

The REG111_LICENSEE Table

Welcome to table **REG111_LICENSEE** of PATSTAT Register. This table contains information about licensees and *rights in rem*.

In particular, we can find information about the country, the name, and the address of the parties.

```
In [2]: from epo.tipdata.patstat import PatstatClient
        from epo.tipdata.patstat.database.models import REG111_LICENSEE
        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(
        REG111_LICENSEE.id
        ).limit(1000)

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

Out [3]:

	id
0	3748392
1	7847136
2	18760228
3	8700916
4	6356013
...	...
995	99933046
996	10162538
997	85901870
998	3744293
999	7290882

1000 rows × 1 columns

CHANGE_DATE

It is the date of when the record was saved in the database.

```
In [4]: change_date = db.query(
        REG111_LICENSEE.change_date,
        REG111_LICENSEE.id
    ).limit(100)

change_date_df = patstat.df(change_date)
change_date_df
```

Out[4]:

	change_date	id
0	2008-07-11	3748392
1	2009-10-23	7847136
2	2024-04-27	18760228
3	2010-01-08	8700916
4	2018-03-09	6356013
...
95	2017-09-15	5739567
96	2022-11-18	18833271
97	2016-10-07	4300571
98	2018-12-07	15797148
99	2013-08-30	10290496

100 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.

```
In [5]: years = db.query(
        REG111_LICENSEE.bulletin_year,
        REG111_LICENSEE.id
    ).limit(1000)

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

Out[5]:

	bulletin_year	id
0	2008	3748392
1	0	7847136
2	0	18760228
3	0	8700916
4	2018	6356013
...
995	0	99933046
996	0	10162538
997	1988	85901870
998	0	3744293
999	0	7290882

1000 rows × 2 columns

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.

```
In [6]: bulletin_nr = db.query(
        REG111_LICENSEE.id,
        REG111_LICENSEE.bulletin_nr,
        REG111_LICENSEE.bulletin_year
    ).limit(100)

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

Out[6]:

	id	bulletin_nr	bulletin_year
0	3748392	33	2008
1	7847136	0	0
2	18760228	0	0
3	8700916	0	0
4	6356013	15	2018
...
95	5739567	42	2017
96	18833271	51	2022
97	4300571	0	0
98	15797148	2	2019
99	10290496	0	0

100 rows × 3 columns

LICENSEE_SEQ_NR

Serial number of license / sub license: the first two digits are the serial number of a main license and the optional other two digits represent the serial number of a sub-license. The maximum serial number for a main license and a sub-license is 10. Some applications can also have their licensee number set to 'deleted'.

```
In [26]: licensee = db.query(
          REG111_LICENSEE.id,
          REG111_LICENSEE.licensee_seq_nr
        ).limit(100)

licensee_df = patstat.df(licensee)
licensee_df
```

Out [26]:

	id	licensee_seq_nr
0	3748392	01 01
1	7847136	02
2	18760228	01
3	8700916	01
4	6356013	10 00
...
95	5739567	01 01
96	18833271	01 00
97	4300571	02
98	15797148	10 00
99	10290496	01

100 rows × 2 columns

The total number of occurrences of the value 'deleted' is 102. This value for this attribute is found only in this table, but not in table REG112_LICENSEE_STATES .

```
In [27]: deleted = db.query(
          REG111_LICENSEE.id,
          REG111_LICENSEE.licensee_seq_nr
        ).filter(
          REG111_LICENSEE.licensee_seq_nr == "deleted"
        )

deleted_df = patstat.df(deleted)
deleted_df
```

Out [27]:

	id	licensee_seq_nr
0	10776976	deleted
1	80900464	deleted
2	10805255	deleted
3	93202199	deleted
4	91114781	deleted
...
97	15163845	deleted
98	91108777	deleted
99	9812454	deleted
100	10807464	deleted
101	11187013	deleted

102 rows × 2 columns

TYPE_LICENSE

The type of the license: exclusive, not exclusive or right *in rem*. The domain consists of 3 ASCII characters:

- 'NOL', i.e. no license (occurs if and only if licensee_seq_nr = 'deleted')
- 'EXC', i.e. exclusive licence
- 'NEX', i.e. non-exclusive licence
- 'RIR', i.e. right in rem

We can check the one-to-one relation between the values 'NOL' and 'deleted', from attributes type_license and licensee_seq_nr respectively.

```
In [31]: nol = db.query(
    REG111_LICENSEE.type_license,
    func.count(REG111_LICENSEE.id).label('number of applications
license')
).group_by(
    REG111_LICENSEE.type_license
).filter(
    REG111_LICENSEE.licensee_seq_nr == 'deleted'
)

nol_df = patstat.df(nol)
nol_df
```

Out [31]:

	type_license	number of applications license
0	NOL	102

We can see that the number of applications without license (i.e. marked as 'NOL') are equal to the number of licensee_seq_nr equal to 'deleted'. Therefore, there are 102 applications without license.

DESIGNATION

Type of designation. Default value apart, i.e. empty string, this attribute can assume two values: 'all' and 'as-indicated'.

```
In [17]: design = db.query(
    REG111_LICENSEE.designation,
    func.count(REG111_LICENSEE.id).label('number_of_application
s')
).group_by(
    REG111_LICENSEE.designation
).order_by(
    func.count(REG111_LICENSEE.id).desc()
)

design_df = patstat.df(design)
design_df
```

Out [17]:

	designation	number_of_applications
0	as-indicated	10440
1	all	4545

VALID_DATE

This attribute indicates the date when the license became valid.

We can check, for example, how many and which licenses became valid after 2015.

```
In [40]: valid_date = db.query(
          REG111_LICENSEE.id,
          REG111_LICENSEE.valid_date
        ).filter(
          REG111_LICENSEE.valid_date > '2015-12-31',
          REG111_LICENSEE.valid_date != '9999-12-31'
        )

valid_date_df = patstat.df(valid_date)
valid_date_df
```

Out [40]:

	id	valid_date
0	18760228	2024-03-19
1	6356013	2018-01-23
2	19199865	2024-05-15
3	18157264	2023-10-10
4	18198362	2022-11-29
...
3199	16705202	2019-02-06
3200	14764306	2018-04-10
3201	20710868	2024-05-15
3202	18199633	2022-03-04
3203	19734195	2022-01-19

3204 rows × 2 columns

CUSTOMER_ID

The identifier of a customer, which can be a legal or natural person. If a customer has several roles (applicant, inventor, representative, licensee, opponent) he usually has several `customer_id` s.

Not every person has a `customer_id` assigned. Persons which are only inventors (and not e. g. also applicants) do not have a `customer_id` .

```
In [32]: customer = db.query(
          REG111_LICENSEE.customer_id
        ).filter(
          REG111_LICENSEE.customer_id != ""
        )

customer_df = patstat.df(customer)
customer_df
```

Out [32]:

	customer_id
0	0101046397
1	0101727164
2	0100726243
3	0101188398
4	0100726297
...	...
5476	0100749387
5477	0102047497
5478	0100725532
5479	0101933622
5480	0100726489

5481 rows × 1 columns

Let's count how many persons are not inventors.

```
In [34]: non_inventors = db.query(
          func.count(REG111_LICENSEE.id).label('non inventors')
        ).filter(
          REG111_LICENSEE.customer_id == ""
        )

non_inventors_df = patstat.df(non_inventors)
print(f"There are {non_inventors_df['non inventors'].item()} appl
      ications without customer ID.")
```

There are 9504 applications without customer ID.

NAME

Name of a person (natural person or legal person).

In showing the names, we have to filter out the empty spaces (default value).

```
In [18]: name = db.query(
          REG111_LICENSEE.name
        ).filter(
          REG111_LICENSEE.name != ""
        ).limit(100)

name_df = patstat.df(name)
name_df
```

Out [18]:

	name
0	Creabilis Therapeutics s.r.l.
1	SARL FLORIADES DE L'ARNON
2	Amedia Pharmaceuticals Limited
3	Albis Technologies AG
4	KAPPA PACKAGNG B.V.
...	...
95	Abbott Laboratories Limited
96	IGR & D
97	BARALAN INTERNATIONAL S.p.A.
98	ImCyse S.A.
99	Abbott Laboratories Vascular Enterprises Limit...

100 rows × 1 columns

Indeed, there are quite many empty names.

```
In [25]: empty = db.query(
          func.count(REG111_LICENSEE.name).label('empty names')
        ).filter(
          REG111_LICENSEE.name == ""
        )

empty_df = patstat.df(empty)
print(f"There are {empty_df['empty names'].item()} empty names.")
```

There are 9504 empty names.

ADDRESS_1

First address line of a person. Default value: empty. In PATSTAT Online these attributes are not populated.

```
In [8]: addr1 = db.query(
        REG111_LICENSEE.id,
        REG111_LICENSEE.name,
        REG111_LICENSEE.address_1
      ).limit(100)

addr1_df = patstat.df(addr1)
addr1_df
```

Out [8]:

	id	name	address_1
0	3748392	Creabilis Therapeutics s.r.l.	BioIndustry Park, Via Ribes 5
1	7847136		
2	18760228		
3	8700916		
4	6356013	SARL FLORIADES DE L'ARNON	Palleau
...
95	5739567	Flax-Slim APS	Høyrups Allé 20
96	18833271	AxLR, SATT du Languedoc Roussillon	Eugène Bataillon
97	4300571		
98	15797148	Phoremest Limited	Office 2055, Building 250
99	10290496		

100 rows × 3 columns

ADDRESS_2

Second address line of a person. Default value: empty. In PATSTAT Online these attributes are not populated.

```
In [9]: addr2 = db.query(
        REG111_LICENSEE.id,
        REG111_LICENSEE.name,
        REG111_LICENSEE.address_2
    ).limit(100)

addr2_df = patstat.df(addr2)
addr2_df
```

Out [9]:

	id	name	address_2
0	3748392	Creabilis Therapeutics s.r.l.	10010 Colletterto Giacosa (TO)
1	7847136		
2	18760228		
3	8700916		
4	6356013	SARL FLORIADES DE L'ARNON	18120 Lury Sur Arnon
...
95	5739567	Flax-Slim APS	2900 Hellerup
96	18833271	AxLR, SATT du Languedoc Roussillon	34095 Montpellier Cedex 5
97	4300571		
98	15797148	Phoremest Limited	Babraham Research Campus
99	10290496		

100 rows × 3 columns

ADDRESS_3

Third address line of a person. Default value: empty. In PATSTAT Online these attributes are not populated.

```
In [35]: addr3 = db.query(
    REG111_LICENSEE.id,
    REG111_LICENSEE.name,
    REG111_LICENSEE.address_3
).filter(
    REG111_LICENSEE.address_3 != ""
)

addr3_df = patstat.df(addr3)
addr3_df
```

Out [35]:

	id	name	address_3
0	19199865	Ocado Solutions Limited	Hatfield Hertfordshire AL10 9UL
1	20177712	Abbott Diabetes Care Limited	Witney
2	9709173	Natural Environment Research Council	Swindon SN2 1EU
3	16886804	SHENZHEN XIANGYUAN TECHNOLOGY CO., LTD	Baoan Central District, Baoan District
4	23150617	Ocado Innovation Limited	Hatfield
...
996	20817266	Ocado Solutions Limited	Hatfield Hertfordshire AL10 9UL
997	20181886	Abbott Laboratories Limited	Vanwall Road
998	6764988	Crafter's Companion Limited	Coundon
999	16705202	Ouest Valorisation	35708 Rennes Cedex
1000	20710868	Ocado Innovation Limited	Hatfield

1001 rows × 3 columns

ADDRESS_4

Forth address line of a person. Default value: empty. In PATSTAT Online these attributes are not populated.

```
In [36]: addr4 = db.query(
    REG111_LICENSEE.id,
    REG111_LICENSEE.name,
    REG111_LICENSEE.address_4
).filter(
    REG111_LICENSEE.address_4 != ""
)

addr4_df = patstat.df(addr4)
addr4_df
```

Out [36]:

	id	name	address_4
0	20177712	Abbott Diabetes Care Limited	OX29 0YL
1	16886804	SHENZHEN XIANGYUAN TECHNOLOGY CO., LTD	Shenzhen City
2	23150617	Ocado Innovation Limited	Hertfordshire AL10 9UL
3	13728500	Goodwin Refractory Services Limited	Stoke-on-Trent
4	20841667	Ocado Group plc	Hertfordshire AL10 9UL
...
578	22199813	Ocado Group plc	Hertfordshire AL10 9UL
579	14705565	Avacta Life Sciences Limited	West Yorkshire LS23 7FZ
580	20181886	Abbott Laboratories Limited	Maidenhead
581	6764988	Crafter's Companion Limited	Bishop Auckland
582	20710868	Ocado Innovation Limited	Hertfordshire AL10 9UL

583 rows × 3 columns

ADDRESS_5

Fifth address line of a person. Default value: empty. In PATSTAT Online these attributes are not populated.


```
In [37]: addr5 = db.query(
          REG111_LICENSEE.id,
          REG111_LICENSEE.name,
          REG111_LICENSEE.address_5
        ).filter(
          REG111_LICENSEE.address_5 != ""
        )

addr5_df = patstat.df(addr5)
addr5_df
```

Out [37]:

	id	name	address_5
0	13728500	Goodwin Refractory Services Limited	Staffordshire ST1 3NR
1	21194269	Abbott Laboratories Limited	Berkshire SL6 4XE
2	17182379	Abbott Laboratories Limited	Berkshire SL6 4XE
3	21211041	Abbott Laboratories Limited	Berkshire SL6 4XE
4	15761276	Eutilex Co., Ltd.	Seoul 08594
...
73	8712839	James White Drinks Limited	Suffolk IP6 9JS
74	20179938	Abbott Laboratories Limited	Berkshire SL6 4XE
75	16794002	Technijet Ltd	Lancashire LA6 2HP
76	20181886	Abbott Laboratories Limited	Berkshire SL6 4XE
77	6764988	Crafter's Companion Limited	Durham DL14 8LG

78 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, according to WIPO ST.3, plus minor additions.

We rank the most frequent countries in the database, avoiding the default value, that is a double space.

```
In [24]: country = db.query(
          REG111_LICENSEE.country,
          func.count(REG111_LICENSEE.id).label('number_of_applications')
        ).filter(
          REG111_LICENSEE.country != " "
        ).group_by(
          REG111_LICENSEE.country
        ).order_by(
          func.count(REG111_LICENSEE.id).desc()
        )

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

Out [24]:

	country	number_of_applications
0	FR	1745
1	GB	1350
2	US	799
3	DE	244
4	NL	183
5	IT	162
6	IE	137
7	CH	129
8	BE	100
9	JP	85
10	KR	65
11	CA	62
12	AT	48
13	SE	43
14	ES	39
15	CN	28
16	DK	28
17	BM	24
18	AU	21
19	LU	20
20	NO	18
21	IL	18

22	FI	17
23	CY	17
24	SG	16
25	GR	12
26	LT	6
27	MY	6
28	CZ	5
29	MC	5
30	PT	4
31	TR	4
32	CO	3
33	NZ	3
34	RS	3
35	IS	3
36	BB	3
37	RU	3
38	LI	3
39	HU	2
40	YU	2
41	SI	2
42	HK	2
43	SA	1
44	KY	1
45	IR	1
46	BH	1
47	MX	1
48	BG	1
49	TH	1
50	BS	1
51	BR	1
52	RO	1
53	PL	1

In []: