The Patent Application Continuation Table (TLS216_APPLN_CONTN)

Welcome to the Application Continuation Table, designated as TLS216_APPLN_CONTN.

In a similar way as the TLS204_APPLN_PRIOR establishes the priority links between applications, the links between parent and child applications for various types relations such as continuation (in part), divisional applications, internal priorities are defined via the TLS216_APPLN_CONTN table. Continuation (in part) is generally only applicable to US patent applications.

In [1]:

```
from epo.tipdata.patstat import PatstatClient

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

# Access ORM
db = patstat.orm()

# Importing the as models
from epo.tipdata.patstat.database.models import TLS216_APPLN_CONTN
```

APPLN_ID

APPLN_ID refers to the continuation application. As for tables TLS204 and TLS205, we join table TLS216 and TLS201 via two different attributes: `parent_appln_id` for table TLS216 and `appln_id` for table TLS201. In this way we can retrieve in table TLS201 previous applications to which `appln_id` s in table TLS216 are linked via the `parent_appln_id` attribute.

In [3]:

```
# Import table TLS201_APPLN
from epo.tipdata.patstat.database.models import TLS201_APPLN

contn_id = db.query(
    TLS201_APPLN.inpadoc_family_id,
    TLS216_APPLN_CONTN.parent_appln_id,
    TLS201_APPLN.appln_id,
    TLS201_APPLN.appln_filing_date,
    TLS201_APPLN.earliest_filing_date
).join(
    TLS201_APPLN, TLS216_APPLN_CONTN.parent_appln_id == TLS201_AP
PLN.appln_id  # Join table TLS201 and TLS216
).limit(10000)

contn_id_df = patstat.df(contn_id)
contn_id_df
```

Out[3]:

	inpadoc_family_id	parent_appln_id	appln_id	appln_filing_date	earliest_filing_date
0	446983955	531109993	531109993	2020-01-24	2014-05-29
1	548168819	548168819	548168819	2020-12-18	2020-12-18
2	524591585	553570488	553570488	2018-07-04	2018-07-04
3	473693356	499760880	499760880	2016-07-07	2015-07-08
4	447651112	509212838	509212838	2016-12-16	2015-12-18
...	...	...	...	...	...
9995	7550437	51620765	51620765	2000-03-21	1996-11-26
9996	423855270	423855270	423855270	2014-05-13	2013-05-15
9997	451089994	456128066	456128066	2016-03-09	2015-03-10
9998	475199553	483100522	483100522	2017-02-24	2016-02-25
9999	14620517	17157628	17157628	1996-04-25	1995-04-26

10000 rows × 5 columns

PARENT_APPLN_ID

PARENT_APPLN_ID refers to the application of which the APPLN_ID is a continuation. The two foreign keys (applications) should be different ones, i.e., there is no "self-continuation".

Also for continuations, we can see that the applications referring to the same prior application belong to the same INPADOC family but possibly to different DOODB families.

In [4]:

```
doc_inpa = db.query(
    TLS201_APPLN.appln_id,
    TLS216_APPLN_CONTN.parent_appln_id,
    TLS201_APPLN.inpadoc_family_id,
    TLS201_APPLN.docdb_family_id
).join(
    TLS201_APPLN, TLS216_APPLN_CONTN.appln_id == TLS201_APPLN.app
ln_id
).filter(
    TLS201_APPLN.inpadoc_family_id == 98645
).order_by(
    TLS216_APPLN_CONTN.parent_appln_id
)

doc_inpa_df = patstat.df(doc_inpa)
doc_inpa_df
```

Out[4]:

```
       appln_id  parent_appln_id  inpadoc_family_id  docdb_family_id
  0   470852042         46108868              98645         37908869
  1   505767121         46108868              98645         37908869
  2   410457093         46108868              98645         37908869
  3   470852042        410457093              98645         37908869
  4   505767121        410457093              98645         37908869
...         ...              ...                ...              ...
908   524407814        909083690              98645         69007671
909   489924383        909083690              98645         61159498
910   498657819        909083690              98645         63168063
911   594864518        909083690              98645         80114271
912   566501132        909083690              98645         80114271

913 rows × 4 columns
```

CONTN_TYPE

The type of continuation describing what relation the later application has to the earlier application.

Note that before 1991, the EPO did not record the so called "linkage type" of priority numbers, that is the EPO did not record which kind of relation a given priority number has (Paris Union priority, continuation, division, etc.). Data in this element prior to 1991 is thus not reliable.

If the continuation type is not known then `contn_type` is filled with 3 spaces. Let's take a look to the continuation applications with a continuation type assigned.

In [12]:

```python
contn_type = db.query(
    TLS216_APPLN_CONTN.parent_appln_id,
    TLS216_APPLN_CONTN.appln_id,
    TLS216_APPLN_CONTN.contn_type
).filter(
    TLS216_APPLN_CONTN.contn_type != '   '  # Filter selecting those applications with contn_type attribute that is not blank (three spaces)
).order_by(
    TLS216_APPLN_CONTN.contn_type
).limit(100000)

contn_type_df = patstat.df(contn_type)
contn_type_df
```

Out[12]:

	parent_appln_id	appln_id	contn_type
0	901355547	6494482	ADD
1	901246511	6312420	ADD
2	901229947	6288602	ADD
3	901175778	6133610	ADD
4	909123017	6449811	ADD
...	...	...	...
99995	376346805	424895787	CIP
99996	909065334	48371380	CIP
99997	470859928	472465053	CIP
99998	534910604	573882632	CIP
99999	53728250	54211342	CIP

100000 rows × 3 columns

In the part of the table that is displayed, we can see two types of continuation: addition (ADD) and continuation in part (CIP).

Changing the filter, we can see that there are many continuation applications with unknown type.

In [15]:

```
# Import func to use count
from sqlalchemy import func

unkwn_contn_type = db.query(
    func.count(TLS216_APPLN_CONTN.contn_type).label('tot_num_of_unknown_types')
).filter(
    TLS216_APPLN_CONTN.contn_type == '   '
)

unkwn_contn_type_df = patstat.df(unkwn_contn_type)
unkwn_contn_type_df = unkwn_contn_type_df['tot_num_of_unknown_types'].item()
print("There are "+str(unkwn_contn_type_df)+" continuation applications with unknown continuation type")
```

```
There are 113940 continuation applications with unknown continuation type
```

In [ ]: