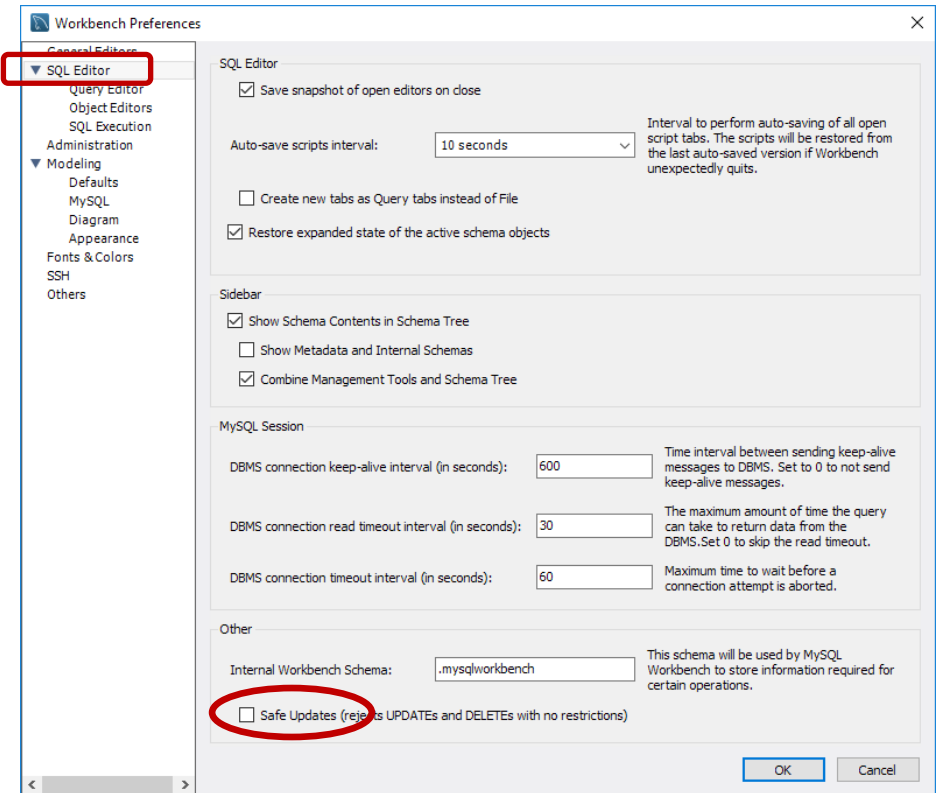
Code	Name	Continent	Region	SurfaceArea	IndepYear	Population	LifeExpectancy	GNP	GNPOLD	LocalName	GovernmentForm	HeadOfState	Capital	Code2
ABW	Aruba	North America	Caribbean	193.00	NULL	103000	78.4	828.00	793.00	Aruba	Nonmetropolitan T...	Beatrix	129	AW
AFG	Afghani...	Asia	Souther...	652090.00	1919	22720000	45.9	5976.00	NULL	Afganistan/A...	Islamic Emirate	Mohammad Omar	1	AF
AGO	Angola	Africa	Central ...	1246700.00	1975	12878000	38.3	6648.00	7984.00	Angola	Republic	JosÃ© Eduard...	56	AO
AIA	Anguilla	North America	Caribbean	96.00	NULL	8000	76.1	63.20	NULL	Anguilla	Dependent Territor...	Elisabeth II	62	AI
ALB	Albania	Europe	Souther...	28748.00	1912	3401200	71.6	3205.00	2500.00	ShqipÃ«ria	Republic	Rexhep Mejdani	34	AL
AND	Andorra	Europe	Souther...	468.00	1278	78000	83.5	1630.00	NULL	Andorra	Parliamentary Copri...		55	AD
ANT	Netherl...	North America	Caribbean	800.00	NULL	217000	74.7	1941.00	NULL	Nederlandse...	Nonmetropolitan T...	Beatrix	33	AN

Table changes

MySQL Workbench setting (repetition)

As mentioned last week: **IMPORTANT** Workbench setting to **turn off**. How-to:

- Edit -> Preferences...
- *Remove mark* in "Safe Updates...".
- Keeping mark here will hinder us from performing some of today's exercises.
 - You may need to scroll down to see this setting in your preferences window.



SQL DDL: changing tables

- DDL = Data Definition Language. Part of SQL. We use the following SQL **commands to modify tables**: (*not to modify content of tables*)
 - CREATE TABLE
 - ALTER TABLE
 - DROP TABLE
- And we use the following SQL **commands to modify content of tables**: (note: this is "regular" SQL, not part of DDL, more on this later)
 - INSERT INTO
 - UPDATE
 - DELETE FROM

DDL, create table

- If we want to create a table, we use the syntax below.
 - This is a *selection* of the possibilities.
 - Other combinations are possible.

```
CREATE TABLE tablename
(
    columnname datatype [UNIQUE|NOT NULL|DEFAULT|...],
    columnname datatype [UNIQUE|NOT NULL|DEFAULT|...],
    ...,
    [PRIMARY KEY (columnname)],
    [FOREIGN KEY (columnname) REFERENCES tablename (columnname)]
);
```

- Not quite getting the syntax? Look it up in the textbook, or google (for example: [w3schools](https://www.w3schools.com/sql/default.asp)) for more details! :-)

Datatypes in MySQL (repetition)

- Note: Datatype names and usage formatting varies slightly from database to database.
- Among MySQL's more common data types are:
 - `char` og `varchar` ← text (char is fixed size, varchar is variable size)
 - `[tiny/small/medium/big] int` ← number without decimals
 - `float` ← decimal number
 - `enum` ← special, user-defined data type
 - `date` ← date in the form: 'YYYY-MM-DD' (*note: this is MySQL-specific*)
 - `timestamp` ← auto-timestamp (*possibly for use with inserts and updates*)
- For a complete overview of data types in MySQL, see for example:
 - [w3schools.com/sql/sql_datatypes.asp](https://www.w3schools.com/sql/sql_datatypes.asp)

DDL, create table – cont.

- Example of a CREATE TABLE statement:

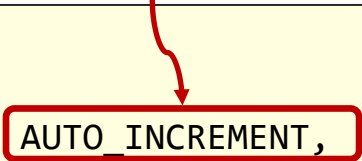
```
CREATE TABLE countryIsh
(
  Code CHAR(3) NOT NULL DEFAULT '',
  Name VARCHAR(60) NOT NULL UNIQUE,
  Continent ENUM('Asia','Europe') NOT NULL,
  Region VARCHAR(30) NOT NULL DEFAULT '',
  SurfaceArea FLOAT(10,2) NOT NULL DEFAULT 0.00,
  IndepYear SMALLINT(6) DEFAULT NULL,
  Population INT(11) NOT NULL DEFAULT 0,
  PRIMARY KEY (Code)
);
```

- NOT NULL means that the column must be filled in.
 - UNIQUE prevents two equal values (receives error message when submitting duplicates).
 - DEFAULT is used to set default values.
- *Feel free to run this SQL in a test-database! :-)*

Automatic ID (increasing serial number)

- An automatically increasing serial number is often used as a primary key.
- This ensures that all values in a column are different.
 - Ergo it also ensures a unique primary key when used like that.
- To specify that a column should use the MySQL-generated, auto-increasing serial number, use the keyword **AUTO_INCREMENT**:

```
CREATE TABLE Person
(
  ID int NOT NULL AUTO_INCREMENT,
  LastName varchar(255) NOT NULL,
  FirstName varchar(255),
  Age int,
  PRIMARY KEY (ID)
);
```



DDL, create table – cont.

- Example # 2: A table with a foreign key towards the countryIsh table:

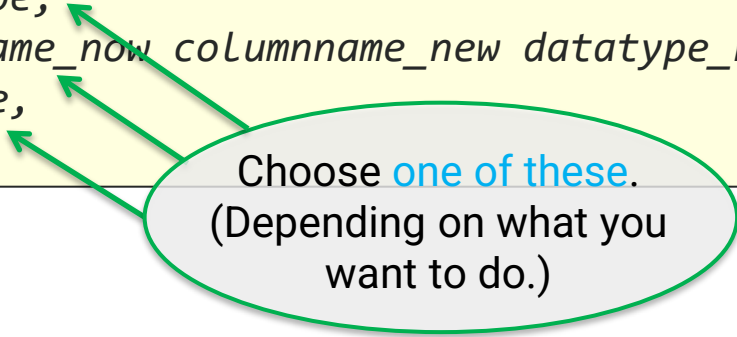
```
CREATE TABLE cityish
(
  id INT(11) NOT NULL AUTO_INCREMENT,
  name VARCHAR(35) NOT NULL DEFAULT '',
  countryCode CHAR(3) NOT NULL,
  district VARCHAR(20) NOT NULL DEFAULT '',
  population INT(11) NOT NULL DEFAULT 0,
  PRIMARY KEY (id),
  CONSTRAINT fk_cityIsh_countryIsh FOREIGN KEY (countryCode) REFERENCES CountryIsh(Code)
);
```

- **AUTO_INCREMENT** means, as already mentioned, an autogenerated serial number.
- **CONSTRAINT**, in this setting, is used as a unique nametag for the foreign key.

DDL, alter table

- Existing tables can be modified with the following syntax: (MySQL specific)

```
ALTER TABLE tablename
ADD columnname datatype,
MODIFY COLUMN columnname_now columnname_new datatype_new,
DROP COLUMN columnname,
...
```



Choose **one of these**.
(Depending on what you want to do.)

