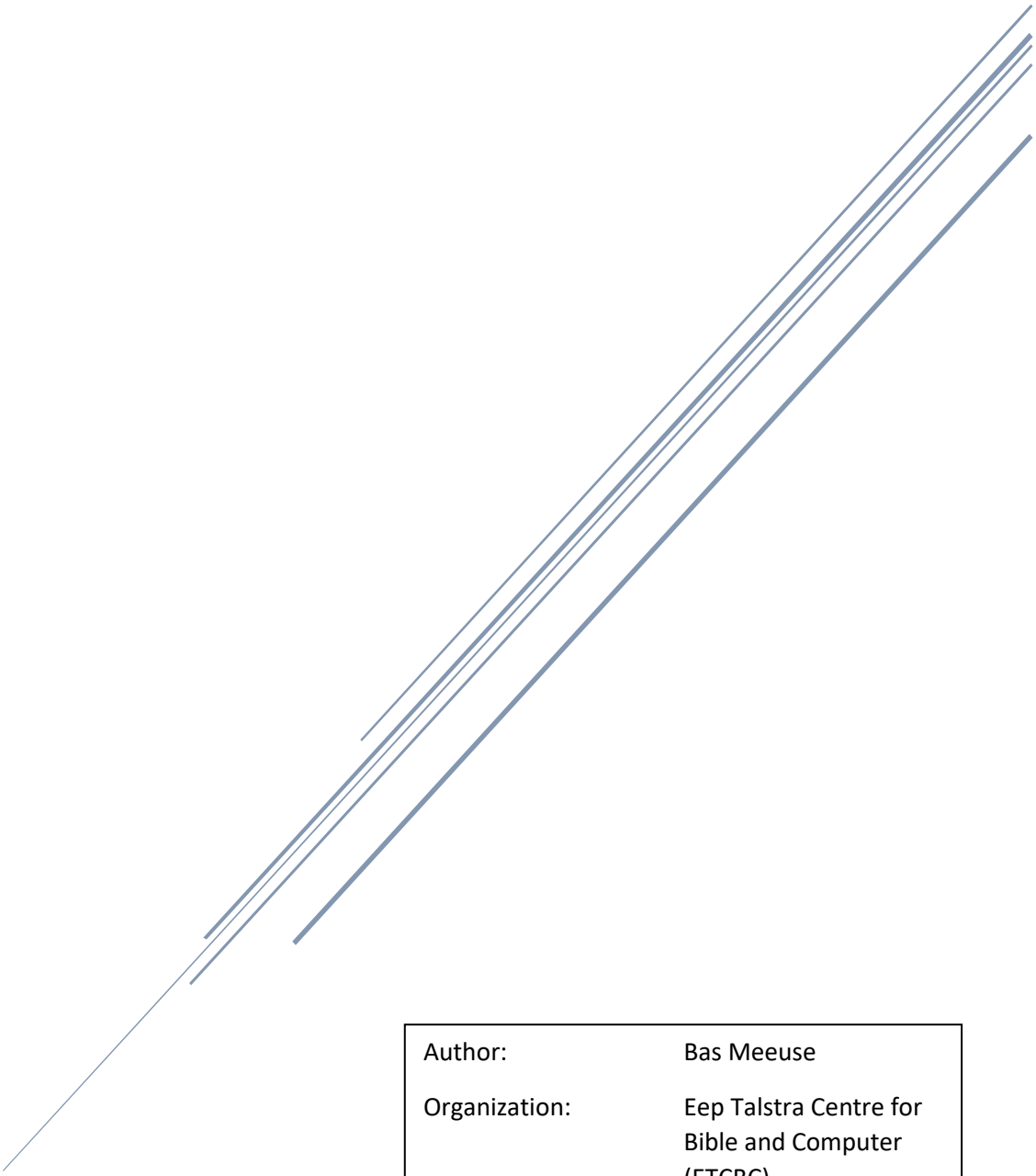
SHEBANQ TUTORIAL 2021

How to start using the BHSA database



Author:	Bas Meeuse
Organization:	Eep Talstra Centre for Bible and Computer (ETCBC)
Date of completion:	22-03-2021

Contents

1 Introduction: What is SHEBANQ?

SHEBANQ stands for 'System for HEBrew Text: Annotations for Queries and Markup'. It is an online environment for studying and doing research regarding the Hebrew Bible.[1] The database contains extensive syntactic and morphological information that can easily be searched and accessed. It is an open-source program, developed by Dirk Roorda (DANS), in conjunction with the Eep Talstra Centre for Bible and Computer (ETCBC), a research center at the Vrije Universiteit Amsterdam, Faculty of Religion & Theology. When SEHBANQ was developed, a complete representation of the ETCBC database was archived at DANS. Later, a Text-Fabric representation of the database was published on GitHub, and this became the BHSA.[2] To execute searches in SHEBANQ, you can write queries that can be shared with other researchers and can be used in academic publications. It is also possible to view earlier published queries by fellow researchers and to check whether there has been written a query on a specific text you are interested in. The following tutorial is written for those who do not have experience with working in SHEBANQ, and for those who are not familiar with the MQL language in which the queries are written. This tutorial explains the basics of query-writing and helps you to get started from the bottom. Getting good at writing queries involves trial and error and some time. However, it will improve the extent of your research, so it will be worth your while.

In this tutorial, first, the database itself will be explained. After this, a fair number of examples will be given, so you can determine if SHEBANQ can indeed help you with your research, translation, and/or exegesis. You don't have to go through all of the examples, but they can be helpful. Then, a tutorial on how to write queries will follow. This tutorial contains helpful examples to get you started, so you will be able to write basic queries after finishing this tutorial and will know how to proceed with writing more advanced queries. At the end of this tutorial, a few appendixes are added to explain additional things and to provide a shortlist of handy operators. Regular references will be made to these appendixes, both in footnotes and in the main text. When you see that this tutorial exists out of a fair number of pages, fear not, for a large part exists of images and query samples with a lot of whitespaces. Don't let the page numbers fool you, following the tutorial is quite manageable.

[1] This text is based on the homepage of https://shebanq.ancient-data.org/.
I want to thank everyone who has helped in realizing this tutorial. First, my internship mentor Wido van Peursen, who has guided me through the process. Second, I want to thank Constantijn Sikkel, who, among other things, has helped me in writing queries and checking all of the example queries. Furthermore, I want to thank everyone who has provided me with examples of translation issues in which SHEBANQ can be useful, and/or examples of queries, or has otherwise contributed to the tutorial (Dirk Roorda, Eep Talstra, Femke Siebesma-Mannings, Janet Dyk, Oliver Glanz, Reinoud Oosting).
[2] Dirk Roorda, "Coding the Hebrew Bible," *Research Data Journal for the Humanities and Social Sciences* 3/1 (2018): 27-41.

2 How is the database built up and how can you use the text?

2.1 The text

The Eep Talstra Centre for Bible and Computer (www.etcbc.nl) develops and maintains an advanced syntactic database of the Hebrew Bible. The methodology of the ETCBC prioritizes syntactic analysis over semantic, literary, or rhetorical analysis. Formal indicators, rather than functional ones, comprise the framework of the database. Linguistic information is encoded hierarchically at the word, phrase, clause, and text levels. The activities of the ETCBC include encoding new texts (data creation) and utilizing already encoded texts for linguistic research.[3]

There are different versions of the database, several of which can still be accessed through the SHEBANQ website. The most recent fixed version is BHSA 2017. Earlier versions are ETCBC3, ETCBC4, and version 4b. Version c is being updated periodically. You are advised to use the most recent fixed version, 2017.

The following YouTube tutorial made by Oliver Glanz can be useful in helping you to get to know how to use the SHEBANQ text: https://www.youtube.com/watch?v=0fc8NJaxSDo&list=PLmvR-VgmhJN1eSSI9qXPj8xovxYi-jM5N. I would advise watching it before moving on with this tutorial, so you know the basics of using the text. Glanz uses an older version in his tutorial, so be aware that the most recent fixed version is now the 2017 version, as mentioned above, not the 4b version. Most functionalities of the text, however, have not changed.

2.2 Introduction to different ways of learning how to use SHEBANQ

For starters, this tutorial document is probably enough to get started with using SHEBANQ and with writing queries. If you have follow-up questions after finishing this tutorial, you can check the other documents that are mentioned in the subsequent part of this section.

If you are seriously engaged with using SHEBANQ for your research, you can send a request to w.t.van.peursen@vu.nl to join the Slack community (etcbc-vu.slack.com). On Slack, you can ask questions about queries and other subjects related to data analyses regarding ancient texts.

For more information and a practical starting point, click on the *Information* symbol on top of your screen when on de SHEBANQ website:

This will lead you to the SHEBANQ GitHub page, which is a useful starting point for reaching different information pages on GitHub regarding SHEBANQ.

On the right side of the GitHub page, you will find a list of links to other useful GitHub pages for different purposes. In this tutorial, I will refer to several of those links. Some of these links can also help you to get started with querying on a basic level (e.g. the 'MQL quick start' guide) or on a more advanced level (e.g. the 'MQL for Programmers' guide).

[3] For more information on the database and its structure, see the articles on the following webpage, especially the article by Wido van Peursen and Cody Kingham (2018), and the article by Eep Talstra and Constantijn Sikkel (2000): http://etcbc.nl/bibliography-copy/.

Another basic PDF specifically on the MQL query language can be found by clicking on ‘MQL Query Guide’ when on the GitHub page.

By the following link, you can also find some guidance on how to start using SHEBANQ: https://github.com/ETCBC/Tutorials. Click on ‘Tutorial Shebanq.pdf’. This might be helpful, but most of the information is also present in what follows.

3 What can you use it for?

This part of the tutorial exists of 10 examples that will show you what you can do with SHEBANQ. The examples are ordered according to their level of difficulty. So the basic queries come first, followed by the more advanced queries. The examples show that you can use SHEBANQ at any level of research, whether you are a student who just wants to make an exegetical assignment or your Hebrew homework, or a skilled researcher who wants to investigate a difficult syntactical question. Explanations about the composition of the queries are included in the tutorial on writing queries (below), in this section, the focus is on their potential for solving all kinds of questions.

3.1 Example 1: Where does Moses start speaking to YHWH?

The first example we will discuss is provided by Eep Talstra in an article in JNSL,[4] about the phrase 'Moses spoke to YHWH' in Ex. 33,12, where we read:

וַיֹּאמֶר מֹשֶׁה אֶל־יְהוָה רְאֵה אַתָּה אֹמֵר אֵלַי הַעַל אֶת־הָעָם הַזֶּה

Moses said to the LORD, "See, you have said to me, 'Bring up this people'…" (NRSV)

The clause itself is not so extraordinary, as we can see when we run the following query:

```
Select all objects where

[clause FOCUS
        [word lex = '>MR[' OR lex = 'DBR['] /*Also possible is '[word ls
                                            =quot]' (lexical set = quotation verb)*/
        [phrase function = Subj
                [word lex = 'MCH=/']
        ]
        ..
        [phrase function IN (Cmpl)
                [word lex = 'JHWH/' OR lex = '>LHJM/']
        ]
]
```
[5]

This query finds all clauses that contain a word regarding speaking, followed by a phrase with a subject function (a query usually searches for the order in which it is written), containing the name *Moses*, then a possible gap to allow space between the two phrases, followed by a phrase with a complement function containing either *YHWH* or *Elohim* to indicate that Moses is indeed speaking to God.

There are eight results, that all reflect Moses speaking to YHWH. In Ex. 33,12, it might be a special case, because Moses himself starts speaking to God. In most of the other cases, Moses speaks to YHWH in reaction to a conversation started by YHWH. When we analyze the results of this query, we can see that it happens only one other time that Moses starts speaking to YHWH, in Num. 11,11. In both instances, this happens in 'the tent of meeting' outside the camp. In no other instance or place does Moses speak to YHWH without being spoken to by YHWH first. So SHEBANQ can be used to check if a certain combination of words is normal, or if something special is going on.

[4] Eep Talstra, "'I and Your People:' Syntax and Dialogue in Exod 33," *Journal of Northwest Semitic Languages* 33/2 (2007): 89–97. For this example, see pp. 91–94.

[5] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4439.

3.2 Example 2: Did Cain receive a sign, or was he made a sign (Genesis 4,15)?

Figure 1 (Ivory, from the Cathedral of Salerno ca. 1384) Louvre, Paris

The next example is taken from an article by Janet Dyk, Oliver Glanz and Reinoud Oosting.[6] The example concerns the Hebrew verb שׂים. The authors of the article discuss the case of Cain who gets a sign from God after killing Abel (Gen. 4,15). In Gen. 4,15 we read:

וַיָּשֶׂם יְהוָה לְקַיִן אוֹת לְבִלְתִּי הַכּוֹת־אֹתוֹ כָּל־מֹצְאוֹ

Then Yahweh put a sign on Cain so that whoever found him would not kill him. (LEB)

The verb שׂים is followed by the preposition ל. If this preposition is followed by another element, such as פנה (presence/face) or עין (eye), it functions as a locator, so the place where something is being put. But without these elements, it indicates for whom or what something is being prepared or put in place. So we see that in Genesis 4,15 the preposition ל is connected with Cain's name. Many translations translate as if the Hebrew contains a different preposition (על), with which שׂים can be translated as something like 'put upon'. So the sign is then 'put on' Cain. See for example the LEB above, and the NRSV:

And the LORD put a mark on Cain, so that no one who came upon him would kill him. (NRSV)

But that is not what the Hebrew tells us, it rather tells us that the sign is set in place on behalf of Cain, not where this sign is placed or what it exactly is.

With the following query, all the cases where שׂים is connected with the preposition ל are selected. When analyzing the results, the way this preposition is used can be determined and similar analyses as shown in the example can be done:

```
select all objects where

[clause
        [word FOCUS lex = 'FJM[']
```

[6] Janet Dyk, Oliver Glanz and Reinoud Oosting, "Het belang van valentiepatronen voor het vertalen van Bijbels Hebreeuwse werkwoorden," *Met Andere Woorden* 32, no. 2 (2013): 23–35. For this example, see pp. 30–32.

```
        ..
        [word FOCUS lex = "L"]
        [word lex <> '<JN/' AND lex <> 'PNH/']
]
```
[7]

This query searches for all clauses that contain first of all the word שִׂים, then a possible gap, followed by the preposition ל. After this, the words פנה and עין are excluded, so we get as few results as possible where the preposition ל functions as a locator.

This analysis leads to a different translation, something like this:

> And YHWH set a sign in place on behalf of Cain, so that no one who came upon him would kill him.

So Cain received a sign for sure, but we do not know where this sign was placed and whether or not it was put “on” him. The Hebrew does not provide us with that information, so we should not imply that in our translation.

[7] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4440.

3.3 Example 3: Israel as YHWH's people or as Moses' people (Exodus 33,16)?

Another example by Eep Talstra, taken from the same article as *Example 1*, is about the case of Moses speaking to God in Ex. 33,16.[8] The tension of the conversation between YHWH and Moses lies in the question of whose people Israel is. We read in Ex 33,16:

וּבַמֶּה יִוָּדַע אֵפוֹא כִּי־מָצָאתִי חֵן בְּעֵינֶיךָ אֲנִי וְעַמֶּךָ הֲלוֹא בְּלֶכְתְּךָ עִמָּנוּ וְנִפְלֵינוּ אֲנִי וְעַמְּךָ מִכֹּל־הָעָם אֲשֶׁר עַל־פְּנֵי הָאֲדָמָה

> And by what will it be known then that I have found favor in your eyes, I and your people? Is it not by your going with us? And so we will be distinguished, I and your people, from all the people who are on the face of the ground. (LEB)

Moses identifies himself with the people of Israel and uses a syntactically highly irregular expression to do this: 'I and your people'. Usually, the personal pronoun and the possessive pronoun are in agreement (E.g. 'I and my people'; 'you and your house'), but here they are not. We can check how often these pronouns are in agreement with the following query:

```
select all objects where

[phrase_atom FOCUS
        [word AS P sp = prps]
        ..
        [word lex = "W" OR lex = ">W"]
        ..
        [word prs !~ "a" AND prs_ps = P.ps]
]
```

[9]

This query looks for a phrase_atom[10], where a personal pronoun (called P, to recall this exact personal pronoun later on in the query), followed somewhere in the same phrase_atom by one of the coordinating conjunctions ו or או and further down a word with a pronominal suffix that agrees in person with the personal pronoun earlier in the query (then assigned P, now recalled).

The results show that indeed, very often (150 verses), both pronouns are in agreement. A query for similar cases as in Ex. 33,16 (note that only the =-operator has changed in the <>-operator)[11], where the pronouns are not in agreement with each other gives only five results:

```
select all objects where

[phrase_atom FOCUS
        [word AS P sp = prps]
        ..
        [word lex = "W" OR lex = ">W"]
        ..
        [word prs !~ "a" AND prs_ps <> P.ps]
]
```

One of the results, where we read 'I and your mother and your brothers' (Gen. 37,10), does not apply because it is a different grammatical situation (the subject 'we' is clarified by this phrase). Another result, Jer. 44,3 does not apply because the irregularity occurs in a listing of people addressed by

[8] Talstra, "Exod 33," 94–97.

[9] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4441.

[10] The notion of phrase_atom, rather than 'phrase' is used to avoid hits that cross the boundary of an apposition or specification. For more information on how the database is built up with regard to phrase_atoms, see the articles in note 2

[11] For operators, see the tutorial below and Appendix A: Operators with examples.

YHWH. Nehemiah 9,16 is also not applicable because both pronouns refer to the same group of people: the ancestors. The other two results are both in Ex. 33,16. Talstra concludes after these findings:

> Therefore, one can understand the phrase "I and your people" to be a purposefully composed phrase, part and parcel of a barely expressed but real conflict. Whose people is this, mine or YHWH's? In other words, Moses is searching for the words to express the fact that Israel is the people of YHWH.[12]

This example shows how it is possible to use SHEBANQ for your exegesis.

[12] Talstra, "Exod 33," 96.

3.4 Example 4: Is 'Barak' a later addition (Judges 5,1)?

The next example is based on an article by Ulrik Sandborg-Petersen.[13] Sandborg-Petersen mentions the case of Judges 5,1, where the well-known Song of Deborah is introduced as follows:

וַתָּשַׁר דְּבוֹרָה וּבָרָק בֶּן־אֲבִינֹעַם בַּיּוֹם הַהוּא לֵאמֹר

On that day Deborah and Barak son of Abinoam sang (f. sg.) this song: (NIV)

One thing that stood out to scholars was the feminine singular finite verb שִׁיר, which agrees very well with Deborah, but not with Barak. Does this challenge Barak's position in the two-member choir? The grammatical disagreement between the singular verb and the plural subject has been put forward as an argument that Barak is a later addition. With the following query, the database can be checked on similar structures, to check if such a grammatical disagreement is indeed so abnormal:

```
select all objects where

[clause
        [phrase FOCUS function=Pred
                [word sp=verb AND nu=sg AND gn=f]
        ]
        ..
        [phrase FOCUS function=Subj
                [word sp=conj]
        ]
]
```
[14]

This query looks for clauses in which there is first a phrase with a predicate function that contains a singular feminine verb, followed by a phrase with a subject function that contains a conjunction (to make sure that the subject is plural).

The query gives 51 results, so apparently, it is not an abnormal construction.[15] With this information, it is difficult to maintain the idea that Barak is a later addition in this text. Or you'd have to argue that every query result contains a later addition, which is unlikely. Of course, we already knew that from e.g. Gesenius-Kautzsch §146g,[16] but you can see that with simple queries, these matters can be easily checked and supported by statistical data.

[13] Ulrik Sanborg-Petersen, "On Biblical Hebrew and Computer Science: Inspiration, Models, Tools, and Cross-Fertilization", in W.Th. van Peursen and J.W. Dyk (eds.), *Tradition and Innovation in Biblical Interpretation. Studies Presented to Professor Eep Talstra on the Occasion of his Sixty-Fifth Birthday.* (Studia Semitica Neerlandica 57; Leiden: Brill, 2011), 261–276.

[14] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=53.

[15] Without the restriction that the that the verb is feminine, just looking for a singular verb and a subject containing a conjunction, the query yields more than 250 results, see Sandborg-Petersen, ibid.

[16] Friedrich Wilhelm Gesenius, *Gesenius' Hebrew Grammar*, ed. E. Kautzsch and A.E. Cowley, 2nd English ed. (Oxford: Clarendon Press, 1910), 468.

3.5 Example 5: How did Abimelech "take" Sarah (Genesis 20,2)?

Another illustration of how SHEBANQ can help you with your exegesis is an example by Oliver Glanz in an article on Bible Software.[17] He mentions a case from Genesis 20, where we read the story of Abraham and Abimelech. Glanz zooms in on verse 2, where we read:

וַיִּשְׁלַח אֲבִימֶלֶךְ מֶלֶךְ גְּרָר וַיִּקַּח אֶת־שָׂרָה

> And King Abimelech of Gerar sent and took Sarah. (NRSV)

Glanz mentions that usually, we would just read over such a verse because the verb לקח is one of the basic five hundred most well-known Hebrew words. Here, however, it occurs with a single object, and the object is a female proper name. With the following query, we can check if this is usual or if there is something extraordinary going on here:

```
select all objects where

[clause
        [phrase FOCUS function IN (Pred,PreC)
                [word lex = "LQX["]
        ]
        ..
        [phrase function = Objc
                [word FOCUS gn =  f AND nu=sg AND sp=nmpr] //this does not yet narrow the
                object down to female proper names!
        ]
][18]
```

This query finds all clauses in which a phrase has a predicate function or a predicate complement function, and within that phrase, the word לקח. This first phrase precedes a phrase with an object-function that contains a singular feminine proper noun.

When investigating the results, it becomes clear that usually לקח + Object, with a female proper name as an object, is followed by the complement 'לְאִשָּׁה' (as women/wife). But not in this case. We would expect that Sarah is taken as Abimelech's wife, but the construction is unusual. This poses all kinds of questions about the way Abimelech takes Sarah, as Glanz shows:

> Due to this observation, one would need to discuss whether the text wants to indicate that Abimelech took Sarah by treating her as if she was an object without rights. At this point, the imagination of the reader has no limits. Did Abimelech rape her? Did he make a maidservant out of her? Or did he "take" her in order to marry her? But if that was the case, why is the construction not rendered in a more straightforward way? The fact is that the text leaves it open to the reader to imagine what Abimelech intended to do with Sarah. At the end of the story, it seems clear that Abimelech married Sarah and had not touched her yet. Could it be that the text wants the reader to imitate the suspicion that Abraham had in thinking that Abimelech was not a God-fearing person? Whatever the answers are, the text contains underdetermined data that triggers exegetical questions that deserve attention.[19]

With SHEBANQ, it can be determined that the Hebrew construction in Gen. 20,2 is indeed unusual.

---

[17] Oliver Glanz, "Bible Software one the Workbench of the Biblical Scholar: Assessment and Perspective," *Andrews University Seminary Studies* 56/ 1 (2018): 22–24.

[18] https://shebanq.ancient-data.org/hebrew/query?id=494. Another query used in studying this text is: https://shebanq.ancient-data.org/hebrew/query?id=493.

[19] Glanz, "Bible Software," 23.

3.6 Example 6: A turning point in Deuteronomy 29,3[4]?

SHEBANQ can also be used to find sentences with the same grammatical structure, for instance in the following example, provided by Eep Talstra.[20] He searches for similar grammatical constructions of Deut. 29,3[4] to determine (amongst other things) whether or not there is a sudden turning point, as some translations might suggest (e.g. The Message and the Dutch NBV). In Deut. 29,3[4] we read:

וְלֹא־נָתַן יְהוָה לָכֶם לֵב לָדַעַת וְעֵינַיִם לִרְאוֹת וְאָזְנַיִם לִשְׁמֹעַ עַד הַיּוֹם הַזֶּה

> But GOD didn't give you an understanding heart or perceptive eyes or attentive ears until right now, this very day. (The Message)
>
> Maar pas vandaag heeft de HEER u werkelijk inzicht gegeven, u de ogen en oren geopend. (NBV)

Talstra wrote the following query, according to the grammatical structure of the sentence as can be found in SHEBANQ:

```
select all objects where

[clause focus
        [phrase function = Nega]
        [phrase typ = VP
                [word ps = p3]
        ]
        [phrase function = Subj]
        ..
        [phrase function = Time
                [word first lex = "<D"]
        ]
][21]
```

This query thus searches for clauses that contain a negating phrase, followed by a verbal phrase that contains a third person, followed by a phrase with a subject function, then a possible gap, and a phrase with a time reference that starts with the word עד.

The query gives nine other results, besides Deut. 29,3[4]. Not a single result indicates a turning point. Two of the results describe a custom that once began and continues to the present of the text (Gen. 32,33; 1 Sam. 5,5). Three of the results describe a special situation that has not occurred before and will stay exceptional (Ex. 10,6; Deut. 34,6; Josh. 23,9). The other four results can also not be seen as texts that indicate a sudden turning point after which the negation would be no longer valid (2 Sam. 12,10; Neh. 13,1; Lev. 19,13 and 1 Kgs. 3,2).

After analyzing the results, Talstra concludes that the turning point suggested by several translations is not valid, due to similar constructions where a turning point is impossible, and different constructions with an infinitive (for this, he used a different query) where there is almost always a turning point. The right translation should therefore be something like this:

> But Yahweh has not given to you a heart to understand, or eyes to see, or ears to hear, even to this day. (LEB)

[20] Eep Talstra, "*Deuteronomium 29:3 in de Nieuwe Bijbelvertaling,"* accessed Februari 2, 2021, https://debijbel.nl/bericht/tekst-taal-en-theologie.
[21] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4442.

3.7 Example 7: Were Solomon's cedars put on rafts (1 Kings 5,23[9])?

Figure 2 (Relief from Sargon's Palace. Sargon transporting cedar logs from Lebanon by sea ca. 716-706 BCE) Louvre, Paris. Photo by Ferrel Jenkins.

The seventh example is from an article by Janet Dyk.[22] In this article, she discusses different valence patterns of the word שׂים, before discussing the main subject of the article about Hos. 1,6. Th0e valence patterns of שׂים she mentions can be recognized with SHEBANQ. One of the examples she mentions is the case of 1 Kings 5,23[9], where Solomon gets cedars from King Hiram of Tyre and says, using שׂים:

וַאֲנִי אֲשִׂימֵם דֹּבְרוֹת בַּיָּם עַד־הַמָּקוֹם

…; I will float them in rafts by sea to the place… (NKJV)

The NKJV translates as if שׂים has one object and thus means something like 'place, put'. In this case where the logs should be placed: in rafts. שׂים with a double object means 'make something to be something else',[23] for example in Is. 50,2:

אָשִׂים נְהָרוֹת מִדְבָּר

[22] Janet Dyk, "Deportation or Forgiveness in Hosea 1.6? Verb Valence Patterns and Translation Proposals," *The Bible Translator* 65, no. 3 (2014): 235–279.
[23] Dyk, "Valence Patterns," 241–242.

…, I make the rivers a desert; … (NRSV)

Within the SHEBANQ database, it is possible to filter all the occurrences of שׂים + double object with the following query:

```
select all objects where

        [clause
                [phrase function IN (PreO, PtcO)
                        [word FOCUS vs = qal AND lex = "FJM["]
                ]
                ..
                [phrase FOCUS function = Objc]
        ]
OR
        [clause
                [phrase FOCUS function = Objc]
                ..
                [phrase function IN (PreO, PtcO)
        [word FOCUS vs = qal AND lex = "FJM["]
                ]
        ]
OR
[clause focus
        [phrase typ = VP AND NOT function IN (PreO, PtcO)
                [word vs = qal AND lex = "FJM["]
        ]
        ..
        [phrase function = Objc]
        ..
        [phrase function = Objc]
]
OR
[clause focus
        [phrase function = Objc]
        ..
        [phrase typ = VP AND NOT function IN (PreO, PtcO)
                [word vs = qal AND lex = "FJM["]
        ]
        ..
        [phrase function = Objc]
]
OR
[clause focus
        [phrase function = Objc]
        ..
        [phrase function = Objc]
        ..
        [phrase typ = VP AND NOT function IN (PreO, PtcO)
                [word vs = qal AND lex = "FJM["]
        ]
]
```
[24]

This query thus searches for five different possible clauses in which שׂים occurs with a double object. The first and the third option will be described, the other options are simply a different order of phrases with the same components. The first option is a clause that contains a phrase with a Qal of the verb שׂים, that at the same time functions either as a participle with an object suffix or a predicate with an object suffix. Then a gap, and then a phrase with an object. The second option has the reversed order of phrases. The third option is a clause that contains first a verbal phrase that does not function as a predicate with an object suffix or a participle with an object suffix. Within that first phrase, the Qal of the verb שׂים occurs. After the first phrase, two phrases with an object follow, separated by a possible gap. The fourth and fifth options have a different phrase order than the third option.

[24] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4334.

One of the results of this query is 1 Kings 5,23[9], a text that is mentioned in the beginning of this example. The construction in 1 Kings 5,23[9] contains two direct objects: 'them' (the cedars) and 'rafts'. Many older (and their younger daughter-) translations translate the second direct object as though it indicates where the cedars were to be placed, namely, *in* rafts. But the Hebrew construction tells us that the cedars are made *into* rafts.[25] Which is of course a sensible way to go about the transportation of wood. The translation should be something along the following lines:

> …, and I will make them into rafts in the sea to float to the place… (LEB)

Many modern translations do translate this verse the right way, as שׂים having two objects, which is a good thing. With SHEBANQ, you can perform comparative syntactical analysis, which can lead to translation improvements or statistical evidence to support translation choices.

[25] Dyk, "Valence Patterns," 242.

3.8 Example 8: What happens to those who walk uprightly (Psalm 84,12[11])?

The next example is provided by Reinoud Oosting in an article on Psalm 84.[26] In this article Oosting discusses several translational issues regarding Psalm 84. The example we will discuss is about the meaning of the verb מנע in combination with the preposition ל. In Psalm 84,12[11], we read:

כִּי שֶׁמֶשׁ וּמָגֵן יְהוָה אֱלֹהִים חֵן וְכָבוֹד יִתֵּן יְהוָה לֹא יִמְנַע־טוֹב לַהֹלְכִים בְּתָמִים

> For the LORD God is a sun and shield; he bestows favor and honor. No good thing does the LORD withhold from those who walk uprightly. (NRSV)

We can see that in Hebrew, the verb מנע is used with the preposition ל. In the ESV and many other translations, this construction is translated as 'withhold from'. However, elsewhere in the Bible whenever this verb has this meaning, it appears with the preposition מן (away … from). E.g. in Neh. 9,20; Prov. 3,27; 23,13. We can check this with the following query:

```
select all objects where

[clause
        [phrase function = Pred OR function = PreC
                [word FOCUS sp = verb AND vs = qal AND lex = "MN<["]
        ]
        ..
        [phrase function = Cmpl
                [word FOCUS sp = prep]
        ]
]
OR
[clause
        [phrase function = Cmpl
                [word FOCUS sp = prep]
        ]
        ..
        [phrase function = Pred OR function = PreC
                [word FOCUS sp = verb AND vs = qal AND lex = "MN<["]
        ]
]
```
[27]

This query searches for two options. Both results will be shown. The first option is a clause that has a phrase with either a predicate function or a predicate complement function, containing a Qal of the verb מנע. Then there is an optional gap, followed by a phrase with a complement function, containing a preposition. The second option is basically the same, except for the fact that the phrases have switched places.

Many lexica offer the same meaning for מנע + ל as for מנע + מן, but the only proof offered for this is Psalm 84,12. This might therefore be a circular argument. Psalm 84,12 can thus be understood differently, in line with Psalm 21,3[2] where we also read מנע and ל:

תַּאֲוַת לִבּוֹ נָתַתָּה לּוֹ וַאֲרֶשֶׁת שְׂפָתָיו בַּל־מָנַעְתָּ

[26] Reinoud Oosting, "'Zelfs de mus vindt *als* huis …'. Een taalkundige kijk op Psalm 84 in de NBV," *Met Andere Woorden* 22, no. 3 (2003): 30–37. For this example, see pp. 34–35.
[27] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=523.

In Psalm 21, however, the preposition ל is connected to the verb נתן ('to give to'). There, the verb מנע only has an object, not a preposition. So it is translated in Psalm 21,3 as follows:

> You have given him his heart's desire and have not withheld the request of his lips. (ESV)

We find the same verb נתן in Psalm 84, which means that in analogy with Psalm 21 the preposition ל can be connected to the verb נתן. The verb מנע then only has an object with it, not a preposition. Linguistically, Psalm 84,12 is thus similar to Psalm 21,3:

> The LORD gives … to // he does not withhold …

In translation then, it looks like this:

> The LORD bestows favor and honor – no good thing does he withhold – to those who walk uprightly.

In this translation, it becomes clear who the recipients of the YHWH's grace and honor are, it is those who walk blamelessly. With the positioning of this phrase at the end of this sentence, it only then becomes clear who the recipients of God's goodness are. When these considerations are incorporated in a translation, it could look like this:

> For the LORD God is a sun and shield;
> No good thing does he withhold,
> but the LORD bestows favor and honor
> to those who walk uprightly.

SHEBANQ thus can help in comparing grammatical constructions in different parts of the Hebrew Bible, which can lead to different translation choices.

3.9 Example 9: Where does the same person start speaking twice (Genesis 20,9–10)?

The ninth example is a query by Oliver Glanz, inspired by Gen. 20,9–10. In these verses, Abimelech starts speaking to Abraham without Abraham answering Abimelech in between:

וַיִּקְרָא אֲבִימֶלֶךְ לְאַבְרָהָם וַיֹּאמֶר לוֹ מֶה־עָשִׂיתָ לָּנוּ וּמֶה־חָטָאתִי לָךְ כִּי־הֵבֵאתָ עָלַי וְעַל־
מַמְלַכְתִּי חֲטָאָה גְדֹלָה מַעֲשִׂים אֲשֶׁר לֹא־יֵעָשׂוּ עָשִׂיתָ עִמָּדִי
וַיֹּאמֶר אֲבִימֶלֶךְ אֶל־אַבְרָהָם מָה רָאִיתָ כִּי עָשִׂיתָ אֶת־הַדָּבָר הַזֶּה

> Then Abimelech called Abraham, and said to him, “What have you done to us? How have I sinned against you, that you have brought such great guilt on me and my kingdom? You have done things to me that ought not to be done.” And Abimelech said to Abraham, “What were you thinking of, that you did this thing? (NRSV)

Usually a direct speech introduction in a dialogue marks a shift of speaker and addressee. This default pattern is even used without explicit mention of this shift (e.g. Gen. 22,7):

וַיֹּאמֶר יִצְחָק אֶל־אַבְרָהָם אָבִיו וַיֹּאמֶר אָבִי וַיֹּאמֶר הִנֶּנִּי בְנִי

> Isaac said to his father Abraham, “Father!” And he said, “Here I am, my son.” (NRSV)

The question posed is as follows: where do we find other cases in which the same direct speech introduction (same subject [e.g. Abimelech] speaks to the same complement [e.g. Abraham]) is repeated after the initial direct speech? Glanz wrote the following query to answer this question: https://shebanq.ancient-data.org/hebrew/query?version=2017&id=491. The query is rather long, so you can check it on SHEBANQ. The results show several places in the Hebrew Scriptures where this pattern occurs. This query is a good example of how you can perform quite specific searches with SHEBANQ for certain patterns, to determine how to deal with these constructions in your exegesis, research, or translation.

3.10 Example 10: The meaning of נתן in the Hebrew Bible

The last example discussed here is provided by Femke Siebesma-Mannens in an article published in *Dead Sea Discoveries* about the verb נתן.[28] She analyses four different patterns of this verb. With SHEBANQ, it can be determined where each case occurs. The four patterns and meanings she provides for the Qal stem are the following:

1. נתן + object	to produce
2. נתן + object + recipient	to give to
3. נתן + object + location	to place
4. נתן+ object + 2nd object	to make into

The following query searches for the first patterns, 'נתן + object'. The comments[29] that are written in the query explain what each part does:

```
/* This query selects clauses with NTN as a predicate with a single
   object and no other complement. Six cases have to be distinguished,
   because (a) the object can precede the predicate, (b) the object can
   be a pronominal suffix, and (c) the semantics of the Kleene star
   conflict with those of the 'first' keyword. */

select all objects where

// The object follows the clause-initial predicate
[clause
        [phrase first function IN (Pred, PreS)
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase FOCUS function = Objc]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]

OR

// The object follows the predicate
[clause
        [phrase first not function IN (Objc, Cmpl)]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase function IN (Pred, PreS)
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase FOCUS function = Objc]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]

OR

// The clause-initial object precedes the predicate
[clause
        [phrase FOCUS first function = Objc]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase function IN (Pred, PreS)
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]

OR

// The object precedes the predicate
[clause
```

[28] Femke Siebesma-Mannens, "Double Object Constructions in DSS Hebrew. *The Case of* ntn," *Dead Sea Discoveries* 27/3 (2020): 372–391.

[29] On comments, see Appendix A: Operators with examples, //.

```
        [phrase first not function IN (Objc, Cmpl)]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase FOCUS function = Objc]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase function IN (Pred, PreS)
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]

OR

// The clause-initial object is a pronominal suffix
[clause
        [phrase FOCUS first function = PreO
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]

OR

// The object is a pronominal suffix
[clause
        [phrase first not function IN (Objc, Cmpl)]
        [phrase not function IN (Objc, Cmpl)] *
        [phrase FOCUS function = PreO
                [word FOCUS vs = qal AND lex = "NTN["]
        ]
        NOTEXIST [phrase function IN (Objc, Cmpl)]
]
```
[30]

Similar queries can be written for patterns 2–4, for example by changing the query in such a way that a second object is no longer excluded, but rather included, or that the query searches for an object and a complement at the same time. The second and the third query would look-similar because both the recipient and the location are tagged as a complement. A possible way to distinguish the two is that the recipient starts often (but not always) with the preposition לְ or אֶל, while a location does not start that way. See for example the following query for pattern 2:

```
select all objects where

[clause focus
        [UNORDEREDGROUP
                [phrase typ = VP AND NOT function IN (PreO, PtcO)
                        [word vs = qal AND lex = "NTN["]
                ]
                [phrase function = Objc]
                [phrase function = Cmpl
                        [word first lex = "L"]
                ]
        ]
]
OR
[clause focus
        [UNORDEREDGROUP
                [phrase function IN (PreO, PtcO)
                        [word vs = qal AND lex = "NTN["]
                ]
                [phrase function = Cmpl
                        [word first lex = "L"]
                ]
        ]
]
```
[31]

This query searches for either the first clause or the second clause. Within the first clause, we find

[30] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4416.
[31] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4437.

the operator UNORDEREDGROUP, which means that the contained blocks can occur in any order.[32] So within the UNORDEREDGROUP, there has to be a verbal phrase that functions neither as a predicate object nor as a participle with an object suffix. This verbal phrase has to contain the Qal of the verb נתן. In the same clause, there also has to be a phrase that functions as an object and another phrase that functions as a complement. The first word of the complement phrase has to be ל. The second clause basically searches for the same things, with the difference that in this case, the phrase with the verb נתן has to function either as a predicate with an object suffix or as a participle with an object suffix. Furthermore, the phrase that functions as an object is left out, because in this clause there are already an object and a complement present.

The fourth case is a little bit more complicated because you would have to write out the three constituent orders. After all, UNORDEREDGROUP only works with different blocks, which is not the case when the patterns occur with two objects (twice [phrase function = Objc]). UNORDEREDGROUP would in that case not specifically search for נתן with two objects. So the fourth query would look like the query from *Example 7*, with the difference that the verb is now נתן instead of שִׂים.

With these queries, it becomes clear that quite specific searches can be done regarding the syntax of the Hebrew Bible.

[32] For more information on UNORDEREDGROUP, see Appendix A.

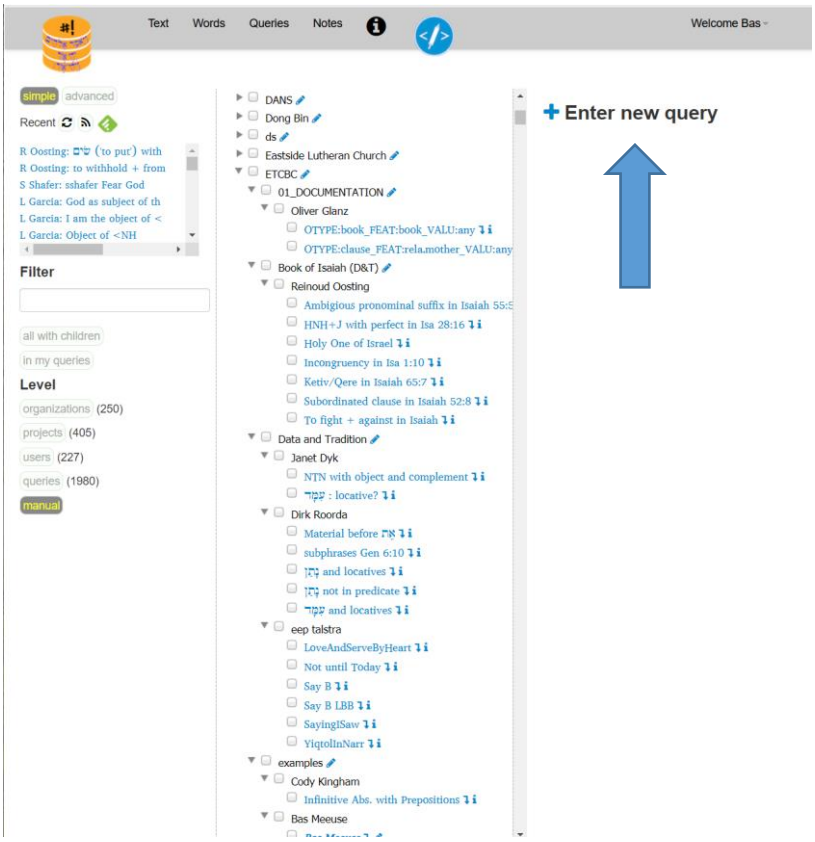
4 How to write queries

In this part of the tutorial, you will start writing queries. My advice is to follow along from the beginning and to test and experiment with the example queries. Try to alter different parts of the query to see what happens and to learn from it.

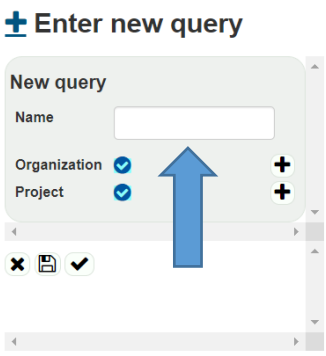
4.1 Enter a new query

First, when on the SHEBANQ website, click on *Queries.*

Click 'Enter new query' (mind that you have to create an account and be logged in to start a new query!):



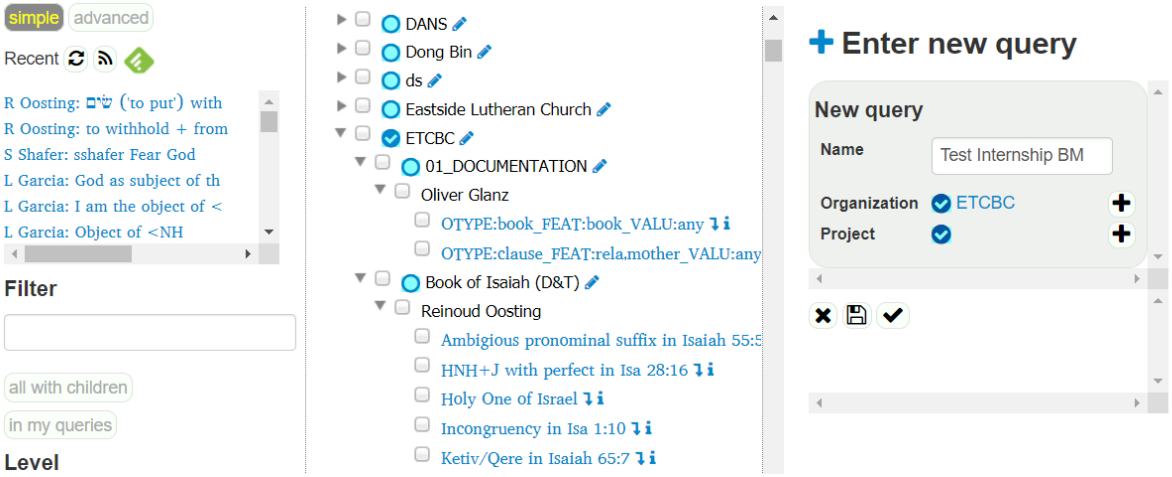
Enter the name of your query:



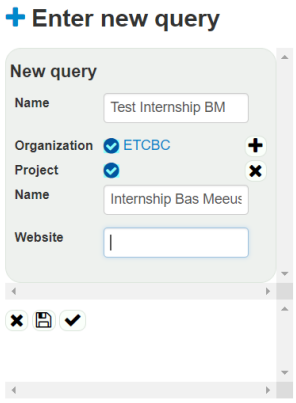
After this, choose an Organization and project by clicking on the now visible blue dots in the query list. You can also search for a project, name, or organization in the 'Filter'-box on the left side of your

screen. After filling in the search command in this box, click ‘all with children’ to search all queries. If you want to start a new organization, or if you are not part of an organization, you can click on the plus sign and enter a new one. You can fill in the name and the website of your organization and add a new organization to the query list.

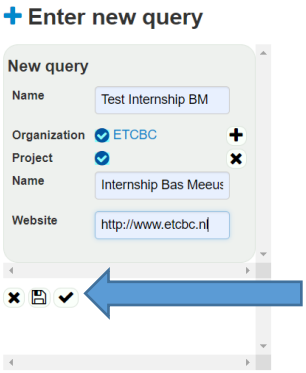
In my case, I choose the ETCBC as an organization by clicking the blue dot next to the organization name:



After choosing your organization, you can do the same for a project, or you can start a new one. In this example, I will start a new project. Click on the plus next to ‘Project’ and fill in the name of the project:

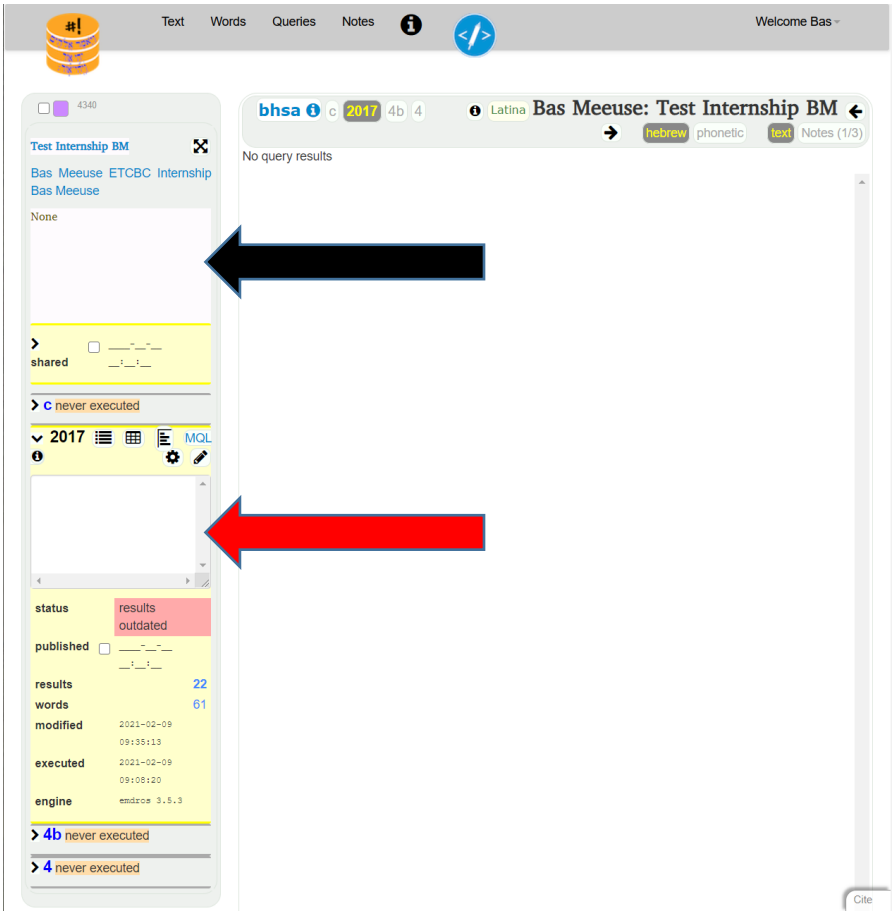


After that, fill in the name of the affiliated website (mind that it has to be the full URL, including the http://www. (or something similar), otherwise you will get an error):



Then, click on the checkmark to enter your query (and project/organization). Your queries will not be visible to others unless you publish them.[33] Your project will appear under 'projects without queries' for other people as long as you have not published any query. For yourself, it will be visible under the selected organization. You can always find your project by typing a search command in the earlier mentioned 'Filter'-box.

After creating a new query, you can click on the blue link of your query under the right project and organization, and you will see this screen:



[33] For sharing queries, see *infra,* How to cite a query.

You can write a description of your query in the box with the black arrow (for yourself, but if you publish your query, it can help others to determine if they can use your query). The box with the red arrow is the one you will be using to write your queries.

4.2 A few simple queries

In this part of the tutorial, a few very basic queries will be shown. Follow along, try to write similar queries yourself, and alter different query parts (other words, other bible books…) to get familiar with the way queries work. After the sample queries, the way objects, features, and values are used in queries will be explained.

To edit a query, click on 

To execute a query, click on 

The overruling principle of MQL is 'The structure of the query mirrors the structure of the objects found with respect to sequence and embedding'.[34]

Always start your query with:

```
select all objects where
```

Finding Gen 1:1:

```
select all objects where

[verse FOCUS book = Genesis AND chapter IN (1) AND verse IN (1)]
```
[35]

Finding attributive clauses:

```
select all objects where

        [clause FOCUS rela IN (Attr)]
```
[36]

Finding the word "Abraham":

```
select all objects where

[word FOCUS lex_utf8 = "אברהם"]
```
[37]

Finding "Abraham" as subject of an attributive clause in Genesis chapter 25–35:

```
select all objects where

[verse book = Genesis AND chapter >=25 AND chapter <=35
        [clause FOCUS rela IN (Attr)
                [phrase function = Subj
                        [word FOCUS lex = ">BRHM/"]
                ]
        ]
]
```
[38]

Finding all occurrences of the root ברא (create):

```
select all objects where

[clause
        [word focus lex_utf8 = 'ברא']
]
```
[39]

[34] MQL Query Guide, 5. On how to get to this guide, see Introduction to different ways of learning how to use SHEBANQ.
[35] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4396.
[36] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4397.
[37] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4398.
[38] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4399.
[39] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4400.

It is also possible to enter the Hebrew words in your query in transcription with Latin script. To do this, you can go to this link: https://annotation.github.io/text-fabric/tf/writing/hebrew.html. Be aware that to write Hebrew words in transcription, you have to add [in case of a verb and / in case of a noun. So for example: 'BRK[' or 'JHWH/'.[40]

The basic outline of a query can be found on this link: https://github.com/ETCBC/shebanq/wiki/MQL-Quickref (also accessible via the GitHub page by clicking on 'MQL quick start') and in Appendix C: Query template.

Now, let's move to a query that is a little bit more advanced because simple word searches are not really what SHEBANQ is designed for.

Suppose that you want to find out in which cases JHWH or Elohim is the object of ברך. This may be important to achieve consistency in Bible translations, where in, for example English, a choice has to be made whether it should be translated with "to bless" or another verb (e.g. Ps. 34:1 אֶת־יְהוָה אֲבָרְכָה NRSV: "I will bless the Lord", but NIV" I will extol the Lord"). In that case, you will have to write a query that contains within a clause both the word ברך and an object containing JHWH or Elohim. This can be achieved by the following query:

```
select all objects where

[clause
        [UNORDEREDGROUP
                [phrase typ = VP
                        [word FOCUS lex_utf8 = 'ברך']
                ]
                [phrase FOCUS function = Objc
                        [word FOCUS lex_utf8 = 'יהוה' OR lex_utf8 = 'אלהים']
                ]
        ]
][41]
```

This query searches within a clause for the word ברך, with either JHWH or Elohim as the object. 'UNORDEREDGROUP' is used in order not to have to write all possible combinations into the query. Mind that when using 'UNORDEREDGROUP' the query does not pay attention to the order in which the parts of the query occur.[42] So when the order of the elements is important for your search, do not use this construct, unless you want to go through all your search results by hand. With few results, this might very well be doable, but definitely not with every search.

[40] For more information on Disambiguation in case of multiple lexical entries, cf. *infra,* Appendix B: Disambiguation.
[41] Cf. https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4168.
[42] Cf. *infra,* Appendix A, UNORDEREDGROUP.

4.3 How to use features

At this moment you might be wondering: but how can I find what sort of features and functions I can use to build such a query? You can find an overview of all the possible operators in the 'MQL Query Guide' on the GitHub page you get to when clicking on the *Information* symbol.[43] An overview of the features and functions can be found on another GitHub page: https://etcbc.github.io/bhsa/features/0_home/#introduction. This GitHub page is also accessible via the following button that can be found on top of any text page:

Most frequently used objects, features, and values:

Object types	Feature	Value
Text section		
verse	Book	Genesis, etc. (Latin names)
	Chapter	1, 2, etc.
	Verse	1, 2 etc.
Syntax		
sentence	Sentence	
clause	Typ	WayX, Way0, etc.
	Rela	Attr, Objc, etc.
	Domain	N, Q, etc.
phrase	Typ	VP, NP, etc.
	Function	Pred, Subj, Objc, etc.
Lexeme/morphology		
word	Lex	">BRHM/"
	lex_utf8	"אברהם"
	Gn	m, f
	Nu	sg, pl, du
	Ps	p1, p2, p3
	St	a, c
	Vs	qal, nif, etc.
	Vt	perf, impf, etc.

When you compose a query, you always start each object block with an open bracket '['. Then you can write the object type, an optional FOCUS (each query has to contain at least one FOCUS, without FOCUS, no results will be displayed), a feature connected with a relational operator (e.g. '=' or '<>') to a value (e.g. 'vs = qal'). Then you close the object block with a closing bracket ']'. You can write multiple object blocks within each other. A larger unit always has to contain a smaller unit. So the object type 'book' can contain everything smaller, such as 'clause' or 'word'. When a larger object type contains a smaller object type, you close the larger object block in such a way that it contains the smaller object block. For example:

```
select all objects where

[A
        [B]
]
```

[43] See also Appendix A for a short overview of frequently used operators with examples.

Let's say that you want to write a query that searches for words with the verbal stem pi'el. You then go to the GitHub page mentioned above, go to morphology and click on 'vs' (verbal stem). Then you can check which code you can use for any object. In this case, the object type is 'word', so you start an object block, write the object type (followed by FOCUS to highlight the results if necessary) and then the feature 'vs' and the value 'piel':

```
select all objects where

[word FOCUS vs = piel][44]
```

Mind that the order within the object block is of crucial importance. You *always* have to start with the object type (followed by a FOCUS), then the feature, connected with an operator to the value.

Another simple example is when you want to find all phrases that function as a predicate. You then go to 'Phrase(-atom) features' and click on 'function'. In this case, the object type is 'phrase' and the feature is 'function'. The values of 'function' are listed, so you can see that the code for 'predicate' is 'Pred'. Now you can write your query:

```
select all objects where

[phrase FOCUS function = Pred][45]
```

With this basic information, you can write whatever query you want. Keep in mind that you always have to work from 'big' to 'small'. So for example, a word is contained in a phrase, is contained in a clause, is contained in a chapter, is contained in a book. The overarching structures have to contain the smaller structures. So for example a word contained in a phrase with the function 'predicate' contained in a clause:

```
select all objects where

[Clause                                          //open object type 'clause'
        [phrase FOCUS function = Pred            //open object type 'phrase'
                [word FOCUS lex = 'BR>[']        //open and close object type 'word'
        ]                                        //close object type 'phrase'
][46]                                            //close object type 'clause'
```

When you write queries in SHEBANQ, you can make this visible by using spaces, so you won't miss closing an object block and have insight into the way your query is built.

4.4 How to cite a query

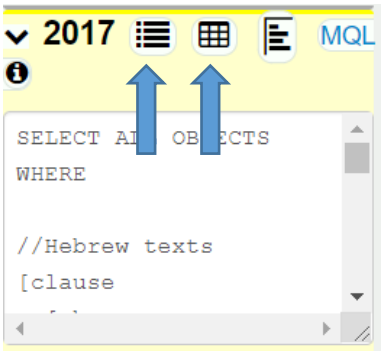
For citing queries, see the following YouTube tutorial by Oliver Glanz: https://bit.ly/3d6wqpV. The webpage he uses is outdated, but the way to cite has stayed the same. You can also check out this GitHub page: https://github.com/ETCBC/shebanq/wiki/Share-Publish.

[44] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4401.
[45] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4402.
[46] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4403.

5 Opening CSV query results in Microsoft Excel

It is also possible to export your query results as a CSV file to Excel.[47]

When you are on a query result page (your own or someone else's), you can see the following buttons:



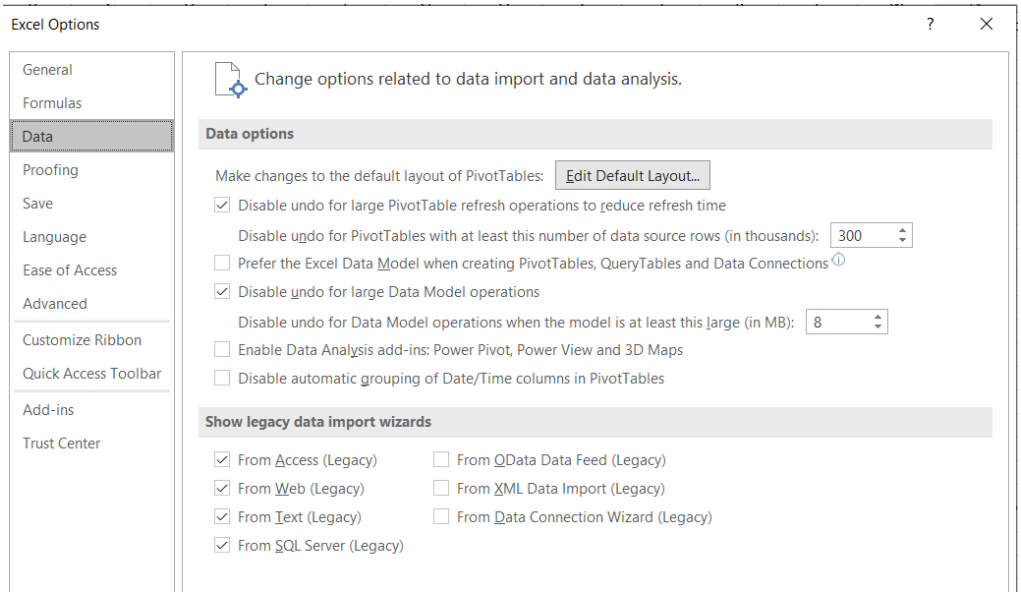
The first export button gives you a standard set of columns for your data.

The second button is a real power button because it allows you to export the current *legend* settings to a CSV file you can use for analysis.[48] Mind that the focused results will be exported, so if you want to export whole verses or clauses, make sure that they are fully focused.[49]

Click one of the buttons to download the CSV-file to your computer.

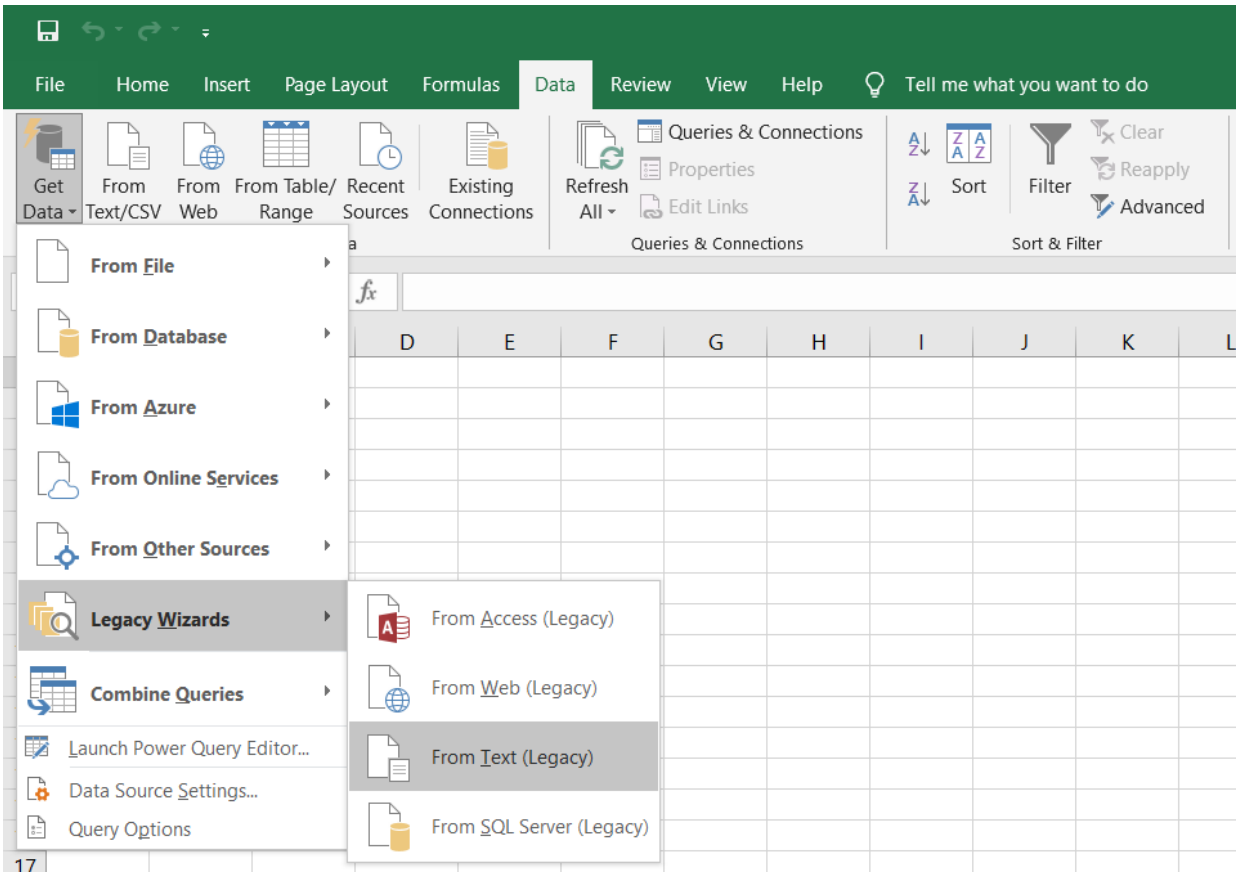
In the following example, I am using a query of Reinoud Oosting.[50]

Open an empty Excel document. Go to the 'Options' menu under 'Files' in the toolbar. When opened, click on 'Data' and select 'From Text (Legacy)':

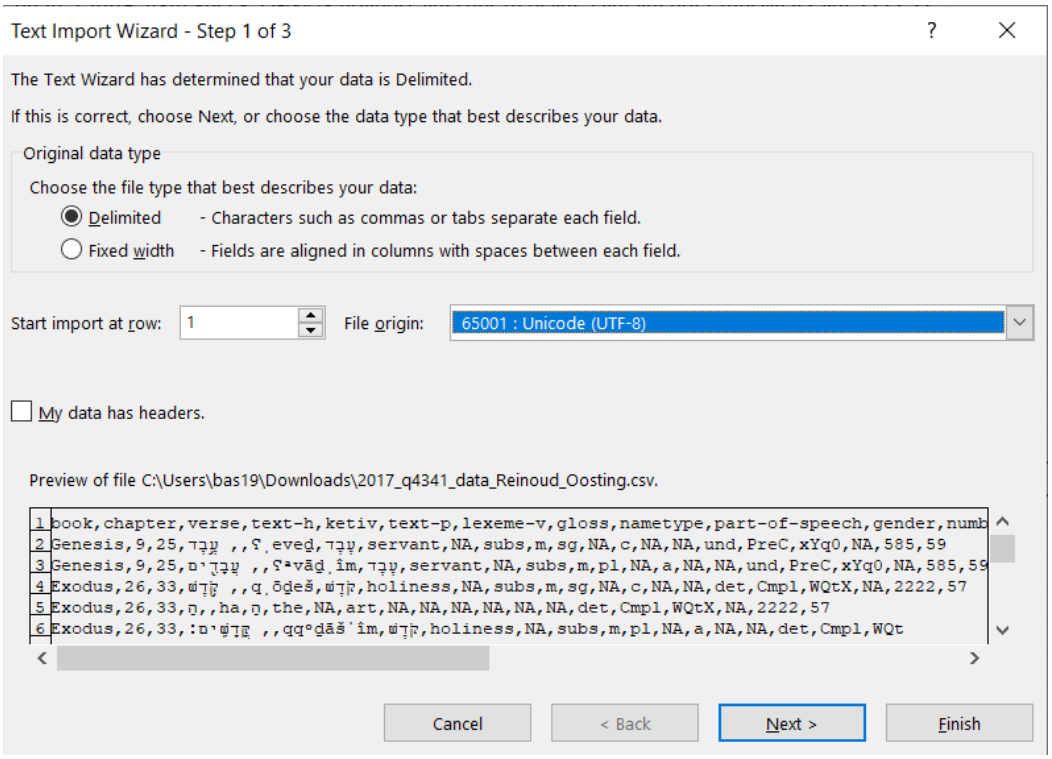


[47] See also https://github.com/ETCBC/shebanq/wiki/Csv-Export.
[48] On how to use the *legend* setting, see the YouTube Tutorial of Glanz under How is the database built up and how can you use the text?, from minute 5:45 and further.
[49] For FOCUS, cf. *Infra,* Appendix A: Operators with examples.
[50] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4341

When you've done this once, you can skip this step the next time you export a CSV-file to Excel. Next, you click 'Data' in the toolbar and go to 'Get Data'. Then, click 'Legacy Wizards' and click 'From Text':

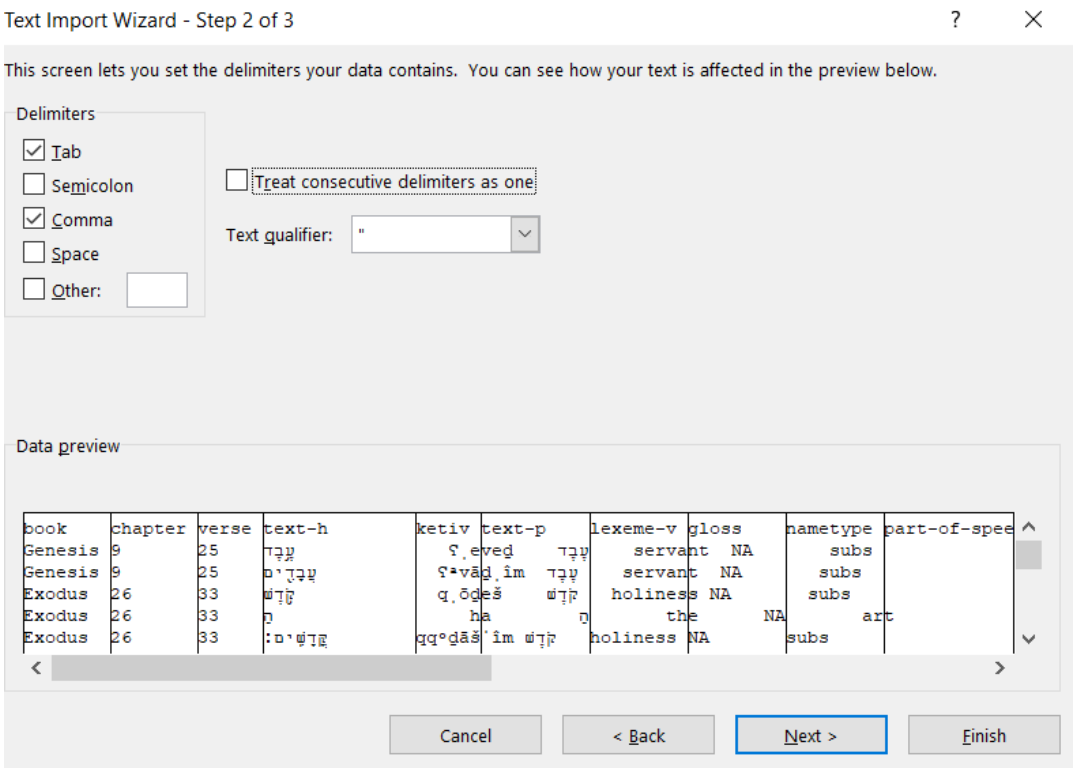


Then, search for the right folder (the downloaded file from SHEBANQ) and click 'import'. The following wizard shows up:

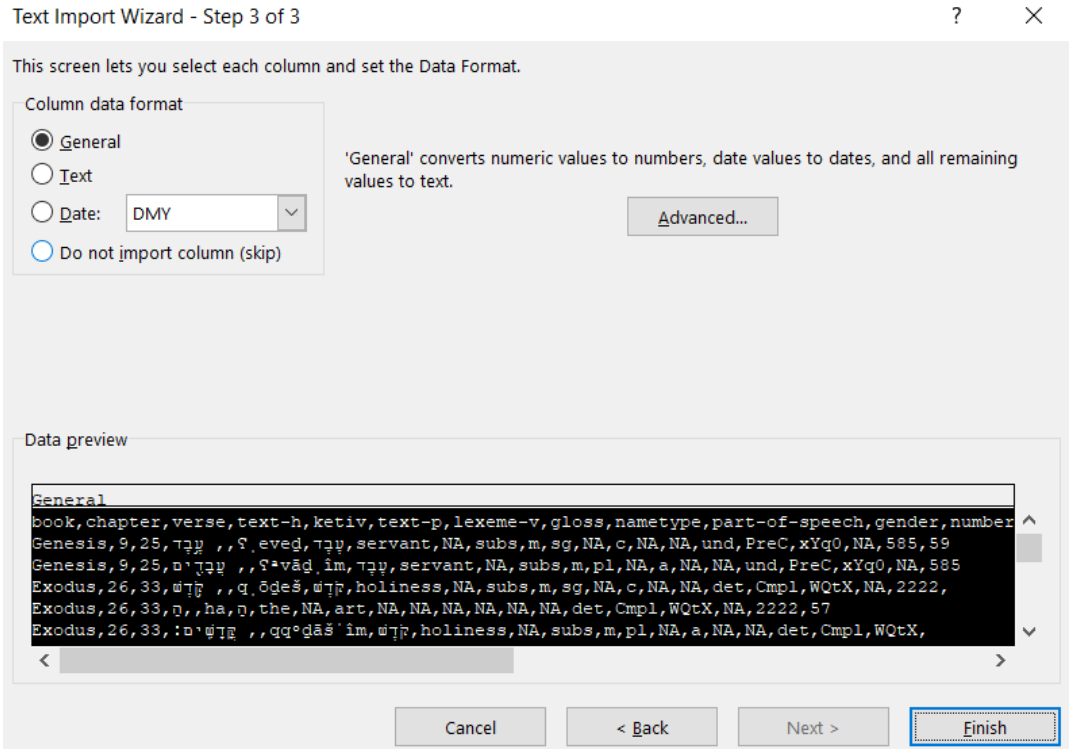


Select ‘Unicode (UTF-8)’ in the selected box, and you will see that Hebrew text appears on the preview screen.

Then you click ‘next’ and select ‘Tab’ and ‘Comma’ (or other things if you want a different layout of your Excel sheet):



Next again and then select ‘General’:



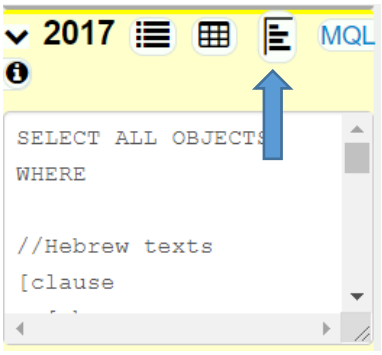
After that, click ‘Finish’, and open in the current sheet and you will have perfectly readable Excel data:

	book	chapter	verse	text-h	ketiv	text-p	lexeme-v	gloss	nametype	part-of-speech	gender	number	person	state	tense	verbal_stem
2	Genesis	9	25	עֶבֶד		ʕˌeveḏ	עֶבֶד	servant	NA	subs	m	sg	NA	c	NA	NA
3	Genesis	9	25	עֲבָדִים		ʕᵃvāḏˌîm	עֶבֶד	servant	NA	subs	m	pl	NA	a	NA	NA
4	Exodus	26	33	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
5	Exodus	26	33	הַ		ha	הַ	the	NA	art	NA	NA	NA	NA	NA	NA
6	Exodus	26	33	הַקֳּדָשִׁים׃		qqºḏāšˈîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
7	Exodus	26	34	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
8	Exodus	26	34	הַ		ha	הַ	the	NA	art	NA	NA	NA	NA	NA	NA
9	Exodus	26	34	הַקֳּדָשִׁים׃		qqºḏāšˈîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
10	Exodus	29	37	קֹדֶשׁ		qˈōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
11	Exodus	29	37	קָדְשִׁים		qˈoḏāšˈîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
12	Exodus	30	10	קֹדֶשׁ־		qˈōḏeš-	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
13	Exodus	30	10	קָדָשִׁים		qˈoḏāšˌîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
14	Exodus	30	29	קֹדֶשׁ		qˈōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
15	Exodus	30	29	קָדָשִׁים		qˈoḏāšˈîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
16	Exodus	30	36	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
17	Exodus	30	36	קָדָשִׁים		qˈoḏāšˌîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
18	Exodus	40	10	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
19	Exodus	40	10	קָדָשִׁים׃		qˈoḏāšˈîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
20	Leviticus	2	3	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
21	Leviticus	2	3	קָדָשִׁים		qˈoḏāšˌîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
22	Leviticus	2	10	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
23	Leviticus	2	10	קָדָשִׁים		qˈoḏāšˌîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
24	Leviticus	6	10	קֹדֶשׁ		qˈōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	c	NA	NA
25	Leviticus	6	10	קָדָשִׁיט֙		qˈoḏāšîm	קֹדֶשׁ	holiness	NA	subs	m	pl	NA	a	NA	NA
26	Leviticus	6	18	קֹדֶשׁ		qˌōḏeš	קֹדֶשׁ	holiness	NA	subs	m	sg	NA	a	NA	NA

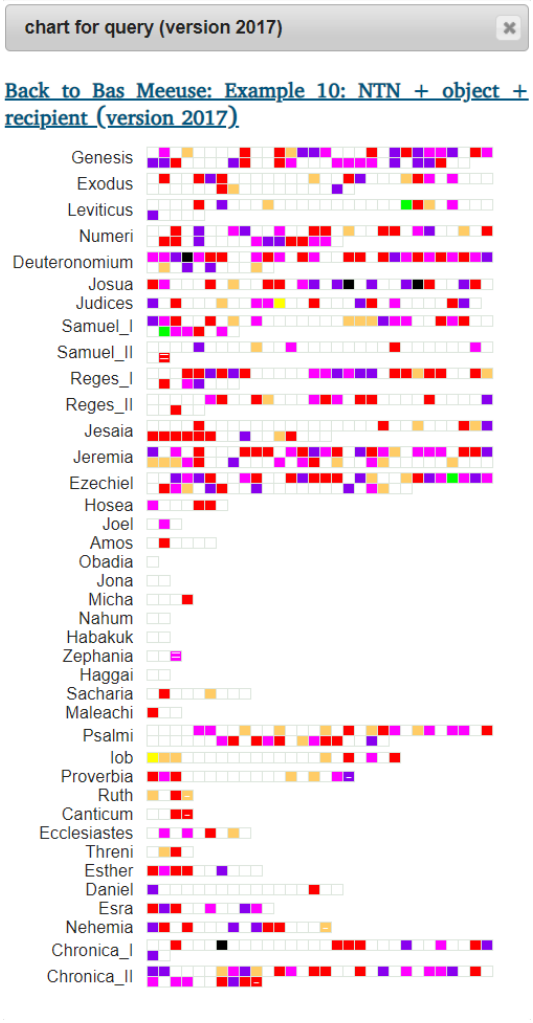
In the future, it might be possible to export Excel files directly from SHEBANQ, but for now, this is the easiest way to do it.

6 Heatmap

Next to the button for exporting query results in CSV files, there is a button that shows you a heatmap of your results:



When you click on it, you can see the heatmap:



The heatmap shows you where your results occur. You can click on each block and it will direct you to the SHEBANQ chapter where that result occurs. The darker the color, the more results in that part of the Hebrew Bible.

7 Text Fabric

Now that you have learned the basics of query writing, you might be interested in more powerful data processing tools, such as Text Fabric. A tutorial for Text Fabric can be found here.

The following Jupiter Notebook goes through all ten examples presented at the beginning of this tutorial, but in Text Fabric, to check whether the results match: https://nbviewer.jupyter.org/github/ETCBC/bhsa/blob/master/primer/tfVersusMql.ipynb.

The Jupiter Notebook in the following link can make you aware of how your queries in SHEBANQ may find other results than intended: https://nbviewer.jupyter.org/github/annotation/tutorials/blob/master/bhsa/searchFromMQL.ipynb.

When you test a hypothesis utilizing a query, you should always ask the question: did I write the right query? When you are in doubt, Text-Fabric gives you the tool to examine cases much more scrupulously than SHEBANQ.

Appendix A: Operators with examples

In this appendix, I will list a few of the most important operators you can use in building queries and give an example with each operator.[51]

AND

Is used to connect conditions that need to be true simultaneously. So the following query looks for words that are verbs as a part of speech, and which verbal tense is imperfect, and which are in the first person, and which are in the plural. So all these things need to be true at the same time to appear in the result list.

```
select all objects where

[word FOCUS sp=verb AND vt=impf AND ps=p1 AND nu=pl][52]
```

AS

Is used to name an object to the features that you want to refer to later in the query. Directly after AS there cannot be an AND operator. The AS keyword must appear right after the object type name ('word' in this case). After AS, you write the name you want to assign to the object. A reference between two objects with AS cannot cross an OR operator. It always has to be within the same string of blocks. The following query looks for two words of the same lexeme in two different clauses within the context of a verse. The first word should be a participle (active or passive) and the second word should be an imperfect form:

```
select all objects where

[verse
                [clause
                        [word AS samelex FOCUS vt IN (ptca,ptcp)]
                ]
                [clause
                        [word FOCUS lex=samelex.lex AND vt IN (impf)]
                ]
][53]
```

FIRST/LAST

Indicates that an object should be the first/last in its enclosing block. For example at the beginning of a clause or the end of a clause/phrase/verse/chapter/book/. The following query looks for instances where Sarah occurs as the last word of the clause (1 case), while Abraham occurs earlier in the clause:

```
select all objects where

[clause
        [word lex_utf8 = "אברהם"]
        ..
        [word FOCUS LAST lex = "FRH=/"]
][54]
```

FOCUS

Makes sure that the elements that you FOCUS are highlighted in your search results. Every query needs at least one FOCUS, without FOCUS, the query will work, but no results are displayed. FOCUS always needs to be written directly after the object

[51] For more information on different operators, check the *MQL Query Guide*, available on the *Information* GitHub page. Many examples are based upon various YouTube tutorials by Oliver Glanz about SHEBANQ. Descriptions are often based on the MQL Query Guide.

[52] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4384.

[53] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4385.

[54] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4386.

name, except with an AS operator. Then it comes right after that. So in the following query, 'Abraham' is focused:

```
select all objects where

[word FOCUS lex_utf8 = "אברהם"]
```
[55]

IN

Is used to indicate that all values within the brackets '()' should be included in the search. So it is used for including feature values and replaces the =-sign. A comma is used to separate values. The following query looks for both first- and second person imperfect verbs:

```
select all objects where

[word FOCUS sp=verb AND vt=impf AND ps IN (p1,p2)]
```
[56]

NOTEXIST

Is used for excluding complete blocks. The following query looks for all chapters in the book of Genesis where the word JHWH does not occur:

```
select all objects where

[chapter FOCUS book = Genesis
        NOTEXIST [word lex = "JHWH/"]
]
```
[57]

OR

Is used to find either this or that. Both results are listed in the results. There are two OR's. One between block strings, as in the following example. The query looks for the same results as the example given with the AND-operator, except that it also looks for cases in which the word is singular instead of plural:

```
select all objects where

[word FOCUS sp=verb AND vt=impf AND ps=p1 AND nu=pl]
OR
[word FOCUS sp=verb AND vt=impf AND ps=p1 AND nu=sg]
```
[58]

The other OR can be used in Boolean expressions, like the AND operator. So for instance, [ps IN (p1,p2)] could also be expressed as [(ps = p1 OR ps = p2)].

UNORDERED-GROUP

Is used in order not to have to write all possible combinations into the query. When using UNORDEREDGROUP the query does not pay attention to the order in which the parts of the query occur. Within UNORDEREDGROUP, you can only use 'bare' object blocks, no '..' or OR operators, and also no Kleene stars. UNORDEREDGROUP always has to contain a string of blocks. Mind that UNORDEREDGROUP is a relatively new and therefore still experimental addition to MQL, so it can only be used for simple queries. The following query looks for instances where both Abraham and an object phrase occur in the same clause, regardless of the order:

```
select all objects where

[clause
```

[55] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4387.
[56] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4388.
[57] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4389.
[58] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4390.

```
        [UNORDEREDGROUP
                [phrase FOCUS function = Objc]
                [word FOCUS lex = '>BRHM/']
        ]
]
```
[59]

= This operator tests for equality of the feature to the value. Variations are '<=' (smaller or equal) or '>=' (larger or equal), these variations can be used in the case of e.g. chapter/verse numbers. For examples of = and variations, see all queries.

<> Is the opposite of =. The following query looks for all imperfect forms which are not a 3rd person (so a 1st or 2nd person):

```
select all objects where

[word FOCUS sp=verb AND vt=impf AND ps<>p3]
```
[60]

~ Matches a regular expression. In the following example, all lexemes wherein the string MLK occurs, are found:

```
select all objects where

[word FOCUS lex ~ "MLK"]
```
[61]

The opposite can be expressed by '!~'. So in the following query, all lexemes without the string MLK are found:

```
select all objects where

[word FOCUS lex !~ "MLK"]
```

For more examples of this operator, see Example 10 and Appendix C: Query Templates.

// These signs are used to switch a part of your query off or to add comments inside your query which are not part of the query. The sign extends to the end of the line. So each new line you want to exclude needs to have new signs. If you want to exclude multiple lines at once, you use /* to start and */ to end.

.. Is used to indicate a possibly infinite distance between two parts of the query. So in the following query, you will find no results, because in no instance the word Abraham directly precedes Sarah:

```
select all objects where

[word FOCUS lex_utf8 = "אברהם"]
[word FOCUS lex = "FRH=/"]
```

But if you add '..', then you make sure that these two words do not necessarily need to be directly next to each other. Make sure that you always use a container when using '..'. If you search for Aaron and Moses without container Exodus 4:16 (Aaron) together with 2 Chron 36:6 (Moses) will constitute one of the exponentially high

[59] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4391.
[60] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4392.
[61] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4438.

number of results. Such an enclosing block is also necessary to prevent these ill-conditioned queries from consuming too much computer power. So in the following query, the container 'verse' is used. This query looks for all instances within the context of a verse where Abraham precedes Sarah:

```
select all objects where

[verse
        [word FOCUS lex_utf8 = "אברהם"]
        ..
        [word FOCUS lex = "FRH=/"]
]
```
[62]

* The Kleene star is a repetition operator. It means 'find me zero or more like this', and it applies to the preceding block. Without restrictions, it means 'zero or more'; with '{}' you can set explicit bounds to the number of repetitions of the preceding block ([word] in case of the first example). So the following query searches for instances where Abraham precedes Sarah with a minimum of 1 and a maximum of 10 words between them:

```
select all objects where

[word FOCUS lex_utf8 = "אברהם"]
[word] *{1-10}
[word FOCUS lex = "FRH=/"]
```
[63]

The next example is part of the query shown in 'Example 10'. The Kleene star in this example applies to `[phrase not function IN (Objc, Cmpl)]` and searches for zero or more cases in which the phrase does not function as an object or complement at that particular part of the clause. This makes sure that there will be no noise in the query results regarding objects and complements which should not be in a particular part of the clause.

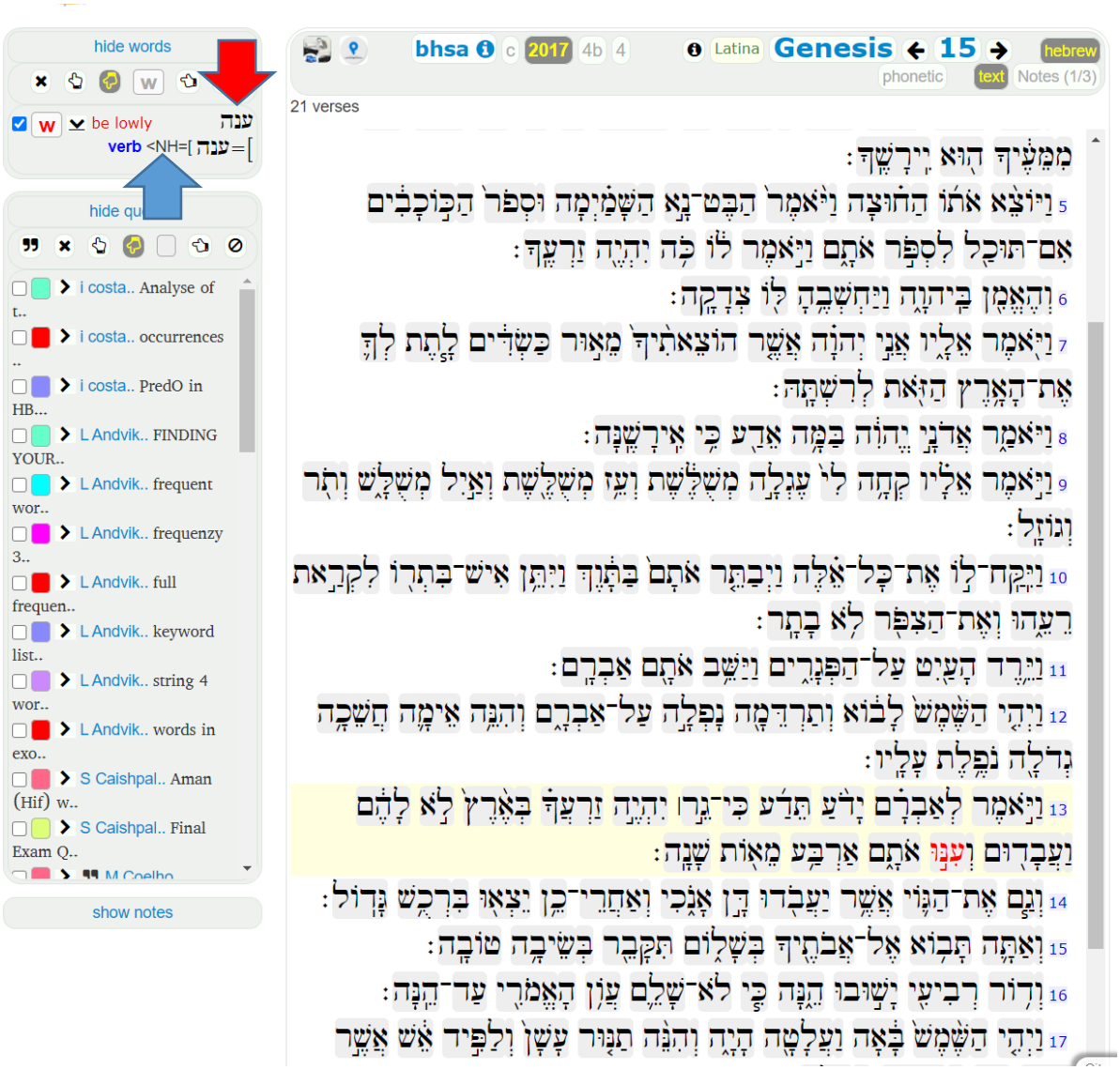
```
select all objects where

// The object follows the predicate
[clause
   [phrase first not function IN (Objc, Cmpl)]
   [phrase not function IN (Objc, Cmpl)] *
   [phrase function IN (Pred, PreS)
      [word lex = "NTN[" AND vs = qal]
   ]
   [phrase not function IN (Objc, Cmpl)] *
   [phrase FOCUS function = Objc]
   NOTEXIST [phrase function IN (Objc, Cmpl)]
]
```
[64]

[62] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4393.
[63] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4394.
[64] https://shebanq.ancient-data.org/hebrew/query?version=2017&id=4395.

Appendix B: Disambiguation

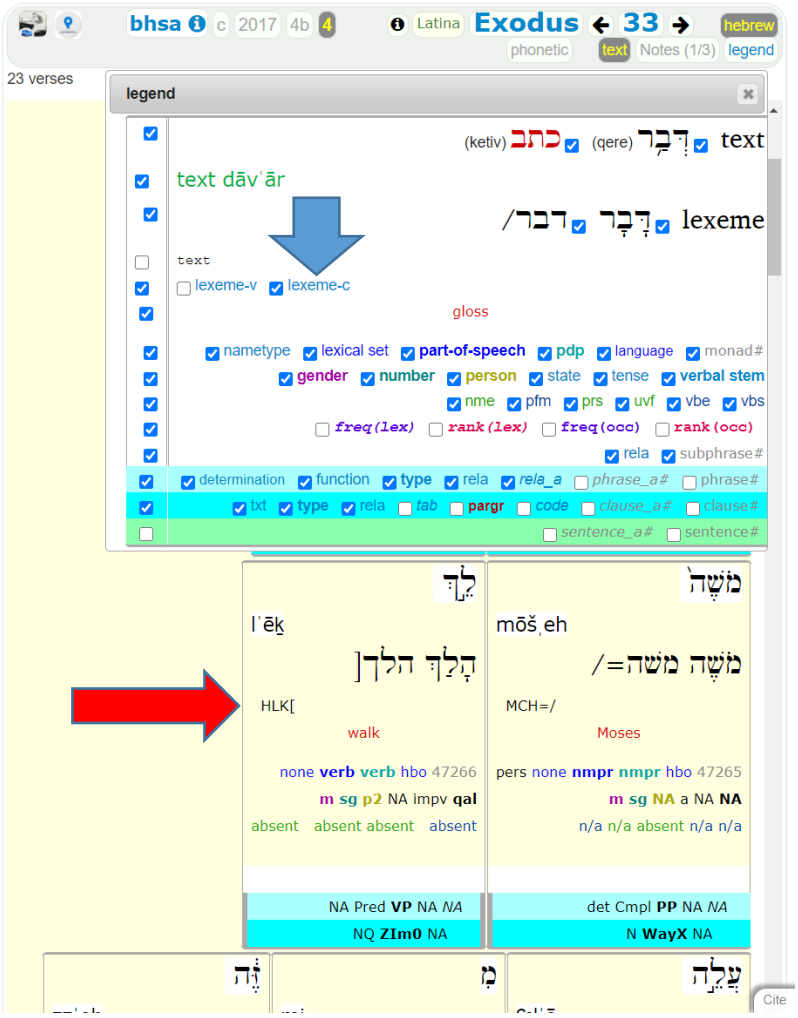
For some common words, there are more lexical entries with the same consonants. This can easily lead to queries that produce too many results or just have a bunch of wrong results mixed in with the right results. One must be aware of this. There are several ways to prevent this from happening, but not with Hebrew input. With the feature 'lex_utf8', it is not possible to enter a disambiguated form in a query (even though the database provides a Hebrew disambiguated form). So when you enter a query with a Hebrew word that has multiple lexical entries (e.g. a verb and a noun), you might get a bunch of misplaced results. However, with transliteration, it is possible to distinguish different lexical entries with the same consonants. In that case, you use the 'lex' feature and the transliterated disambiguated form. You can find the disambiguated form in the SHEBANQ lexicon or by clicking on a Hebrew word in the text. For example in the following image you can see that the disambiguated form of ענה, *be lowly*, '<NH=[' is (blue arrow):



You can simply copy-paste this form in your query. It is also possible to click on the Hebrew root by the red arrow, which will lead you to the following screen in which you can also see the disambiguated transliterated form:

As said before, it is also possible to find the disambiguated form simply via the lexicon (which can be entered by clicking 'to lexicon' on the screen you see left, or by clicking 'words' on top of the SHEBANQ webpage). When clicking on a word in the lexicon, you will see a similar information box for that word as shown on the left.

Yet another way to find the disambiguated form is to toggle on the 'lexeme-c' (blue arrow) *legend* setting:[65]



The disambiguated form (red arrow) then appears in the text information. You can simply copy-paste it into your query.

[65] On how to use the *legend* setting, see the YouTube Tutorial of Oliver Glanz under How is the database built up and how can you use the text?, from minute 5:45 and further.

Appendix C: Query templates

Standard query template[66]

The general shape of a query is as follows (the line numbers are not part of the query):

```
1     select all objects where
2       [sentence
3               [phrase
4                       [word sp=verb]
5                       [word sp=subs]
6               ]
7               ..
8               [phrase
9                       [word g_cons = 'H']
10                      [word focus]
11              ]
12      ]
```

Template explanation:

1. All queries start this way.
2. Look for a **sentence** ...
3. with a **phrase** inside ...
4. which contains a **word** having the feature **sp** (=part of speech[67]) with the value **verb** .
5. directly followed by a **word** having the feature **sp** (=part of speech) with the value **subs** (=noun).
6. and no more constraints on this phrase ...
7. followed by any material within the same sentence ...
8. followed by another **phrase** ...
9. which contains a **word** having value **H** for feature **g_cons** ...
10. directly followed by another **word**, and we say **focus** to require that this word will be highlighted in the results ...
11. and no more constraints for this phrase ...
12. and no more constraints for this sentence.

[66] https://github.com/ETCBC/shebanq/wiki/MQL-Quickref, slightly adapted.
[67] Features and values can be found in the feature documentation (see above, section 4.3) https://etcbc.github.io/bhsa/features/0_home/#introduction.

Advanced example: Verbal morphology

The query template for complex verbal morphology looks as follows:

```
select all objects where

//section01: part of speech
       [word FOCUS sp = verb
              //section02: preformative
                     AND g_pfm ~ "\![J][.][A]\!"
              //section03: lexeme
                     AND  lex ~ "^[>BGDHWZXVJKLMNS<PYQRCT]{3}=*\[" AND lex !~ "^.[WJ]"
              //section04: verbal ending
                     AND g_vbe = "[W."
              //section05a: verbal stem
                     AND vs IN (qal,nif,piel,pual,hit,hif,hof)
                     //section05b
                            //AND vbs = "HT" //alternatively one could use vbs
("absent"=qal,piel,pual; "HT"=hit; "N"=nif, "H"=hif,hof)
                     //section05c
                            //AND g_vbs ~ "\][N][IE]\]" (this would for example search for
all cases in which we would find the stem signs for the niphal ] and ]
              //section06: pronominal suffixes
                     AND prs !~ "a"//this finds any suffix form
]
```

Template explanation:

This query is intended to showcase how to build complex morphological queries. Here are some explanations that should help anybody to modify the query for him/herself:

//section01: here we define the part of speech (sp) as verbs (verb).

//section02: here the graphical performative (that is a phenomenological, not paradigmatic description of the performative) [g_pfm] is defined with the help of a regular expression (~). Each pfm starts with "!" and ends with "!". Since I am using regular expressions, I need to put a "\" before each "!" to make sure that the exclamation marks are not read as regular expressions but as ETCBC database values. The "[J]" stands for a Yod, the [.] stands for a Dagesh, and the [A] stands for a Patach. A full transcription list for vowels and other Masoretic markers can be found here: https://annotation.github.io/text-fabric/tf/writing/hebrew.html

//section03: here we define the lexeme (lex) with the help of regular expressions. In the sample above, we find a block that contains all the transliterated letters of the Hebrew alphabet. With the expression {3}, we make sure that the lexeme consists out of three consonants. To exclude II"yod/waw verbs, the regular expression lex !~ "^.[WJ]" is used, to make sure that the second consonant is not a waw or a yod. The dot is used to allow a consonant before the 'forbidden' letter. The content within the square brackets allows for "OR" functionality. Thus, [BK] would stand for either "B" or "K". [>BGDHWZXVJKLMNS<PYQRFCT] then stands for one of the consonants of the alphabet. Before the block, the anchor "^" is placed to indicate that this is the very beginning of the lexeme. At the end of the block, you find "\[" marking the lexeme as a verb. The normal sign of verbs is "[". However, since we are using the regular expression (regular expressions are enabled by the "~" signed after "lex") we have to make sure that "[" is not read as a regular expression but as a normal sign. This we do by putting "\" before "[".

//section04: here the graphical verbal ending (that is a phenomenological, not paradigmatic description of the performative) (g_vbe) is defined. Each g_vbe has to start with "[" and can then be

followed by any consonant and vowel. In my case, I have defined the "W" and "." (Dagesh) to define the ה ending.

//section05a: here the verbal stem (vs) is defined.

//section05b: alternatively to "vs" one could also choose for "vbs" (still paradigmatic representation) and "g_vbs" if one wants to look only for the graphical/phenomenological representation of the verbal stem and not for the paradigmatic information. E.g. vbs = "HT" (the three following values are available: "absent"=qal,piel,pual; "HT"=hit; "N"=nif, "H"=hif,hof)

//section05c: alternatively to "vs" or "vbs" (both paradigmatic representations) one could search for "g_vbs" if one wants to look only for the graphical/phenomenological representation of the verbal stem and not for the paradigmatic information. E.g. g_vbs ~ '\][N][IE]\]' (this would for example search for all cases in which we would find the stem signs for the Niphal נִ and נְ.

//section06: here we define the presence of pronominal suffixes. A list of the paradigmatic forms of the pronominal suffix (in Hebrew) is as follows:

```
1sgC = "NJ", "J"
2sgM = "K"
2sgF = "K="
3sgM = "W", "HW"
3sgF = "H"
1plC = "NW"
2plM = "KM"
2plF = "KN"
3plM = "HM", "M", "MW"
3plF = "HN", "N"
```

The alphabet for paradigmatic forms of the suffixes exists only of six letters: HWJKMN. The set of suffixes with at least one consonant contains all suffixes. Therefore, prs ~ "[HWJKMN]" is equal to prs !~ "a". Suffixes with at least two consonants can also be matched as follows: prs ~ "[HWJKMN]{2}".

Template example

A possible query could look like this:

```
select all objects where

[word FOCUS sp IN (verb,subs,prep) AND prs !~ 'a'] // this finds all suffixes that come with
at least one consonant (two consonants are possible)
OR
[word FOCUS sp IN (verb,subs,prep) AND prs = "J"] // this finds all suffixes that have only
J as consonant and no other consonant following.
OR
[word FOCUS sp IN (verb,subs,prep) AND prs ~ 'K='] // this finds the homographic 2sg suffix
OR
[word FOCUS sp IN (verb,subs,prep) AND prs ~ '[HWJKMN]{2}'] // this finds all suffixes that
come with two consonants
```