Introduction on SQL



Content

Basics of relational databases

Your first query

Querying a single table

Querying multiple tables

Grouping, counting and aggregating rows

Basics of relational databases

Data Storage

Pre-computer age data storage

Today: Relational Databases

- Ideal for structured data
- It's all in tables
- SQL query language

Tables, Rows, Columns - the easy way

person_nr	person
1	Markus
2	Natasa
3	Tudor
4	Isabella

person_nr	food_nr
1	1
2	2
3	3
4	4
2	5
1	6
2	7
4	5
3	2
4	6
2	6
...	...

food_nr	food
1	Fries
2	Rice
3	Cookies
4	Spring rolls
5	Sushi
6	Beer
7	Chocolate

Who likes most different kinds of food ?

Who likes beer ?

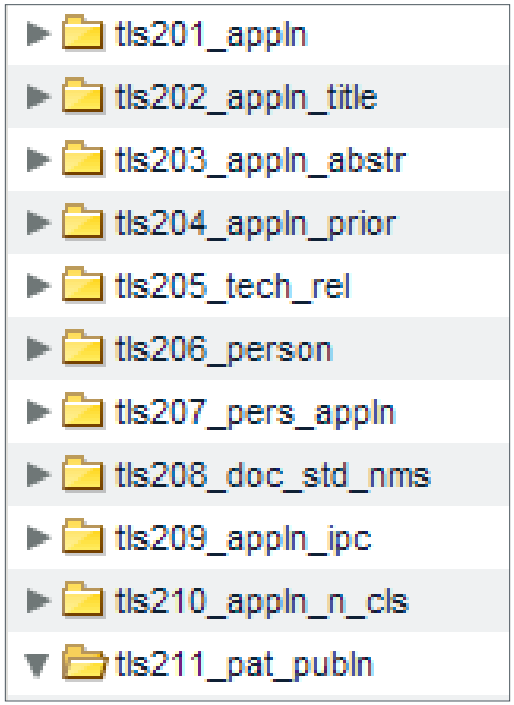
Tables, Rows, Columns

A relational database is a collection of linked tables.

A table consists of columns and rows.

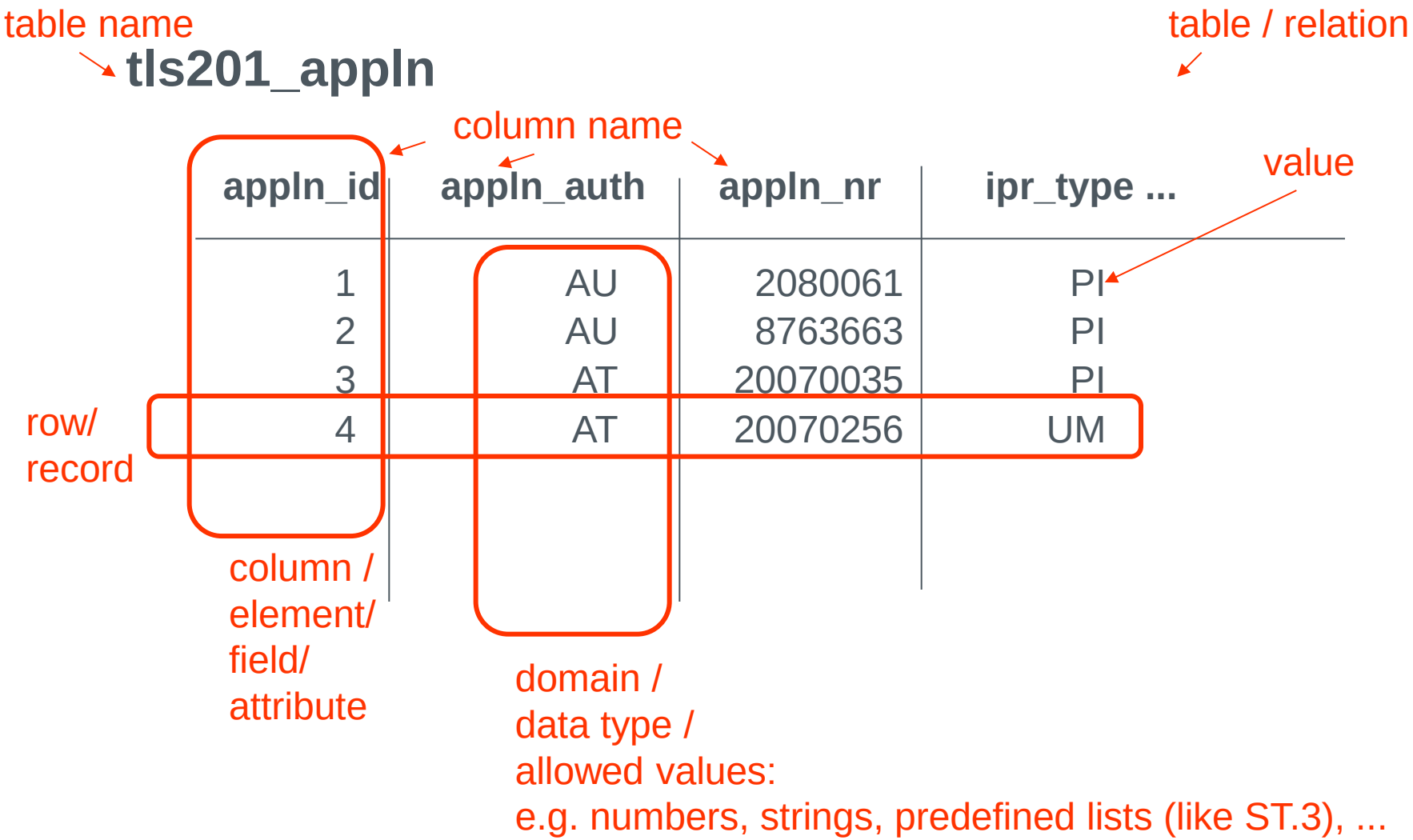
The cells contain the data.

Each row is a record.



appln_id	appln_auth	appln_nr	appln_kind	appln_filing_da...	ipr_type
0				9999-12-31	
1	EP	00103094	A	2000-02-15	PI
2	EP	00107845	A	1992-12-02	PI
3	EP	00202556	A	2000-07-17	PI
4	EP	00300208	A	2000-01-13	PI
5	EP	00310305	A	2000-11-20	PI

Relational Database concepts



A relational database

- ... is a collection of linked tables
- Each field of a row contains an elementary data element, e. g.
  - one IPC symbol
  - one application number
  - one abstract (regarded as one string, not as a sequence of words!)
- ... is well suited for structured data, but not for text or images (because e. g. text is usually treated as a complex data element, containing paragraphs, sentences, words, ...)

Exercise:

Use the Data catalogue to find out the allowed values (= domain) of the element `appln_kind` of the table `tls201_appln`

What applications do we retain with the condition

where appln_kind = 'W'

A key uniquely identifies a row

- A key is an attribute (or a set of attributes) which can uniquely identify a single row of its own table

- Examples
  - `tls201_appln.appln_id`
    (short for: column `appln_id` in `tls201_appln`)
  - in table `tls204_appln_prior`:
    columns `appln_id` and `prior_appln_id`

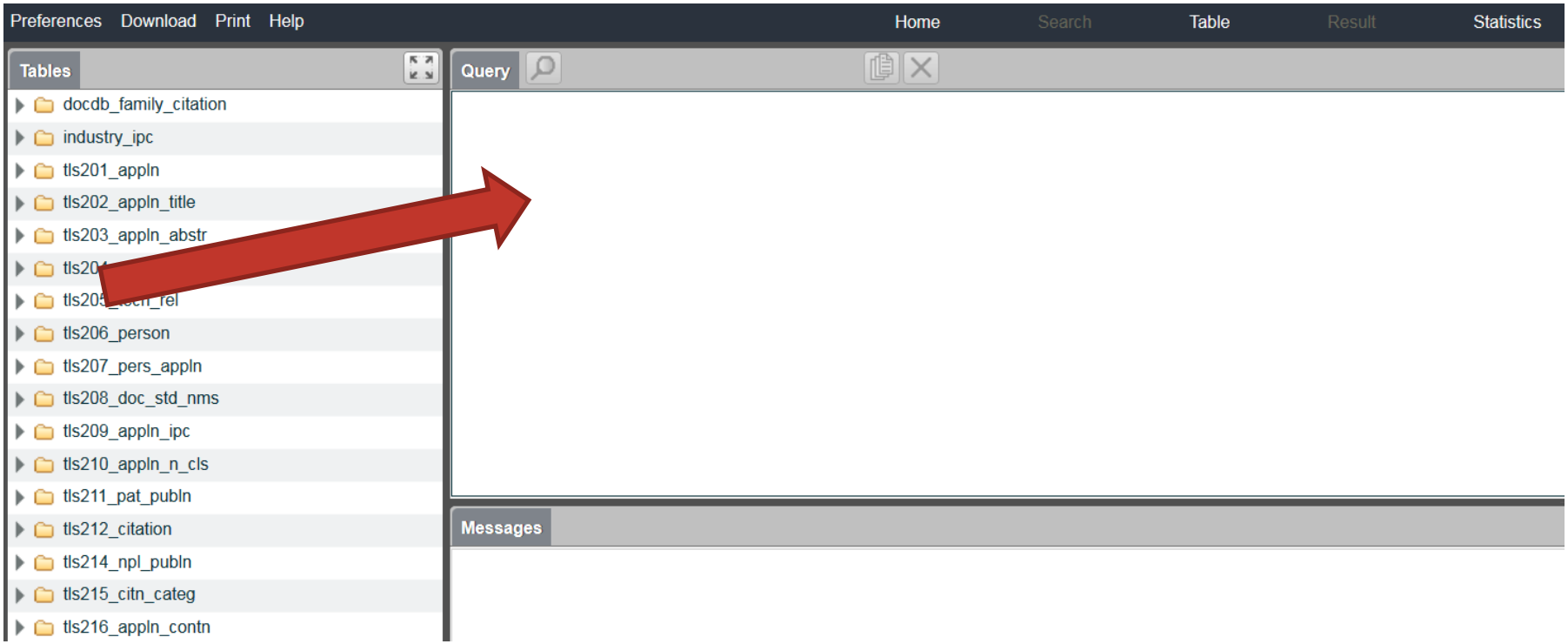
- By convention, keys are usually the leftmost columns of a table

Exercise:

Use the Data Catalog to find the keys of table tls211_pat_publn.

Querying a single table

Your first query



SQL: Your first query

Example:

```
SELECT appln_id
FROM tls201_appln
WHERE appln_auth = 'IE'
```

retrieves all application IDs from the table tls201_appln where the application authority = ‘IE’ (Ireland)

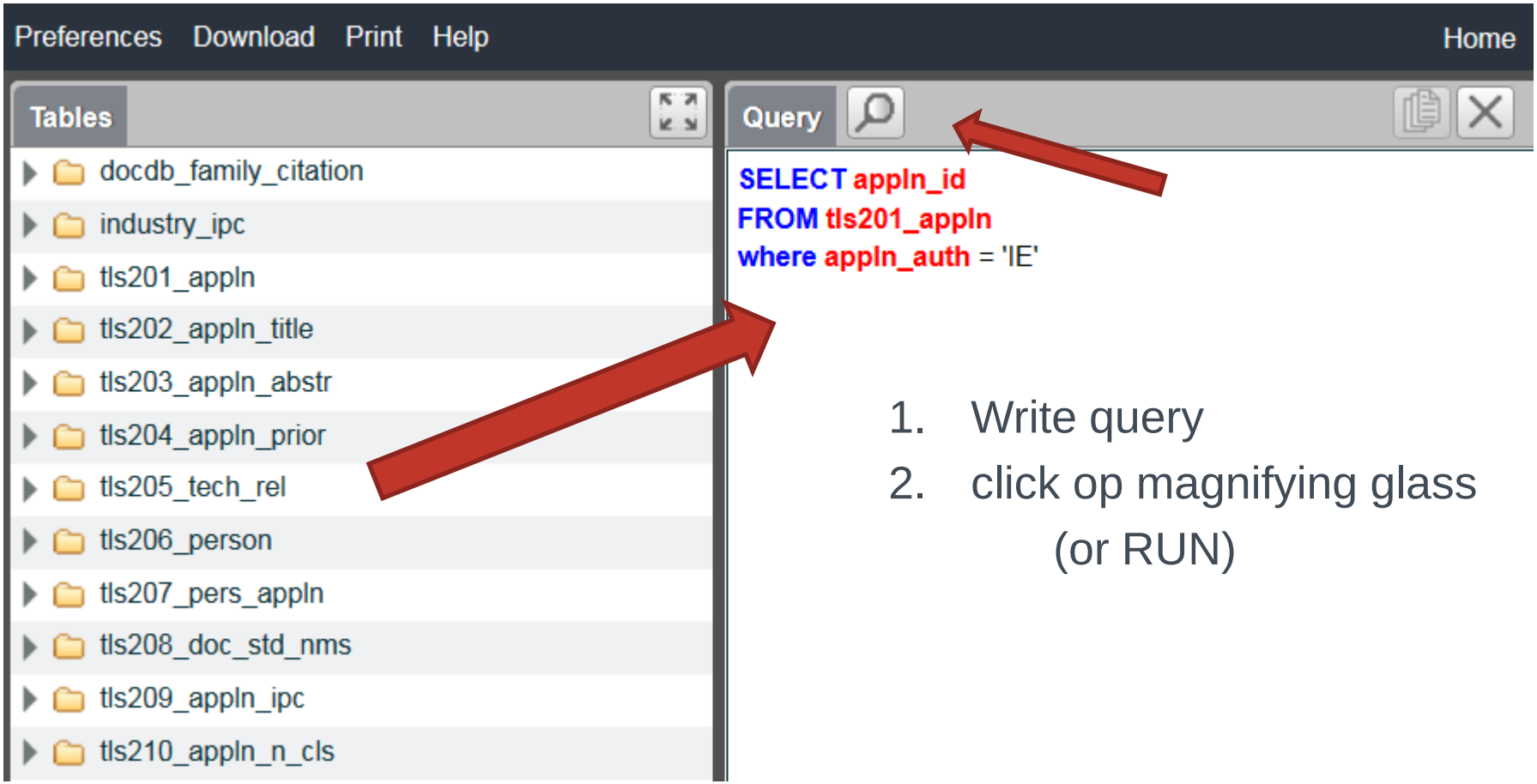
Or in non data base language:

retrieves all application ID's from the applications table for patents that were filed in Ireland

SELECT <attribute name> ← define result field(s)
FROM <table name> ← define table(s) used
WHERE <condition> ← optional; condition (which rows?)

Note: SQL language is not case-sensitive

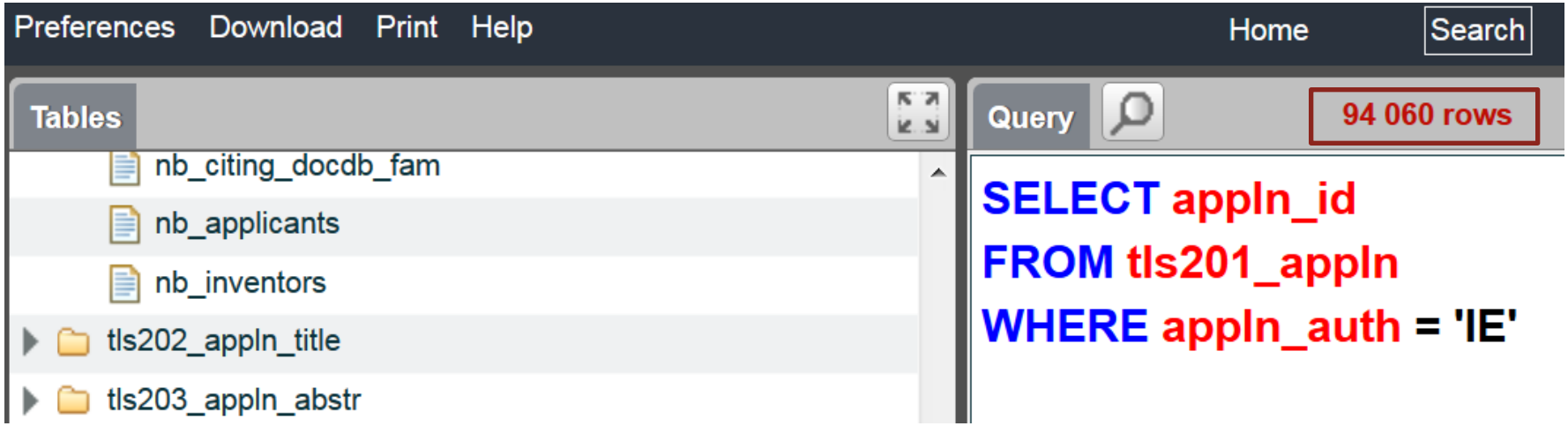
Your first query



1. Write query
2. click op magnifying glass
   (or RUN)

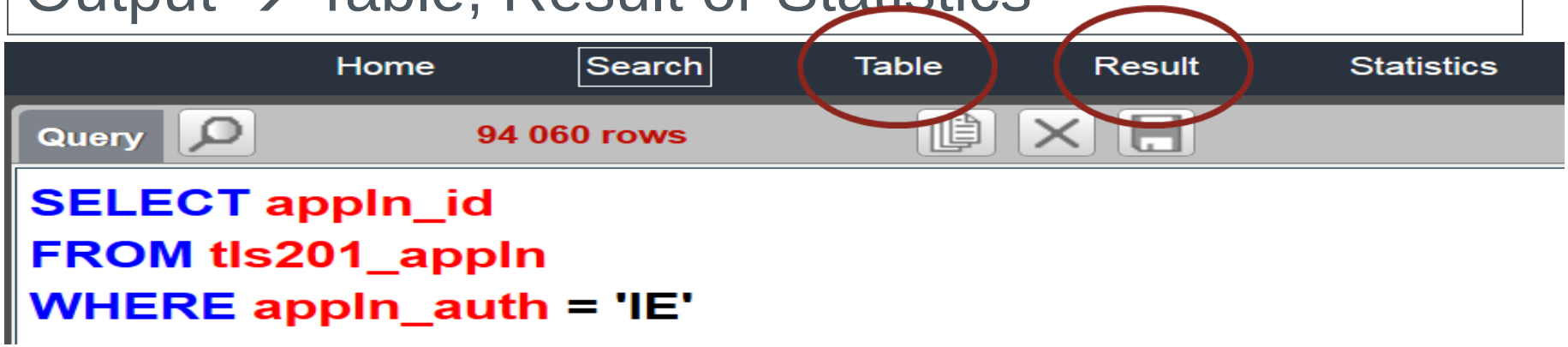
Exercise 1: SQL

Type the query of the last slide in the query window and execute it (or run it)

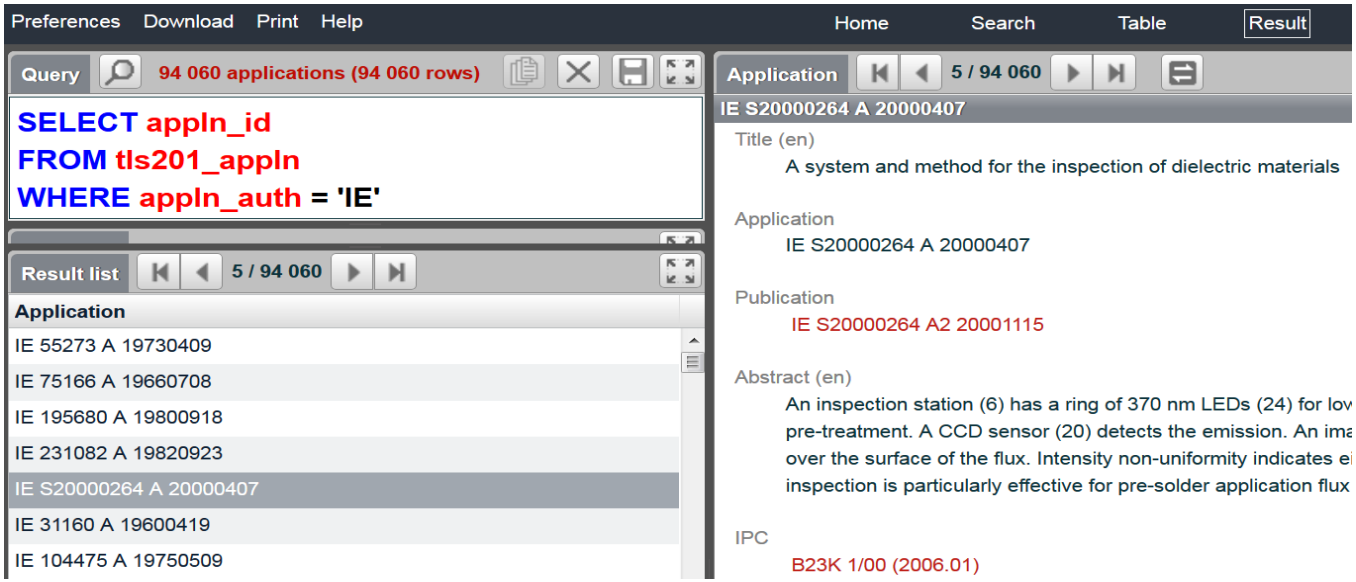


Exercise 1: Query design

Output → Table, Result or Statistics



Result table	1 / 94 060
Row	appln_id
1	14208
2	27402
3	27404
4	46188
5	46190
6	46192



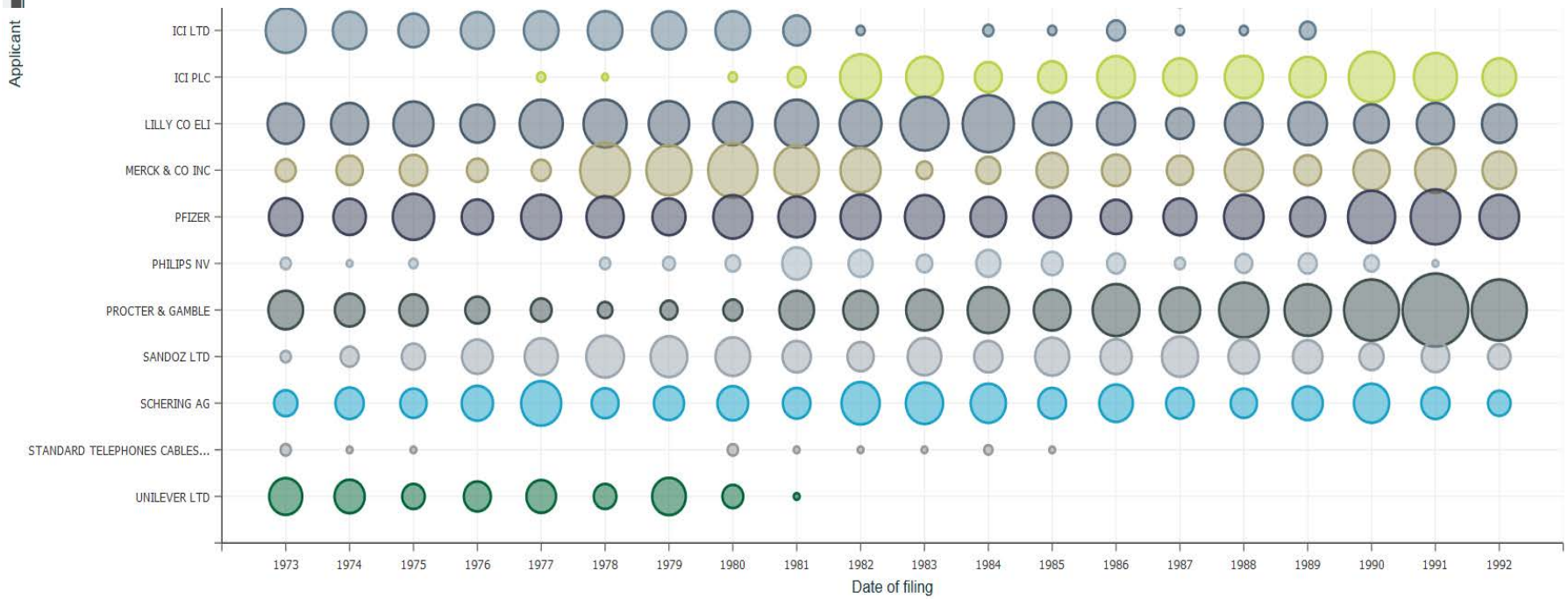
Exercise 1: Query design

Output → Statistics

Home | Search | Table | Result | Statistics

Query | 93 801 rows

```
SELECT appln_id
FROM tls201_appln
where appln_auth = 'IE'
```



The 3 main SQL keywords: SELECT - FROM - WHERE

Query

Every query starts with SELECT

- SELECT lists the **columns** (attribute) you want to see in your result
- FROM identifies the **table** you want to query
- WHERE defines the **rows** you want to retrieve

Result

The result is always another table, which contains

- all the rows which fit the condition of the WHERE clause
- exactly the columns you listed (same order) in your SELECT clause

Exercise 2: SQL

```
SELECT appln_auth, appln_nr, appln_kind, appln_filing_date
FROM tls201_appln
WHERE tls201_appln.appln_auth = 'UY' ;
```

Result table (based on table tls201_appln)

FROM
Where from ?

Exercise2

appln_auth	appln_nr	appln_kind	appln_filing_date
UY	4014	U	03.05.2004
UY	25666	A	17.08.1999
UY	28432	A	22.07.2004
UY	28634	A	24.11.2004
UY	31960	A	03.07.2009
UY	31626	A	03.02.2009
UY	25379	A	03.02.1999

SELECT
What do you see

WHERE
Limit the results

SELECT clause

The select clause may contain

- a single attribute
- a list of attributes, separated by comma
- an asterisk (*), which stands for "all attributes of the table(s)“
- an attribute with a function count(appln_id), max(appln_filing_date)

Examples

- SELECT appln_id from ...
- SELECT appln_auth, appln_nr, appln_kind, appln_filing_date from ...
- SELECT * from ....
- SELECT appln_auth, count(appln_id) from ...
- SELECT appln_auth, max(publn_date) from ...

Exercise 3: SQL

Use the previous query to retrieve

- several attributes of table tls201_appln
- all attributes of this table, but use the countries Brazil & Mexico

```
SELECT *
FROM tls201_appln
WHERE  (appln_auth= 'BR') or  (appln_auth= 'MX') ;
```

Exercise 4: SQL

List names of persons (applicants or inventors) + address that have a Turkish country code.

What table ?

What fields ?

What country code ?

```
SELECT person_name, person_address
FROM tls206_person
WHERE person_ctry_code =  'TR' ;
```

WHERE clause (1)

The WHERE clause contains one or more conditions.

The general form of a condition is:

<attribute or constant> <comparison operator> <attribute or constant>

The most common **comparison operators** and their meanings are

- = equal
- <>, != not equal
- >, >= larger, larger or equal
- <, <= smaller, smaller or equal

WHERE clause (2)

Typically, an attribute is compared to a constant.
Examples:

- appln_auth = 'IE' ← Strings are always enclosed by '
- appln_id <> 1234711 ← Number are not enclosed in quotes
- appln_filing_date >= '2001-01-01' ← Dates are also enclosed by '

Careful with dates: some data base platforms use # to enclose dates; some platforms “auto format” dates to dd-mm-yyyy.

You can also compare 2 attributes:

Example:

- appln_filing_year = publn_earliest_year ← = Applications which are filed and published in the same year (is an indicator for what ???)

Comparison operator: IN

Some of the following operators can be prefixed with the keyword NOT, which reverses its meaning.

IN, NOT IN

- Tests if the value of an attribute is contained in a list of values.

Examples:

- appln_auth IN ('EP', 'JP', 'US') ← limit to trilateral offices
  = (appln_auth='EP' or appln_auth='JP' or appln_auth='US')

appln_auth NOT IN ('US', 'CA') ← all offices except US and CA

Comparison operator: BETWEEN

BETWEEN, NOT BETWEEN

- Tests if the value of an attribute is between 2 values

Example:

appln_filing_date BETWEEN '2001-04-01' **AND** '2001-04-30'

```
SELECT appln_auth, appln_nr,appln_filing_date
FROM tls201_appln
WHERE appln_filing_date BETWEEN
                        '2001-04-01' AND '2001-04-30'
```

Alternative:

```
WHERE appln_filing_date >= '2001-04-01' AND
appln_filing_date <= '2001-04-30'
```

Comparison operator: LIKE

LIKE, NOT LIKE

- Tests whether 2 strings are similar.
  The %-wildcard represents zero, one or more characters (any characters)

Examples:

- person_name LIKE 'SIEMENS%'
- publn_kind LIKE 'A%'
- ipc_class_symbol like 'B60K%'
- appln_title LIKE '%hybrid%motor%'

Exercise 5: SQL

Create a list of the all the titles that contain 'Hybrid' and 'Motor'

What table ?

What fields ?

Criteria ?

```
SELECT appln_title, appln_id
FROM tls202_appln_title
WHERE tls202_appln_title.appln_title LIKE '%hybrid%motor%';
```

Same ? WHERE tls202_appln_title.appln_title LIKE '%hybrid%' and tls202_appln_title.appln_title LIKE '%motor%';

WHERE clause (3)

The WHERE clause can contain more than 1 condition.

AND, OR and brackets can be applied as usual

Example:

- WHERE (appln_auth = 'EP'
  AND appln_filing_year = 2008)
  OR (appln_auth <> 'EP'
  AND appln_filing_year = 2010)

Exercise 6: SQL

Create a query to retrieve all PCT applications which were filed through the Austrian patent office since 01.01.2005.

Hints:
- It is sufficient to use table tls201_appln only.
- When in doubt, check the Data Catalog

```
SELECT appln_auth, appln_nr, appln_kind, appln_filing_date

FROM tls201_appln

WHERE (appln_auth = 'AT' ) AND (appln_kind = 'W')

            AND (appln_filing_date> '2005-01-01' );
```

Exercise 7: SQL

Create a query to retrieve all application ID's from patents that were classified with the CPC code used for wind energy. (7b: Smart Grids)

Hints:

- How to find the appropriate CPC code(s) ?
- Be sure to include all applications !
- How would you do the same for the relevant IPC code(s)

```
SELECT tls224_appln_cpc.appln_id
FROM tls224_appln_cpc
WHERE tls224_appln_cpc.cpc_class_symbol  LIKE 'Y02E  10/7%'
```

ORDER BY clause

The "ORDER BY" clause allows you to "sort" the output of your query according to one (or more) columns in a descending or ascending way.

Create a query to retrieve from tls206_person a table that contains the person name and harmonised name that contain "EINSTEIN" in the harmonised name.

Result ? Add order by person_name to the query.

```
SELECT person_name, hrm_l2
FROM tls206_person
WHERE hrm_l2 like ('einstein%')
order by person_name desc
```

DISTINCT clause

As you could see from the result of the previous query, there are "duplicates" in the result list.

person_name	hrm_l2
Einstein IP Limited	EINSTEIN
EINSTEIN	EINSTEIN
EINSTEIN IP LIMITED	EINSTEIN
EINSTEIN IP LIMITED	EINSTEIN
EINSTEIN IP LIMITED	EINSTEIN
EINSTEIN & VOGLER	EINSTEIN & VOGLER
EINSTEIN & VOGLER	EINSTEIN & VOGLER
EINSTEIN ALBERT	EINSTEIN ALBERT
EINSTEIN ALFRED C.	EINSTEIN ALFRED C.

Duplicates can be removed by adding the "DISTINCT" in the select clause

```
SELECT DISTINCT person_name, hrm_l2
FROM tls206_person
WHERE hrm_l2 like ('einstein%')
order by person_name desc
```

Querying multiple tables

Links between tables

- In PATSTAT, patent data is broken into multiple tables. So, typically you have to query multiple tables to get useful answers.
- Luckily, tables can be joined. The JOIN is a very powerful feature of SQL and you will use it for most queries.
- One JOIN clause can connect 2 tables.
  To connect multiple tables, multiple JOIN clauses must be used.
- A join makes 1 table out of multiple tables based on common elements (foreign keys).

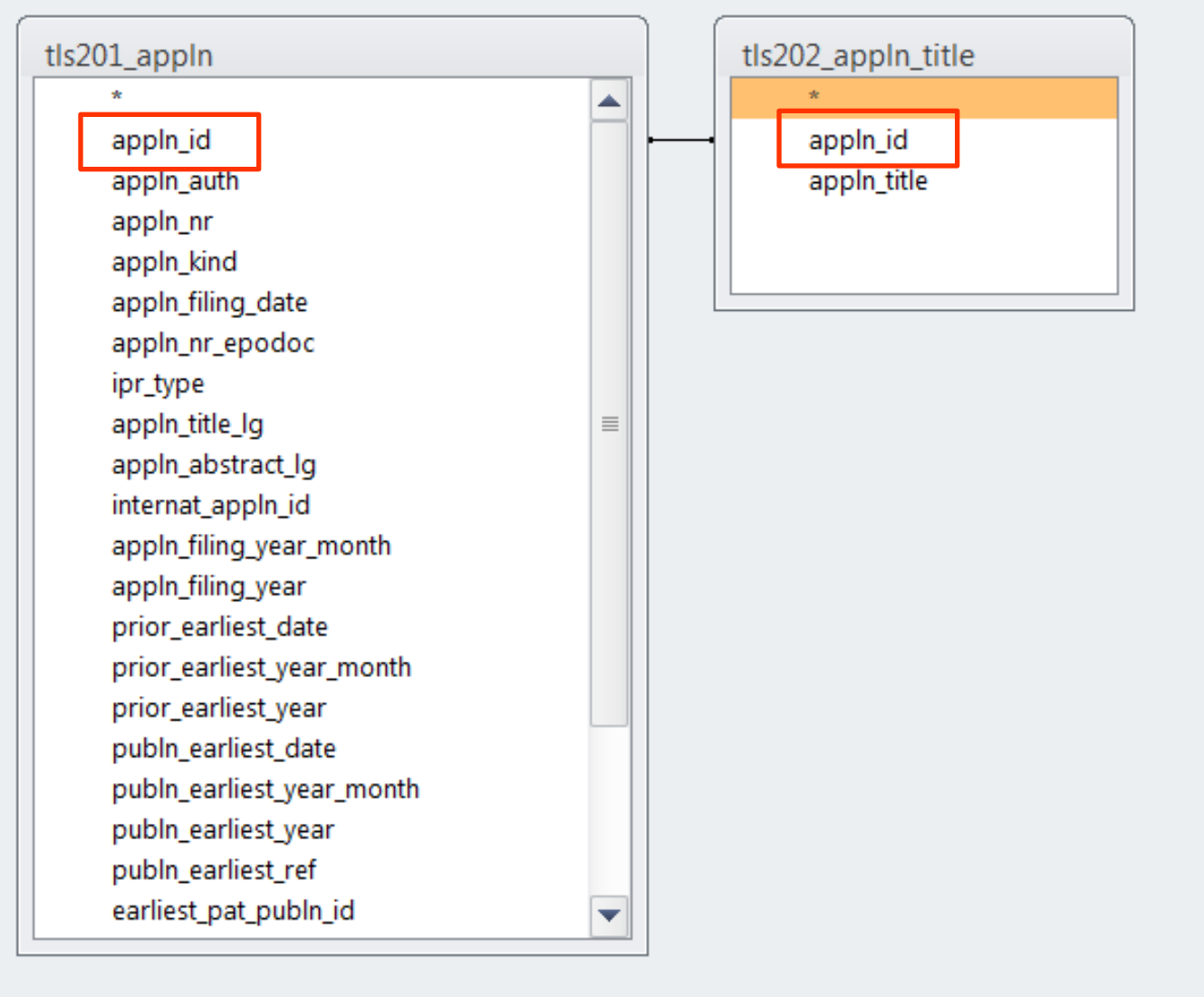
Links between tables

- How tables can be joined in a proper way, is defined in the Data Catalog and the data base model

TLS202_APPLN_TITLE(			
APPLN_ID	NOT NULL		
APPLN_TITLE	NOT NULL		
PRIMARY KEY	APPLN_ID		
FOREIGN KEY	APPLN_ID	REFERENCES	TLS201_APPLN (APPLN_ID)

TLS204_APPLN_PRIOR(			
APPLN_ID	NOT NULL		
PRIOR_APPLN_ID	NOT NULL		
PRIOR_APPLN_SEQ_NR			
PRIMARY KEY	(APPLN_ID, PRIOR_APPLN_ID)		
FOREIGN KEY	APPLN_ID	REFERENCES	TLS201_APPLN (APPLN_ID)
FOREIGN KEY	PRIOR_APPLN_ID	REFERENCES	TLS201_APPLN (APPLN_ID)
)			

Links between tables



Links between tables: person tables

Fragment of the real database

tls201_appln

appln_id	appln_auth	appln_nr	appln_kind	appln_filing_date
56	EP	4005863	A	12-03-2004

tls207_pers_appln

person_id	appln_id	applt_seq_nr	invt_seq_nr
232	56	1	0
233	56	0	1
234	56	0	2
235	56	0	3

tls206_person

person_id	person_name	person_address
232	Solvay Fluor GmbH	Hans-Böckler-Allee 20,30173 Hannover
233	Seseke-Koyro, Ulrich, Dr.	Eichendorffstrasse 3F,30916 Isernhagen
234	Frehse, Joachim	Breithauptstrasse 2,30625 Hannover
235	Becker, Andreas	Martin-Ottens-Ring 7,29331 Lachendorf

JOIN clause

```
SELECT appln_auth, appln_nr, appln_kind, appln_title
FROM tls201_appln  INNER JOIN tls202_appln_title ON tls201_appln.appln_id
= tls202_appln_title.appln_id
WHERE  appln_auth='US' AND appln_title LIKE '%rotor brake%';
```

- The JOIN clause specifies the **table to be joined** with the other tables in the queries.
- The **join condition** (the ON part of the JOIN clause) specifies the attribute(s) which should be used to join the tables.

appln_auth	appln_nr	appln_kind	appln_title
US	12418998	A	Helicopter rotor brake
US	77597707	A	Wind energy plant with a hydraulically actuated rotor brake
US	77598107	A	Wind energy plant with a hydraulically actuated rotor brake and method for the hydraulic control of a rotor brake
US	94004504	A	Rotor brake as well as rotor and rotary-wing aircraft with such a rotor brake

Table Alias - Giving tables a short name

- Table names are quite long
- You can assign a shorter **alias** name to tables
- Aliases are specified in the query and only valid within the scope of the query
- Aliases have to be written directly after the table name
- → Queries are easier to read

```
SELECT a.appln_auth, a.appln_nr, a.appln_kind, t.appln_title
FROM tls201_appln a inner  JOIN tls202_appln_title t ON a.appln_id = t.appln_id
WHERE a.appln_auth="US" AND t.appln_title like '%rotor brake%';
```

INNER JOIN vs. LEFT JOIN

- The **INNER JOIN** returns corresponding rows in 2 tables. A row in the left table with no corresponding row in the right table (or vice versa) is not retrieved.
- The **LEFT JOIN** also returns rows which do not have a corresponding row in the other table.

```
SELECT a.appln_id, a.appln_auth, a.appln_nr, a.appln_kind, t.appln_title
FROM tls201_appln a LEFT JOIN tls202_appln_title t ON a.appln_id = t.appln_id
where a.appln_id between 30004029 and 30012832
```

appln_id	appln_auth	appln_nr	appln_kind	appln_title
30005510	JP	9485278	U	
30012248	JP	9498391	U	
30012832	JP	9499576	A	WIND FORCE MOTOR
30004029	JP	9482298	A	FLUID TYPE POWER GENERATION APPARATUS
30009950	JP	9493897	A	GENERATING DEVICE UTILIZING TIDE OF SEA WATER

Exercise 12: SQL

Retrieve all US applications filed between BETWEEN '2015-04-01' AND '2015-04-30‘ with the respective IPC codes and order by appln_id and IPC code.

Hints:

- You need to link 2 tables....

- When in doubt, check the Data Catalog

```
SELECT tls201_appln.appln_auth, tls201_appln.appln_nr,
   tls201_appln.appln_kind, tls209_appln_ipc.ipc_class_symbol

FROM tls201_appln JOIN tls209_appln_ipc ON tls201_appln.appln_id
   = tls209_appln_ipc.appln_id

WHERE tls201_appln.appln_auth ='US'  and appln_filing_date
   between '2015-04-01' and '2015-04-30'

order by tls201_appln.appln_id, ipc_class_symbol
```

JOIN clause

```
SELECT a.appln_auth, a.appln_nr, a.appln_kind, a.appln_filing_date,
i.appln_id, i.ipc_class_symbol, i.ipc_class_level
FROM tls201_appln AS a LEFT JOIN tls209_appln_ipc AS i ON a.appln_id =
i.appln_id
WHERE a.appln_id=146;
```

- 1 applications may have 0, 1, or many IPC symbols (1:n relationship)
- Restricting the query to 1 application, still retrieves several rows, with an identical application part

appln_auth	appln_nr	appln_kind	appln_filing_date	appln_id	ipc_class_symbol	ipc_class_level
EP	07015148	A	02-Aug-07	146	F03D 9/00	A
EP	07015148	A	02-Aug-07	146	H02J 3/18	A
EP	07015148	A	02-Aug-07	146	F03D 7/04	A

Exercise 13: SQL

Retrieve all EP applications that have the word 'shell' in the applicant name. Use % wildcards to capture all names.

Hints:

- You need to link 3 tables....

- When in doubt, check the Data Catalog

```
SELECT tls201_appln.appln_auth, tls201_appln.appln_nr,
   tls201_appln.appln_kind, tls206_person.person_name

FROM (tls201_appln INNER JOIN tls207_pers_appln ON
   tls201_appln.appln_id =  tls207_pers_appln.appln_id) INNER JOIN
   tls206_person ON tls207_pers_appln.person_id =
   tls206_person.person_id

WHERE tls201_appln.appln_auth = 'EP' AND
   tls206_person.person_name like '%shell%' and applt_seq_nr > 0 ;
```

Grouping, counting and aggregating rows

GROUP BY clause

```
SELECT a.appln_auth, a.appln_filing_year
FROM tls201_appln a
where a.appln_filing_year between 2005 and 2008
GROUP BY a.appln_auth, a.appln_filing_year
ORDER BY a.appln_auth, a.appln_filing_year;
```

- The GROUP BY is followed by one or more attributes.
- All rows which have the same values for these attributes are compressed into 1 row

appln_auth	appln_filing_year
AM	2007
AP	2005
AP	2006
AP	2007
AP	2008
AR	2005
AR	2006
AR	2007
AR	2008
AT	2005
AT	2006

GROUP BY clause

```
SELECT a.appln_auth, a.appln_filing_year, count (*) as number_of_patents
FROM tls201_appln a
where a.appln_filing_year between 2005 and 2008
GROUP BY a.appln_auth, a.appln_filing_year
ORDER BY a.appln_auth, a.appln_filing_year;
```

The count(*) returns the number of rows (here: applications) which are compressed into a single row by the GROUP BY clause (filing year and office)

appln_auth	appln_filing_year	number_of_records
AP	2005	1
AP	2006	3
AP	2007	6
AP	2008	3
AR	2005	18
AR	2006	11
AR	2007	15
AR	2008	10
...	2005	64

In plain words:
This query computes the number of applications per office and filing year.

GROUP BY clause

Restrictions:

- When using GROUP BY the SELECT clause should only contain
  - attributes listed in the GROUP BY clause or
  - aggregated attributes, like COUNT

Aggregate functions

Aggregate functions work on a set of values and return a single value.

- When there is no GROUP BY, the function works on the complete result table
- When there is a GROUP BY CLAUSE, the function works on each group separately

- **COUNT()** Number of rows containing a Non-null value
- **SUM()** Sum
- **MIN()** Minimum
- **MAX()** Maximum

Exercise 14: SQL

Make a sorted list of applications that have 50 or more inventors. The maximum number of inventors per application should also be shown.

Hints:

- You need to link 2 tables....

- When in doubt, check the Data Catalog

```
SELECT a.appln_auth, a.appln_nr, a.appln_kind, max(pa.invt_seq_nr)

FROM tls201_appln a INNER JOIN tls207_pers_appln pa ON
   a.appln_id =  pa.appln_id

WHERE pa.invt_seq_nr >= 50

GROUP BY a.appln_auth, a.appln_nr, a.appln_kind

ORDER BY a.appln_auth, a.appln_nr;
```

Wrap up

General structure of SELECT

SELECT [DISTINCT] attribute(s) or expression(s)
FROM table
JOIN table **ON** join condition(s) -- JOIN clause may repeat
WHERE condition(s)
GROUP BY attribute(s) or expression(s)
(**HAVING** group condition(s))
ORDER BY attribute(s) or expression(s)

String Functions

String functions work with string attributes

- **LEFT(**ipc_class_symbol, 4**)**
  returns the first 4 characters (= IPC subclass)

  Often used on IPC and CPC column

- **CONCAT**(publn_auth, publn_nr, ' ', publn_kind**)**
  returns a string with the arguments concatenated

Comparison operator: BETWEEN

BETWEEN, NOT BETWEEN

- Tests if the value of an attribute is between 2 values

Example:

- appln_filing_date BETWEEN '2001-04-01' **AND** '2001-09-30'

Instead BETWEEN, you could use the operators >= and <=

- appln_filing_date >= '2001-04-01' **AND** appln_filing_date <= '2001-09-30'

Priorities

Other than you possibly expected, the priority table tls204_appln_prior is just a linking table. It links an application with each of its (Paris convention) priorities

```
SELECT a1.appln_id, a1.appln_auth, a1.appln_nr, a1.appln_kind,
pr.prior_appln_seq_nr, a2.appln_id AS prio_id, a2.appln_auth AS prio_auth,
a2.appln_nr AS prio_nr, a2.appln_kind AS prio_kind, a1.appln_nr_epodoc,
a1.appln_filing_date
FROM (tls201_appln AS a1 LEFT JOIN tls204_appln_prior AS pr ON
a1.appln_id = pr.appln_id) LEFT JOIN tls201_appln AS a2 ON pr.prior_appln_id
= a2.appln_id
WHERE a1.appln_auth='ep'  AND a1.appln_filing_date= '2013-06-25'
```

appln_id	appln_auth	appln_nr	appln_kind	prior_appln_seq_nr	prio_id	prio_auth	prio_nr	prio_kind	appln_nr_epodoc	appln_filing_date
407428468	EP	2013063315	W	2	405276150	EP	13161860	A	WO2013EP63315	25-06-2013
407428468	EP	2013063315	W	1	405381521	PL	39967812	A	WO2013EP63315	25-06-2013
407584025	EP	2013001864	W	1					WO2013EP01864	25-06-2013

Exercise 17: SQL-functions

Retrieve all applications and their key data (authority, application number, filing date, priority date) which have a (Paris convention) priority date in January 2008 and where the difference with the filing date (between priority and later filing) is > 365 days

```
SELECT tls201_appln.appln_auth, tls201_appln.appln_nr,
   tls201_appln.appln_kind, tls201_appln.appln_filing_date,
   tls201_appln_1.appln_filing_date, (tls201_appln.appln_filing_date-
   tls201_appln_1.appln_filing_date) AS difference

FROM (tls201_appln INNER JOIN tls204_appln_prior ON
   tls201_appln.appln_id = tls204_appln_prior.APPLN_ID) INNER JOIN
   tls201_appln AS tls201_appln_1 ON
   tls204_appln_prior.PRIOR_APPLN_ID = tls201_appln_1.appln_id

WHERE (((tls201_appln_1.appln_filing_date) Between '2008-01-01' And
   '2008-01-31') AND (((tls201_appln.appln_filing_date-
   tls201_appln_1.appln_filing_date))>365));
```

Families

Each application belongs to exactly one DOCDB and exactly one INPADOC family[1].
The query below retrieves all INPADOC family members of application EP08010999. (appln_nr)

```
SELECT a.appln_id, a.appln_auth, a.appln_nr, a.appln_kind,
b.appln_id, b.appln_auth, b.appln_nr, b.appln_kind,
b.appln_filing_date
FROM (tls201_appln AS a JOIN tls201_appln as b on
a.inpadoc_family_id = b.inpadoc_family_id)
WHERE a.appln_auth='ep'  AND a.appln_nr= '08010999'
```

The application itself is of course part of the family

appln_id	appln_auth	appln_nr	appln_kind	appln_id	appln_auth	appln_nr	appln_kind	appln_filing_date
33082	EP	08010999	A	55878477	US	21407408	A	2008-06-16
33082	EP	08010999	A	55668652	CN	200810131494	A	2008-06-23
33082	EP	08010999	A	33082	EP	08010999	A	2008-06-18
33082	EP	08010999	A	57898442	ES	200701738	A	2007-06-22

Thank you for your attention. Questions ?

patstat@epo.org

Geert Boedt

Acknowledgements
Martin Kracker (EPO)
Davide Lingua (EPO)
Christian Soltmann (EPO)