The REG108_APPLICANT_STATES Table

Welcome to table **REG108_APPLICANT_STATES**. In some cases the applicant does not seek patent protection in all countries which are designated in the application (see table `REG109_DESIGNATED_STATES`), but only in some of these countries. In these cases this table lists the countries for which a specific applicant seeks protection.

Let's start by initialising the client and importing what is needed.

In [2]:
```
from epo.tipdata.patstat import PatstatClient
from epo.tipdata.patstat.database.models import REG108_APPLICANT_
STATES
from sqlalchemy import func
import pandas as pd

# Initialise the PATSTAT client
patstat = PatstatClient(env='PROD')

# Access ORM
db = patstat.orm()
```

ID (Primary Key)

Technical identifier for an application, without business meaning. Its values will not change from one PATSTAT edition to the next.

In [3]:
```
i = db.query(
    REG108_APPLICANT_STATES.id
).limit(1000)

df = patstat.df(i)
df
```

Out[3]:

	id
0	89201077
1	5822663
2	4029926
3	87104717
4	86103512
...	...
995	19725162
996	83201528
997	5756866
998	7857336
999	17722025

1000 rows × 1 columns

SET_SEQ_NR

All applicants (one or more) of an application at a certain point of time are regarded as a “set” of applicants. The applicant(s) may change over time. Every change in these applicants results in another, newer set of applicants. This attribute contains the descending number for each set of party.

In [3]:
```
set_seq = db.query(
    REG108_APPLICANT_STATES.id,
    REG108_APPLICANT_STATES.set_seq_nr
).limit(1000)

set_seq_df = patstat.df(set_seq)
set_seq_df
```

Out[3]:

	id	set_seq_nr
0	89301221	3
1	19707367	2
2	18857350	1
3	20965446	1
4	99400551	1
...	...	...
995	90901626	1
996	12177863	1
997	11743540	3
998	15759804	1
999	89200528	4

1000 rows × 2 columns

BULLETIN_YEAR

For actions that have been published in the EPO Bulletin, it is the year of the publication in the bulletin. The default value is 0, used for applications that are not published or for which the year is not known. The format is YYYY otherwise.

The applications in this table are the ones seeking protection only in some designated states. Therefore, they are not published in the EPO bulletin. Indeed, if we search for applications with `bulletin_year` different from 0 we find no results.

In [4]:
```
years = db.query(
    REG108_APPLICANT_STATES.bulletin_year,
    REG108_APPLICANT_STATES.id
).filter(
    REG108_APPLICANT_STATES.bulletin_year != 0
)

years_df = patstat.df(years)
years_df
```

Out[4]:

BULLETIN_NR

This is the issue number of the EPO Bulletin for actions that have been published in it. The Bulletin number indicates the calendar week the Bulletin has been published. The default value 0 is used when the attribute `bulletin_year` is 0.

We found out that the bulletin year is always 0, hence, by definition, `bulletin_nr` is always equal to 0 as well.

In [5]:
```
bulletin_nr = db.query(
    REG108_APPLICANT_STATES.id,
    REG108_APPLICANT_STATES.bulletin_nr,
    REG108_APPLICANT_STATES.bulletin_year
).filter(
    REG108_APPLICANT_STATES.bulletin_nr != 0
)

bulletin_nr_df = patstat.df(bulletin_nr)
bulletin_nr_df
```

Out[5]:

TYPE

Type of party ("A" for applicant / "I" for inventor / "R" for legal representative). Since this table is about applicants, we expect to find only "A" as type.

In [9]:
```
type_party = db.query(
    REG108_APPLICANT_STATES.type,
    func.count(REG108_APPLICANT_STATES.id).label('number_of_appli
cations')
).group_by(
    REG108_APPLICANT_STATES.type
).order_by(
    func.count(REG108_APPLICANT_STATES.id).desc()
)

type_party_df = patstat.df(type_party)
type_party_df
```

Out[9]:

	type	number_of_applications
0	A	1850548

SEQ_NR

Sequence number of party (concerns applicant, inventor and legal representative). All applicants (one or more) of an application at a certain point of time are regarded as a “set” of applicants. The `seq_nr` defines the order of applicants within this set.

In [4]:
```
seq = db.query(
    REG108_APPLICANT_STATES.id,
    REG108_APPLICANT_STATES.seq_nr,
    REG108_APPLICANT_STATES.set_seq_nr
).limit(1000)

seq_df = patstat.df(seq)
seq_df
```

Out[4]:

	id	seq_nr	set_seq_nr
0	89301221	2	3
1	19707367	1	2
2	18857350	2	1
3	20965446	1	1
4	99400551	1	1
...	...	...	...
995	90901626	2	1
996	12177863	2	1
997	11743540	1	3
998	15759804	1	1
999	89200528	1	4

1000 rows × 3 columns

COUNTRY

Two-letter country/territory code for patent parties (applicant/inventor/agent), designated states of applicant, or country of licensees. Default value: empty. The domain consists of up to 2 alphabetic characters, acoording to WIPO ST.3, plus minor additions.

In [7]:
```
country = db.query(
    REG108_APPLICANT_STATES.country,
    func.count(REG108_APPLICANT_STATES.id).label('number_of_applications')
).group_by(
    REG108_APPLICANT_STATES.country
).order_by(
    func.count(REG108_APPLICANT_STATES.id).desc()
)

country_df = patstat.df(country)
country_df
```

Out[7]:

	country	number_of_applications
0	DE	99674
1	GB	96750
2	FR	95568
3	IT	80906
4	NL	72084
5	CH	67826
6	SE	66464
7	ES	65310
8	LI	65288
9	BE	65244
10	AT	64040
11	DK	56561
12	GR	56464
13	LU	54631
14	IE	53393
15	FI	50062
16	PT	49546
17	CY	48490
18	MC	48189
19	BG	42044
20	CZ	41834
21	EE	41295
22	TR	41223
23	HU	40351

24	SK	37606
25	RO	36457
26	SI	36257
27	PL	34400
28	IS	33194
29	LT	32755
30	LV	29758
31	MT	24918
32	NO	23088
33	HR	22781
34	MK	20415
35	AL	19674
36	SM	19575
37	RS	16216
38	ME	217

For example, we can retrieve the applications from one particular country/territory, together with name and address of the applicant. This can be done by joining this table with table REG107_PARTIES. Let's check out Germany (country code DE).

In [12]:
```
from epo.tipdata.patstat.database.models import REG107_PARTIES

de = db.query(
    REG108_APPLICANT_STATES.id,
    REG107_PARTIES.name,
    REG107_PARTIES.address_1
).join(
    REG108_APPLICANT_STATES, REG107_PARTIES.id == REG108_APPLICAN
T_STATES.id
).filter(
    REG108_APPLICANT_STATES.country == 'DE',
    REG107_PARTIES.name != ""
).order_by(
    REG107_PARTIES.name
)

de_df = patstat.df(de)
de_df
```

Out[12]:

	id	name	address_1
0	13766700	'T HART, Johan	c/o High Tech Campus
1	13821162	'T HART, Johan	c/o High Tech Campus 5
2	1985366	'T HOOFT, Cor	Diverseylever Maarssenbroeksedijk 2
3	1985366	'T HOOFT, Cor	Diverseylever Maarssenbroeksedijk 2
4	1985366	'T HOOFT, Cor	Diverseylever Maarssenbroeksedijk 2
...	...	...	...
1200736	93119065	Üffinger, Gerhard, Dr.	Leinäckerstrasse 71
1200737	93119065	Üffinger, Gerhard, Dr.	Leinäckerstrasse 71
1200738	93119066	Üffinger, Gerhard, Dr.	Leinäckerstrasse 71
1200739	93119066	Üffinger, Gerhard, Dr.	Leinäckerstrasse 71
1200740	93119066	Üffinger, Gerhard, Dr.	Leinäckerstrasse 71

1200741 rows × 3 columns

In [ ]: