need to define "kind"

I pulled the section out of the manual you wrote listing the kinds found in AQL. Two comments:

1. This is not really a definition. Can you suggest a definition? I wrote “AQL contains a variety of constructions that build on on previously defined constructions. Following the programming language literature, we will refer to such constructions as kinds.” I think this comes closer to a definition, but by your remark I think you want more.
2. I am concerned that what is in the manual is not complete. For example, what about sigma, delta, pi, eval, coeval, unit and counit. If they are not kinds, then what do we call them?

There are “plain instance quotients” in AQL (where the equations are given by pairs of generators in JDBC or pairs of AQL terms written in the AQL file) - should these be used instead of instance coequalizers?

This sounds like a topic the will require an interactive discussion. Let’s schedule some time.

bullet point 13)  listed twice fixed.

mention don't need to understand all 13 points now - it is like a table of contents

"alow"

may want to list all AQL 'kinds' - can get them from the manual Done

what you call a "block", the manual and java code call a "section" Did a find and replace.

change "by definition automatically" to "by definition" Done

missing period after "designate C\_n" Done

"skolum" -> "Skolem" Done

"creating a structure" -> "creating a set of tables" or "**creating a database instance**" Done

need to define "generator" Done

need disclaimer that the tutorial will take some study to understand and you don't need to

understand everything before moving on (or do you?) I added some additional discussion in the introduction to this effect. We may want to elaborate on this point depending on the feedback we get from people like Mark Williams and Ken Webb.

"takes the product of two instances obtained by reversing all of the arrows" - it's ambiguous what arrows are being

reversed here and only one possible meaning is correct, and instance products are not introduced yet,

so suggest removing this sentence Done.

"as presentations of" -> "as a presentation of" Done

"schemas is of" -> 'schemas are of' Done

"query keyword" -> "query kind" Done

'no optional' -> 'not optional' Done

'Generally mappings that have the same source and target are called endomaps.', yes but so what? Removed

'it's novel' -> 'its novel' Done

"that vary over the the generators" actually the from-bound variables range over all the generators in the saturation/tables Rewrote to reflect this.

"whose type is schema\_colimit" -> "whose kind is schema\_colimit" Done

"Dela" -> "Delta" Done

"schema. so" -> "schema, so" Done

This:

 \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*WARNING!!!!\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* If we enable this block of code the surrogate ids created by AQL in the export will change. Since

 \* the literal transforms tLnkToSTSpnImp and tLnkToSPSpnImp defined below depend on the specific

 \* values of these surrogates those transforms will no longer work.

 \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 worries me if it isn't caused by the fact that the tutorial is self-contained.   Moreover, I'm not sure this this:

  \* In the first implementation of the instance we merging records using literal transforms. This is

 \* the most transparent of the three methods, but is not recommended. Beware, this is VERY brittle

 \* and we only illustrate this here help bridge to the later strategies.   Consider that if you

 \* re-export the instances all of the ids will change and this code segment will break.

 is a good idea - it's "evil" in the category theory sense.  There may be a way to obtain these transforms

 from  literal transforms by means of eval.  The comment "Clearly this approach does not scale." is useful to

 us, but not to users of the tool, who shouldn't be doing this at all.

" This is very different from the default behavior of the AQL import

 \* process. " - hopefully this is because the tutorial needs to be self-contained, not because the

 AQL import process is broken?

On reflection I completely agree. I got rid of the first member of this sequence.

 JDBC stuff just starts up fast and without warning or connection to the rest of the tutorial

 the ending is rushed I added an entry to the table of contents and a section describing pragmas, imports and exports broadly. I also added some additional explanation here and there. I think more is needed, but it is a start. One thing that would help is a full list of all of the options you now have for imports and exports. I think the manual is out of date. If you can provide me with a comprehensive set of options we might consider expanding this section to do a better job of illustrating what all of the various options do. I must admit I am a currently a bit bewildered by all of the options and not at all comfortable with my understanding of how to use them.

"coproduct\_sigma"

 \* <SkillClass[100.fk\_IsType.fk\_HasClass]> -> 122 - is this displayed in the viewer somewhere? No it is not, hence the lead in “We can conclude:” Do you have any suggestions for how to improve this? I did throw it together quickly.

https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif