Folie 2 (Questions?):

--- OUTER JOIN

--- NOT IN

https://www.w3resource.com/sqlite/in-operator.php

--- NOT EXISTS

https://www.sqlitetutorial.net/sqlite-exists/

--- EXCEPT

https://www.sqlitetutorial.net/sqlite-except/

--- EXCEPT ALL

Folie 3 (Database Decomposition):

--- Have a table, that has NULL-values (CREATE TABLE xy)

```
CREATE TABLE names (Id INTEGER PRIMARY KEY, Last text NOT NULL, First text NOT NULL, DoB text);
```

```
INSERT INTO names (id, last, first, dob) VALUES (1, “Smith”, “Joe”, “6.7.1985”), (2, “Blum”, “Jane”, NULL), (3, “Smith”, “Jonas”, “1.9.2001”), (4, “Naish”, “Robby”, NULL);
```

// Note that single int values don’t need quotation marks (when you use text instead of integer)

--- Find a way to separate the table into sub-tables without NULLS

--- Rename original table (ALTER TABLE xy RENAME TO xy_new)

```
ALTER TABLE names RENAME TO names_original;
```

--- Create the sub-tables (CREATE TABLE xy)

```
CREATE TABLE names (Id INTEGER PRIMARY KEY, Last text NOT NULL, First text NOT NULL);
```

```
CREATE TABLE "birthdays" (Id INTEGER PRIMARY KEY, DoB text NOT NULL);
```

--- Fill each sub-table (INSERT INTO)

https://www.sqlitetutorial.net/sqlite-insert/

```
INSERT INTO names select Id, Last, First from names_original;
```

```
INSERT INTO birthdays select Id, DoB from names_original;
```

--- check if everything worked out (OUTER JOIN)

```
select * from names full outer join birthdays;
```

doesn’t look right (ON statement missing!)

```
select * from names n full outer join birthdays b ON n.id = b.id;
```

still doesn’t look the same as names_original (There is a names.id and a birthdays.id that we have to select for)

```
select n.id, last, first, dob from names n full outer join birthdays b ON n.id = b.id;
```

--- prove it $\mathrm{A} - B = 0 \wedge B - A = 0 \Rightarrow A = B$ (EXCEPT)

A := SELECT * FROM names_original;

B := SELECT n.id, last, first, dob FROM names n FULL OUTER JOIN birthdays b ON n.id = b.id;

A – B (A EXCEPT B):

select * from names_original EXCEPT select n.id, last, first, dob from names n full outer join birthdays b on n.id = b.id;

B – A (B EXCEPT A):

select n.id, last, first, dob from names n full outer join birthdays b on n.id = b.id EXCEPT select * from names_original;

Both return an empty set ⇒ $A = B$

Folie 4 (Using our knowledge):

```
--- Find every entry in names, that doesn‘t have a first name, that starts with ‚J‘:
select * from names where id not in (select id from names where first like "J%");


--- Find every person, that has an entry in birthdays:
select * from names n where exists (select * from birthdays where Id = n.Id);


--- Find every person, that has no entry in birthdays
select Id from names except select Id from birthdays;
--- Bonus:
select * from names natural join (select Id from names except select Id from birthdays);
```

Folie 5 (Last weeks queries):

--- Which movie made the most profit and when was it released? (Avatar, 2009, 2.553.439.092)

First way, that comes to mind:

```
select originaltitle, year, grossworldwide - budget as profit from movies natural join movies2grossworldwide natural join movies2budget order by profit desc limit 20;
```

Use max function:

```
select originaltitle, year, max(grossworldwide - budget) as profit from movies natural join movies2grossworldwide natural join movies2budget;
```

--- Who directed it? (James Cameron)

```
select name, originaltitle from persons natural join directors2movies natural join movies where originaltitle='Avatar’;
```

You may realize, that it’s important, that ‘=‘ is case sensitive, while ‘like’ is NOT

```
select name, originaltitle from persons natural join directors2movies natural join movies where originaltitle like 'avatar';
```

--- Which movie has the most directors and how many? (36)

```
select movieid, count(*) as cnt from directors2movies group by movieid order by cnt desc limit 20;
```

Weird Number, what happened?

```
select originaltitle, title, year, movieid, count(*) as cnt from directors2movies natural join movies group by movieid order by cnt desc limit 20;
```

Let’s check that on the IMovieDB:

https://www.imdb.com/

https://www.imdb.com/title/tt0973844/?ref_=nv_sr_srsg_0_tt_1_nm_0_q_Chacun%2520son%2520cin%25C3%25A9ma%2520ou%2520Ce%2520petit%2520coup

--- How many movies have the same title (1093)?

Subquerie:

```
select originaltitle, count(*) as cnt from movies group by originaltitle having cnt > 1 order by cnt desc;
select count(*) as duplicatecnt from (select originaltitle, count(*) as cnt from movies group by originaltitle having cnt > 1 order by cnt desc);
```

--- What are the

most popular actor names?

subquerie (List of names (distinct since personid) and in how many movies they played in):

```
select name, count(*) as cnt from persons natural join actors2movies group by personid limit 20;
```

better:

```
select name from persons natural join actors2movies group by personid limit 20;
```

solution REMEMBER TO NOT LIMIT THE SUBQUERIE:

```
select name, count(*) as cnt from (select name, personid from persons natural join actors2movies group by personid) group by name order by cnt desc limit 20;
```

Top 10 Movies with the least/most actors? (1, 1301)

```
select originaltitle, count(*) as cnt from movies natural join actors2movies group by movieid order by cnt limit 10;
```

```
select originaltitle, count(*) as cnt from movies natural join actors2movies group by movieid order by cnt desc limit 10;
```

Top 20 directors who made the most movies (Alfred Hitchcock)

```
select name, count(*) as cnt from persons natural join directors2movies group by personid order by cnt desc limit 20;
```

Top 20 directors with the most awards? (Steven Spielberg

```
select name, count(*) as awardcnt from persons natural join directors2movies natural join awards2movies natural join awards where awardoutcome='Winner' group by personid order by awardcnt desc limit 20;
```

Top 10 Actors that played in the most movies?

What could be a problem while finding this solution? (An actor could be listed more than once for a given movie)

Thats why we group by personid:

```
select name, count(*) as cnt from persons natural join actors2movies group by personid order by cnt desc limit 10;
```