Schnelleinstieg in ANNIS

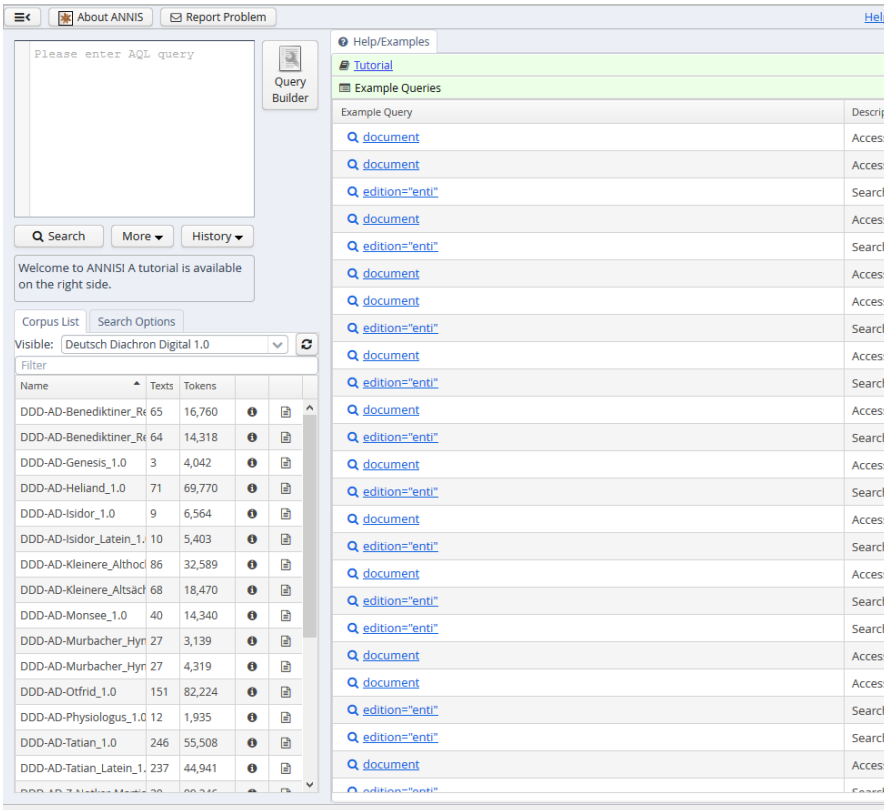
Inhalt

Über die Schnittstelle ANNIS ist eine wachsende Zahl an Korpora zugänglich, die für die historische Sprachwissenschaft des Deutschen hochinteressant sind. Da sich die Referenzkorpora Altdeutsch, Mittelhochdeutsch und (demnächst) Frühneuhochdeutsch dieser Plattform bedienen, ist ihre Bedeutung in den vergangenen Jahren sehr gewachsen. Nur wenige Tage vor Veröffentlichung dieses Tutorials ist auch die 2017er Version des Bonner Frühneuhochdeutschkorpus erschienen, die sich ebenfalls der ANNIS-Plattform bedient. Daher soll im Folgenden ein kurzer Einstieg gegeben werden, der natürlich die weitaus ausführlicheren Tutorials, die auf der ANNIS-Seite sowie auf den Seiten der HU Berlin verfügbar sind, nicht ersetzen kann.
Der Titel dieses Tutorials ist eigentlich ein Oxymoron, denn ein Schnelleinstieg in ANNIS ist ebenso wenig möglich wie z.B. ein Schnelleinstieg in R (was mich nicht davon abgehalten hat, auch das Tutorial zu R so zu nennen). Das Verständnis von ANNIS kann man jedoch erheblich beschleunigen, wenn man die zugrundeliegenden Prinzipien versteht. Dazu will dieser Text beitragen.
Im Folgenden wähle ich als Beispiel das Referenzkorpus Altdeutsch (REA), das mit dem Referenzkorpus Mittelhochdeutsch (REM), dem Referenzkorpus Mittelniederdeutsch/Niederrheinisch (REN) und dem noch nicht veröffentlichten Referenzkorpus Frühneuhochdeutsch zur Familie der „Deutsch Diachron Digital“-Referenzkorpora gehört.[1] Es ist über ANNIS frei und ohne Registrierung zugänglich, und zwar unter https://korpling.german.hu-berlin.de/annis3/ddd.

[1] In einer früheren Version dieses Tutorials hatte ich die Wahl des Beispiels auch damit begründet, dass die Texte des REA im Unterschied zum REM nicht herunterladbar seien. Tatsächlich jedoch sind die Daten über das LAUDATIO-Repository verfügbar: http://www.laudatio-repository.org/repository/corpus/ddd%3Addd-ad/TEI-header_version1_Schema7_2018-06-22T07%3A33%3A54%3A457Z

1. Die Suchmaske

Wenn Sie den o.g. Link aufrufen, sehen Sie zunächst Folgendes:



Die Seite enthält oben links das Suchfenster, in das wir unsere Suchanfragen eingeben müssen. Unten links sehen wir die Korpusliste: Hier müssen wir mindestens ein Korpus auswählen. Rechts sehen wir eine Reihe von Beispielanfragen. Mit Klick auf „Tutorial“ können wir hier auch ein Tutorial aufrufen, in dem das ANNIS-Interface eingehend erklärt wird.

2. Korpusauswahl

Bevor wir uns der Suchsyntax zuwenden, wählen wir zunächst ein Korpus bzw. mehrere Korpora aus. Wie Sie in der Korpusliste unten links sehen, besteht das Referenzkorpus Altdeutsch aus vielen einzelnen Subkorpora. Theoretisch können wir alle auswählen (indem wir einen auswählen und dann mit gedrückter Shift-Taste den Pfeil nach unten betätigen), aber oft mag es auch Gründe geben, nicht alle zu benutzen (z.B. die lateinischsprachigen Texte auszusparen). Ist kein Text ausgewählt, funktioniert die Suche nicht.

3. Suchanfragen erstellen

Suchanfragen müssen in der sog. Annis Query Language (AQL) formuliert werden und können entweder manuell eingegeben oder mit Klick auf „Query Builder“ interaktiv zusammengestellt werden. Gerade bei den Referenzkorpora Altdeutsch und Mittelhochdeutsch kann es sehr sinnvoll sein, den Query Builder zu benutzen – gerade dann, wenn man mit der Lemmaebene arbeitet: Hier kommen sehr viele Sonderzeichen zum Einsatz, und das Lemma, nach dem man suchen möchte, ist oft nicht aus dem Neuhochdeutschen vorhersagbar (und auch nicht aus den

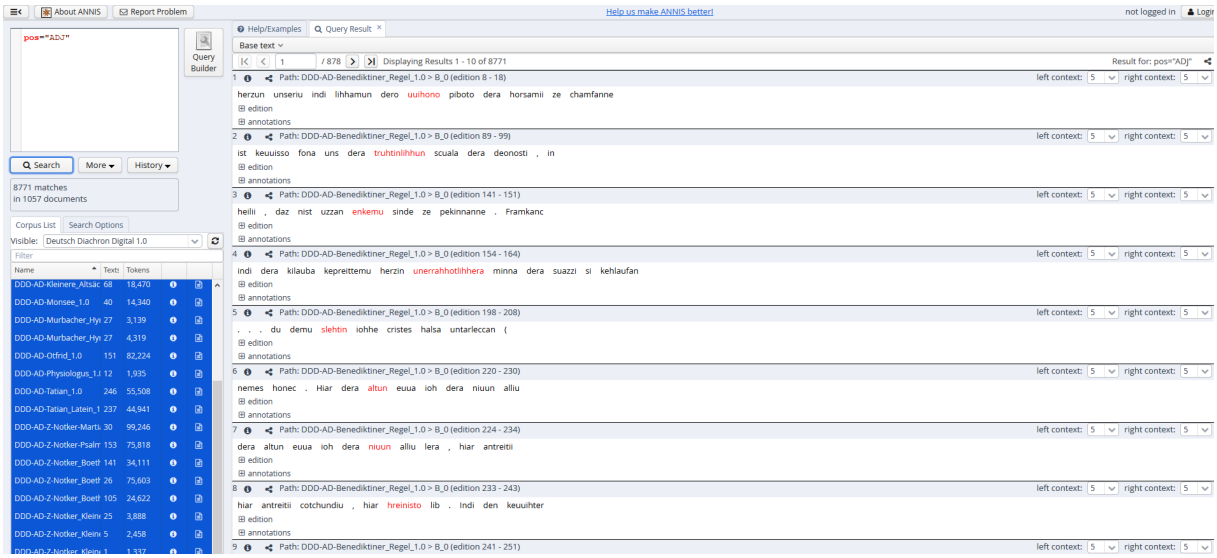
eventuell vorhandenen Kenntnissen des „Normalalthochdeutschen“ oder „Normalmittelhochdeutschen“).
Um die Suchsyntax zu verstehen, ist jedoch sinnvoll, sich zunächst anhand einfacher Beispiele mit der manuellen Eingabe vertraut zu machen.

3.1 Nach einem einzigen Token[2] suchen

Die Suchabfragesyntax folgt dabei einem einfachen Prinzip: Es müssen jeweils Attribut-Wert-Paare angegeben werden, also z.B.

pos	=	"ADJ"
Attribut		Wert

Wenn wir genau diese Suchanfrage ins ANNIS-Suchfeld eingeben, erhalten wir alle Tokens, die mit dem POS-Tag „ADJ“ versehen sind:



Wie aus der Ergebniszusammenfassung unter dem Suchfenster hervorgeht, finden sich insgesamt um die 8700 Treffer. Für ein Korpus mit über 650.000 Wörtern ist das eine relativ geringe Anzahl: Es wurden nämlich bei weitem nicht alle Adjektive gefunden, wie man vielleicht auf den ersten Blick denken könnte, sondern nur diejenigen, die genau mit dem Tag „ADJ“ versehen sind. Viele tragen aber auch den Tag „ADJA“ (attributives Adjektiv) oder „ADJD“ (determinatives Adjektiv). Hier kommen wieder **reguläre Ausdrücke** ins Spiel. Um reguläre Ausdrücke in ANNIS benutzen zu können, muss man – ähnlich übrigens wie beim Deutschen Textarchiv – Slashes verwenden, z.B.

pos = /ADJ.?/
(findet alle Tokens, bei deren POS-Annotation auf ADJ noch maximal genau ein Zeichen folgt)

oder

[2] Genau genommen sucht ANNIS nicht nach „Tokens“, sondern vielmehr nach *nodes* und *edges*. Hier vereinfache ich aber ganz bewusst und verweise für detailliertere Informationen auf die Dokumentation unter http://corpus-tools.org/annis/aql.html (zuletzt abgerufen November 2017).

pos = /ADJ.*/
(findet alle Tokens, bei deren POS-Annotation auf ADJ beliebig viele Zeichen oder kein Zeichen mehr folgt).

3.2 Nach mehreren Tokens suchen
Natürlich können wir z.B. auch nach Belegen suchen, bei denen auf ein Adjektiv ein Substantiv folgt. Dafür müssen wir wissen, wie man in AQL **Abstandsoperatoren** einsetzt. Wenn wir zum Beispiel Belege finden wollen, in denen ein Substantiv *unmittelbar* auf das Adjektiv folgt, können wir den Abstandsoperator . verwenden. Dass der Punkt als Abstandsoperator dient, ist ein Spezifikum von AQL und zunächst vielleicht etwas verwirrend, weil wir ihn ja schon bei den regulären Ausdrücken als Platzhalter kennengelernt (und oben unter 3.1 auch so eingesetzt) haben. In AQL bedeutet ein Punkt, der ohne weitere Modifikation als Abstandsoperator eingesetzt wird, so viel wie „das, was nach dem Punkt steht, folgt unmittelbar auf das, was vor dem Punkt steht", also z.B.:

pos=/ADJ.?/ . pos=/N./

Lies: Ein als Nomen (NA = „Nomen Appellativum"[3], oder NE = Eigenname) getaggtes Token folgt unmittelbar auf ein als Adjektiv (ADJ, ADJA oder ADJD, s.o.) getaggtes Token.
Was ich soeben gezeigt habe, ist die sogenannte **verkürzte Schreibweise**. Alternativ gibt es noch die sog. **Klauselschreibweise**, in der wir zuerst die einzelnen Attribut-Wert-Paare spezifizieren und dann festlegen, wie sie zueinander in Relation stehen:

verkürzte Schreibweise	Klauselschreibweise
pos=/ADJ.?/ . pos=/N./	pos=/ADJ.?/ & das ist #1 pos=/N./ & das ist #2 #1 . #2 #1 vor #2

Die Klauselschreibweise – die ich hier mit grauen Erläuterungen versehen habe, die natürlich nicht mit eingegeben werden dürfen! – ist etwas umständlicher, hat aber auch ihre Vorteile. Gerade bei komplexen Suchanfragen kann es manchmal zur Übersichtlichkeit beitragen, wenn man zuerst die einzelnen Attribut-Wert-Paare (um der Lesbarkeit willen eines pro Zeile) spezifiziert und sie anschließend „verkettet". Wie Sie an der farblichen Kodierung erkennen können, werden die einzelnen Elemente der Suchanfrage bei der Klauselschreibweise zunächst über & verbunden. Dem & kommt aber keine weitere Funktion zu, als ANNIS zu sagen, dass alle genannten Elemente in die Suche einbezogen werden sollen. Erst der letzte Teil der Anfrage gibt dann noch die Information, wie die beiden Tokens, nach denen gesucht wird, zueinander in Verbindung stehen.
Selbstverständlich können wir nicht nur nach direkt aufeinanderfolgenden Tokens suchen. Wir können z.B. auch nach Adjektiven suchen, denen im Abstand von 1 bis 2 Wörtern ein Substantiv folgt. Dafür können wir einfach nach dem Punkt einen Mindest- und einen Höchstabstand definieren:

[3] Dieser Tag ist zugegebenermaßen etwas verwirrend und problematisch, denn normalerweise bedeutet NA „Not Available"; wenn man vorhat, Exportdateien aus ANNIS in R einzulesen, muss man daher aufpassen, dass R die Zellen, die den Wert NA enthalten, nicht als „leere" bzw. ungültige Zellen behandelt.

verkürzte Schreibweise	Klauselschreibweise
pos=/ADJ.?/ .1,2 pos=/N./	pos=/ADJ.?/ & pos=/N./ & #1 .1,2 #2

3.3 Nach einem Token suchen, das mehrere Kriterien zugleich erfüllt

Die Methode, die wir in 3.2 kennengelernt haben, können wir auch verwenden, um nach einem Token zu suchen, das mehrere Kriterien zugleich erfüllt. Angenommen, wir wollen nach dem Lemma *man* suchen, aber nur die Ergebnisse finden, in denen *man* als Nomen Appellativum (NA) getaggt ist.

verkürzte Schreibweise	Klauselschreibweise
lemma="man" _=_ pos ="NA"	lemma="man" & pos ="NA" & #1 _=_ #2

_=_ bedeutet, dass die Annotationen, nach denen man sucht, genau die gleiche Spanne abdecken (*identical coverage*). Wir können uns also merken: x _=_ y bedeutet „x ist gleich y“, x . y bedeutet „x kommt vor y“, wobei man Letzteres, wie oben gesehen, noch mit genaueren Abständen modifizieren kann. Das Cheat Sheet am Ende des Tutorials zeigt noch weitere Optionen, mit denen man z.B. Tokens in einem bestimmten Abstand vor *oder* nach einem anderen Token suchen kann.

Natürlich kann man auch nach Tokens suchen, die mehr als zwei Kriterien gleichzeitig erfüllen; der Komplexität der Suchanfrage ist keine andere Grenze gesetzt als die, die durch die verfügbaren Annotationen des jeweiligen Korpus vorgegeben ist.

3.4 Arbeit mit dem Query Builder

Wenden wir uns nun dem bereits erwähnten Query Builder zu, mit dem man die oben erwähnten Suchanfragen interaktiv „basteln“ kann. Wenn man weiß, wonach man suchen muss, ist es nach meiner Erfahrung einfacher und schneller, die Suche manuell einzugeben, aber wie ebenfalls bereits erwähnt, kommt man gerade bei der Suche nach bestimmten Lemmata nicht besonders weit, wenn man einfach nach neuhochdeutschen Wörtern sucht, wie beispielsweise ein Blick auf die „Lemma“-Ebene zu Beginn des REA-Korpustexts „Kleinere althochdeutsche Denkmäler“ zeigt:

1 Path: DDD-AD-Kleinere_Althochdeutsche_Denkmäler_1.0 > AB_AltbairischeBeichte (edition 1 - 6) left context: 5 right

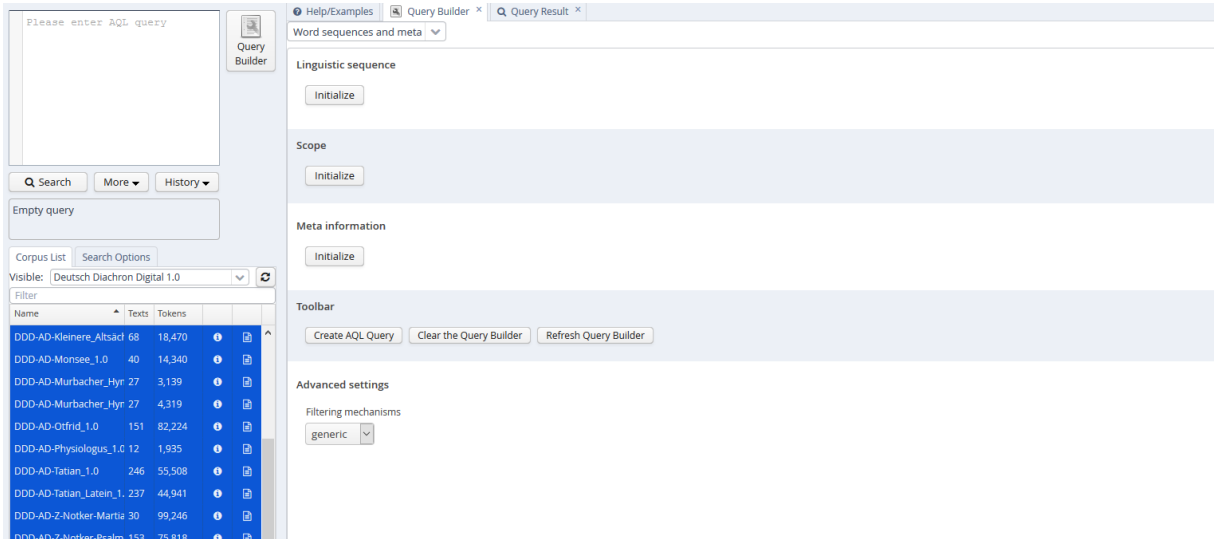
Truhtin , dir uuirdu ih pigihtik

annotations

edition	Truhtin	,	dir	uuirdu	ih	pigihtik
text	Truhtin	,	dir	uuirdu	ih	pigihtik
lemma	truhtin		du	werdan	ih	bijihtig
posLemma	NA	$,	PPER	VA	PPER	ADJ
pos	NA	$,	PPER	VAFIN	PPER	ADJD
inflectionClassLemma	A_MASC			ST3B		A,O
inflectionClass	A_MASC			ST3B		A,O
inflection	SG_NOM		SG_DAT_2	IND_PRES_SG_1	SG_NOM_1	POS
lang	goh		goh	goh	goh	goh
clause	CF_CS_U_M					
line	1					
translation	Herr		du	werden	ich	bījihtīg wērdan: (seine Sünden) bekennen, beichten

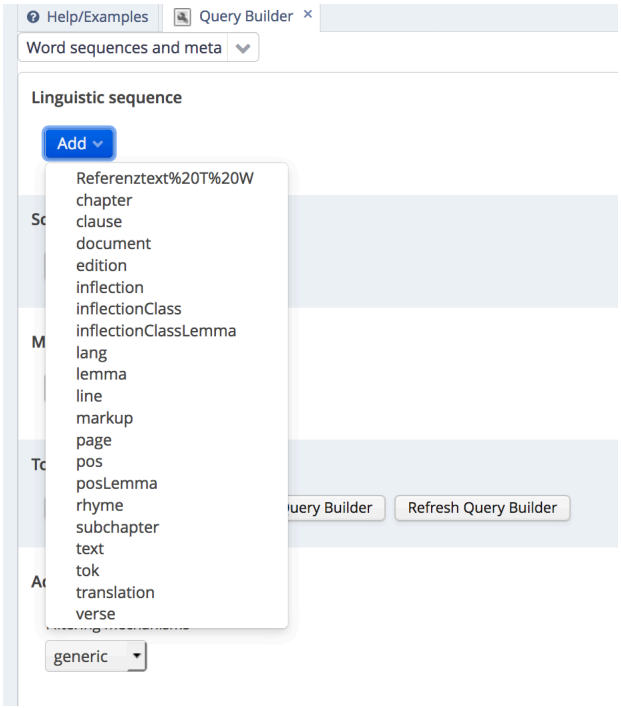
edition

Die Lemma-Ebene orientiert sich, wie wir sehen, auch (ortho)graphisch relativ stark am Althochdeutschen; im REM ist sie sogar noch weniger intuitiv, da die Lemmata extrem viele Sonderzeichen aufweisen (*be-gègenen, wërden, er-wèl(e)t*, um nur einige Beispiele zu nennen – auch die Klammern sind unmittelbar übernommen und sind natürlich unvorhersagbar, wenn man nicht sehr gut mit der Lemmatisierung vertraut ist). Gerade für die Lemmasuche empfiehlt sich somit in den Referenzkorpora zu den älteren Sprachstufen die Benutzung des Query Builders.
In den Query Builder gelangen wir durch Klick auf den fast unübersehbaren Button rechts vom Suchfenster. Im Query Builder sehen wir zunächst eine ganze Reihe von Optionen:

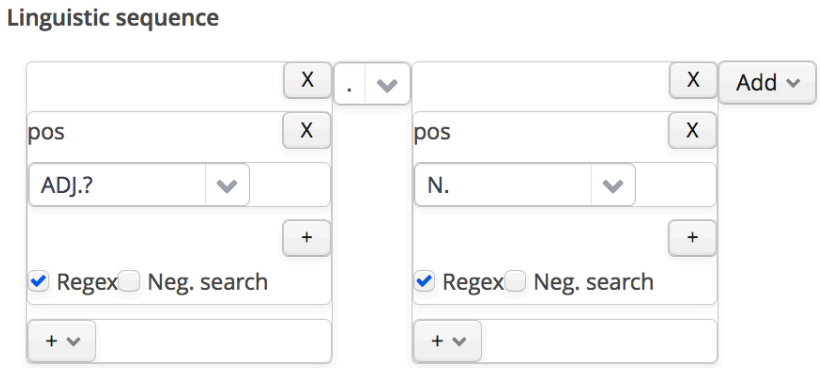


Im Dropdown-Menü ganz oben können wir zunächst zwischen zwei verschiedenen „Varianten“ des Query Builders wählen: „Word Sequences and Meta Information“ einerseits, „General (TigerSearch-like)“ andererseits. Die letztgenannte Ansicht ist v.a. für **Baumbanken** hilfreich, weil man dort gezielt nach „Knoten“ und „Kanten“ suchen kann (vgl. Kap. 6 in Hartmann 2018). Bei den Referenzkorpora, mit denen wir uns in diesem Tutorial beschäftigen, handelt es sich nicht um Baumbanken, deshalb werden wir diese Option hier ignorieren und uns ausschließlich auf die „Word Sequences and Meta Information“-Version konzentrieren (für eine hervorragende Anleitung zur Suche in Baumbanken vgl. Dipper 2015).
Wenn wir die oben manuell eingegebene Suchanfrage mit dem Query Builder kreieren möchten, dann müssen wir eine Reihe von „Linguistic sequences“ kombinieren; ganz oben sehen wir das entsprechende Feld, das wir zunächst mit Klick auf „Initialize“ initialisieren müssen. Im

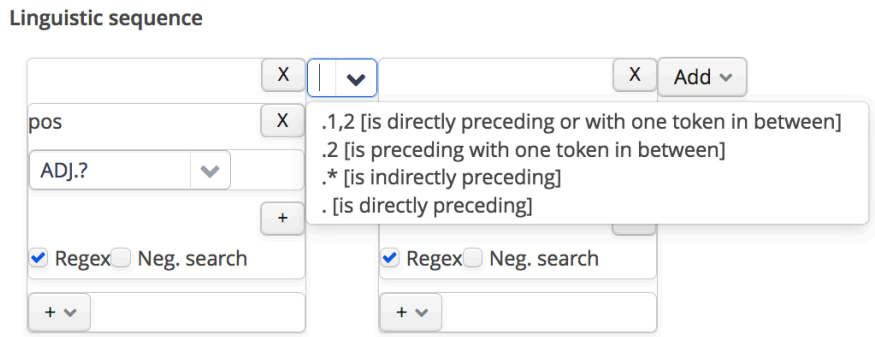
Dropdown-Menü „Add“ können wir dann eine Ebene auswählen, auf der wir suchen möchten, z.B. die Lemma- oder POS-Ebene:



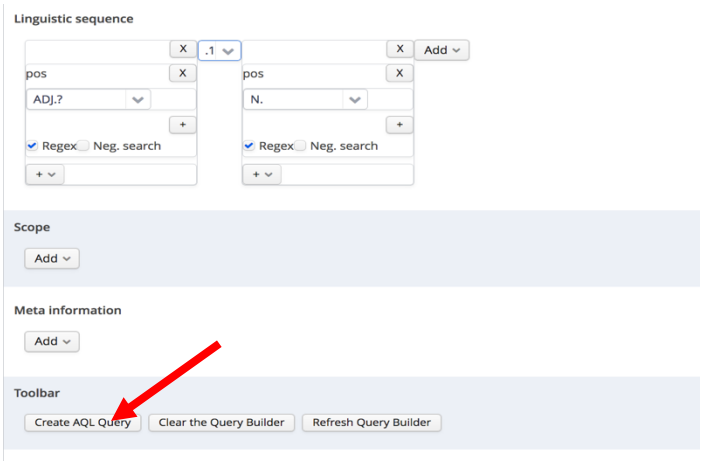
Versuchen wir, die Anfrage pos=/ADJ.?/ .1,2 pos=/N./ nachzubilden. Dafür wählen wir aus der Liste zunächst „pos“ aus und geben dann ADJ. ins entsprechende Feld ein. Anschließend klicken wir noch einmal auf „Add“, wählen aus dem Dropdown-Menü erneut „pos“ aus und geben dann ins neu erschienene Feld „N.“ ein.



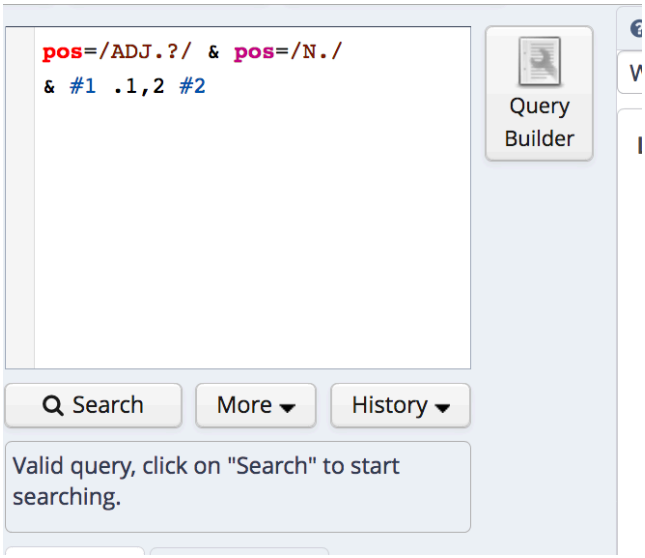
Im Dropdown-Menü zwischen den beiden Konstituenten können wir nun auswählen, in welchem Verhältnis die beiden zueinander stehen sollen; defaultmäßig ausgewählt ist, wie wir im obigen Screenshot sehen, das Aufeinanderfolgen (also der Punkt, der uns bereits begegnet ist). Wir wollen aber eine flexiblere Präzedenzrelation (Abstand von 1 bis 2 Tokens), deshalb müssen wir aus dem Dropdown-Menü die entsprechende Option auswählen, nämlich .1,2:



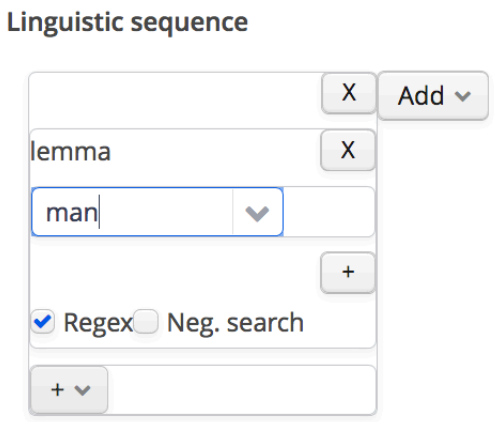
Jetzt können wir die Suchanfrage „bauen“, indem wir auf den entsprechenden Button „Create AQL Query“ klicken.



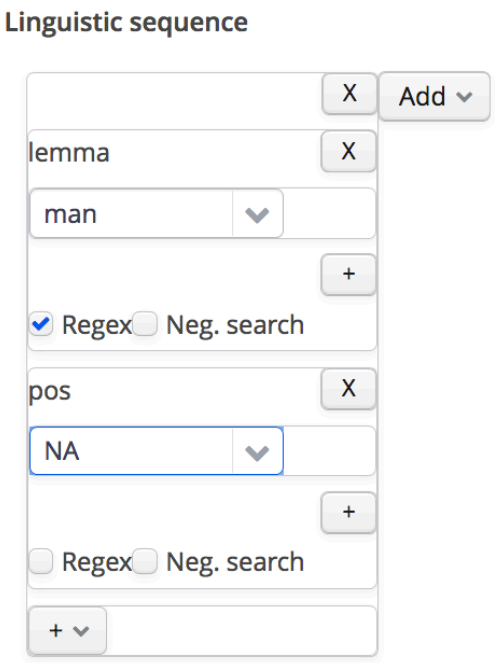
Durch Klick auf den Button wird die interaktiv erstellte Suchanfrage ins Suchfenster übertragen und sieht nun so aus wie die manuell erstellte Anfrage in der Klausel-Syntax:



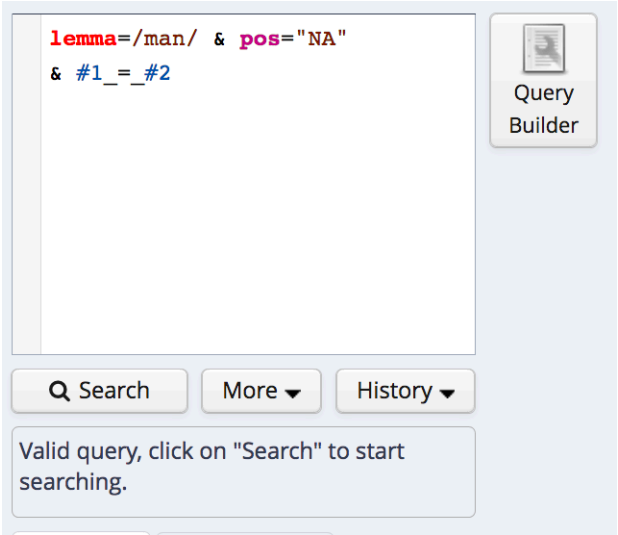
Versuchen wir nun noch, die zweite Beispiel-Anfrage von oben, lemma="man" _=_ pos ="NA", nachzubilden. Dafür klicken wir zunächst auf „Clear the Query Builder“, um im interaktiven Anfrageinterface quasi *tabula rasa* zu machen und wieder von vorn zu beginnen. Erneut klicken wir bei „Linguistic Sequence“ auf „Initialize“, dann auf „add“ und wählen diesmal „lemma“ aus dem Dropdown-Menü aus, um anschließend „man“ in das Fenster einzugeben (oder es aus dem Dropdown-Menü auszuwählen, das alle verfügbaren Lemmata anzeigt, was, wie gesagt, bei Lemmatisierungen mit unvorhersagbaren Eigenschaften wie Sonderzeichen, Klammern usw. hilfreich ist). Nun sehen wir, dass es neben dem „Add“-Button rechts von der eben eingegebenen Konstituente noch ein kleines „+“ unten links im Fenster gibt:



Mit diesem „+“ können wir nun eine weitere Annotationsebene auswählen, in diesem Fall „pos“, und dort dann „NA“ eingeben. Im Query Builder müssen wir also ein wenig umdenken: Bei der manuellen Suchanfrage macht es im Grunde keinen Unterschied, ob die beiden Elemente, nach denen wir suchen, in einer Präzdenzrelation stehen oder dieselbe Spanne abdecken; wir müssen nur den Operator zwischen den beiden Elementen, nach denen wir suchen, ändern. Im Query Builder hingegen müssen wir unterschiedlich vorgehen, je nachdem, in welchem Verhältnis die beiden Konstituenten zueinander stehen. Sicherlich haben Sie auch schon den Sinn hinter diesen unterschiedlichen Wegen erkannt: Wenn wir auf das „Add“ rechts vom Fenster klicken, erscheint ein neues Fenster rechts von der ersten Konstituente, die wir erstellt haben. Diese horizontale Darstellung trägt der Tatsache Rechnung, dass wir in diesem Fall nach zwei *verschiedenen* Tokens suchen, die aufeinanderfolgen oder zumindest nah beieinander stehen. Mit dem kleinen +-Zeichen hingegen, das wir jetzt gewählt haben, machen wir klar, dass wir *dasselbe* Token noch näher spezifizieren wollen. Deshalb erscheint der nächste Teil der Suchanfrage auch ganz ikonisch unterhalb des ersten Teils:



Wenn wir wieder auf „Create AQL Query“ klicken, erscheint erneut die Suchanfrage, wie wir sie oben in der Klausel-Syntax formuliert haben, im Suchfeld:

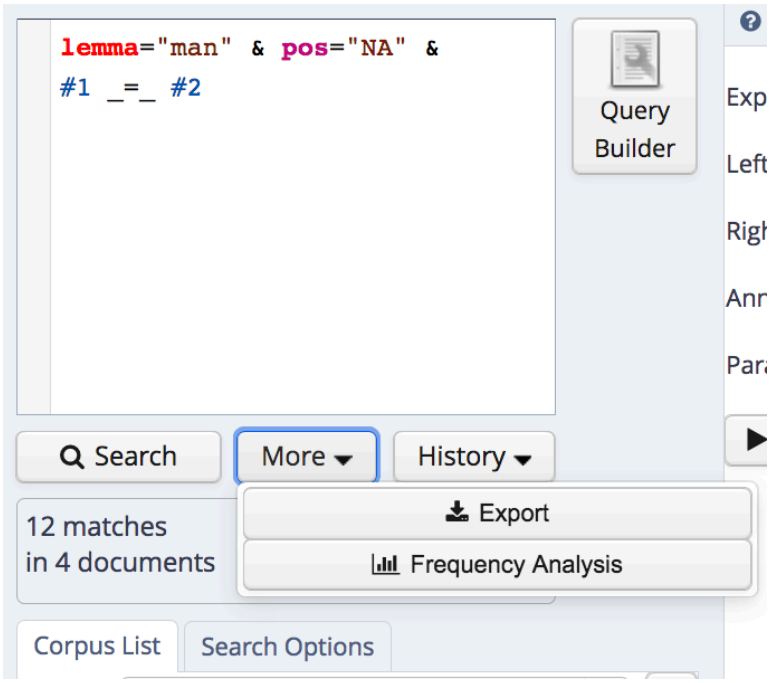


Die Tatsache, dass die Suchanfrage ins Suchfenster übertragen wird, gibt auch die Möglichkeit, sie dort noch flexibel anzupassen, denn nicht alle Suchanfragen, die in AQL möglich sind, lassen sich direkt über den Query Builder generieren. So haben wir oben gesehen, dass das Dropdown-Menü zu Präzedenzrelationen nur die vier Optionen ., .1,2, .2 und .* zulässt; so etwas wie .1,7 (Abstand von 1 bis 7 Wörtern) ist hier nicht vorgesehen, kann aber im Suchfeld natürlich problemlos manuell nachgetragen werden.

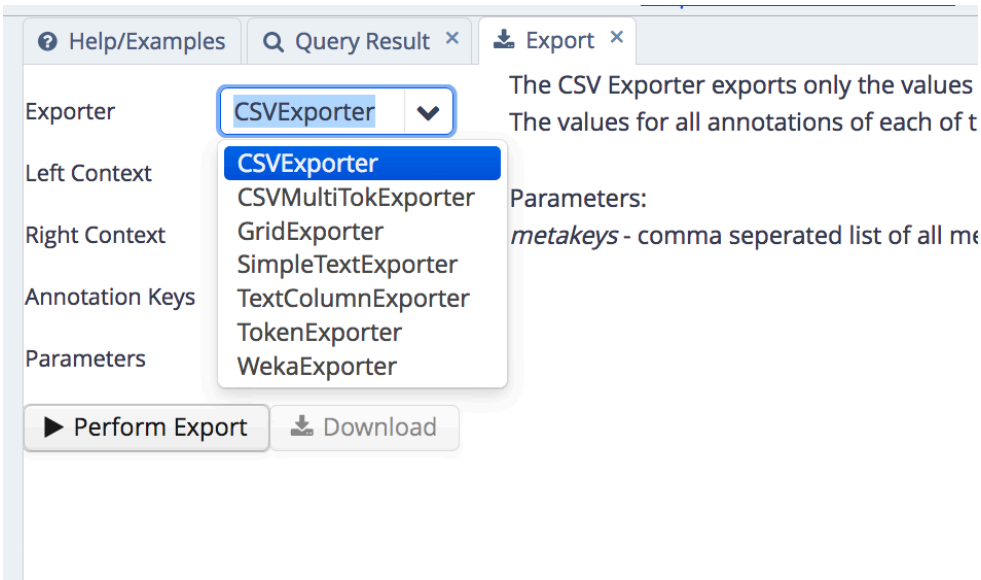
4. Export aus ANNIS

Die Exportfunktionen von ANNIS sind einerseits sehr vielfältig, andererseits gerade für AnfängerInnen z.T. nicht ganz unproblematisch, denn während einige Exportmöglichkeiten zu simpel sind (etwa der SimpleTextExporter), ist der Output mancher der Exporter, die ANNIS anbietet,

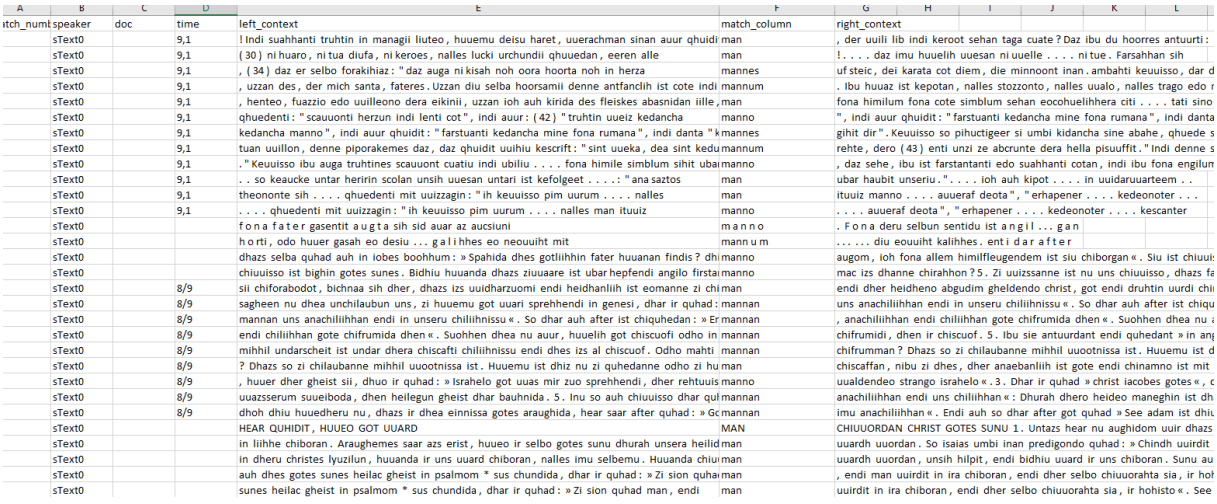
zu komplex. Glücklicherweise gibt es seit kurzem auch den TextColumnExporter, mit dem man relativ bequem KWIC-Konkordanzen exportieren kann, die sich dann in Spreadsheet-Programmen öffnen und weiter bearbeiten lassen. Leider ist dieser Exporter noch nicht in allen Korpora, die ANNIS verwenden, implementiert. Im REA ist er vorhanden, in REM und im Bonner Frühneuhochdeutschkorpus derzeit noch nicht.
Schauen wir uns zunächst an, wie man im REA zum TextColumnExporter gelangt, ehe wir uns mit den „problematischeren“ Korpora auseinandersetzen.
Unterhalb des Suchfensters sehen wir den „More“-Button, mit dem wir zur Export-Option gelangen.



Im Export-Fenster können wir nun zwischen verschiedenen Exportern wählen:



Rechts neben dem Auswahlmenü wird eine kurze Erläuterung des jeweiligen Exporters angezeigt, sodass sich eine genauere Erläuterung der einzelnen zur Verfügung stehenden Exportvarianten an dieser Stelle erübrigt. Wichtig zu wissen ist, dass viele der Exporter den Kontext ignorieren, also gerade keinen „Key Word In Context"-Export ermöglichen. Je nachdem, was man mit den Daten vorhat, kann das unter Umständen genau das sein, was man als Benutzerin oder als Benutzer will. In diesen Tutorials gehen wir aber von dem Standardszenario aus, dass wir – gerade bei historischen Sprachdaten – mit der KWIC-Konkordanz weiterarbeiten möchten. Deshalb ist der erwähnte TextColumnExporter für unsere Zwecke ideal. Die csv-Datei, die er generiert, lässt sich so wie in Tutorial „01-Grundlegendes" beschrieben in einem Tabellenkalkulationsprogramm wie Excel oder Calc öffnen:



itch_numl	speaker	doc	time	left_context	match_column	right_context
	sText0		9,1	! Indi suahhanti truhtin in managii liuteo , huuemu deisu haret , uuerachman sinan auur qhuidi	man	, der uuili lib indi keroot sehan taga cuate ? Daz ibu du hoorres antuurti :
	sText0		9,1	(30) ni huaro , ni tua diufa , ni keroes , nalles lucki urchundii qhuuedan , eeren alle	man	! . . . . daz imu huuelih uuesan ni uuelle . . . . ni tue . Farsahhan sih
	sText0		9,1	, (34) daz er selbo forakihiaz : " daz auga ni kisah noh oora hoorta noh in herza	mannes	uf steic , dei karata cot diem , die minnoont inan . ambahti keuuisso , dar d
	sText0		9,1	, uzzan des , der mich santa , fateres . Uzzan diu selba hoorsamii denne antfanclih ist cote indi	mannum	. Ibu huuaz ist kepotan , nalles stozzonto , nalles uualo , nalles trago edo r
	sText0		9,1	, henteo , fuazzio edo uuilleono dera eikinii , uzzan ioh auh kirida des fleiskes abasnidan iille ,	man	fona himilum fona cote simblum sehan eocohuelihhera citi . . . . tati sino
	sText0		9,1	qhuedenti : " scauuonti herzun indi lenti cot " , indi auur : (42) " truhtin uueiz kedancha	manno	" , indi auur qhuidit : " farstuanti kedancha mine fona rumana " , indi danta
	sText0		9,1	kedancha manno " , indi auur qhuidit : " farstuanti kedancha mine fona rumana " , indi danta " k	mannes	gihit dir " . Keuuisso so pihuctigeer si umbi kidancha sine abahe , qhuede s
	sText0		9,1	tuan uuillon , denne piporakemes daz , daz qhuidit uuihiu kescrift : " sint uueka , dea sint kedu	mannum	rehte , dero (43) enti unzi ze abcrunte dera hella pisuuffit . " Indi denne s
	sText0		9,1	. " Keuuisso ibu auga truhtines scauuont cuatiu indi ubiliu . . . . fona himile simblum sihit ubai	manno	, daz sehe , ibu ist farstantanti edo suahhanti cotan , indi ibu fona engilum
	sText0		9,1	. . so keaucke untar heririn scolan unsih uuesan untari ist kefolgeet . . . . : " ana saztos	man	ubar haubit unseriu . " . . . . ioh auh kipot . . . . in uuidaruuarteem . .
	sText0		9,1	theononte sih . . . . qhuedenti mit uuizzagin : " ih keuuisso pim uurum . . . . nalles	man	ituuiz manno . . . . auueraf deota " , " erhapener . . . . kedeonoter . . .
	sText0		9,1	. . . . qhuedenti mit uuizzagin : " ih keuuisso pim uurum . . . . nalles man ituuiz	manno	. . . . auueraf deota " , " erhapener . . . . kedeonoter . . . . kescanter
	sText0			f o n a f a t e r gasentit a u g t a sih sid auar az aucsiuni	m a n n o	. F o n a deru selbun sentidu ist a n g i l . . . g a n
	sText0			h o r ti , odo huuer gasah eo desiu . . . g a l i hhes eo neouuiht mit	mann u m	. . . . . diu eouuiht kalihhes . ent i d a r a f t e r
	sText0			dhazs selba quhad auh in iobes boohhum : » Spahida dhes gotliihhin fater huuanan findis ? dh	manno	augom , ioh fona allem himilfleugendem ist siu chiborgan « . Siu ist chiuui
	sText0			chiuuisso ist bighin gotes sunes . Bidhiu huuanda dhazs ziuuaare ist ubar hepfendi angilo firsta	manno	mac izs dhanne chirahhon ? 5 . Zi uuizssanne ist nu uns chiuuisso , dhazs fa
	sText0		8/9	sii chiforabodot , bichnaa sih dher , dhazs izs uuidharzuomi endi heidhanliih ist eomanne zi chi	man	endi dher heidheno abgudim gheldendo christ , got endi druhtin uurdi chi
	sText0		8/9	sagheen nu dhea unchilaubun uns , zi huuemu got uuari sprehhendi in genesi , dhar ir quhad :	mannan	uns anachiliihhan endi in unseru chiliihnissu « . So dhar auh after ist chiqu
	sText0		8/9	mannan uns anachiliihhan endi in unseru chiliihnissu « . So dhar auh after ist chiquhedan : » Er	mannan	, anachiliihhan endi chiliihhan gote chifrumida dhen « . Suohhen dhea nu
	sText0		8/9	endi chiliihhan gote chifrumida dhen « . Suohhen dhea nu auur , huuelih got chiscuofi odho in	mannan	chifrumidi , dhen ir chiscuof . 5 . Ibu sie antuurdant endi quhedant » in ang
	sText0		8/9	mihhil undarscheit ist undar dhera chiscafti chiliihnissu endi dhes izs al chiscuof . Odho mahti	mannan	chifrumman ? Dhazs so zi chilaubanne mihhil uuootnissa ist . Huuemu ist d
	sText0		8/9	? Dhazs so zi chilaubanne mihhil uuootnissa ist . Huuemu ist dhiz nu zi quhedanne odho zi hu	man	chiscaffan , nibu zi dhes , dher anaebanliih ist gote endi chinamno ist mit
	sText0		8/9	, huuer dher gheist sii , dhuo ir quhad : » Israhelo got uuas mir zuo sprehhendi , dher rehtuuis	manno	uualdendeo strango israhelo « . 3 . Dhar ir quhad » christ iacobes gotes « ,
	sText0		8/9	uuazsserum suueiboda , dhen heilegun gheist dhar bauhnida . 5 . Inu so auh chiuuisso dhar qui	mannan	anachiliihhan endi uns chiliihhan « : Dhurah dhero heideo maneghin ist dh
	sText0		8/9	dhoh dhiu huuedheru nu , dhazs ir dhea einnissa gotes araughida , hear saar after quhad : » Gc	mannan	imu anachiliihhan « . Endi auh so dhar after got quhad » See adam ist dhiu
	sText0			HEAR QUHIDIT , HUUEO GOT UUARD	MAN	CHIUUORDAN CHRIST GOTES SUNU 1 . Untazs hear nu aughidom uuir dhazs
	sText0			in liihhe chiboran . Araughemes saar azs erist , huueo ir selbo gotes sunu dhurah unsera heilid	man	uuardh uuordan . So isaias umbi inan predigondo quhad : » Chindh uuirdit
	sText0			in dheru christes lyuzilun , huuanda ir uns uuard chiboran , nalles imu selbemu . Huuanda chiui	man	uuardh uuordan , unsih hilpit , endi bidhiu uuard ir uns chiboran . Sunu au
	sText0			auh dhes gotes sunes heilac gheist in psalmom * sus chundida , dhar ir quhad : » Zi sion quhai	man	, endi man uuirdit in ira chiboran , endi dher selbo chiuuorahta sia , ir hol
	sText0			sunes heilac gheist in psalmom * sus chundida , dhar ir quhad : » Zi sion quhad man , endi	man	uuirdit in ira chiboran , endi dher selbo chiuuorahta sia , ir hohisto « . See

Literatur

Dipper, Stefanie. 2015. Annotierte Korpora für die Historische Syntaxforschung: Anwendungsbeispiele anhand des Referenzkorpus Mittelhochdeutsch. *Zeitschrift für germanistische Linguistik* 43(3). 516–563.

Hartmann, Stefan. 2018. *Deutsche Sprachgeschichte. Grundzüge und Methoden.* Tübingen: Francke.

Cheat Sheet für ANNIS

Einfache Lemmasuche

"Mann"	findet Wortform *Mann*
lemma="Mann"	findet Lemma *Mann*, falls Korpus lemmatisiert ist.

Lemmasuche mit POS

"Mann" _=_ pos=/N.*/	findet alle substantivischen Instanzen von *Mann* einschl. Eigennamen wie *Thomas Mann.*

Suche nach Wortfolgen

"Mann" . "oder" . "Frau"	findet Wortfolge *Mann oder Frau* in genau dieser Reihenfolge.
"Mann" . "oder" .* "Frau"	findet *Mann oder Frau* in dieser Reihenfolge, wobei zwischen *oder* und *Frau* beliebig viele weitere Wörter stehen können.
"Mann" . "oder" .2,4 "Frau"	findet *Mann oder Frau* in dieser Reihenfolge, wobei zwischen *oder* und *Frau* mindestens zwei und maximal vier weitere Wörter stehen.
"Mann" ^3,7 "Frau"	findet *Mann* im Abstand von 3-7 Wörtern vor oder nach *Frau*.
"Mann" . pos=/V.*/	findet *Mann* unmittelbar gefolgt von einem Verb.

Suche mit regulären Ausdrücken

/.ehen/	findet z.B. *gehen* und *sehen* - . steht für irgendein Zeichen.
/.*ehen/	findet Wörter mit 0 oder mehr Zeichen vor *ehen,* z.B. *begehen*
/.+ehen/	findet Wörter mit 1 oder mehr Zeichen vor *ehen,* z.B. *begehen*
"Mann" | "Frau"	findet *Mann* ODER *Frau*

Suche nach Annotationsebenen

Die Benennung der Annotationsebenen ist **korpusspezifisch**: So heißt z.B. die Wortartenebene in manchen Korpora pos, in anderen POS (groß geschrieben). Solche Informationen können i.d.R. der Dokumentation des jeweiligen Korpus entnommen werden. Im Zweifelsfall wenden Sie einen Trick an: Suchen Sie nach irgendeinem Lemma, exportieren Sie die Ergebnisse und schauen Sie in der exportierten Textdatei nach, wie die Annotationsebene benannt ist.