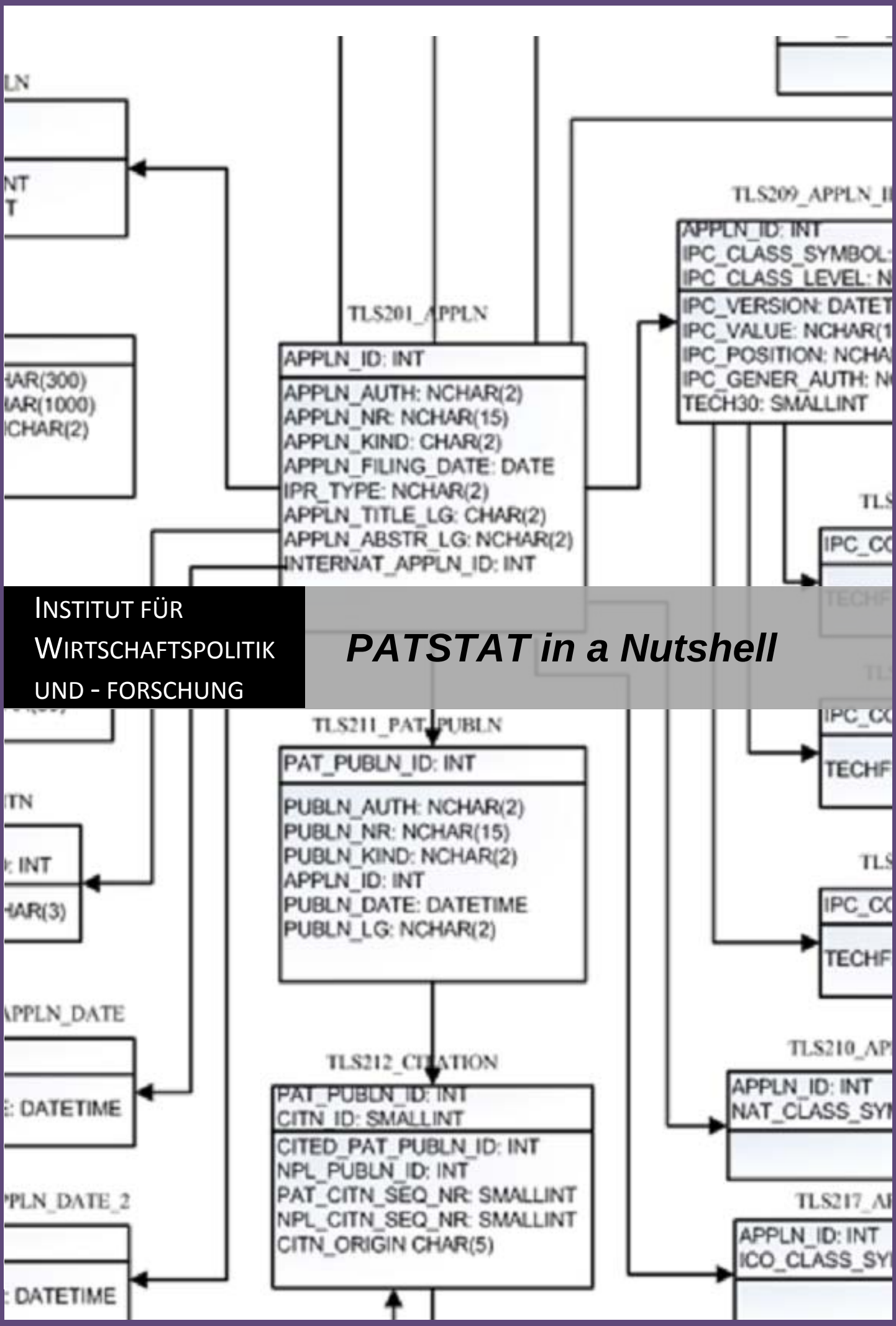


INSTITUT FÜR WIRTSCHAFTSPOLITIK UND - FORSCHUNG

PATSTAT in a Nutshell

PATSTAT in a Nutshell

Eine *kurze* Anleitung zur Nutzung der *EPO Worldwide Patent Database* und eine Einführung in SQL

Juni 2010

Chung Anh Tran, Franziska Beyer, Markus Leible und Timm Teubner
Universität Karlsruhe (TH) - KIT
Institut für Wirtschaftsforschung und -politik (IWW)

Inhaltsverzeichnis

INHALTSVERZEICHNIS 4

ABBILDUNGSVERZEICHNIS 5

TABELLENVERZEICHNIS 5

1 EINFÜHRUNG 6

2 PATSTAT 7

2.1 Hintergrund 7

2.2 Aufbau 7
2.2.1 PATSTAT-IWW – Eine Modifikation 7

3 ARBEITEN MIT *PATSTAT* UND SQL 9

3.1 Relationale Datenbankstrukturen – Eine Kurzeinführung 9

3.2 Structured Query Language (SQL) 9

3.3 Grundsätzliches zur Mengenlehre 15

3.4 Beispiele und Übungen 18

3.5 Exkurs: Validierungstabellen 38
3.5.1 Motivation und Grundidee 38
3.5.2 Aufbau 38
3.5.3 Gewichtung 40
3.5.4 Datensäuberung 40
3.5.5 Datenbank 42
3.5.6 SQL Queries 43
3.5.7 Verwendung 44

4 STANDARDISIERTE ABFRAGEN 45

GLOSSAR 47

LITERATUR 50

Abbildungsverzeichnis

Abbildung 1: PATSTAT-IWW 1.1 8
Abbildung 2: Datenimport Excel 25

Tabellenverzeichnis

Tabelle 1: Beispielabfrage (Tabelle X) 10
Tabelle 2: Kürzel für die PATSTAT Tabellen 11
Tabelle 3: Abfrage-Ergebnis 11
Tabelle 4: Abfrage-Ergebnis 13
Tabelle 5: Beispiel Tabellen für INNER JOIN 14
Tabelle 6: Abfrage-Ergebnis 15
Tabelle 7: Weiterführende Links zu SQL 15
Tabelle 8: tls201_appln 16
Tabelle 9: tls209_appln_ipc 16
Tabelle 10: tls206_person, tls207_pers_appln 16
Tabelle 11: Ergebnis eines INNER JOIN aller Tabellen 17
Tabelle 12: Dimensionen der Klassifizierung 39
Tabelle 13: Beibehaltene Ländercodes 41
Tabelle 14: Angepasste Ländercodes 41
Tabelle 15: Gelöschte Ländercodes 41
Tabelle 16: PATSTAT-Tabellen und Kürzel 42

1 Einführung

Wissenschaftliche Arbeiten und Untersuchungen benötigen eine Argumentationsgrundlage, um ihre Aussagen bzw. Hypothesen zu stützen. Hinter diesen Arbeiten stehen Forschungsfragen, die es gilt mit den richtigen Methoden, Analysen, Daten etc. zu beleuchten. Eine verlässliche Datenquelle ist dafür entscheidend. Die Aussagekraft einer Untersuchung steht und fällt oft mit den Daten. Daher ist eine wichtige Aufgabe eine klare und aussagekräftige Datengrundlage zu schaffen.
Im Bereich der Innovationsforschung sind Patentdaten eine der am häufigsten verwendeten Datenquellen. Denn diese Daten sind nicht nur relativ einfach zu erhalten, sondern in der Regel bereits in einer aufgearbeiteten Form in Datenbanken zu finden. Schließlich muss jedes Patentamt Informationen zu seinen Patenten offenlegen. Patentdaten können zu den unterschiedlichsten Fragestellungen verwendet werden. Diese lassen sich grundsätzlich auf die drei Fundamentaleigenschaften eines Patentes zurückführen. Die rechtliche Funktion, die Informationsfunktion und die Funktion als FuE-Indikator (Grupp (1998)).
Mit der rechtlichen Funktion wird der Schutz eines Patentes auf eine technische Erfindung bezeichnet. Für die Offenlegung einer Erfindung erhält der Patentanmelder ein temporäres Monopol[1], um die Kosten und Mühen, die zur Entwicklung einer Erfindung nötig waren, wieder amortisieren zu können. Somit wird ein Anreiz geschaffen, neues Wissen bzw. neue Produkte zu generieren.
Die Informationsfunktion bezeichnet die Gegenseite des Monopolschutzes. Hier wird das Wissen bzw. die Information auf einem Patent beschrieben, das der Allgemeinheit offengelegt wird. Dieser „Tausch“, Monopolschutz gegen Wissen, ist die entscheidende Funktion eines Patentes. Volkswirtschaftlich gesprochen soll hiermit die gesamte Wohlfahrt der Gesellschaft erhöht werden, da mit dem offengelegten Wissen neues Wissen generiert werden kann und das der Allgemeinheit dient.
Für die Innovationsforschung ist die letzte Funktion, die Funktion als FuE-Indikator, von größter Bedeutung. Das Patent wird dabei verwendet, um den erfolgreichen Ausgang von FuE zu erfassen. Hierzu können die unterschiedlichsten Betrachtungsweisen in Angriff genommen werden. Es ist nicht nur möglich regionale, sektorale und temporale Innovationsaktivitäten zu analysieren. Vielmehr können diese in unterschiedlichsten Aggregationsstufen untersucht werden. D.h. es können Forschungsfragen im mikroökonomischen, mesoökonomischen und makroökonomischen Bereich abgedeckt werden.

Eine bekannte Datenbank zu Patenten ist die EPO Worldwide Statistical Patent Database, die sogenannte PATSTAT Datenbank[2]. Sie beinhaltet Patentdaten zu fast allen Patentämtern der Welt und lässt somit eine globale Analyse von Patenten zu. Um mit PATSTAT jedoch arbeiten zu können, müssen nicht nur einige Grundlagen zu Patentindikatoren bekannt sein, sondern es ist ebenso notwendig einige Grundlage zu relationalen Datenbanken und der Datenbankenabfragesprache SQL zu erlernen. Daher werden im Folgenden diese Grundlagen vermittelt, um einen Einstieg in das Arbeiten mit PATSTAT zu ermöglichen. Diese Einführung beginnt in Kapitel 2 mit einer Anleitung zu der Datenbank PATSTAT. Daraufhin werden grundsätzliche Aspekte von relationalen Datenbanken und SQL erläutert, die mit einigen Übungen, die direkt auf PATSTAT aufbauen, abgerundet werden. Ein Exkurs erläutert die vom IWW erarbeiteten Validierungstabellen. Mit diesen ist es möglich anfängliche, oberflächlichere Abfragen zu kontrollieren. In Kapitel 4 sollen daraufhin wichtige Punkte erarbeitet werden, mit denen Patentabfragen zu einem bestimmten Qualitätsstandard geführt werden sollen. Im letzten Kapitel wird mit zusätzlichen Informationen und fortführenden Hinweisen abgeschlossen.

[1] In der Regel sind das 20 Jahre. Dies kann zwischen unterschiedlichen Patentämtern variieren.
[2] Aktuelle Version am IWW: Oktober 2007

2 PATSTAT

2.1 Hintergrund

PATSTAT ist eine Patentdatenbank mit Informationen aus fast allen Patentämtern der Welt. In erster Linie deckt die Datenbank Informationen zu Patenten ab, wie Anmeldedatum, Patentamt, Erteilung etc. Die Besonderheit von PATSTAT liegt jedoch darin, dass zusätzlich auch Informationen zu Anmeldern, Erfindern, Patentklassifizierung, Abstracts, Patentzitaten und vieles mehr in einer sehr konzentrierten Form erhältlich sind.

Für eine genauere Aufschlüsselung der Daten wird jedoch auf die Beschreibung von PATSTAT verwiesen. Diese ist bei jeder Version beiliegend vorhanden.

2.2 Aufbau

Der Aufbau von PATSTAT basiert auf dem Prinzip der relationalen Datenbanken (mehr dazu im nächsten Kapitel). Die Informationen sind somit verteilt auf unterschiedliche Datenbanktabellen, um eine kleinstmögliche Datenredundanz zu schaffen. Insgesamt hat die ursprüngliche PATSTAT-Version 16 Tabellen, die über 500 Mio. Datensätze enthalten. Um eine schnelle Performance der Abfragen zu garantieren wurde daher die Datenbank vom IWW ausgelagert und durch das Rechenzentrum des KIT betreut. Als Datenbankmanagementsystem wird ein Microsoft SQL Server 2005 verwendet.

Da eine genauere Beschreibung der Datenbankstruktur wiederum in der Anleitung von PATSTAT zu finden ist, beschränkt sich die folgende Beschreibung auf die Modifikationen, die am IWW an PATSTAT vorgenommen wurden.

2.2.1 PATSTAT-IWW – Eine Modifikation

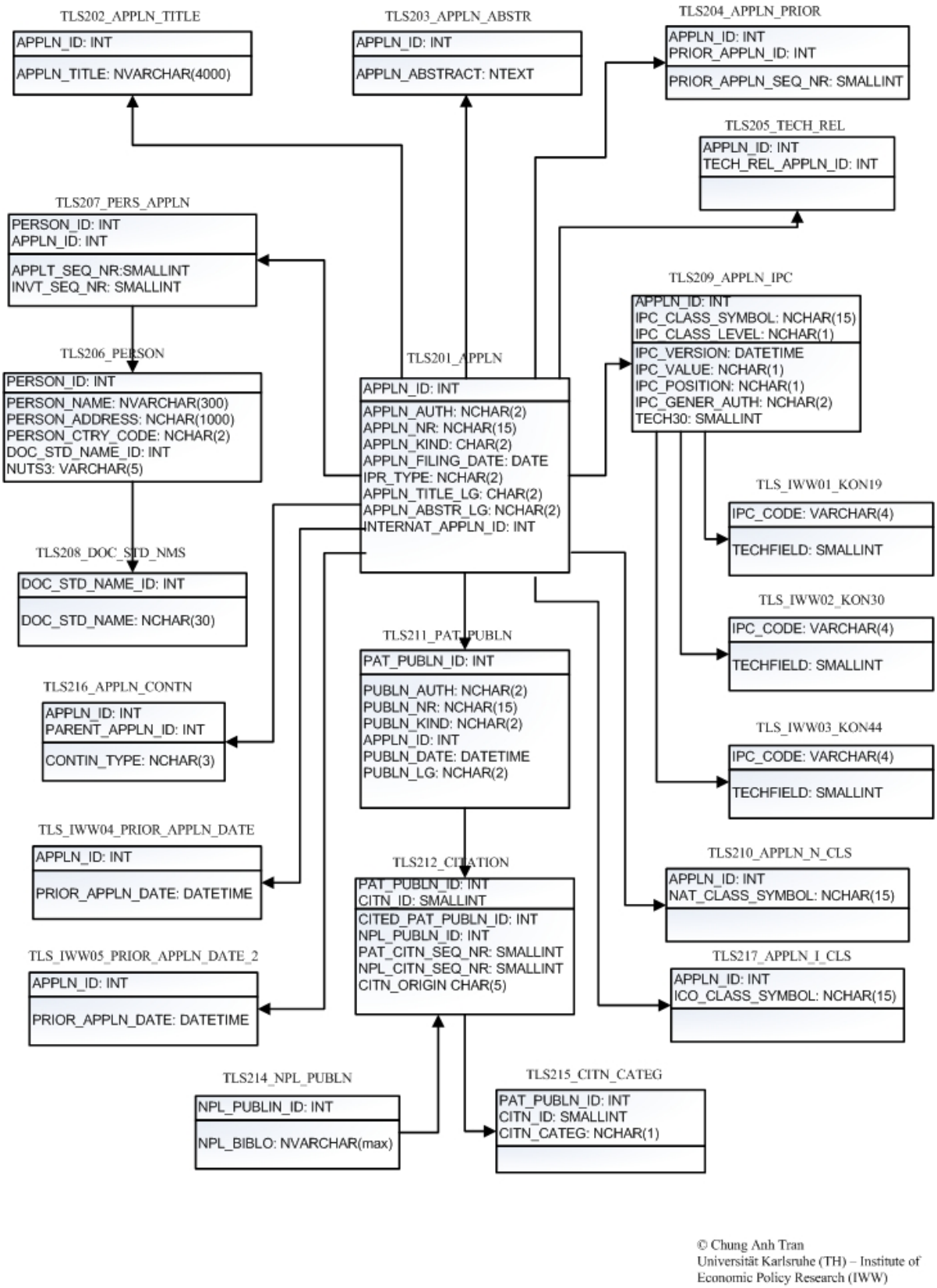
Für die Arbeit am IWW wurde die bestehende Datenbank überarbeitet, um mit den Fragestellungen des Institutes gerecht werden zu können. Dafür wurden Informationen mit den Patentdaten gekoppelt, die es erleichtern direkt auf besondere regionale, sektorale und zeitliche Aspekte zugreifen zu können.

Der regionale Aspekt wird durch die Zuordnung der NUTS3-Codes gegeben. Dazu wurde bei allen Adressen der Anmelder und Erfinder der NUTS3-Code, wenn es möglich ist, zugewiesen. Die Tabelle tls206_person wurde daher mit einer zusätzlichen Spalte versehen. Für sektorale Untersuchungen wurden mehrere Konkordanzen eingeführt, die ein Patent direkt in ein bestimmtes Technologiefeld einordnen. Tls_iww01_kon19, tls_iww02_kon30[3], tls_iww03_kon44[4] und tls_iww04_kon35[5] bauen dabei auf unterschiedliche Ausgangssituationen zur Klassifizierung von Technologien auf.
Für eine Vereinfachung der zeitlichen Betrachtung wurde eine direkte Möglichkeit eingeführt das Prioritätsdatum für alle Patente (tls_iww04_prior_appln_date) bzw. für alle EPO- und WIPO-Patente zu erfragen (tls_iww05_prior_appln_date2). Die zweite Modifikation wurde eingerichtet, um schnellere Abfragezeiten zu generieren, da eine Abfrage über alle Daten sehr zeitintensiv ist.
Eine detaillierte Darstellung der Datenbank ist in

[3] Siehe Schmoch et al. (2003)
[4] Siehe Hinze et al. (1997)
[5] Siehe Schmoch (2008)

Abbildung 1 zu sehen.
Das nächste Kapitel soll nun eine kurze und kompakte Einführung in Datenbanktheorie und SQL geben.

Abbildung 1: PATSTAT-IWW 1.1



3 Arbeiten mit *PATSTAT* und SQL

3.1 Relationale Datenbankstrukturen – Eine Kurzeinführung

Um große Datenmengen zu verwalten, also zu speichern, abzurufen und ggf. neu zu strukturieren sind relationale Datenbankstrukturen[6] unverzichtbar. Im Rahmen der Arbeit mit PATSTAT soll hier eine kurze Einführung dazu gegeben werden.

Zunächst einmal besteht eine Datenbank aus mehreren Tabellen. Die Tabellen sind allgemein so aufgebaut, wie sie nach dem Wortsinn zu vermuten wären. Es gibt eine feste Spaltenstruktur und eine variable Anzahl an Zeilen. Jede Zeile entspricht dabei einem Datensatz, dem sogenannten Tupel. Die Spaltenstruktur, also das Gerüst der Tabelle ändert sich über die Lebensdauer der Tabelle in der Regel nicht, der Inhalt der Tabelle, also die Anzahl und die Werte der Datensätze dagegen sehr wohl. Für die Arbeit mit PATSTAT ist es wichtig zu wissen, dass einige der Tabellen über 20 Millionen Datensätze, also Zeilen enthalten. Eine Bearbeitung „per Hand“ ist also aufgrund der Größe von Vornherein ausgeschlossen.

Die verschiedenen Tabellen der Datenbank stehen in einem gewissen Verhältnis zueinander, siehe Abbildung 1, die ist das Grundprinzip der relationalen Datenbankentheorie. Während zum Beispiel in einer Tabelle eine Liste aller Anmeldungen und einem Verweis auf den Anmelder hinterlegt ist, wird in einer anderen Tabelle eine Liste aller Anmelder und deren Wohnorte gespeichert. Die Informationen sind dabei über logische Schlüssel miteinander verknüpft. Die Verknüpfung der Daten ist dabei so gestaltet, dass Redundanzen und damit assoziierte Probleme wie Einfüge-, Editier- und Löschanomalien vermieden werden. Man sagt auch, die Datenbank ist in Normalform[7]. Darauf soll hier aber nicht näher eingegangen werden, es wird auf die allgemeine Datenbanktheorie verwiesen.

Ganz allgemein erfolgt die eindeutige Identifikation und Zuordnung von Datensätzen über Primär- und Fremdschlüssel. Diese sind in den PATSTAT Tabellen meist als IDs bezeichnet und können daher schnell und leicht identifiziert werden.

Von einer detaillierte Auseinandersetzung mit Datenbankensystemen und Datenbankenmanagementsystemen wird an dieser Stelle abgesehen. Dies würde nicht dem Verständnis und der Arbeit mit PATSTAT dienen. Für weitere Informationen kann aber auf einschlägige Fachliteratur zurückgegriffen werden. Nach den theoretischen Grundlagen soll in den folgenden Abschnitten eine praktische Einführung in die Arbeit mit PATSAT gegeben werden.

3.2 Structured Query Language (SQL)

Da für den Umgang mit der Patentdatenbank eine Grundkenntnis der Datenbankabfragesprache SQL unabdingbar ist, soll dieses kurze Tutorial Nutzern als Einstieg dienen, den Umgang mit SQL insbesondere im Zusammenhang mit der Datenbank zu erlernen. Es werden dabei die grundlegenden Elemente von SQL anhand PATSTAT-spezifischer Beispiele erläutert.

[6] Kemper und Eickler (2004)
[7] Codd (1990)

Am IWW hat sich eingebürgert, die SQL-Befehle groß zu schreiben und die Namen klein.

`SELECT FROM WHERE`

Die mit Abstand wichtigsten Operatoren in SQL sind `SELECT`, `FROM` und `WHERE`, wobei eine typische Abfrage wie folgt aufgebaut ist[8]:

```
SELECT          year, auth, title
FROM            Tabelle X
WHERE           id >= 10 AND id <=15;
```

Tabelle 1: Beispielabfrage (Tabelle X)

		SELECT			
	id	year	auth	title	ipc
	1	1992	EP	#N/A	A
	4	1992	EP	#N/A	A
	6	1992	EP	#N/A	B
	7	1992	EP	#N/A	B
WHERE	10	1992	EP	#N/A	B
	11	1992	EP	#N/A	C
	13	1993	IB	#N/A	D
	15	1993	IB	#N/A	D
	20	1993	IB	#N/A	D
	22	1993	IB	#N/A	E
	24	1993	IB	#N/A	E
	25	1993	IB	#N/A	E

Der `SELECT` Befehl selektiert stets die Spalten einer Tabelle. Dabei können beliebig viele Spalten gewählt werden. Um alle Spalten zu wählen, kann kurz `SELECT *` geschrieben werden. Nichtselektierte Spalten erscheinen im Ergebnis – stets wieder eine Tabelle – nicht. Die Reihenfolge der Spalten wird mit der Reihenfolge der Spaltennamen nach dem `SELECT`-Befehl festgelegt.

Der `FROM` Befehl wählt die eigentliche Tabelle aus, auf die Operationen bezogen werden. Eine Liste der PATSTAT-Tabellen und Bezeichnungen ist in **Tabelle 2** gegeben. Auch hier können mehrere Tabellen ausgewählt werden, diese werden dann in einer Aufzählung durch Kommata getrennt. Besonders bei größeren Abfragen ist die Übersichtlichkeit dieser sehr wichtig. Deshalb sollte jede Tabelle eindeutig und identifizierbar benannt werden. Für den Umgang mit PATSTAT werden folgende dreistelligen Kürzel vorgeschlagen:

[8] Die angegebenen Beispiele sind nicht immer die einzige Lösung. Gerade unter SQL können öfter mehrere Befehl-Codes zum gleichen Ziel führen.

Tabelle 2: Kürzel für die PATSTAT Tabellen

Tabelle		Kürzel
dbo.tls201_APPLN		APP
dbo.tls202_APPLN_TITLE		TTL
dbo.tls203_APPLN_ABSTR		ABS
dbo.tls204_APPLN_PRIOR		PRR
dbo.tls205_TECH_REL		TCH
dbo.tls206_PERSON		PRS
dbo.tls207_PERS_APPLN	*)	PRSAPP
dbo.tls208_DOC_STD_NMS		DOC
dbo.tls209_APPLN_IPC		IPC
dbo.tls210_APPLN_N_CLS		NCL
dbo.tls211_PAT_PUBLN		PUB
dbo.tls212_CITATION		CIT
dbo.tls214_NPL_PUBLN		NPC
dbo.tls215_CITN_CATEG		CCT
dbo.tls216_APPLN_CONTN		CTN
dbo.tls217_APPLN_I_CLS		ICL
IWW-interne Tabellen		
dbo.tls_iww_prior_appln_date		PRI
dbo.tls_iww_kon19		K19
dbo.tls_iww_kon30		K30
dbo.tls_iww_kon44		K40

*) einzige Ausnahme, hier sechs-stelliges Kürzel

Die vollständigen Bezeichnungen der Tabellennamen können in der Patstatanleitung nachgelesen werden.

Der `WHERE` Befehl wertet die Zeilen hinsichtlich einer nachgestellten Bedingung aus. Ist wie im Beispiel oben durch die `WHERE` Klausel gefordert, dass die ID zwischen 10 und 15 liegen muss, so werden eben nur solche Zeilen selektiert; alle anderen Zeilen fallen heraus. Das Ergebnis der obigen Abfrage sähe nun entsprechend wie folgt aus:

Tabelle 3: Abfrage-Ergebnis

	year	auth	title
	1992	EP	#N/A
	1992	EP	#N/A
	1993	IB	#N/A
	1993	IB	#N/A

Die wesentlichen Grundprinzipien sind damit gegeben. Natürlich reicht die Mächtigkeit des `SELECT FROM WHERE` Konstrukts noch bei weitem nicht aus, um komplexe Fragestellungen

zu beantworten. Hierzu werden weitere Operatoren und Konstruktionen benötigt, von denen einige in der Folge behandelt werden sollen.

LIKE

Der LIKE Operator dient dazu, Texte (Strings) auf das Vorkommen bestimmter Begriffe hin zu untersuchen.

```
SELECT          *
FROM            Tabelle X
WHERE           X.ipc_class_symbol LIKE 'E04%';
```

Diese Abfrage würde nun alle Zeilen auswählen, die in der Spalte *ipc_class_symbol* einen Eintrag der Form E04… enthalten, d.h. mit E04 beginnen. Das %-Zeichen steht hierbei als Platzhalter bzw. *Wildcard* für beliebig viele weitere Zeichen. Als Platzhalter für genau ein Zeichen kann der Tiefstrich „_“ verwendet werden. Der LIKE Operator ist besonders für den Umgang mit Text-Strings wie Patent-Abstracts oder -titeln wichtig, weil hier häufig die Texte auf Schlüsselworte etc. hin untersucht werden. Auch für die Klassifizierung der IPC-Codes kann der LIKE-Operator sinnvoll verwendet werden.

COUNT

Mit COUNT wird, wie der Name schon sagt, die Anzahl der gefundenen Einträge gezählt.

```
SELECT          COUNT(*)
FROM            Tabelle X
WHERE           id=1;
```

Mit dieser Abfrage werden alle Zeilen mit id = 1 selektiert, gezählt, und das Ergebnis in Form einer einzigen (aggregierten) Zahl ausgegeben.

GROUP BY

Mit der GROUP BYKlausel werden Abfrageergebnisse anhand von bestimmten Kriterien zusammengefasst. Betrachten wir hierzu noch einmal die Tabelle von oben.

```
SELECT          ipc, COUNT(*)
FROM            Tabelle X
WHERE           1=1
GROUP BY        X.ipc;
```

Der GROUP BYOperator wird dabei im Code noch nach die WHERE Klausel gesetzt. In diesem Beispiel hätte man die WHERE-Bedingung auch weglassen können, da sie keine Einschränkung darstellt; die Trivial-Bedingung 1=1 ist stets erfüllt, so dass sie keine Einschränkung darstellt. Wir gruppieren nun unsere Suchergebnisse anhand der IPC-Sektion, die in der Spalte ipc angegeben ist. Hier ist zu beachten, dass alle Spalten, die im SELECT Teil vorkommen, nicht jedoch im GROUP BY Teil, durch einen Aggregator zusammengefasst werden müssen. Gleiche Werte, hier A, B, C, usw. werden dabei zusammengefasst und kommen nur noch einmal vor – sie werden auf einen einzigen Vertreter aggregiert. Das Ergebnis der obigen Abfrage würde folgendermaßen aussehen:

Tabelle 4: Abfrage-Ergebnis

	ipc	COUNT(*)
	A	2
	B	3
	C	1
	D	3
	E	3

Mögliche Aggregatoren sind `COUNT, SUM, MIN, MAX, AVG`, und viele weitere mehr. Vergleiche dazu **Tabelle** 7: Weiterführende Links zu SQL.

`DISTINCT`

Mit `DISTINCT` wird angegeben, dass nur verschiedenwertige Suchergebnisse in die Ergebnis Tabelle aufgenommen werden.

```
SELECT          DISTINCT ipc
FROM            Tabelle X
WHERE           1=1;
```

Bezogen auf die obige Tabelle wären das die fünf verschiedenen Ausprägungen A bis E. Alle „doppelten“ fallen weg. Wichtig dabei ist, dass sich `DISTINCT` stets auf alle im `SELECT` Teil genannten Spalten bezieht. Mehrdimensionale Tupel müssen also hinsichtlich jedes Wertes gleich sein, um als gleich zu gelten.

`ORDER BY`

Mit `ORDER BY`lässt sich eine Ergebnistabelle anhand von gewünschten Kriterien (Spalten) sortieren.

```
SELECT      *
FROM        dbo.tls206_person PRS
ORDER BY    person_ctry_code ASC, person_id DESC;
```

Dabei wird der `ORDER BY` Term stets ganz ans Ende des Queries gestellt, im Falle der gemeinsamen Verwendung mit `GROUP BY`also auch noch nach diesem Ausdruck. Ist eine Sortierung anhand mehrerer Kriterien gewünscht, werden diese durch Kommata getrennt aufgelistet. Die Priorität der Sortierung richtet sich dabei nach der Reihenfolge der Aufzählung. Die Schlüsselworte `ASC` und `DESC` geben an, ob auf- oder absteigend sortiert werden soll.

`YEAR`

Mit `YEAR` können Datumsinformationen, die in der Datenbank stets im etwas sperrigen DATE-Format hinterlegt sind, auf das Kalenderjahr reduziert werden. Dabei wird der Funktion `YEAR` der entsprechende Verweis im Argument übergeben:

```
SELECT      APP.appln_id, YEAR(APP.appln_filing_date)
FROM        dbo.tls201_appln APP;
```

Diese Vereinfachung dient vor allem der geclusterten Zuordnung von Patenten im Kontext zeitlicher Analysen und kann als etabliert angesehen werden.

INSERT INTO

Mit `INSERT INTO` können zuvor erstellte Tabellen mit Inhalt gefüllt werden. Oft ist es zweckmäßig, Zwischenergebnisse in zusätzlichen Tabellen in der Datenbankstruktur abzulegen, um ggf. später wieder schnell darauf zurückgreifen zu können, oder um ein hohes Maß an Ordnung und Übersichtlichkeit zu gewährleisten. Oft ist ein solches Vorgehen aufgrund der Aufgabenstellung sogar notwendig, da die Möglichkeiten einer einzigen Abfrage begrenzt sind und daher „Zwischenschritte“ eingebaut werden müssen. In diesem Fall empfiehlt es sich, zusätzliche – temporäre – Tabellen anzulegen. Soll nun das Ergebnis eines Queries direkt in eine solche Tabelle geschrieben werden, geschieht dies mit `INSERT INTO`. Wichtig: Dabei müssen die Tabellenstruktur und das Ausgabeformat des Queries genau übereinstimmen!

```
INSERT INTO titemp_SBS01
SELECT      APP.appln_id,
            APP.appln_auth

FROM        dbo.tls201_appln APP;
```

Der `INSERT INTO` Befehl mit der Tabelle, in die eingefügt werden soll, wird dem Querie dabei einfach voran gestellt. Das Erstellen eigener Tabellen erfolgt dabei in der Regel direkt im SQL Server Management Programm und wird an dieser Stelle nicht behandelt.

INNER JOIN

Der `INNER JOIN` ist das am häufigsten verwendete Schlüsselwort, um Tabellen zu kombinieren.

Tabelle 5: Beispiel Tabellen für INNER JOIN

X	id	year
	1	1992
	2	1992
	3	1993
	4	1993

Y	id	ipc
	1	A
	3	A
	3	B
	5	C

```
SELECT          X.id, X.year, Y.id, Y.ipc
FROM            Tabelle X
INNER JOIN      Tabelle Y
ON              X.id = Y.id
WHERE           1=1;
```

Die Tabellen X und Y werden hierbei mit einem `INNER JOIN` zusammengefügt. Zu jedem `INNER JOIN` gehört eine `ON`-Bedingung, die angibt, welche Kriterien für die Kombination zweier Datensätze aus den beiden Tabellen erfüllt sein müssen. Man kann sich den `INNER JOIN` folgendermaßen vorstellen. Jede Zeile von Tabelle X wird mit jeder Zeile von Tabelle Y kombiniert und im Falle, dass alle Bedingungen aus `ON`- und `WHERE`-Klausel erfüllt sind, beibehalten, ansonsten verworfen. Aus den 4 x 4 = 16 sich ergebenden Zeilen erfüllen genau

drei die Bedingung, dass X.id und Y.id identisch sind. Das Ergebnis dieses `INNER JOIN` sähe deshalb wie folgt aus:

Tabelle 6: Abfrage-Ergebnis

	X.id	X.year	Y.id	Y.ipc
	1	1992	1	A
	3	1993	3	A
	3	1993	3	B

Natürlich können JOINS auch mehrfach hintereinander und geschachtelt verwendet werden.

Prinzipiell sollte stets der `INNER JOIN` verwendet werden. Die alternative Möglichkeit, das oben angesprochene „Kreuzprodukt“ der Tabellen durch einfaches Selektieren zu bilden, birgt große Nachteile und erhöht den Rechen- und damit Zeitaufwand ungemein. Dieses Vorgehen sollte daher vermieden werden!

Weiterführende Tutorials:
Für weitere Informationen zur Verwendung von SQL können Online-Tutorials zur Hilfe genommen werden. Es stehen im Internet eine Vielzahl guter Lehrgänge und Online-Handbücher zu den Grundelementen und speziellen Funktionen von SQL – in aller Regel kostenlos – zur Verfügung. Dem Vertiefungsgrad sind dabei keine Grenzen gesetzt. Einige Beispiele sind in Tabelle 7 aufgeführt.

Tabelle 7: Weiterführende Links zu SQL

	URL	Sprache	Art
1	http://www.sql-und-xml.de/sql-tutorial/	DE	Tutorial
2	http://sql.1keydata.com/de/	DE	Tutorial
3	http://www.sql-tutorial.net/	EN	Tutorial
4	http://www.sql-und-xml.de/server-daten/sql-befehle/	DE	Referenz
5	http://www.oreilly.de/catalog/sqlnutger/chapter/ch04.html	DE	Referenz

3.3 Grundsätzliches zur Mengenlehre

Analysen mit PATSTAT können riesige Datenmengen enthalten. Ergebnistabellen mit mehreren Millionen Zeilen sind keine Seltenheit.

Dies ist auf die Struktur und den Aufbau einer relationalen Datenbank zurückzuführen. Werden zwei Tabellen mit Hilfe eines INNER JOIN kombiniert, folgt als Ergebnis eine Tabelle, die sowohl die ursprünglichen Elemente der einen wie auch die der anderen Tabelle enthält. Bei der Kombination dieser Tabelle mit einer dritten Tabelle, wächst die neue Tabelle wieder um die Elemente der dritten Tabelle.

Da es bei einer Patentanalyse je nach Komplexität der Abfrage durchaus vorkommen kann, dass sechs oder mehr Tabellen mit Hilfe von Joins verknüpft werden müssen, sind die Tabellen mit denen das Datenbankmanagement-System intern umgehen muss sehr groß. Auch

die spätere Nutzung der Ergebnistabellen ist nur wenig sinnvoll, wenn sie in einem Übermaß Elemente enthält, die für die eigentliche Analyse nicht gebraucht werden.

Das folgende Beispiel soll nun den Sachverhalt der auftretenden Matrixmultiplikationen näher erläutern:

In einem vereinfachten Fall wird angenommen, dass lediglich zwei Patente in der Datenbank gespeichert sind. Diese sind an unterschiedlichen Patentämtern angemeldet worden. Patent 1 wurde am Europäischen Patentamt und Patent 2 am Deutschen Patent- und Markenamt angemeldet (siehe Tabelle 8).

Tabelle 8: tls201_appln

tls201_appln	
appln_id	appln_auth
1	EP
2	DE

Des Weiteren decken beide Patente jeweils mehrere verschiedene IPC-Klassen ab und wurden folglich auch separat in diesen Kategorien angemeldet. Patent 1 wurde in den IPC-Klassen A61B, G01N und G10L angemeldet, während Patent 2 in den Klassen B01J und F26B angemeldet wurde (siehe Tabelle 9).

Tabelle 9: tls209_appln_ipc

tls209_appln_ipc	
appln_id	ipc_class_symbol
1	A61B
1	G01N
1	G10L
2	B01J
2	F26B

Hinsichtlich der Erarbeitung der Patente waren insgesamt fünf unterschiedliche Personen beteiligt, die Personen mit den IDs 1-3 an Patent 1 und die Personen mit den IDs 4 und 5 an Patent 2. Dabei ist Person 1 lediglich Anmelder, nicht aber (Ko)-Erfinder von Patent 1, was sich an der Inventor-Nummer 0 zeigt. Die anderen vier beteiligten Personen hingegen sind Erfinder der jeweiligen Patente.

Tabelle 10: tls206_person, tls207_pers_appln

tls206_person			tls207_pers_appln		
appln_id	person_id		appln_id	person_id	invt_seq_nr
1	1		1	1	0
1	2		1	2	1
1	3		1	3	2
2	4		2	4	1
2	5		2	5	2

Nun wird ein `INNER JOIN` von allen Tabellen durchgeführt. Die nächste Abbildung zeigt das Ergebnis der Operation.

```
SELECT    APP.appln_id, appln_auth, PRS.person_id, invt_seq_nr,
          ipc_class_symbol
FROM      tls201_appln APP INNER JOIN tls207_pers_appln PRSAPP ON
          APP.appln_id = PRSAPP.appln_id INNER JOIN tls209_appln_ipc IPC ON
          APP.appln_id = IPC.appln_id INNER JOIN tls206_person PRS ON
          PRSAPP.person_id = PRS.person_id
ORDER BY  APP.appln_id, invt_seq_nr, ipc_class_symbol;
```

Dabei wird ersichtlich, dass sich die Menge an ausgegebenen Zeilen erheblich vermehrt hat. Waren es zu Beginn vermeintlich „nur“ zwei unterschiedliche Patente, so hat man nun bereits 13 verschiedene Zeilen. Dies ist darauf zurückzuführen, dass es bei Patent 1 zu einer Matrixmultiplikation der 3 beteiligten Personen mit den 3 verschiedenen IPC-Klassen kommt. Somit ergeben sich alleine neun unterschiedliche Zeilen. Da das Patentamt (EP) ein eindeutiges Attribut ist, ergibt sich hier keine weitere Zunahme an Ausgabezeilen.

Tabelle 11: Ergebnis eines INNER JOIN aller Tabellen

Ergebnis				
appln_id	appln_auth	person_id	invt_seq_nr	ipc_class_symbol
1	EP	1	0	A61B
1	EP	1	0	G01N
1	EP	1	0	G10L
1	EP	2	1	A61B
1	EP	2	1	G01N
1	EP	2	1	G10L
1	EP	3	2	A61B
1	EP	3	2	G01N
1	EP	3	2	G10L
2	DE	4	1	B01J
2	DE	4	1	F26B
2	DE	5	2	B01J
2	DE	5	2	F26B

Patent 2 schlägt in dem Beispiel mit vier Zeilen zu Buche, die daraus resultieren, dass sich eine Matrixmultiplikation der 2 beteiligten Personen mit den 2 unterschiedlichen IPC-Klassen ergibt. Das Patentamt (DE) ist in diesem Fall wieder eindeutig und führt somit nicht zu weiteren Zeilen in der Ausgabetabelle. Ein `COUNT` -Befehl würde hier bei falschem Einsatz schnell falsche Ergebnisse liefern. Daher muss die Wahl der gewünschten Ausgabevariablen durchdacht erfolgen. Oftmals ergibt sich ein umfangreicheres Ergebnis, als gewünscht. Wie an den einzelnen Ausgangstabellen zu sehen ist, ist es oftmals sinnvoll, eine Anfrage in mehrere Teilabfragen aufzuspalten. Eine Analyse der IPC-Klassen anhand der Tabelle tls209_appln_ipc ist sehr viel einfacher, als wenn lediglich die Ergebnistabelle vorhanden ist.

3.4 Beispiele und Übungen

Um die Grundlagen zu verfestigen sollen in diesem Abschnitt Beispiele aufgeführt werden. Die folgenden Beispiele und Aufgaben dienen zur Übung und Übersicht für das Arbeiten mit PATSTAT. Dabei ist zu berücksichtigen, dass es bei der Lösung der Aufgaben stets mehrere Lösungswege gibt. Zunächst wird ein einfaches Beispiel betrachtet, welches im Folgenden erweitert wird, so dass der Aufbau einer Patentabfrage verständlich wird.

Beispiel 1:

Gesucht werden die Patente, die bei der EPO (European Patent Office) zwischen den Jahren 1999 und 2005 angemeldet wurden (Anmeldenummer = appln_id). Es sollen die Anmeldenummern ausgegeben werden.

```
SELECT      APP.appln_id
FROM        dbo.tls201_appln APP
WHERE       APP.appln_auth LIKE 'EP'
AND         YEAR(APP.appln_filing_date) BETWEEN 1999 AND 2005;
```

Lösung:

appln_id
14531092
14531117
14531124
14531131
14531142
14531256
14531260
14676274
15010304
15240868
16279634
16280784
...

Dies ist ein Ausschnitt der Lösungsmenge, da die Liste 917785 Zeilen umfasst. An dieser Stelle sind nur die ersten 12 ‚appln_id' angegeben.
Bei der Abfrage wird immer eine andere Reihenfolge ausgegeben. Um eine gleich bleibende Sortierung zu erhalten, müsste noch eine Zeile am Schluss hinzugefügt werden.

```
ORDER BY APP.appln_id ASC
```

Mit `ORDER BY` kann angegeben werden, nach was sortiert werden soll (siehe S.13). Das `ASC` (ascending) gibt hierbei die ‚appln_ids' aufsteigend an. Da das der Standard ist, kann das `ASC` auch weggelassen werden. `DESC` gibt dabei absteigende Sortierung an.

Ergebnis (mit `ORDER BY`):

appln_id
14423554
14423555
14423556
14423557
14423558
14423559
14423560
14423561
14423562
14423563
14423564
14423565
...

Für Tabelle `dbo.tls201_appln` wird das Tabellenalias APP angewendet, da es sehr lang wäre, den ganzen Tabellennamen anzuhängen. Ohne den Aliasnamen sähe die erste Zeile wie folgt aus:

```
SELECT dbo.tls201_appln.appln_id
```

Die Zeile ist deutlich länger. Bei der Auswahl mehrerer Spalten wäre die gesamte Abfrage sehr unübersichtlich. Die Tabelle mit den Abkürzungen befindet sich auf Seite 11.

Zusatz: Um sich einen Überblick über die Tabelleninhalte zu verschaffen, kann folgende Abfrage verwendet werden:

```
SELECT TOP 10 *
FROM dbo.tls201_appln;
```

Beispiel 2:

Mit einer kleinen Veränderung können diese Patente gezählt werden. Die Abfrage sieht dann folgendermaßen aus:

```
SELECT      COUNT(APP.appln_id) AS Anzahl_appln_id
FROM        dbo.tls201_appln APP
WHERE       APP.appln_auth LIKE 'EP'
AND         YEAR(APP.appln_filing_date) BETWEEN 1999 AND 2005;
```

Ergebnis:

Anzahl_appln_id
917785

Mit dem AS – Alias wird eine Spalte eingeführt, die den Namen ‚Anzahl_appln_id' trägt. Dieser Name kann frei gewählt werden.
Wenn die Datensätze Bedingungen erfüllen sollen, können Aggregatfunktionen verwendet werden, wie COUNT (Name), MIN (Name), MAX (Name), AVG (Name).

Beispiel 3:

Es wird die Anzahl der Patente, die beim Europäischen Patentamt (EPO) angemeldet sind, gesucht, unter der Bedingung, dass sie auch in der IPC Klassifizierungstabelle (dbo.tls209_appln_ipc)stehen:

```
SELECT          COUNT(IPC.appln_id) AS Anzahl_appln_id
FROM            dbo.tls209_appln_ipc IPC
WHERE           IPC.appln_id IN
                (SELECT  APP.appln_id
                FROM            dbo.tls201_appln APP
                WHERE           APP.appln_auth LIKE 'EP'
                AND             YEAR(APP.appln_filing_date) BETWEEN 1999
                                AND 2005);
```

Ergebnis:

Anzahl_appln_id
6156981

Hier wird eine Unterabfrage verwendet. Die innere Abfrage liefert eine Menge von Datensätzen zurück. Die Datensätze sind somit eine Selektion der gewählten Tabelle. Auf diesen ausgewählten Datensätzen arbeitet dann die äußere Abfrage. Der Operator IN vergleicht die Spaltenwerte aus der Spalte ‚appln_id' mit jedem Element aus der inneren Abfrage.

Das Ergebnis von 6.156.981 Anmeldungen ist deutlich größer als im vorangegangenen Beispiel mit 917.785. Dies ist der Fall, da eine ‚appln_id' in mehreren IPC-Klassen/Gruppe/Untergruppen (International Patent Classification) vertreten sein kann.

Beispiel 4:

Gesucht werden nun die Anzahl der Patente, die beim Europäischen Patentamt 1986 angemeldet wurden. Dabei sollen nur die Anmelder und nicht die Erfinder berücksichtigt werden.

```
SELECT      COUNT(APP.appln_id)
FROM        dbo.tls201_appln APP INNER JOIN dbo.tls207_pers_appln PRSAPP ON
            APP.appln_id = PRSAPP.appln_id
WHERE       APP.appln_auth LIKE 'EP'
AND         PRSAPP.applt_seq_nr > 0
AND         YEAR(APP.appln_filing_date) = 1986;
```

Diesmal wird ein INNER JOIN verwendet. Die Tabellen werden nur dann zusammengeführt, wenn die angegebenen Kriterien komplett erfüllt sind. Sobald eines der Kriterien nicht erfüllt ist, entsteht kein Datensatz in der Ergebnismenge.
Der INNER JOIN kann auch implizit geschrieben werden:

```
FROM        dbo.tls201_appln APP, dbo.tls207_pers_appln PRSAPP
WHERE       APP.appln_id = PRSAPP.appln_id
```

Es ist zu erwähnen, dass die Reihenfolge in der die Tabellen genannt werden beim `INNER JOIN` keine Rolle spielt. Beim Join können die Bedingungen durch logische Operatoren wie `AND`, `OR` oder `NOT` verfeinert werden. Gegebenenfalls sind weitere Varianten des JOINS nötig, um spezifische Aufgabenstellungen zu beantworten. Für detaillierte Erklärungen von z.B. LEFT oder RIGHT JOIN wird auf SQL-Tutorials verwiesen.

Ergebnis:

Anzahl_appln_id
46881

Zusatz: Anmelder sind diejenigen, die eine appln_id besitzen und deren applt_seq_nr >0 ist. Erfinder sind diejenigen, die eine person_id besitzen und deren invt_seq_nr>0 ist.

Beispiel 5 (Einordnung nach Technologie 30):

Gesucht werden die Patentanmeldungen zwischen 1994 und 1996, die in das Technologiefeld 30 (Bauwesen) fallen. Dabei sollen die Spalten mit der ‚appln_id', den relevanten Daten der Person, die Anmeldernummer (applt_seq_nr), die Erfindernummer (invt_seq_nr) und das Anmeldedatum aufgeführt und in einer geeigneten Weise sortiert werden. Es sollen weiterhin nur Personen (person_name) aufgeführt werden, die eine Adresse (person_address) besitzen.

```
SELECT      PRSAPP.appln_id, PRS.person_id, PRS.person_name,
            PRS.person_address, PRS.person_ctry_code,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr,
            YEAR(APP.appln_filing_date)
FROM        dbo.tls206_person PRS, dbo.tls207_pers_appln PRSAPP,
            dbo.tls201_appln APP
WHERE       PRS.person_id = PRSAPP.person_id
AND         PRSAPP.appln_id = APP.appln_id
AND         YEAR(APP. appln_filing_date) BETWEEN 1994 AND 1996
AND         PRSAPP.appln_id IN
            (SELECT IPC.appln_id
            FROM dbo.tls209_appln_ipc IPC
            WHERE IPC.tech30 = 30)
AND         PRS.person_address <> ' '
ORDER BY    YEAR(APP. appln_filing_date), PRSAPP.appln_id,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr;
```

Ergebnis:

appln_id	person_id	person_name	person_address	person_ctry_code	applt_seq_nr	invt_seq_nr	Year
14407220	1281258	ARNO ILVES	Aia 42, EE3300 Kuressaare, ESTONIA	EE	1	1	1994
14407220	18661341	MATTIA SEPPEL	Pikk 60A-12, EE3300 Kuressaare, ESTONIA	EE	2	2	1994
14407220	1363170	AS SIMA V	Aia 42, EE3300 Kuressaare, ESTONIA	EE	3	0	1994
14407225	25877811	SEIJA ROEMAE	Kuhilaskuja 6 B 10, 80140 Joensuu, FINLAND	FI	0	1	1994
14407225	16269076	KYAESTI NEVALAINEN	Kalevankatu 13A B 10, 80110 Joensuu, FINLAND	FI	0	2	1994
14407225	22352509	PEKKA VEHVILOEINEN	Liperintie 119, 83100 Liperi, FINLAND	FI	0	3	1994
14407225	235457	ABLOY SECURITY LTD OY	Rajasampaanranta 2, FIN-00560 Helsinki, FINLAND	FI	1	0	1994
14407235	13827979	JUKKA PITKOENEN	Sahamyllynkatu 38 , 80160 Joensuu, FINLAND	FI	0	1	1994
14407235	235457	ABLOY SECURITY LTD OY	Rajasampaanranta 2, FIN-00560 Helsinki, FINLAND	FI	1	0	1994
14407237	20761764	NIELS ADELHOLM LARSEN	21B Pilemosevej, DK-2610 R°dovre, DENMARK	DK	0	1	1994
14407237	29905221	V.KANN RASMUSSEN INDUSTRI A/S	10 Tobaksvejen, DK-2860 S°borg, DENMARK	DK	1	0	1994
14407252	327019	ADOLF AMBROSCH	Wiesenstrasse 39, D-64331 Weiterstadt, GERMANY	DE	0	1	1994
14407252	27743410	SUEBA COOPERATION GESELLSCHAFT FUER BAUFORSCHUNG, BAUENTWICKLUNG UND FRANCHISINGMBH	Neustadter Strasse 5+7, D-64331 Weiterstadt, GERMANY	DE	1	0	1994
14407253	12175549	ILMAR SAUL	Kudina küla, Palamuse vald, EE2362 Jogevamaa, ESTONIA	EE	1	1	1994

Anhand dieses Beispiels ist sehr gut zu sehen, dass es mehrere Anmelder und mehrere Erfinder zu einem Patent geben kann. Der Erfinder kann auch gleichzeitig der Anmelder sein. Dies ist bei der Auswertung der Patentanmeldungen zu berücksichtigen (siehe 3.5.3 Gewichtung).

Beispiel 6 (Einordnung nach Technologie 44):

Gesucht werden die Patentanmeldungen zwischen 1994 und 1996. Im Gegensatz zum vorherigen Beispiel soll hier aus der Konkordanz 44 ausgewählt werden und zwar die Technologiefelder zwischen 28 und 41. Auswahl und Sortierung der Spalten erfolgt wie in Beispiel 5.

```
SELECT      PRSAPP.appln_id, PRS.person_id, PRS.person_name,
            PRS.person_address, PRS.person_ctry_code,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr,
            YEAR(APP.appln_filing_date)
FROM        dbo.tls206_person PRS, dbo.tls207_pers_appln PRSAPP,
            dbo.tls201_appln APP
WHERE       PRS.person_id = PRSAPP.person_id
AND         PRSAPP.appln_id = APP.appln_id
AND         YEAR(APP.appln_filing_date) BETWEEN 1994 AND 1996
AND         PRSAPP.appln_id IN
            (SELECT     IPC.appln_id
            FROM        dbo.tls209_appln_ipc IPC,
                        dbo.tls_iww03_KON44 K44
            WHERE       SUBSTRING(IPC.ipc_class_symbol,1,4)= K44.IPC_code
                        collate SQL_Latin1_General_CP850_CI_AS
            AND         K44.TechField BETWEEN 28 AND 41)
ORDER BY    YEAR(APP.appln_filing_date), PRSAPP.appln_id,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr;
```

Der Unterschied zu Beispiel 5 ist, dass in der Unterabfrage ein Vergleich durchgeführt werden muss, da in die Tabelle *dbo.tls209_appln_ipc* nur die Tech30 aufgenommen wurden. Mit `SUBSTRING(IPC.ipc_class_symbol,1,4)` werden die ersten 4 Zeichen ausgewählt und mit den IPC_codes in Tabelle *dbo.tls_iww03_KON44* verglichen. Das `Collate` legt fest die Sortierungsreihenfolge fest. (näheres unter technet.microsoft.com)

Ergebnis:

appln_id	person_id	person_name	person_address	person_ctry_code	applt_seq_nr	invt_seq_nr	Year
28	11067180	HEROUNI, SOUREN PARISOVICH		AM	1	1	1994
684	2614745	BLACK CHRISTOPHER		US	1	1	1994
684	29258920	TOSI PIERRE-FRANCOIS		US	2	2	1994
684	1450758	ATKIN ANDREW		US	3	3	1994
684	16649393	LAZARTE JAIME E		US	4	4	1994
684	20747752	NICOLAU YVES CLAUDE		US	5	5	1994
684	11536159	HONE BASIL T		US	6	6	1994
694	10058417	HAACK AN KARL WERNER		DE	1	1	1994
726	9573497	GRANELL JAIME BAUCELLS			1	0	1994
727	18331173	MARTENS HARALD AAGAARD		NO	0	1	1994
727	23656209	REBERG JAN OTTO		NO	0	2	1994
727	12083712	IDT INC.		US	1	0	1994
728	17024853	LEVESQUE YVON		CA	0	1	1994
728	5010035	COTE SYLVAINE		CA	0	2	1994
728	8645548	GALLAGHER DENIS		CA	0	3	1994
728	13543221	JOHNSON & JOHNSON, INC.		CA	1	0	1994

(Ausschnitt von 4343704 Zeilen)

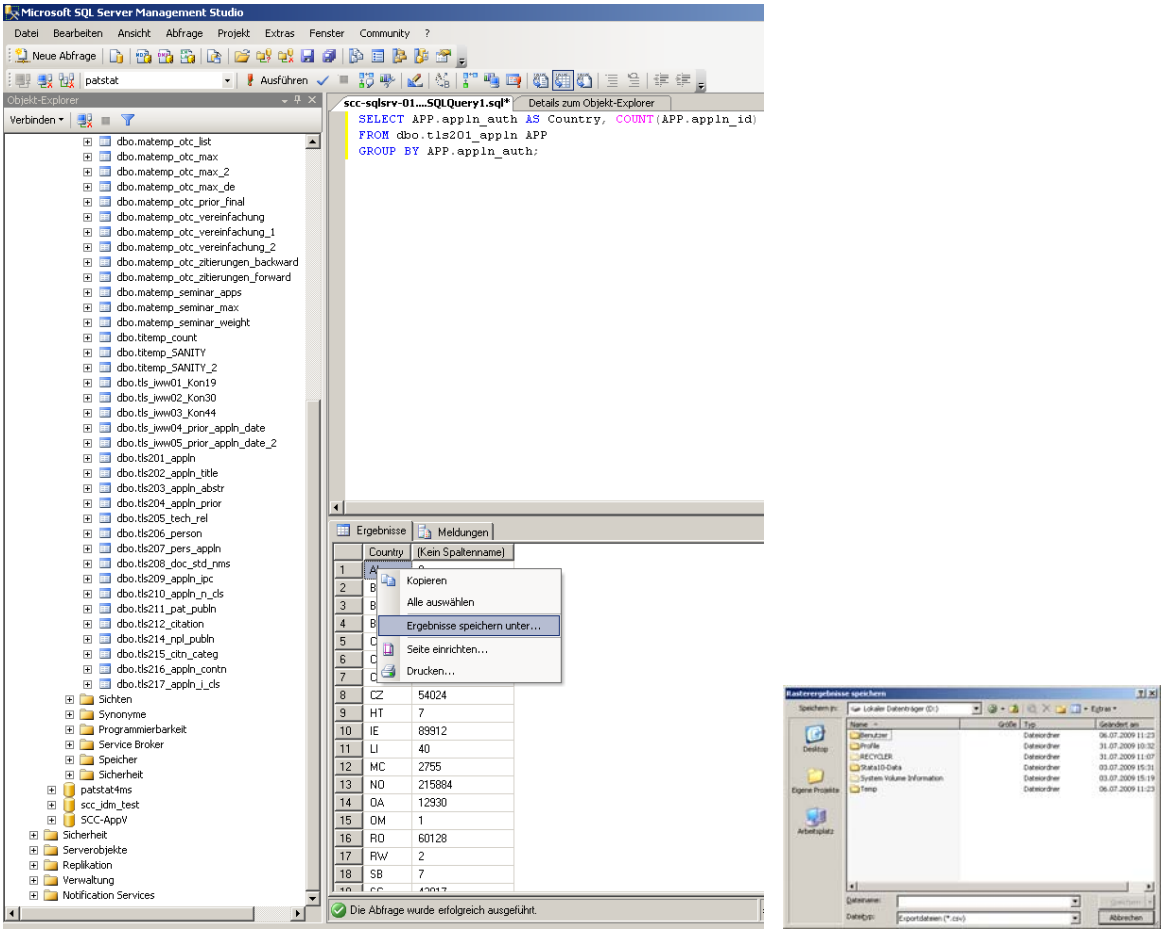
Hinweis:
Wenn die Daten zu groß sein sollten, um sie direkt in Excel zu speichern, müssen sie geteilt werden (oder es sollte eine Möglichkeit gefunden werden, die Abfrage anders zu formulieren oder die Abfrage unterteilt zu gestalten). Die Abfrage kann z.B. unterteilt werden, indem die Jahre einzeln abgefragt werden (nicht von 1999-2009, sondern 1999, 2000, ..., 2009).

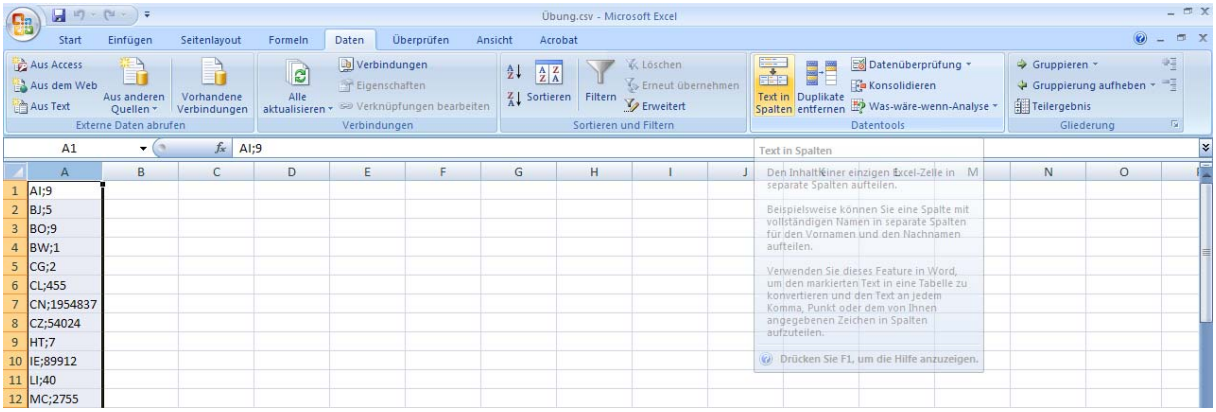
Um die Daten abzuspeichern gibt es folgende Vorgehensweise:

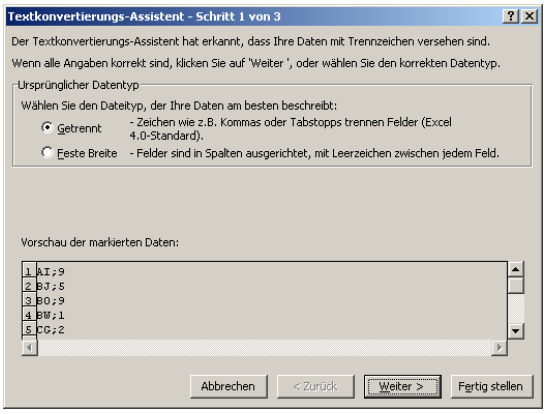
- ➔ Rechte Mausklick auf „Ergebnisse speichern unter...“ in der Ergebnistabelle des Microsoft SQL Server Management Studio
- ➔ Daten als *.csv abspeichern
- ➔ *.csv-Datei in Excel öffnen
- ➔ Unter Daten „Text in Spalten“ anklicken
- ➔ „Getrennt“ ankreuzen, auf „weiter“ klicken
- ➔ Z.B. „Semikolon“ ankreuzen
- ➔ Datei als *.xls speichern

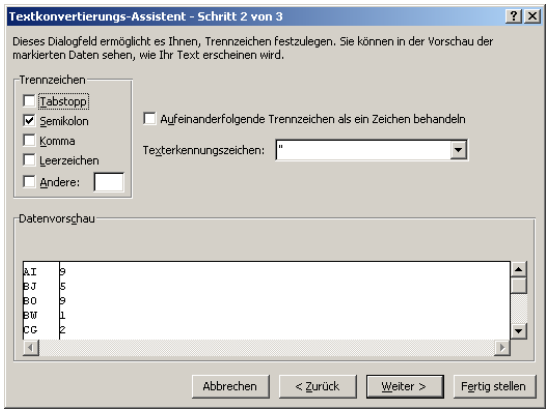
Generell ist es besser die Daten erst einmal als *.csv-Datei abzuspeichern.

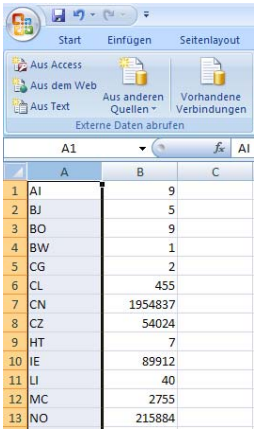
Abbildung 2: Datenimport Excel




Übung.csv - Microsoft Excel
Start
Einfügen
Seitenlayout
Formeln
Daten
Überprüfen
Ansicht
Acrobat
Text in Spalten
Duplikate entfernen
Datenüberprüfung
Konsolidieren
Was-wäre-wenn-Analyse
Gruppieren
Gruppierung aufheben
Teilergebnis
Verbindungen
Eigenschaften
Verknüpfungen bearbeiten
Sortieren
Filtern
Löschen
Erneut übernehmen
Erweitert
Externe Daten abrufen
Verbindungen
Sortieren und Filtern
Datentools
Gliederung
A1
AI;9
Text in Spalten
Den Inhalt einer einzigen Excel-Zelle in separate Spalten aufteilen.
Beispielsweise können Sie eine Spalte mit vollständigen Namen in separate Spalten für den Vornamen und den Nachnamen aufteilen.
Verwenden Sie dieses Feature in Word, um den markierten Text in eine Tabelle zu konvertieren und den Text an jedem Komma, Punkt oder dem von Ihnen angegebenen Zeichen in Spalten aufzuteilen.
Drücken Sie F1, um die Hilfe anzuzeigen.
AI;9
BJ;5
BO;9
BW;1
CG;2
CL;455
CN;1954837
CZ;54024
HT;7
IE;89912
LI;40
MC;2755


Textkonvertierungs-Assistent - Schritt 1 von 3
Der Textkonvertierungs-Assistent hat erkannt, dass Ihre Daten mit Trennzeichen versehen sind.
Wenn alle Angaben korrekt sind, klicken Sie auf 'Weiter', oder wählen Sie den korrekten Datentyp.
Ursprünglicher Datentyp
Wählen Sie den Dateityp, der Ihre Daten am besten beschreibt:
Getrennt - Zeichen wie z.B. Kommas oder Tabstopps trennen Felder (Excel 4.0-Standard).
Feste Breite - Felder sind in Spalten ausgerichtet, mit Leerzeichen zwischen jedem Feld.
Vorschau der markierten Daten:
1 AI;9
2 BJ;5
3 BO;9
4 BW;1
5 CG;2
Abbrechen
< Zurück
Weiter >
Fertig stellen


Textkonvertierungs-Assistent - Schritt 2 von 3
Dieses Dialogfeld ermöglicht es Ihnen, Trennzeichen festzulegen. Sie können in der Vorschau der markierten Daten sehen, wie Ihr Text erscheinen wird.
Trennzeichen
Tabstopp
Semikolon
Komma
Leerzeichen
Andere:
Aufeinanderfolgende Trennzeichen als ein Zeichen behandeln
Texterkennungszeichen: "
Datenvorschau
AI 9
BJ 5
BO 9
BW 1
CG 2
Abbrechen
< Zurück
Weiter >
Fertig stellen


Start
Einfügen
Seitenlayout
Aus Access
Aus dem Web
Aus Text
Aus anderen Quellen
Vorhandene Verbindungen
Externe Daten abrufen
A1
AI
AI 9
BJ 5
BO 9
BW 1
CG 2
CL 455
CN 1954837
CZ 54024
HT 7
IE 89912
LI 40
MC 2755
NO 215884

Beispiel 7:

Gesucht wird die Anzahl der Patentanmeldungen von Deutschland beim Europäischen Patentamt zwischen 1989 und 2007. Die Anzahl soll nach Jahren gruppiert werden. In der Ergebnisstabelle sollen die Jahre und die Anzahl der Patentanmeldungen aufgeführt werden.

```
SELECT      YEAR(APP.appln_filing_date) AS Jahr,
            COUNT(APP.appln_id) AS Anzahl_appln_id
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRS.person_id = PRSAPP.person_id
AND         APP.appln_id IN
            (SELECT APP2.appln_id
             FROM     dbo.tls201_appln APP2, dbo.tls206_person
                      PRS2, dbo.tls207_pers_appln PRSAPP2
             WHERE    APP2.appln_id = PRSAPP2.appln_id
             AND      PRS2.person_id = PRSAPP2.person_id
             AND      PRS.person_ctry_code = 'DE'
             AND      APP2.appln_auth = 'EP'
             AND      YEAR(APP2.appln_filing_date) BETWEEN 1989
                      AND 2007)
GROUP BY    YEAR(APP.appln_filing_date)
ORDER BY    YEAR(APP.appln_filing_date);
```

Ergebnis:

Jahr	Anzahl_appln_id
1989	42010
1990	46347
1991	39618
1992	43129
1993	44874
1994	46682
1995	52735
1996	57386
1997	68736
1998	78518
1999	88145
2000	99497
2001	107997
2002	110086
2003	111563
2004	119970
2005	125865
2006	78105
2007	7791

Mit `GROUP BY` wird angegeben, nach was gruppiert werden soll (siehe S.12). Beim `GROUP BY` müssen alle Spalten aufgeführt werden außer derjenigen, die sich auf den arithmetischen Operator bezieht. Hier in diesem Beispiel ist es eine Spalte.

Beispiel 8:

Gesucht werden die Patentanmeldungen beim Europäischen Patentamt zwischen 1985 und 2005 bei denen die Personen aus dem Bezirk SG8 in Großbritannien kommen. Auswahl der Spalten frei wählbar.

```
SELECT      APP.appln_id, YEAR(APP.appln_filing_date) AS Jahr,
            APP.appln_auth,PRSAPP.*, PRS.person_ctry_code,
            PRS.person_address
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRS.person_id = PRSAPP.person_id
AND         APP.appln_id IN
            (SELECT     APP2.appln_id
             FROM       dbo.tls201_appln APP2, dbo.tls206_person
                        PRS2, dbo.tls207_pers_appln PRSAPP2
            WHERE       APP2.appln_id = PRSAPP2.appln_id
            AND         PRS2.person_id = PRSAPP2.person_id
            AND         PRS.person_address LIKE '%SG8%'
            AND         APP2.appln_auth = 'EP'
            AND         YEAR(APP2.appln_filing_date) BETWEEN 1985
                        AND 2005)
ORDER BY    YEAR(APP.appln_filing_date);
```

Ergebnis:

appln_id	Jahr	appln_auth	person_id	appln_id	applt_seq_nr	invt_seq_nr	person_ctry_code	person_address
15470588	1985	EP	26883763	15470588	0	2	GB	6A Dickasons Melbourn,Royston Hertfordshire SG8 6EL
15457653	1985	EP	20128488	15457653	0	2	GB	6 Middle Street,Triplow Royston Herts SG8 7RD
15460168	1985	EP	28526576	15460168	0	1	GB	4 Cedar Close Melbourn,Royston Hertfordshire SG8 6BL
15458378	1985	EP	27607649	15458378	1	0	GB	York Way Royston,Hertforshire SG8 5HJ
15457798	1985	EP	17272671	15457798	1	0	GB	Baldock Road Royston,Hertfordshire SG8 5BQ
15476087	1985	EP	5301672	15476087	0	1	GB	21 Cherry Drive,Royston Hertfordshire, SG8 7DL
15476088	1985	EP	5301671	15476088	0	1	GB	21 Cherry Drive,Royston Hertfordshire SG8 7DL
15476405	1985	EP	23606294	15476405	0	3	GB	The Old Orchard Chapel Lane,Melbourn Royston Herts. SG8 6CN
15457895	1985	EP	1700845	15457895	0	1	GB	1 Silver Street Litlington,Nr. Royston Hertfordshire SG8 OQE
15465183	1985	EP	23191394	15465183	0	2	GB	94 Green Drift,Royston Hertfordshire SG8 5BT
15464599	1985	EP	6160788	15464599	1	0	GB	Roysia House,Royston, Herts. SG8 9JJ
15466251	1985	EP	6160788	15466251	1	0	GB	Roysia House,Royston, Herts. SG8 9JJ
15473203	1985	EP	6160788	15473203	1	0	GB	Roysia House,Royston, Herts. SG8 9JJ
15500468	1986	EP	16435269	15500468	0	1	GB	4, Clydesdale Road Royston,Hertfordshire, SG8 9JA
15504971	1986	EP	20633705	15504971	0	4	GB	3, Russett Way Melbourn,Royston Herts SG8 6HF
15506229	1986	EP	23191393	15506229	0	2	GB	94 Green Drift Royston,Hertfordshire SG8 5BT
15501504	1986	EP	5433731	15501504	0	2	GB	8 Brooke Road,Royston Hertfordshire SG8 7DR
15517282	1986	EP	23606293	15517282	0	3	GB	The Old Orchard Chapel Lane,Melbourn Royston Herts SG8 6CN
15516742	1986	EP	27603130	15516742	0	2	GB	Little Briairs Church Street,Thripolow Herts SG8 7RE
15501541	1986	EP	24278325	15501541	0	1	GB	1 Silver Street Litlington,Royston Herts, SG8
15512456	1986	EP	26883763	15512456	0	2	GB	6A Dickasons Melbourn,Royston Hertfordshire SG8 6EL
15518024	1986	EP	12912242	15518024	0	3	GB	82 Downlands Baldock Road,Royston Hertfordshire SG8 5BY
15558088	1987	EP	26883770	15558088	0	2	GB	6A, Dickasons Melbourn,Royston Hertfordshire SG8 6EL
15558285	1987	EP	26883763	15558285	0	4	GB	6A Dickasons Melbourn,Royston Hertfordshire SG8 6EL

(Ausschnitt von 1118 Zeilen)

Beispiel 9:

Beispiel 9_1:
Gesucht werde die Patente die beim Europäischen Patentamt (EP) und beim weltweitem Patentamt (WO) zwischen 1999 und 2001. Dabei soll mindestens eine Person (egal ob Anmelder oder Erfinder) aus Deutschland stammen. In der Ergebnistabelle sollen neben den Personendaten, den Anmeldernummern, den Erfindernummern ebenfalls die Zuordnungen in die Technologie 30 aufgelistet werden.

```
SELECT      DISTINCT APP.appln_id, YEAR(APP.appln_filing_date) AS Jahr,
            PRSAPP.person_id, PRS.person_name, PRS.person_address,
            PRS.person_ctry_code,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr, APP.appln_auth AS Amt,
            IPC.Tech30
FROM        dbo.tls201_appln APP INNER JOIN dbo.tls209_appln_ipc IPC ON
            APP.appln_id = IPC.appln_id,
            dbo.tls206_person PRS, dbo.tls207_pers_appln PRSAPP
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRSAPP.person_id = PRS.person_id
AND         APP.appln_auth IN ('EP', 'WO')
AND         YEAR(APP.appln_filing_date) BETWEEN 1999 AND 2001
AND         PRSAPP.appln_id IN
            (SELECT     PRSAPP2.appln_id
            FROM        dbo.tls207_pers_appln PRSAPP2
            WHERE       PRSAPP2.person_id IN
                        (SELECT     PRS.person_id
                        FROM        dbo.tls206_person PRS
                        WHERE       PRS.person_ctry_code = 'DE'
                        AND         PRS.person_address LIKE
                                    '%Karlsruhe%'
                        )
            )
ORDER BY    APP.appln_id, PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr,
            IPC.Tech30;
```

(Ausschnitt von 4059 Zeilen)

Was hier neu dazu gekommen ist, ist das DISTINCT. Es sollen hier nur unterschiedliche Zeilen ausgewählt werden. Durch zusammenführen von Tabellen kann es zu Redundanzen kommen was mit dem Wort DISTINCT ausgeschlossen wird.

Beispiel 9_2:
Gesucht werden die Patente die beim Europäischen Patentamt (EP) und beim weltweitem Patentamt (WO) zwischen 1999 und 2001. Dabei sollen die Personen (egal ob Anmelder oder Erfinder) nur aus Deutschland stammen. In der Ergebnistabelle sollen neben den Personendaten, den Anmeldernummern, den Erfindernummern ebenfalls die Zuordnungen in die Technologie 30 aufgelistet werden.

```
SELECT      DISTINCT APP.appln_id, YEAR(APP.appln_filing_date) AS Jahr,
            PRSAPP.person_id, PRS.person_name, PRS.person_address,
            PRS.person_ctry_code,
            PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr, APP.appln_auth AS Amt,
            IPC.Tech30
FROM        dbo.tls201_appln APP INNER JOIN dbo.tls209_appln_ipc IPC ON
            APP.appln_id = IPC.appln_id,
            dbo.tls206_person PRS, dbo.tls207_pers_appln PRSAPP
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRSAPP.person_id = PRS.person_id
AND         APP.appln_auth IN ('EP', 'WO')
AND         YEAR(APP.appln_filing_date) BETWEEN 1999 AND 2001
AND         PRSAPP.appln_id IN
            (SELECT     PRSAPP2.appln_id
            FROM        dbo.tls207_pers_appln PRSAPP2
            WHERE       PRSAPP2.person_id IN
                        (SELECT     PRS.person_id
                        FROM        dbo.tls206_person PRS
                        WHERE       PRS.person_ctry_code = 'DE'
                        )
            )
AND         PRS.person_address LIKE '%Karlsruhe%'
ORDER BY    APP.appln_id, PRSAPP.applt_seq_nr, PRSAPP.invt_seq_nr,
            IPC.Tech30;
```

Beide Abfragen sind bis auf die veränderte Position von ‚**AND PRS.person_address LIKE '%Karlsruhe%'**' gleich. Einmal steht es in der inneren Abfrage und einmal außerhalb. Das Ergebnis verändert sich erheblich; statt 4059 Zeilen sind nur 1204 Zeilen betroffen. Es ist somit wichtig genau zu wissen was gesucht wird. Im Beispiel 9_1 ist gut zu erkennen, dass nicht nur andere Orte sondern auch andere Länder in der Ergebnistabelle auftauchen können.

Hinweis:
Beim Suchen z.B. nach „Karlsruhe“ wird exakt nach der Schreibweise „Karlsruhe“ gesucht, die falschen Eingaben wie Karslruhe, Kalsruhe werden nicht mitberücksichtigt. Nach diesen muss speziell gesucht werden.

Ergebnis 9_1:

appln_id	Jahr	person:id	person_name	person_address	person_ctry_code	applt_seq_nr	invt_seq_nr	Amt	Tech30
14539276	2000	2574952	Birnbaum, Roland	Röthardterstrasse 9,73433 Aalen	DE	0	1	EP	1
14539276	2000	2574952	Birnbaum, Roland	Röthardterstrasse 9,73433 Aalen	DE	0	1	EP	6
14539276	2000	10803916	Heimüller, Hans-Jost	Am Schafgarten 9,67373 Dudenhofen	DE	0	2	EP	1
14539276	2000	10803916	Heimüller, Hans-Jost	Am Schafgarten 9,67373 Dudenhofen	DE	0	2	EP	6
14539276	2000	19055457	Merkle, Klaus	Ebersteinstrasse 3,75015 Bretten	DE	0	3	EP	1
14539276	2000	19055457	Merkle, Klaus	Ebersteinstrasse 3,75015 Bretten	DE	0	3	EP	6
14539276	2000	3364900	Buck, Carsten	Wörishofferstrasse 2,76189 Karlsruhe	DE	0	4	EP	1
14539276	2000	3364900	Buck, Carsten	Wörishofferstrasse 2,76189 Karlsruhe	DE	0	4	EP	6
14539276	2000	29601840	TYCO Electronics Logistics AG	Ampèrestrasse 3,9323 Steinach / SG	CH	1	0	EP	1
14539276	2000	29601840	TYCO Electronics Logistics AG	Ampèrestrasse 3,9323 Steinach / SG	CH	1	0	EP	6

Ergebnis 9_2:

appln_id	Jahr	person:id	person_name	person_address	person_ctry_code	applt_seq_nr	invt_seq_nr	Amt	Tech30
14539276	2000	3364900	Buck, Carsten	Wörishofferstrasse 2,76189 Karlsruhe	DE	0	4	EP	1
14539276	2000	3364900	Buck, Carsten	Wörishofferstrasse 2,76189 Karlsruhe	DE	0	4	EP	6

(Ausschnitt von 1204 Zeilen)

Beispiel C (mit temporären Tabellen und Gewichtungen)

Gesucht wird folgendes
/*
Anzahl der Patente
Anzahl der unterschiedlichen Anmelder
Anzahl der unterschiedlichen Erfinder
Anzahl der Patente gewichtet nach Erfindern aus dem Land
Anzahl der Patente gewichtet nach Anmelder aus dem Land
für jedes Land und jedes Jahr für die IPC-Klasse C
*/

Zum Schluss soll alles in einer Tabelle aufgezeigt werden.

Bei diesem längeren Beispiel ist es sinnvoll temporäre Tabellen zu erstellen. In der PATSTAT-Datenbank ist eine Tabelle zum Anschauen unter dbo.iww_beispiel angelegt worden. Die Tabelle wird erstellt indem mit der rechten Maustaste auf „neue Tabelle“ erstellen geklickt wird und angibt wie der oder die Spaltennamen lauten sollen, von welchem Typ die Daten sein sollen und ob die „NULL“ zugelassen werden soll. Die Tabelle kann dann unter dem gewünschten Namen gespeichert werden.
Um zum Ergebnis dieses Beispiels zu gelangen, müssen alle temporären Tabellen erstellt werden. Hier in dem Beispiel C sind es alle Tabellen, die lauten dbo.iww_beispiel_*

Schritt C1:
Gesucht werden zunächst alle Patente, die beim EPO, Wipo, in den USA und in Japan zwischen 1980 und 2005 angemeldet wurden. Diese Daten werden in Tabelle iww_beispiel1 gespeichert.

/* Anzahl der Patente */

```
INSERT      INTO dbo.iww_beispiel1
SELECT      PRS.person_ctry_code,
            YEAR(APP.appln_filing_date) AS year,
            COUNT(DISTINCT APP.appln_id) AS patents
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP, dbo.tls209_appln_ipc IPC
WHERE       YEAR(APP.appln_filing_date) BETWEEN 1980 AND 2005
AND         APP.appln_auth IN ('EP', 'WO', 'IB', 'US', 'JP')
AND         APP.appln_id = PRSAPP.appln_id
AND         APP.appln_id = IPC.appln_id
AND         PRSAPP.person_id = PRS.person_id
GROUP BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date)
ORDER BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date);
```

/* Ende Abschnitt */
Anmerkung: Die Länderkürzel stehen in der Tabelle tls201_appln in der Spalte appln_auth. Die Kürzel können im Internet nachgeschaut werden. Die Kürzel für die Patentämter können in der Patstatanleitung nachgelesen werden.

Schritt C2:
Es werden nun die Patente rausgesucht bei denen es mindestens einen Anmelder gibt. Ansonsten gleiche Bedingungen wie in Schritt C1.

```
/* Anzahl der unterschiedlichen Anmelder */
INSERT      INTO dbo.iww_beispiel2
SELECT      PRS.person_ctry_code,
            YEAR(APP.appln_filing_date) AS year,
            COUNT(DISTINCT PRS.person_id) AS applicants
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP, dbo.tls209_appln_ipc IPC
WHERE       YEAR(APP.appln_filing_date) BETWEEN 1980 AND 2005
AND         APP.appln_auth IN ('EP', 'WO', 'IB', 'US', 'JP')
AND         PRSAPP.applt_seq_nr > 0
AND         APP.appln_id = PRSAPP.appln_id
AND         APP.appln_id = IPC.appln_id
AND         PRSAPP.person_id = PRS.person_id
GROUP BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date)
ORDER BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date);
/* Ende Abschnitt */
```

Schritt C3:
Es soll nun mindestens ein Erfinder aufgelistet sein, ansonsten wieder gleiche Bedingungen wie in Schritt C1.

```
/* Anzahl der unterschiedlichen Erfinder */
INSERT      INTO dbo.iww_beispiel3
SELECT      PRS.person_ctry_code,
            YEAR(APP.appln_filing_date) AS year,
            COUNT(DISTINCT PRS.person_id) AS inventors
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP, dbo.tls209_appln_ipc IPC
WHERE       YEAR(APP.appln_filing_date) BETWEEN 1980 AND 2005
AND         APP.appln_auth IN ('EP', 'WO', 'IB', 'US', 'JP')
AND         PRSAPP.invt_seq_nr > 0
AND         APP.appln_id = PRSAPP.appln_id
AND         APP.appln_id = IPC.appln_id
AND         PRSAPP.person_id = PRS.person_id
GROUP BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date)
ORDER BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date);
/* Ende Abschnitt*/
```

Schritt C4:
Die Patente sollen in der IPC-Klasse C (Chemie; Hüttenwesen) aufgelistet sein.

```
/* Alle relevanten ‚appln_ids' auslesen */
INSERT      INTO dbo.iww_beispiel_Apps
SELECT      DISTINCT APP.appln_id
FROM        dbo.tls201_appln APP, dbo.tls206_person PRS,
            dbo.tls207_pers_appln PRSAPP, dbo.tls209_appln_ipc IPC
WHERE       YEAR(APP.appln_filing_date) BETWEEN 1980 AND 2005
AND         APP.appln_auth IN ('EP', 'WO', 'IB', 'US', 'JP')
AND         IPC.ipc_class_level = 'C'
AND         APP.appln_id = PRSAPP.appln_id
AND         APP.appln_id = IPC.appln_id
AND         PRSAPP.person_id = PRS.person_id
ORDER BY    APP.appln_id;
/* Ende Abschnitt */
```

Schritt C5:

```
/* Maximum der ‚applt_seq_nr' und ‚invt_seq_nr' obiger ‚appln_ids' bestimmen */
INSERT      INTO dbo.iww_beispiel_MAX
SELECT      APP.appln_id, MAX(PRSAPP.applt_seq_nr),
            MAX(PRSAPP.invt_seq_nr)
FROM        dbo.tls201_appln APP, dbo.tls207_pers_appln PRSAPP,
            dbo.tls206_person PRS, dbo.iww_beispiel_Apps IWWAP
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRSAPP.person_id = PRS.person_id
AND         APP.appln_id = IWWAP.appln_id
GROUP BY    APP.appln_id
ORDER BY    APP.appln_id;
/* Ende Abschnitt */
```

Schritt C6:

```
/* Gewichtung */
INSERT      INTO dbo.iww_beispiel_weight
SELECT      PRS.person_ctry_code, YEAR(APP.appln_filing_date),
            APP.appln_id,
CASE        WHEN PRSAPP.applt_seq_nr = 0 THEN 0
            WHEN MAX.applt_seq_nr != 0 THEN (1.0/MAX.applt_seq_nr) END,
CASE        WHEN PRSAPP.invt_seq_nr  = 0 THEN 0
            WHEN MAX.invt_seq_nr != 0 THEN (1.0/MAX.invt_seq_nr)
END
FROM        dbo.tls201_appln APP, dbo.tls207_pers_appln PRSAPP,
            dbo.tls206_person PRS, dbo.iww_beispiel_MAX MAX
WHERE       APP.appln_id = PRSAPP.appln_id
AND         PRSAPP.person_id = PRS.person_id
AND         APP.appln_id = MAX.appln_id
ORDER BY    PRS.person_ctry_code, YEAR(APP.appln_filing_date),
            APP.appln_id;
/* Ende Abschnitt */
```

Schritt C6:

```
/* Aggregieren */
INSERT      INTO dbo.iww_beispiel_weight_final
SELECT      WEI.person_ctry_code, WEI.appln_filing_date,
            SUM(WEI.applt_seq_nr), SUM(WEI.invt_seq_nr)
FROM        dbo.iww_beispiel_weight WEI
GROUP BY    WEI.person_ctry_code, WEI.appln_filing_date;
/* Ende Abschnitt */
```

C_ENDE:

```
/* Zusammenfügen */
INSERT      INTO dbo.iww_beispiel_ende
SELECT      IWW1.person_ctry_code, IWW1.appln_filing_date, IWW1.patents,
            IWW2.applicants, IWW3.inventors, WF.app_weight, WF.inv_weight
FROM        dbo.iww_beispiel1 IWW1, dbo.iww_beispiel2 IWW2,
            dbo.iww_beispiel3 IWW3, dbo.iww_beispiel_weight_final WF
WHERE       IWW1.person_ctry_code = IWW2.person_ctry_code
AND         IWW1.person_ctry_code = IWW3.person_ctry_code
AND         IWW1.person_ctry_code = WF.person_ctry_code
AND         IWW1.appln_filing_date = IWW2.appln_filing_date
AND         IWW1.appln_filing_date = IWW3.appln_filing_date
AND         IWW1.appln_filing_date = WF.year
ORDER BY    IWW1.person_ctry_code, IWW1.appln_filing_date;
/* Ende Abschnitt */
```

```
SELECT      *
FROM        dbo.iww_beispiel_ende;
```

Endergebnis:

person_ctry_code	appln_filing_date	patents	applicants	inventors	app_weight	inv_weight
DE	1905-06-04 00:00:00.000	12115	4299	18547	10546	4825
DE	1905-06-05 00:00:00.000	13347	4764	20342	11507	5290
DE	1905-06-06 00:00:00.000	13445	4779	20471	11511	5282
DE	1905-06-07 00:00:00.000	13170	4847	20257	11319	5504
DE	1905-06-08 00:00:00.000	15571	5802	23605	13622	6752
DE	1905-06-09 00:00:00.000	16513	6024	25252	14358	6829
DE	1905-06-10 00:00:00.000	18039	6553	27495	15789	7454
DE	1905-06-11 00:00:00.000	18736	6918	29496	16358	7567
DE	1905-06-12 00:00:00.000	20954	7469	32816	18458	8211
DE	1905-06-13 00:00:00.000	22552	8015	35508	19652	8688
DE	1905-06-14 00:00:00.000	23465	8052	38491	20465	8778

(Ausschnitt von 2678 Zeilen)

3.5 Exkurs: Validierungstabellen

3.5.1 Motivation und Grundidee

Der souveräne Umgang mit Patentdaten aus Datenbanken, insbesondere der PATSTAT-Datenbank, ist für die wissenschaftliche Arbeit auf dem Gebiet der Innovationsforschung essentiell. Doktoranden, Assistenten, wissenschaftliche Hilfskräfte und Studenten legen die so gewonnenen Daten ihren Arbeiten zugrunde, oft werden Daten dabei zuarbeitend von Dritten erhoben. Schnelle und verlässliche Validierbarkeit der Ergebnisse ist daher von großer Bedeutung. Mit dieser Arbeit soll ein Werkzeug geschaffen werden, das es erlaubt, schnell – also mit wenigen Mausklicks und insbesondere ohne den Einsatz von SQL oder der eigentlichen PATSTAT Datenbank – Datensätze auf ihre Plausibilität hin zu überprüfen. Es werden dabei vorab die Daten aus PATSTAT extrahiert, in Microsoft Excel hinterlegt und können dann über eine Pivot-Tabelle bedarfsspezifisch strukturiert werden. Der einzelne Anwender kann dann schnell, unkompliziert und vor allem ohne Vorkenntnisse in Datenbank-Query-Sprachen wie SQL o.a. auf die grundlegenden Informationen zugreifen.

3.5.2 Aufbau

Diese Arbeit beschränkt sich in ihrer Herangehensweise auf „das gewichtete Zählen von Patentapplikationen“. Wie das im Einzelnen geschieht, wird in der Folge erläutert. Die wesentliche, damit assoziierte Tabelle der PATSTAT Datenbank ist die Liste der applications, also der Patentanmeldungen. Der Primärschlüssel ist hier die *appln_id* – eine laufende Identifikationsnummer. Es hat sich gezeigt, dass die meisten Abfragen, die im Rahmen der Innovationsforschung durchzuführen sind, gerade in dieser Tabelle ihren zentralen Bestandteil haben. Es soll nun also möglich sein, spezielle Abfragen rund um die Patentapplikationen auf ihre Plausibilität hin zu überprüfen. Dabei muss klar sein, dass Plausibilität in diesem Zusammenhang nicht als Korrektheit verstanden werden kann. Der Vergleich mit Referenzdaten kann keinesfalls die Richtigkeit einer Abfrage beweisen oder widerlegen. Vielmehr soll eine grundlegende Datenbasis geschaffen werden, anhand der Ergebnisse schnell und einfach geprüft und somit lediglich in Bezug auf ihre Größenordnung hin eingeschätzt werden können.

Die Differenzierung der Betrachtung hat häufig einen zeitlichen, räumlichen oder technologiebasierten Bezug. Es liegt daher nahe, auch eine Standardtabelle, wie sie in dieser Arbeit angestrebt wird, nach eben diesen Kriterien auszurichten. Es werden daher folgende Faktoren für die Klassifikation herangezogen:

Jahr

Der Längsschnitt durch Patentdaten dient der Innovationsforschung insbesondere zur Untersuchung und Identifikation von Trends und Entwicklungen. Vereinfachend wird in dieser Arbeit das genaue Datum jedes Datensatzes auf das Kalenderjahr reduziert. Auch beschränkt sich der Horizont auf drei Dekaden. Insbesondere wird bei dieser Analyse zur zeitlichen Zuordnung von Applikationen das Prioritätsdatum verwendet, welches in vielen Fällen vom „offiziellen“ Anmeldedatum abweicht, den tatsächlichen Innovationszeitpunkt aber besser widerspiegelt.

Zu beachten ist bei der Arbeit mit der Komponente Zeit, dass die PATSTAT Daten nicht über den gesamten, hier abgedeckten Zeitraum konsistent sind. Genauere Informationen dazu sind dem PATSTAT Handbuch zu entnehmen.

Patentbüro

Hinsichtlich der Anmeldestelle, also dem Büro, bei welchem die Anmeldung eingereicht wurde, wird hier zwischen dem Europäischen (European Patent Office (EPO)) und dem Internationalen Bureau (World Intellectual Property Organisation (WIPO)) unterschieden. Diese Reduktion ist für detaillierte Recherchen hinsichtlich der Anmeldestelle und insbesondere für Abfragen mit Fokus auf die europäischen Patentbesonderheiten nicht genau genug, für viele allgemeiner formulierte Aufgabenstellungen jedoch ausreichend.

IPC Klasse

Jeder Patentapplikation ist über die Tabelle IPC eine oder mehrere Technologieklassen zugeordnet. Das Klassensystem ist hierarchisch aufgebaut, soll hier im Einzelnen aber nicht weiter erläutert werden. Es werden für jede *appln_id* die *ipc_section* sowie die *ipc_class*, also die beiden höchsten Hierarchiestufen des (Core-) *ipc_class_symbols* hinterlegt. Bereits die nächsten Stufen lassen die Anzahl der zu unterscheidenden Datengruppen derart ansteigen, dass eine angemessene Handhabung in Excel nicht mehr möglich ist.

Land (Anmelder)

Jeder Patenapplikation ist über die Tabellen **dbo.tls207_pers_appln** und **dbo.tls206_person** eine oder mehrere Personen (natürlich oder juristisch) und deren Nationalitäten zugeordnet. Diese wird anhand der zweistelligen ISO Ländercodes hinterlegt.

NUTS3 Code

Für detailliertere räumliche Betrachtungen auf regionaler Ebene sind die Datenfragmente zusätzlich hinsichtlich des NUTS3-Codes, also einer Kennzeichnung etwa auf Landkreisgröße, differenziert. Dieser ist für viele Daten zwar nicht vorhanden, besonders für Deutschland, Frankreich und Großbritannien sind die Werte jedoch weitestgehend verfügbar.

Tabelle 12: Dimensionen der Klassifizierung

Kriterium	**Bezeichnung in PATSTAT**	**Datenbereich**
Jahr	*prior_filing_date*	1979 bis 2009
Patentbüro	*appln_auth*	IB, EP[9]
IPC Klasse	*ipc_section*	A, B, C, D, E, F, G, H
	ipc_class	01, 02, 03, ... , 99
Land (Anmelder)	*person_country_code*	DE, FR, ...
NUTS 3 Code	*nuts3*	DE225, ...

Für jeden Schnitt dieser vier Hauptdimensionen soll nun eine gewichtete Größe hinterlegt werden, die angibt, wie viele Patentapplikationen auf den entsprechenden Bereich entfallen. Auf das Thema Gewichtung wird in 3.5.3 eingegangen. Es ist zu beachten, dass die sechs Kriterien sich auf lediglich vier unabhängige Dimensionen reduzieren, da eine echte funktionale Abhängigkeit

[9] WO wurde aufgrund von mangelnder Datenbasis hier ausgelassen

zwischen *nuts3* und *person_country_code* und eine unechte, nur logische Abhängigkeit zwischen *ipc_section* und *ipc_class* besteht. Das heißt, dass der NUTS3-Code stets auch den *country_code* determiniert.

Die Klassifizierung bezüglich der IPC-Klasse ist etwas komplizierter. Hier ist eine Einteilung anhand der vollständigen *ipc_class_symbols* nicht sinnvoll, weil die Datenmenge schlichtweg zu groß werden würde; eine solche Einteilung wäre zu feingliedrig. Die Beschränkung auf die ersten beiden logischen Elemente des IPC-Class-Symbols, und damit die Einteilung in „größere Klassen" erscheint daher sinnvoller. Nun ist aber die *ipc_class* nicht ohne die *ipc_section* sinnvoll zu verstehen, d.h. ein Schnitt über Patentdaten hinsichtlich einer *ipc_class* über alle *ipc_sections* macht keinen Sinn, da die *ipc_classes* in verschiedenen *ipc_sections* sehr unterschiedliche Bedeutungen haben. Insofern stellt das Kriterium *ipc_class* immer nur eine Verfeinerung des Kriteriums *ipc_section* dar und erzeugt deshalb auch keine weitere (echte) Dimension.

3.5.3 Gewichtung

Es ist nun möglich, dass sich zu einer bestimmten *appln_id* mehrere Einträge finden, wenn zum Beispiel mehrere Anmelder zugeordnet sind, weil das Patent auf eine Gruppe von Erfindern zurückgeht. Jede *appln_id* darf (auch bei mehrfachem Vorkommen) aus Konsistenzgründen jedoch nur mit dem Gewicht „1" gezählt werden. Sie entspricht nur einem realen Patent, bzw. einer „Innovationseinheit". Sind den unterschiedlichen Anmeldern nun unterschiedliche Nationalitäten zugeordnet, so wird die Anmeldung linear gewichtet und die resultierenden reellen Werte aus dem Intervall [0, 1] der entsprechenden Kategorie zugeordnet. Gleiches gilt für Aufteilungen auf verschiedene Technologieklassen.
Es ist zu beachten, dass den Applikationen zur korrekten Bestimmung des Innovationszeitpunktes, die korrespondierenden Jahreszahlen der *prior_filing_dates*, also das Datum, der „ursprünglichen" Innovation zugeordnet wird.
Weiterhin gilt zu beachten, dass bezüglich der Technologieklassen stets nur die Core-IPC Einträge berücksichtigt werden.
Es werden insgesamt 573.986 einzelne Datensätze exportiert, die die Grundlage für die Pivot-Tabelle bilden. An dieser Stelle ergibt sich eine Beschränkung durch Excel 2003, da die maximale Anzahl an Zeilen ca. 65.000 beträgt. In Excel 2007 gilt diese Beschränkung nicht, das Limit liegt dort bei knapp über einer Million Zeilen. Für die Arbeit auf den Instituts-Rechnern, die lediglich über Microsoft Excel 2003[10] verfügen, wurde deshalb eine verkürzte Version mit ca. 18.000 Datensätzen zusammengestellt. Es entfallen dort die Kategorien *nuts3* und *ipc_class*. Alle Applikationen, über alle Dimensionen hinweg haben ein Gesamtgewicht von 2.032.101. Dies gilt sowohl für die umfangreiche, als auch für die reduzierte Version.

3.5.4 Datensäuberung

Um eine gewisse Konsistenz hinsichtlich der Daten zu gewährleisten, wurden folgende Schritte zur Säuberung und Vereinheitlichung vorgenommen. Zunächst einmal wurden den Länderkürzeln zur besseren Verständlichkeit die ausgeschriebenen Ländernamen anhand des ISO-3166-Standards zugeordnet (Stand 2006).

Obwohl in der verwendeten Version der ISO-3166 Tabelle nicht mehr geführt, werden die folgenden Länderkürzel beibehalten: Es wird angenommen, dass es, obwohl die Länder in dieser Form nicht mehr existieren, von Interesse sein kann, Innovationstätigkeit gerade diesen politischen Bereichen zuzuordnen.

[10] Stand 2009

Tabelle 13: Beibehaltene Ländercodes

ISO Code	Land	# Datensätze
BU	Burma	5
DD	DDR	262
RS	Serbien	3
SU	UdSSR	408
YU	Jugoslawien	299
ZR	Zaire	3

Folgende Länderkürzel sind veraltet und wurden durch ihre aktuellen Äquivalente ersetzt.

Tabelle 14: Angepasste Ländercodes

Altes Kürzel	Neues Kürzel	Land	# Datensätze
SF	FI	Finnland	7
UK	GB	Großbritannien	14

Folgende Länderkürzel haben keine Bedeutung, und werden als nicht zuordenbare Tippfehler interpretiert und somit gelöscht. Die Datensätze bleiben bestehen und haben lediglich in der Kategorie *country_code* ein Leerzeichen.

Tabelle 15: Gelöschte Ländercodes

Code	Land	# Datensätze
W	?	2
D	?	1
U	?	2
EP	?	2
EN	?	3
JA	?	7
KO	?	7
RH	?	4
SP	?	2

Insgesamt treten, inklusive der Löschung der 30 oben genannten ungültigen Kürzel, 2552 Leerzeichen in der Kategorie *country_code* auf. Das entspricht, gemessen an der Gesamtzahl der Datensätze einem Anteil von 0,446%. Die Leerzeichen wurden in der Folge einheitlich durch #N/A ersetzt.

Im Bereich der NUTS3 Codes wurden 1514 nicht-echte Codes, also Codes, die beispielsweise nur den Ländercode enthielten einheitlich durch NN ersetzt.

3.5.5 Datenbank

Weil typischerweise mehrere Personen an den Datenbankabfragen beteiligt sind, empfiehlt es sich, einen universellen Standard hinsichtlich der Begrifflichkeiten einzuführen. Ein erster Schritt, um Abfragen für verschiedene Anwender leicht einsehbar und verständlich zu machen, ist die einheitliche Nomenklatur der Tabellen im Abfrage-Code. Aufbauend auf vorangegangen Abfragen und Erfahrungen wird mit dieser Arbeit folgende Benennung vorgeschlagen:

Tabelle 16: PATSTAT-Tabellen und Kürzel

Tabelle		Kürzel
dbo.tls201_APPLN		APP
dbo.tls202_APPLN_TITLE		TTL
dbo.tls203_APPLN_ABSTR		ABS
dbo.tls204_APPLN_PRIOR		PRR
dbo.tls205_TECH_REL		TCH
dbo.tls206_PERSON		PRS
dbo.tls207_PERS_APPLN		PRSAPP
dbo.tls208_DOC_STD_NMS		DOC
dbo.tls209_APPLN_IPC		IPC
dbo.tls210_APPLN_N_CLS		NCL
dbo.tls211_PAT_PUBLN		PUB
dbo.tls212_CITATION		CIT
dbo.tls214_NPL_PUBLN		NPC
dbo.tls215_CITN_CATEG		CCT
dbo.tls216_APPLN_CONTN		CTN
dbo.tls217_APPLN_I_CLS		ICL
dbo.tls_iww_kon19		K19
dbo.tls_iww_kon30		K30
dbo.tls_iww_kon44		K40
dbo.tls_iww_prior_appln_date		PRI

(Rot hinterlegte Tabellen wurden für die Validierungstabellen verwendet)

3.5.6 SQL Queries

Hier sind die Queries aufgeführt, die für die Erstellung der Validierungstabelle verwendet wurden. Die Erstellung erfolgt dabei in drei Schritten. Zunächst werden die notwendigen Tabellen miteinander verknüpft und in der (sehr umfangreichen) Tabelle **dbo.titemp_SANITY** umfassend hinterlegt. Im nächsten Schritt wird eine Hilfstabelle **dbo.titemp_count** angelegt, in der für jede verwendete *appln_id* die Anzahl ihres Vorkommens hinterlegt wird. Dieses Gewicht wird im nächsten Schritt dann reziprok in die erste Tabelle zurückgegeben und in einer neuen Tabelle **dbo.titemp_SANITY_2** gespeichert. Diese wird abschließend dann nach den gewünschten Kriterien gruppiert und die Gewichte entsprechend aufsummiert. Die hieraus resultierenden Ergebnisse werden dann in Excel importiert und bilden die Grundlage für die Pivot-Tabelle.

```
DELETE FROM titemp_SANITY;
INSERT INTO titemp_SANITY

SELECT      APP.appln_id,
            APP.appln_auth,
            YEAR(PRI.prior_appln_date),
            PRS.person_ctry_code,
            PRS.nuts3,
            SUBSTRING(IPC.ipc_class_symbol,1,1),
            SUBSTRING(IPC.ipc_class_symbol,2,2),
            SUBSTRING(IPC.ipc_class_symbol,4,1),
            IPC.ipc_class_symbol

FROM

(
(
dbo.tls201_appln APP

INNER JOIN
       dbo.tls_iww04_prior_appln_date PRI
ON          APP.appln_id = PRI.appln_id
      AND YEAR (PRI.prior_appln_date) BETWEEN 1979 AND 2009
      AND (APP.appln_auth = 'EP' OR APP.appln_auth = 'IB'))

INNER JOIN

dbo.tls209_appln_ipc IPC

ON APP.appln_id = IPC.appln_id AND IPC.ipc_class_level = 'C'
)

INNER JOIN

      (
      dbo.tls206_person PRS

      INNER JOIN
            dbo.tls207_pers_appln PRSAPP
      ON PRS.person_id = PRSAPP.person_id
      )

ON APP.appln_id = PRSAPP.appln_id

ORDER BY appln_id;


/* Count appearance of every applnID */
```

```
DELETE FROM dbo.titemp_count;
INSERT INTO dbo.titemp_count

SELECT      appln_id,
            COUNT (*)

FROM        dbo.titemp_SANITY

GROUP BY    appln_id;


/* Match weight of applnID to main data list */

DELETE FROM dbo.titemp_SANITY_2;
INSERT INTO dbo.titemp_SANITY_2

SELECT      SNTY.*,
            (1.0/CNT.nbr)

FROM        dbo.titemp_SANITY SNTY
INNER JOIN  dbo.titemp_count CNT

ON          SNTY.appln_id = CNT.appln_id ;


/* Generate output for PIVOT table - (573.986 entries) */

SELECT      appln_auth,
            year,
            person_ctry_code,
            nuts3, ipc_section,
            ipc_class,
            SUM (w)

FROM        dbo.titemp_SANITY_2

GROUP BY    appln_auth,
            year,
            person_ctry_code,
            nuts3,
            ipc_section,
            ipc_class;
```

3.5.7 Verwendung

Die beiden Dokumente sind im Verzeichnis **`waldhornstrasse\validierung\`** abgelegt. Sie sollten dort niemals verändert oder dort benutzt/geöffnet werden, sondern immer lokal abgespeichert und dann erst verwendet werden. Die umfangreiche Version (**`2009_validierung_XL.xlsx`**) ist etwa 43 MB groß, die gekürzte Version (**`2009_validierung_S.xlsx`**) noch ca. 1,3 MB. Zusätzlich finden sich in diesem Verzeichnis weitere Informationen zur Verwendung der Tabellen.

4 Standardisierte Abfragen

Zur einfacheren und bis zu einem gewissen Grad standardisierten Bearbeitung von Patentanalysen mit Hilfe der in einem früheren Abschnitt erwähnten PATSTAT-Datenbank wurde am IWW ein Fragebogen mit dem Titel „Arbeiten mit PATSTAT“ erarbeitet.

Dieser vereinfacht es, Patentanalysen in Auftrag zu geben, da er ohne große Kenntnis von SQL und der PATSTAT-Datenbank verwendet werden kann. Somit dient der Fragebogen als Schnittstelle zwischen dem Auftraggeber und der Person, die die Analyse dann letztlich mit SQL abarbeitet.

Dabei ist es aber nicht beabsichtigt, jegliche Kommunikation und Abstimmung zwischen Auftraggeber und der bearbeitenden Person überflüssig zu machen. Der Fragebogen ist lediglich eine erste Grundlage für den Einstieg in die jeweilige Abfrage und zieht somit auch Kommunikation und Abstimmung nach sich.

Mit Hilfe von Adobe Acrobat Professional kann das Formular sowohl ausgefüllt gespeichert und verschickt werden, als auch ausgedruckt und von Hand übergeben werden.

Der Fragebogen gliedert sich in die 5 Abschnitte Abstract, Kriterien/Input, Ausgabe, Bemerkungen und Beispiel für eine Ausgabetabelle, worauf nun näher eingegangen wird.

Abstract

Im „Abstract“-Bereich besteht die Möglichkeit, die Abfrage kurz und in wenigen Worten zusammenzufassen. Hier kann deutlich gemacht werden, welches das Ziel der Abfrage ist und wie sie sich in den Gesamtkontext eines übergeordneten Projektes einordnen lässt.

Kriterien

Danach werden im Abschnitt „Kriterien“ die Inputfaktoren mit denen die Analyse durchgeführt wird bestimmt. Zuallererst muss der für die Untersuchung relevante Zeitraum angegeben werden. Hier ist zwischen Anmeldedatum, Publikationsdatum und Prioritätsdatum zu unterscheiden. Weiterhin besteht die Möglichkeit, mehrere Jahre zu Abschnitten gruppiert anzugeben, da die Betrachtung sonst jedes Jahr einzeln analysiert.

Hinsichtlich der Patentklassifizierung kann auf die IPC-Klassen oder die Technologiefeld-Konkordanzen 19/30/44 zurückgegriffen werden. Bei einer Analyse auf IPC-Basis kann noch die gewünschte Genauigkeit angegeben werden.
Anzumerken ist, dass standardmäßig immer IPCs in der Core-Klassifikation zur Analyse herangezogen werden. Sollen auch Patente in der „advanced level“-Klassifikation betrachtet werden, so ist dies unter Sonstiges explizit zu erwähnen.

Wichtig ist die Angabe der gewünschten Länderzuordnung von Patenten, die anhand der beteiligten Akteure geschieht. Dies kann entweder auf Erfinderbasis, auf Anmelderbasis oder auf einer Kombination beider passieren.
Des Weiteren gibt es in PATSTAT mehrere Ansätze, um Personen räumlich zuzuordnen. So kann dies mit Hilfe des Ländercodes, des NUTS3-Codes oder des Adressfeldes aus der Personen-Tabelle erfolgen.

Ausgabe

Im Abschnitt „Ausgabe“ werden die gewünschten Ausgabevariablen definiert. Hierbei ist es anzuraten, nur die wirklich benötigten Elemente ausgeben zu lassen. Die Ausgabe aller, respektive einiger nicht benötigter Elemente führt zu längeren Abfragen und größeren Ergebnistabellen.
Vor dem Hintergrund eventuell später auftretender Folgeabfragen, sollte die Wahl der Variablen sorgfältig und überlegt erfolgen.

Die Anordnung der möglichen Variablen ist ähnlich gegliedert wie die PATSTAT-Datenbank selbst. Es gibt patentspezifische Informationen wie Patentnummer, Amt und Anmelde-/Prioritäts-/Publikationsdatum und auch rein personenspezifische Informationen wie Personennummer, Name und Ländercode. Erfinder-Nummer und Anmelder-Nummer beschreiben, an welcher Position die jeweilige Person als Erfinder respektive Anmelder auf einem Patent genannt wurde. Mit Hilfe der Punkte „Zitate“ und „Zitierungen“ kann ausgeben werden, welche Patente das jeweilige Patent zitiert hat, bzw. von welchen Patenten es zitiert wurde.

Beispiel für eine Ausgabetabelle

Im letzten Abschnitt besteht die Möglichkeit, das erwartete Ergebnis darzustellen. So können die Elemente der gegebenen Tabelle mit Beispieldaten und Werten gefüllt werden, damit direkt ersichtlich wird, welches Resultat von der Abfrage erwartet wird.
Dies kann sowohl direkt im Programm erfolgen, als auch handschriftlich, sofern das Dokument ausgedruckt wurde.
Generell hat ein Ausdruck des Fragebogens gegenüber dem Versand per Email den Vorteil, dass es auf jeden Fall zu einem persönlichen Gespräch zwischen Auftraggeber und bearbeitender Person kommt. Dieses ist, wie bereits erwähnt, unabdingbar und sollte vom Auftraggeber auch dann gesucht werden, wenn der Fragebogen auf elektronische Weise der bearbeitenden Person zugänglich gemacht wird.

Glossar

- A
  - **Anmeldedatum (filinge date)** – Datum der Vorlage eines Patentes bei einem Patentamt.
- C
  - **COLLATE** – Eine Klausel, die auf eine Datenbankdefinition oder eine Spaltendefinition angewendet werden kann, um die Sortierung zu definieren, oder auf einen Zeichenfolgenausdruck, um eine Sortierungsumwandlung anzuwenden.
  - **Core IPC und Advanced IPC** – IPC Klassifizierungsstufen (siehe PATSTAT-Anleitung)
  - **COUNT** – SQL-Operator. Zählt die gefundenen Ergebnisse.
- D
  - **DISTINCT** – SQL-Operator. Erweitert SELECT derart, dass nur unterschiedliche Elemente ausgegeben und redundante Zeilen ausgespart werden.
- E
  - **EPO** – European Patent Organisation (dt. EPA)
- F
  - **Filing Date** – siehe Anmeldedatum
  - **FROM** – SQL-Operator. Wählt die Tabelle aus, auf die die Operationen bezogen werden.
  - **FuE** – Forschung und Entwicklung
- G
  - **Gewicht** – Zur korrekten Aufsummierung von Patenten können diese anteilhaft aufgeteilt werden. Am Häufigsten geschieht dies auf Anmelder/Erfinder, IPC-Klassen und Technologiefelder.
  - **GROUP BY** – SQL-Operator. Gruppiert Abfrageergebnisse und fasst sie nach gegebenen Kriterien zusammen.
- I
  - **INNER JOIN** – SQL-Operator. Verknüpft zwei Tabellen anhand eines Attributes, das in beiden Tabellen vorkommt. Bei PATSTAT ist dies im Allgemeinen die appln_id.
  - **INSERT INTO** – SQL-Operator. Fügt das Ergebnis einer Abfrage in eine Tabelle ein, anstatt es auszugeben.
  - **IPC-Klasse** – International Patent Classification. Klassifizierung der technischen Inhalte von Patenten in unterschiedliche Klassen.
- K
  - **Konkordanz 19, 30 oder 44** – Klassifizierung von Patent, die eine Zuweisung zu einem der gegebenen Technologiefelder (unterschieden in 19, 30 oder 44 Technologiefelder) zulassen.

- L
    - **LIKE** – SQL-Operator. Wertet Textfelder hinsichtlich eines gegebenen Begriffes aus.
- N
    - **NUTS** – europäisches System zur geografischen Klassifizierung entsprechend nationaler der Verwaltungsgliederung. NUTS3 beschreibt dabei Regionen und Städte, in Deutschland z. B. die Landkreise und kreisfreien Städte.
- O
    - **ORDER BY** – SQL-Operator. Sortiert die Ergebnistabelle.
- P
    - **Patentamt** – Behörde, die Schutz durch Patente genehmigt. National hervorzuheben sind DPMA (Deutschland), USPTO (Vereinigte Staaten), international hervorzuheben sind EPO (Europäische Patentorganisation) und WIPO (Weltorganisation für geistiges Eigentum).
    - **PATSTAT** - EPO Worldwide Patent Statistical Database, Patentdatenbank für den akademischen und institutionellen Gebrauch.
    - **Primärschlüssel** – Dient der eindeutigen Identifizierung einer Datenzeile aus einer Gruppe von Wertekombinationen.
    - **Prioritätsdatum** – Datum der ersten Vorlage eines Patentes, unabhängig des Patentamtes.
    - **Publikationsdatum** – Datum der Veröffentlichung eines Patentes nach dessen Erteilung.
- Q
    - **Query** – Datenbankabfrage.
- R
    - **Relationale Datenbank** – etablierter Standard für Datenbanken, basierend auf relationaler Algebra. Relationen werden durch Tupel beschrieben, die die eigentlichen Datensätze beschrieben.
- S
    - **SELECT** – SQL-Operator. Einschränkung der Ergebnismenge auf gewünschte Attribute, bestimmt auszugebende Spalten einer Tabelle
    - **SQL** – Structured Query Language. Einfach aufgebaute Datenbankabfragesprache, die sich an die englische Sprache anlehnt. Wichtigste Operatoren sind SELECT, FROM und WHERE.
- T
    - **Tabelle** – Relation in einer relationalen Datenbank.
    - **TechField** – Zuordnung von IPC-Klassen in bestimmte Technologiefelder, um eine generellere Klassifizierung zu ermöglichen.
    - **Tech30** – (siehe Konkordanz 30)
    - **Tupel** – eigentlicher Datensatz in einer Tabelle.

- Y
    - **YEAR** – SQL-Operator. Wandelt Datumsangaben vom ausführlichen DATETIME-Format in das Kalenderjahr um.
- W
    - **WHERE** – SQL-Operator. Wertet Zeilen hinsichtlich einer nachgestellten Bedingung aus.
    - **WIPO** – World Intellectual Property Organisation

Literatur

Codd, E.F. (1990): *The Relational Model for Database Management*. Version 2. Addison-Wesley, Reading u.a. 1990

Grupp, H. (1998), *Foundations of the Economics of Innovation. Theory, Measurement and Practice*, Chetenham, UK: Edward Elgar

Hinze, S., Reiss, T., Schmoch, U. (1997), *Statistical Analysis on the Distance Between Fields of Technology*, Report for European Commission TSER Project, September.

Kemper, A. und Eickler, A. (2004), *Datenbanksysteme. Eine Einführung*. Oldenbourg, München 2004

Schmoch, U., Laville F., Patel P. and Frietsch R. (2003), *Linking Technology Areas to Industrial Sectors*, Final Report to the European Commission, DG Research

Schmoch, U. (2008), *Concept of a Technology Classification for Country Comparisons*, Final Report to the World Intellectual Property Organisation (WIPO), June 2008