The TLS212_CITATION Table

This notebook explores the `TLS212_CITATION` table, which plays a crucial role in understanding the interconnections between patent publications, applications, and non-patent literature documents through citation analysis. The `TLS212_CITATION` table establishes links that shed light on the citation dynamics in the patent landscape, including both forward and backward citations. Forward and backward citations are defined as well as the citation generating authority (e.g., search authority) and the procedural step in which the citation was created (e.g., search report or opposition procedure).

Within this table, the `PAT_PUBLN_ID` attribute refers to the citing publication, indicating which patent is referencing another work. Conversely, the `CITED_PAT_PUBLN_ID` identifies the patent publication that is being cited. Additionally, the `CITED_APPLN_ID` refers to a patent application being cited; this application is independent of any publication cited by `CITED_PAT_PUBLN_ID`, thus providing a valid citation on its own. The `CITED_NPL_PUBLN_ID` attribute pertains to non-patent literature being cited, which may contain "hidden" references to patent publications.

Importantly, the business rules specify that the `PAT_PUBLN_ID` and `CITED_PAT_PUBLN_ID` must refer to different publications, thereby prohibiting self-citation. This prohibition of self-citation means that a publication cannot cite itself, ensuring that citation relationships remain clear and distinct. This structure allows for the establishment of many-to-many relationships between publications, meaning that one publication can cite multiple others, and a single publication can be cited by multiple sources.

The table distinguishes three types of citations based on the values of specific attributes. A patent citation occurs when either a patent publication or a patent application is cited, which is indicated by a non-zero value in `CITED_PAT_PUBLN_ID` or `PAT_CITN_SEQ_NR`. In contrast, a non-patent literature (NPL) citation arises when `CITED_APPLN_ID` has a non-zero value, signifying that the citation is derived from non-patent sources. Finally, an NPL citation that refers to a patent publication is represented by non-zero values in both `CITED_NPL_PUBLN_ID` and `NPL_CITN_SEQ_NR`, while `CITED_PAT_PUBLN_ID` remains zero.

The `TLS212_CITATION` table references the `TLS211_PAT_PUBLN` table to identify the citing publications through the `PAT_PUBLN_ID` field, which captures the patents or applications that reference other works. Additionally, the `CITED_PAT_PUBLN_ID` and `CITED_APPLN_ID` fields link to patent publications and applications, respectively, indicating the sources being cited. This table provides information about the patent applications that may be cited in the `TLS212_CITATION` table, establishing a connection that allows for a deeper understanding of the context in which applications are referenced. The `CITED_NPL_PUBLN_ID` field allows for the integration of non-patent literature (`TLS214_NPL_PUBLN`) into the citation framework, providing insights into how external research influences patent activity.

The `TLS212_CITATION` table categorizes citations into three distinct types based on the values of specific attributes. These types are defined by whether the citations refer to patent publications, patent applications, or non-patent literature (NPL), as indicated by the values in relevant columns.

1. Patent Citation: a patent citation occurs when a citing publication references another patent publication or a patent application. This type of citation is identified by checking the following attributes:

- `CITED_PAT_PUBLN_ID` : If this value is greater than zero, it indicates that a patent publication is being cited.
- `PAT_CITN_SEQ_NR` : A non-zero value in this attribute also signifies a citation of a patent.

In summary, if either CITED_PAT_PUBLN_ID or PAT_CITN_SEQ_NR has a non-zero value, the citation is classified as a patent citation. This means that the citing publication is referencing existing patent literature or applications, highlighting the direct connections within patent documentation.

2. Non-Patent Literature (NPL) Citation: a non-patent literature (NPL) citation arises when the citation comes from non-patent sources. This type is indicated by:

- `CITED_APPLN_ID` : If this value is greater than zero, it signifies that the citation is derived from non-patent literature. In this case, the citing publication is referencing an application that is not related to any patent publication.

Thus, the presence of a non-zero CITED_APPLN_ID implies that the citation refers specifically to non-patent sources, such as research articles or other documentation that does not fall under patent categories.

3. NPL Citation Referring to a Patent Publication: the third type of citation is a more specific category of NPL citation, which refers to a patent publication. This situation is identified by:

- `CITED_NPL_PUBLN_ID` : This attribute must have a non-zero value, indicating that the citation is derived from non-patent literature.
- `NPL_CITN_SEQ_NR` : A non-zero value in this column indicates the sequence number of the citation from the non-patent literature.
- `CITED_PAT_PUBLN_ID` : This attribute should be zero, confirming that the citation does not refer to another patent publication.

In this case, the citation originated from non-patent literature but specifically mentions a patent publication. The citation in the non-patent literature may contain references to patent documents, indicating a connection between the non-patent source and the patent literature.

In [1]:
```python
from epo.tipdata.patstat import PatstatClient
from epo.tipdata.patstat.database.models import (
    TLS201_APPLN,
    TLS206_PERSON,
    TLS212_CITATION,
    TLS211_PAT_PUBLN,
)
from sqlalchemy import and_, case, func, select

# Initialise the PATSTAT client
patstat = PatstatClient(env="TEST")

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

In [2]:
```python
query_citations = (
    db.query(
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_replenished,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth,
    )
    .filter(
        TLS212_CITATION.cited_appln_id != 0,  # Exclude cited_app
ln_id = 0
        TLS212_CITATION.cited_appln_id < 900000000  # filter for
application ID < 900000000
    )
    .order_by(TLS212_CITATION.citn_id.desc())  # Sort by CITN_ID
in descending order
)

# Convert the result to a DataFrame
citations_res = patstat.df(query_citations)

# Display the resulting DataFrame
citations_res
```

Out[2]:

	pat_publn_id	citn_replenished	citn_id	citn_origin	cited_pat_publn_id	cited_appln_id
0	590912575	0	2336	APP	0	482700078
1	590912575	0	2335	APP	0	472639143
2	511879958	0	2285	APP	0	417285474
3	511879958	0	2284	APP	0	417285474
4	496003161	0	2224	APP	0	417285474
...	...	...	...	...	...	...
2405	510930241	0	1	APP	0	489142429
2406	578100589	0	1	APP	0	48371388
2407	291724442	0	1	SEA	0	7184830
2408	471426537	0	1	SEA	0	470413045
2409	489652413	0	1	SEA	0	7626690

2410 rows × 10 columns

Key Fields in the TLS212_CITATION Table

Together, the three fields (`PAT_PUBLN_ID` , `CITN_REPLENISHED` , `CITN_ID`) form a composite primary key that ensures each citation record is unique.

- `PAT_PUBLN_ID` links the citation to a specific patent publication.
- `CITN_REPLENISHED` ensures that any updates to the citation data can be tracked separately from the original record.
- `CITN_ID` identifies the specific citation related to that publication, whether it's a patent or non-patent literature citation.

PAT_PUBLN_ID

This field represents the patent **publication making** the citation, in other words it refers to the **citing** publication. It identifies the patent document that includes a citation to another patent or non-patent literature. `PAT_PUBLN_ID` identifies a specific patent publication. Each patent publication has a unique `PAT_PUBLN_ID` in the `TLS211_PAT_PUBLN` table, which is used in `TLS212_CITATION` to associate citations with that publication.

CITN_REPLENISHED

The `CITN_REPLENISHED` attribute in the PATSTAT database refers to a special type of citation that is "replenished" or copied from one patent publication to another, for example in the context of European Patent Office (EPO) and international (PCT) applications. It is meant to fill in the citation information that might be missing from European publications but is present in the corresponding international publication. When a European patent application (Euro-PCT) is based on an international (PCT) application, the EPO typically does not repeat the citations from the international search report. However, these citations are still relevant to understanding the Euro-PCT application, so PATSTAT "replenishes" the citation list by adding citations from the corresponding international PCT application. It applies to any patent publication where citations are carried over from previous publications, whether they are from the same authority or from an international stage (such as PCT).

In [3]:
```
# Step 1: Query to get citations with CITN_REPLENISHED > 0 and vi
ew authorities
query_replenished_citations_authorities = (
    db.query(
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_replenished,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth,   # Include generating a
uthority field
        TLS201_APPLN.appln_auth  # Add the authority of the cited
application
    )
    .join(TLS201_APPLN, TLS201_APPLN.appln_id == TLS212_CITATION.
cited_appln_id)
    .filter(
        TLS212_CITATION.citn_replenished > 0  # Filter for replen
ished citations
    )
    .order_by(TLS212_CITATION.citn_id.desc())  # Sort by CITN_ID
in descending order
)

# Step 2: Convert the result to a DataFrame
replenished_citations_authorities_res = patstat.df(query_replenis
hed_citations_authorities)

# Step 3: Display the resulting DataFrame
replenished_citations_authorities_res
```

Out[3]:

	pat_publn_id	citn_replenished	citn_id	citn_origin	cited_pat_publn_id	cited_appln_id
0	438941429	414267447	67	APP	0	337892844
1	504203495	488182001	64	APP	0	456163830
2	504203538	488184975	64	APP	0	456163830
3	587262303	560464616	40	APP	0	43354643
4	473305534	445016972	39	APP	0	240243
...	...	...	...	...	...	...
346	415483449	378682213	1	APP	0	352306838
347	437723248	412764517	1	APP	0	380769816
348	531964463	508293364	1	APP	0	905598766
349	577017155	551914838	1	APP	0	905828622
350	424937548	410958001	1	APP	0	267285082

351 rows × 11 columns

CITN_ID

It ensures that each citation, whether it's a patent citation or non-patent literature (NPL) citation, is uniquely identifiable within the context of a single publication. This ID allows multiple citations to be associated with one publication, avoiding duplication and keeping the records organized. The `CITN_ID` is assigned sequentially to distinguish each citation listed in a patent publication, starting from 0. For example, the first citation in a document gets `CITN_ID = 1`, the second gets `CITN_ID = 2`, and so on.

The number is purely a running number and has no special meaning beyond helping to distinguish citations within that particular citing publication.

To analyse the mechanism of `CITN_ID`, you can run a query that retrieves the active citations made by a specific publication of a patent application. In this case, you will focus on the application with the number **473378055** and the publication with the number **515503477**. This query will return a DataFrame containing the publications that are cited by the specified publication, along with the `CITN_ID` for each citation. The `CITN_ID` is a sequential identifier that distinguishes the citations within the same publication, meaning its value increases with each citation.

In [4]:
```
# Query to fetch the data with joins and filters
query = (
    db.query(
        TLS201_APPLN.appln_nr_epodoc,
        TLS201_APPLN.appln_nr,
        TLS211_PAT_PUBLN.appln_id,
        (TLS211_PAT_PUBLN.publn_auth + TLS211_PAT_PUBLN.publn_n
r).label('publn_full_number'),
        TLS211_PAT_PUBLN.publn_kind,
        TLS211_PAT_PUBLN.publn_date,
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_replenished,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth
    )
    .join(TLS211_PAT_PUBLN, TLS201_APPLN.appln_id == TLS211_PAT_P
UBLN.appln_id)
    .join(TLS212_CITATION, TLS211_PAT_PUBLN.pat_publn_id == TLS21
2_CITATION.pat_publn_id)
    .filter
    (TLS201_APPLN.appln_id == 473378055,  # Replace with your app
ln_id
        TLS212_CITATION.pat_publn_id == 515503477  # Filter for s
pecific pat_publn_id
    )  # Replace with your appln_id
    .order_by(TLS212_CITATION.citn_id.asc())
)

# Execute the query and convert the result to a DataFrame
result_df = patstat.df(query)

# Display the resulting DataFrame
result_df
```

Out[4]:

	appln_nr_epodoc	appln_nr	appln_id	publn_full_number	publn_kind	publn_da
0	None	201514753150	473378055	US10337490	B2	2019-07-(
1	None	201514753150	473378055	US10337490	B2	2019-07-(
2	None	201514753150	473378055	US10337490	B2	2019-07-(
3	None	201514753150	473378055	US10337490	B2	2019-07-(
4	None	201514753150	473378055	US10337490	B2	2019-07-(
...	...	...	...	...	...	
142	None	201514753150	473378055	US10337490	B2	2019-07-(
143	None	201514753150	473378055	US10337490	B2	2019-07-(
144	None	201514753150	473378055	US10337490	B2	2019-07-(
145	None	201514753150	473378055	US10337490	B2	2019-07-(
146	None	201514753150	473378055	US10337490	B2	2019-07-(

147 rows × 16 columns

In [5]:
```
# Remove duplicates based on 'cited_pat_publn_id' and 'citn_origin'
distinct_df = result_df.drop_duplicates(subset=['cited_pat_publn_id', 'citn_origin'])

# Optionally, you can check the distinct entries by grouping
grouped = distinct_df.groupby('cited_pat_publn_id').size()

# Display the distinct DataFrame without double counting
distinct_df
```

Out[5]:

	appln_nr_epodoc	appln_nr	appln_id	publn_full_number	publn_kind	publn_da
0	None	201514753150	473378055	US10337490	B2	2019-07-(
1	None	201514753150	473378055	US10337490	B2	2019-07-(
2	None	201514753150	473378055	US10337490	B2	2019-07-(
3	None	201514753150	473378055	US10337490	B2	2019-07-(
4	None	201514753150	473378055	US10337490	B2	2019-07-(
...	...	...	...	...	...	
141	None	201514753150	473378055	US10337490	B2	2019-07-(
142	None	201514753150	473378055	US10337490	B2	2019-07-(
143	None	201514753150	473378055	US10337490	B2	2019-07-(
144	None	201514753150	473378055	US10337490	B2	2019-07-(
145	None	201514753150	473378055	US10337490	B2	2019-07-(

146 rows × 16 columns

CITN_ORIGIN

The `CITN_ORIGIN` attribute provides essential information about the source or phase from which a citation originates in the context of patent documents. In other words, this attribute indicates the provenance of a citation within the patent publication process.

The `CITN_ORIGIN` attribute can take several values, each signifying the origin of the citation:

- APP: Citations introduced by the applicant. This indicates that the applicant referenced this publication during the application process.
- SEA: Citations introduced during the search phase, specifically from a search report. This reflects citations identified during the initial patent search.
- ISR: Citations from the International Search Report. These citations come from the formal report generated during the international patent application process.
- SUP: Citations from the Supplementary Search Report. These citations are additional references provided after the initial search report.
- PRS: "PRe-Search" citations. These are preliminary citations available before official publication, typically applicable to U.S. applications.
- EXA: Citations introduced during the examination phase. This indicates that the examiner has referenced this publication during the examination of the application.
- OPP: Real opposition documents cited by the opposition division, which are included in the published European Patent Specification (EP-B2).
- APL: Citations introduced when an appeal is filed by the applicant, proprietor, or patentee.
- FOP: Citations introduced by the opponent or the proprietor when an opposition has been filed.
- TPO: Citations resulting from Third Party Observations (Art 115 EPC).
- CH2: Citations introduced during the Chapter 2 phase of the Patent Cooperation Treaty (PCT).

In [6]:
```
from sqlalchemy import func  # Import func from sqlalchemy

# Query to fetch distinct CITN_ORIGIN types for a specific public
ation
query_origin_types = (
    db.query(
        TLS212_CITATION.citn_origin,
        func.count(TLS212_CITATION.citn_id).label('citation_coun
t')  # Count of citations for each origin
    )
    .join(TLS211_PAT_PUBLN, TLS212_CITATION.pat_publn_id == TLS21
1_PAT_PUBLN.pat_publn_id)
    .filter(
        TLS211_PAT_PUBLN.pat_publn_id == 515503477  # Replace wit
h your desired patent publication ID
    )
    .group_by(TLS212_CITATION.citn_origin)  # Group by CITN_ORIGI
N to get distinct types
    .order_by(TLS212_CITATION.citn_origin.asc())  # Sort by CITN_
ORIGIN in ascending order
)

# Execute the query and convert the result to a DataFrame
origin_types_df = patstat.df(query_origin_types)

# Display the resulting DataFrame
origin_types_df
```

Out[6]:

	citn_origin	citation_count
0	APP	139
1	SEA	8

To avoid double counting citations due to different CITN_ORIGIN values, you can select distinct citations based on the combination of cited_pat_publn_id and CITN_ORIGIN. Here's how you can filter out duplicates:

Patent Literature

CITED_PAT_PUBLN_ID

The `CITED_PAT_PUBLN_ID` is a crucial surrogate key used to uniquely identify a cited patent publication within patent databases. It plays a significant role in patent literature, where it is necessary to reference other patents and applications that cite a particular document. This attribute serves a vital function by ensuring that each cited publication can be accurately and consistently identified across various sources and applications.

When a patent or application references a publication, this reference signifies that the publication is a document publicly disclosing a specific patent. This could include information such as the patent's claims, specifications, and other relevant data essential for understanding the scope and implications of the patent. The citation process is essential for establishing the context of a patent's claims and for evaluating its novelty and non-obviousness against prior art.

It is important to note that self-citations, where a publication cites itself, are deliberately excluded from this identification process. This exclusion helps maintain the integrity of the citation data and prevents skewed counts of cited publications.

This mechanism facilitates the accurate tracking and analysis of patent citations, contributing to a clearer understanding of the relationships between patents and their cited references in the broader context of patent law and intellectual property.

CITED_APPLN_ID

`CITED_APPLN_ID` is a surrogate key that uniquely identifies a cited patent application. This attribute is used when an application (i.e., a submission for a patent) is being cited. It is independent of any publication referenced by `CITED_PAT_PUBLN_ID`. The `CITED_APPLN_ID` is a significant surrogate key that uniquely identifies a cited patent application within patent databases. This attribute is essential when a patent application references another application, indicating that the cited application is relevant to the context of the citing application. Unlike `CITED_PAT_PUBLN_ID`, which pertains specifically to cited patent publications, `CITED_APPLN_ID` focuses on the unique identification of applications within the patent system.

PAT_CITN_SEQ_NR

The `PAT_CITN_SEQ_NR` is a sequential number assigned to patent citations within a specific combination of publication and citation origin, indicating the order of the citation in that context. It helps to uniquely identify and categorize citations without suggesting duplicate counts, as the numbering resets for each origin of citations.

Non-patent Literature

CITED_NPL_PUBLN_ID

`CITED_NPL_PUBLN_ID` refers to a non-patent-literature being cited, which in turn may contain "hidden" references to patent publications. The `CITED_NPL_PUBLN_ID` is a surrogate key that uniquely identifies non-patent literature (NPL) publications that have been cited within patent documents. This attribute serves as a reference to external sources, such as scientific articles or research papers, that are relevant to the patent application or publication. It can contain up to 32 ASCII characters, and a value of 0 is used to indicate that no NPL publication has been cited. By providing a distinct identifier for each cited NPL, this attribute facilitates the tracking and analysis of references to external literature within the context of patent citations.

NPL_CITN_SEQ_NR

It is a sequence number that uniquely identifies non-patent literature (NPL) citations within a specific series of NPL citations for a given publication or origin combination. Similar to the `PAT_CITN_SEQ_NR`, the numbering for `NPL_CITN_SEQ_NR` starts at 1 for each origin of citations and is sequentially assigned, allowing for easy tracking of NPL references.

This attribute is essential for organizing and referencing NPL citations within patent documents, and it helps distinguish between NPL citations and patent citations. If a citation is not an NPL citation but rather a patent citation, the `NPL_CITN_SEQ_NR` will be set to 0. It's important to note that the sequence number does not indicate the order in which the NPL citations appear but rather serves as a unique identifier for each citation within the specific context of a publication's origin.

The `NPL_CITN_SEQ_NR` is similar to the `PAT_CITN_SEQ_NR` in that both are sequence numbers used to uniquely identify citations within their respective contexts.

Other Attributes

CITN_GENER_AUTH

The attribute `CITN_GENER_AUTH` serves as an identifier for the International Search Authority (ISA) associated with Patent Cooperation Treaty (PCT) search reports, including supplementary search reports, as well as for national or regional search authorities in other contexts. This attribute is represented by a two-character country code, following the WIPO ST.3 standard, and it defaults to spaces when not applicable.

It provides a country code that specifies the authority responsible for conducting the search, thereby indicating which organization reviewed the patent application in question.

- ISR or SUP, then identifying the (Supplementary) International Search Authority (ISA)

or

- SEA, EXA or PRS, then identifying a national / regional search authority.

In [8]:
```
query_citations_with_authorities = (
    db.query(
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth,   # Include generating a
uthority field
        TLS201_APPLN.appln_auth  # Add the authority of the cited
application
    )
    .join(TLS201_APPLN, TLS201_APPLN.appln_id == TLS212_CITATION.
cited_appln_id)
    .filter(
        TLS212_CITATION.citn_gener_auth != '  '  # Filter for non
-null generating authorities
    )
    .order_by(TLS212_CITATION.citn_id.desc())
)


citations_with_authorities_res = patstat.df(query_citations_with_
authorities)


citations_with_authorities_res
```

Out[8]:

	pat_publn_id	citn_id	citn_origin	cited_pat_publn_id	cited_appln_id	pat_citn_seq_nr	c
0	528103587	57	SEA	0	960055248	57	
1	528103587	56	SEA	0	960038473	56	
2	528103587	55	SEA	0	960007628	55	
3	528103587	54	SEA	0	960041687	54	
4	528103587	53	SEA	0	960013042	53	
5	528103587	52	SEA	0	960050466	52	
6	528103587	51	SEA	0	960053250	51	
7	528103587	50	SEA	0	960041813	50	
8	528103587	49	SEA	0	960043456	49	
9	552547296	46	SEA	0	419122235	44	
10	573337737	18	SEA	0	519722239	18	
11	541517071	16	SEA	0	960054903	16	
12	534798305	15	SEA	0	960023472	15	
13	558563181	12	SEA	0	960036671	12	
14	495855058	5	SEA	0	419801180	5	
15	523798264	5	ISR	0	21525091	5	
16	592020905	4	SEA	0	414768222	4	
17	517852504	4	ISR	0	960011210	4	
18	482170247	3	SEA	0	415630242	3	
19	579500166	3	SEA	0	960005734	3	
20	498739268	3	SEA	0	42672659	3	
21	502103472	2	SEA	0	315707591	2	
22	573405462	1	EXA	0	537020359	1	
23	504080149	1	SEA	0	501673157	1	
24	471426537	1	SEA	0	470413045	1	
25	489652413	1	SEA	0	7626690	1	

Forward Citation Analysis

This script is designed to interactively retrieve and display forward citation data related to a specific patent invention, identified by its DOCDB family ID. In the context of patent analysis, forward citations refer to citations made by later patents to a given invention, while backward citations refer to references within a patent to prior works. The terms "forward" and "backward" merely describe the perspective: when Patent A cites Patent B, Patent B is a backward citation for A, but A becomes a forward citation for B. This interconnected relationship provides insight into the influence of an invention over time.

By focusing on forward citations related to a specific DOCDB family (which groups patent documents across different jurisdictions for the same invention), this script enables the user to assess the invention's impact. Forward citations are valuable indicators of an invention's relevance, as they can highlight how widely and in what ways subsequent technologies have built upon it. This approach also allows users to explore citation metadata, such as publication dates, citation origins, and technological classifications. These details can support further analysis, such as identifying trends in innovation, pinpointing groundbreaking inventions, and assessing the invention's influence across technological fields.

To analyze the forward citations of an invention effectively, we need to track citations across all publications and applications associated with the invention’s DOCDB family. Here's the two-step approach:

- Citations Directly to Applications: First, identify citations made directly to any application associated with the invention. These citations target specific applications, providing a first layer of forward citation data that indicates which later patents reference the application itself.
- Citations to Publications Linked to Applications: Next, look at citations that reference the patent publications related to those applications. Since multiple publications can stem from a single application (especially across different jurisdictions), tracking citations at the publication level ensures a broader capture of forward citation activity associated with the invention.

Combining citations from both the application and publication levels creates a comprehensive view of an invention’s forward citations. This approach helps assess the invention's overall impact by revealing how widely and in what ways subsequent technologies cite it across various stages and jurisdictions.

In this interactive environment, users can enter a DOCDB family ID, click a button to retrieve citation data, and explore the results in real-time, making it easier to conduct targeted analyses on an invention's impact in the technological landscape.

In [9]:
```
# Query to find all application IDs with the specified DOCDB fami
ly ID
query_applications = (
    db.query(TLS201_APPLN.appln_id)
    .filter(TLS201_APPLN.docdb_family_id == 56289336 )  # 5628933
6
)

# Execute the query and convert the result to a DataFrame
applications_res = patstat.df(query_applications)

# Step 2: Extract the list of application IDs
application_ids = applications_res['appln_id'].tolist()
```

In [10]:
```
# Step 1: Query to get citations where cited application IDs belo
ng to the DOCDB family
query_cited_application_citations = (
    db.query(
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_replenished,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth
    )
    .filter(TLS212_CITATION.cited_appln_id.in_(application_ids))
# Filter by application IDs
)

# Step 2: Convert the result to a DataFrame
cited_application_citations_res = patstat.df(query_cited_applicat
ion_citations)
```

In [11]:
```
# Step 1: Query to get publication IDs associated with the applic
ation IDs
query_publications = (
    db.query(
        TLS211_PAT_PUBLN.pat_publn_id
    )
    .join(TLS201_APPLN, TLS201_APPLN.appln_id == TLS211_PAT_PUBL
N.appln_id)
    .filter(TLS201_APPLN.appln_id.in_(application_ids))  # Use th
e application IDs
)

# Step 2: Convert the result to a list of publication IDs
publication_ids = [row.pat_publn_id for row in query_publication
s]

# Step 3: Query to filter citations for the publication IDs
query_filtered_citations = (
    db.query(
        TLS212_CITATION.pat_publn_id,
        TLS212_CITATION.citn_replenished,
        TLS212_CITATION.citn_id,
        TLS212_CITATION.citn_origin,
        TLS212_CITATION.cited_pat_publn_id,
        TLS212_CITATION.cited_appln_id,
        TLS212_CITATION.pat_citn_seq_nr,
        TLS212_CITATION.cited_npl_publn_id,
        TLS212_CITATION.npl_citn_seq_nr,
        TLS212_CITATION.citn_gener_auth
    )
    .filter(
        TLS212_CITATION.cited_pat_publn_id.in_(publication_ids)
# Use the publication IDs
    )
)

# Step 4: Convert the result to a DataFrame
filtered_citations_res = patstat.df(query_filtered_citations)
```

In [12]:
```
import pandas as pd

# Step 1: Concatenare i DataFrame con citazioni trovate
combined_citations_res = pd.concat([filtered_citations_res, cited
_application_citations_res], ignore_index=True)

# Step 2: Visualizzare il DataFrame combinato
combined_citations_res
```

Out[12]:

	pat_publn_id	citn_replenished	citn_id	citn_origin	cited_pat_publn_id	cited_appln_id
0	556029637	0	138	APP	473378056	0
1	542922977	0	140	APP	473378056	0
2	545479418	0	138	APP	473378056	0
3	566885436	0	136	APP	473378056	0
4	585634322	0	139	APP	473378056	0
...	...	...	...	...	...	...
62	564110666	0	6	APP	0	473378055
63	590029961	564110666	6	APP	0	473378055
64	545264221	521625720	10	APP	0	473378055
65	537014092	513718600	7	APP	0	473378055
66	513718551	0	7	APP	0	473378055

67 rows × 10 columns

In [13]:
```
import ipywidgets as widgets
from IPython.display import display

# Define widgets
docdb_family_input = widgets.Text(
    description='DOCDB Family ID:',
    placeholder='Enter DOCDB Family ID',
)

button = widgets.Button(description="Retrieve Forward Citations")
output = widgets.Output()

def retrieve_data(docdb_family_id):
    # Step 1: Query application IDs for the given DOCDB family
    query_applications = (
        db.query(TLS201_APPLN.appln_id)
        .filter(TLS201_APPLN.docdb_family_id == docdb_family_id)
    )
    applications_res = [row.appln_id for row in query_application
s]
```

```
    if not applications_res:
        return pd.DataFrame()  # Return empty DataFrame if no app
lications found

    # Step 2: Query to get citations for the application IDs
    citations_data = []

    for appln_id in applications_res:
        # Query 1: Get citations where the cited application ID m
atches
        query_cited_application_citations = (
            db.query(
                TLS212_CITATION.pat_publn_id,
                TLS212_CITATION.citn_replenished,
                TLS212_CITATION.citn_id,
                TLS212_CITATION.citn_origin,
                TLS212_CITATION.cited_pat_publn_id,
                TLS212_CITATION.cited_appln_id,
                TLS212_CITATION.pat_citn_seq_nr,
                TLS212_CITATION.cited_npl_publn_id,
                TLS212_CITATION.npl_citn_seq_nr,
                TLS212_CITATION.citn_gener_auth
            )
            .filter(TLS212_CITATION.cited_appln_id == appln_id)
            )
        citations_data += patstat.df(query_cited_application_cita
tions).to_dict(orient='records')

        # Query 2: Get publication IDs for the application
        query_publications = (
            db.query(TLS211_PAT_PUBLN.pat_publn_id)
            .join(TLS201_APPLN, TLS201_APPLN.appln_id == TLS211_P
AT_PUBLN.appln_id)
            .filter(TLS201_APPLN.appln_id == appln_id)
        )
        publication_ids = [row.pat_publn_id for row in query_publ
ications]

        # Filter citations for the publication IDs
        if publication_ids:
            query_filtered_citations = (
                db.query(
                    TLS212_CITATION.pat_publn_id,
                    TLS212_CITATION.citn_replenished,
                    TLS212_CITATION.citn_id,
                    TLS212_CITATION.citn_origin,
                    TLS212_CITATION.cited_pat_publn_id,
                    TLS212_CITATION.cited_appln_id,
                    TLS212_CITATION.pat_citn_seq_nr,
                    TLS212_CITATION.cited_npl_publn_id,
                    TLS212_CITATION.npl_citn_seq_nr,
                    TLS212_CITATION.citn_gener_auth,
                    TLS211_PAT_PUBLN.publn_date,  # Include publi
```

```
cation date
                    TLS211_PAT_PUBLN.appln_id,
                    TLS201_APPLN.docdb_family_id
                )
                .join(TLS211_PAT_PUBLN, TLS211_PAT_PUBLN.pat_publ
n_id == TLS212_CITATION.pat_publn_id)
                .join(TLS201_APPLN, TLS201_APPLN.appln_id == TLS2
11_PAT_PUBLN.appln_id)
                .filter(TLS212_CITATION.cited_pat_publn_id.in_(pu
blication_ids))
            )
            citations_data += patstat.df(query_filtered_citation
s).to_dict(orient='records')

    # Step 3: Create a DataFrame from the collected data
    combined_citations_df = pd.DataFrame(citations_data)
    combined_citations_df['docdb_family_id'] = combined_citations
_df['docdb_family_id'].astype('Int64')
    combined_citations_df['appln_id'] = combined_citations_df['ap
pln_id'].astype('Int64')
    return combined_citations_df

# Button click handler
def on_button_clicked(b):
    with output:
        output.clear_output()
        docdb_family_id = docdb_family_input.value
        if docdb_family_id:
            # Retrieve data
            combined_citations_df = retrieve_data(int(docdb_famil
y_id))

            # Display data
            if not combined_citations_df.empty:
                display(combined_citations_df)
            else:
                print(f"No citations found for DOCDB Family ID: {
docdb_family_id}")

# Attach button click handler
button.on_click(on_button_clicked)

# Display widgets
display(docdb_family_input, button, output)
```

```
Text(value='', description='DOCDB Family ID:', placeholder='Enter
DOCDB Family ID')

Button(description='Retrieve Forward Citations', style=ButtonStyl
e())

Output()
```

Examples of inventions: 56289336, 34743063