- Example:

```
ALTER TABLE person
ADD birthdate date;
```

DDL, drop table

- Deleting a table is short and sweet:

```
DROP TABLE tablename;
```

- *Note:* But make sure you delete the correct table!
- It will be gone forever and ever. ;-)

- Example:

```
DROP TABLE email;
```


Changing the data

SQL, adding new rows of data

- Adding rows to a table can be done in two ways.
 - With specifying the columns: (for a single row)

```
INSERT INTO tablename  
VALUES (value1, value2, value3, ..., valueN);
```

or without specifying the columns: (for a single row or multiple rows)

```
INSERT INTO tablename (columnname1, columnname2, ...)  
VALUES (value1, value2, ...),  
      (value1, value2, ...),  
      ...
```

- *Note:* Text fields (such as CHAR and VARCHAR) should be in apostrophes ('fnutter').
 - *Remember:* Primary key content must be unique!

INSERT INTO, example

```
INSERT INTO countryIsh (Code, Name, Continent)
VALUES ('WWW', 'OnlineLand', 'Asia');
```

```
INSERT INTO cityIsh (name, countryCode, population)
VALUES ('CloudCity', 'WWW', 12345);
```

- *Note:* We do not include information for all columns.
 - We can leave out columns that allow NULL and columns with a DEFAULT value.
- *Question:* Is it a coincidence that new rows are added to countryIsh before new rows are added to cityIsh?

SQL, update data in a table

- Updating the content of columns can be done for all rows, or with the help of a where-clause for a selection of rows:

```
UPDATE tablename  
SET columnname1 = value1, columnname2 = value2, ...  
[WHERE ...]
```

- Example:

```
UPDATE countryIsh  
SET Region = 'FarOutThere', Population = 74  
WHERE Code = 'WWW';
```

SQL, remove data rows from a table

- Removal of contents in a table can be done for all rows, or with a where-clause for only some rows: (as for updates)

```
DELETE FROM tablename  
[WHERE ...]
```

- Example:

```
DELETE FROM countryIsh  
WHERE Code = 'WWW';
```

- *Note:* As a security feature, MySQL Workbench will by default prevent deletion of rows without a where-clause to a key-reference.
 - This can be changed in Preferences, as previously shown.

What? DELETE didn't work?

```
Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint f... 0.000
```

- Vi are not allowed to delete from countryIsh.
 - We are not allowed because the city we entered still has a foreign key to this country.
- We can omit this problem by deleting the city-row with the FK first.
- But let's try something else:
 - We can use the keywords DELETE CASCADE on a FK in a table to allow this behavior.
 - But then we first have to remove the current FK from the table, then put it back in with the DELETE CASCADE keywords. (Shown over the coming slides.)

DELETE CASCADE, example

- First remove existing FK: (by using the constraint-nametag we gave it at creation)

```
ALTER TABLE cityIsh  
DROP FOREIGN KEY fk_cityIsh_countryIsh;
```

- Then re-create the foreign key, now with ON DELETE CASCADE added to it:

```
ALTER TABLE cityIsh  
ADD CONSTRAINT fk_cityIsh_countryIsh  
FOREIGN KEY (countryCode) REFERENCES CountryIsh(Code) ON DELETE CASCADE;
```

- Then trying to delete the row again. And, voila! It works. :-)

```
DELETE FROM countryIsh  
WHERE Code = 'WWW';
```

DDL, drop table examples

- *Note:* There is an important difference between:
 - Deleting content from a table (deleting rows of data).
 - Dropping (removing) the table itself (including any data in it).
- If we want to clean up after ourselves in the example from these slides, we need to drop our test tables:

```
DROP TABLE cityIsh;
```

```
DROP TABLE countryIsh;
```


Update tables through Workbench GUI

- What we now have learned, can be handled "easier" through the Workbench GUI:
 - There are GUI (graphical user interface) options that allow us to do the same things without writing a single line of DDL or SQL statements.
- HOWEVER:
 - What happens when we want to create a large database? For example, World from "world_schema.sql". We can't manually create 4000-5000 rows by hand.
 - And what about distributing our database solution to end users: We can't manually create the rows at each and every customer's machine.
- DDL and SQL are often not the easiest way to do small modifications to the db.
 - But: **For larger changes, DDL and SQL is the only practical way!**

Restore (the) World

- It may be that you delete (or change) content in the World database that should not have been deleted (or changed).
- Do not despair! To restore (the) World:
 - Cut and paste everything from the world_schema.sql file into Workbench, then run it with the "Lightning" button.
 - Or run the script from the Workbench menu, as explained in week 2.
- DONE: (the) World is restored! ;-)



Today's exercises & next lesson

- Now: 2 hours of exercises.
- Exercises are found on Canvas. Short summary:
 - Tasks for both the **DDL statements** and the **changing data statements**.
- Main contents for the next lesson:
 - More advanced queries than before: Using JOIN, which means selecting data from multiple tables at once.

The

