Item 173
git_comments:

1. **comment:** In a case of making text search query for concrete subject adding uri predicate will make query much more efficient
   label: code-design
2. Cache-key does not matter if lang or graphURI are null

git_commits:

1. **summary:** JENA-1645: Use uri predicate in concrete subject query.
   message: JENA-1645: Use uri predicate in concrete subject query.

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Poor performance with full text search (Lucene)
   description: Situation: half of a million of an indexed by Lucene documents(emails actually), searching for emails by sender/receiver and some text. If to put text filter in the start of SPARQL query it executes once but in a case of very common words here are a lot of results(100 000+) that leads to poor performance, limiting results count may and up with missed results. If to put text search as the last condition it executes once per each already found subject. That's completely OK but text search completely ignores subject URI. I found two methods in TextQueryPF class: variableSubject(...) for the first case, and concreteSubject(...) for the second one. The question is: why can't subject URI be used as a constraint in the text search?

jira_issues_comments:

1. **body:** I have written some code that uses subject URI as an additional constraint and it works much faster in my case, but not sure if there can be any problems in more general cases.
   label: code-design
2. It would be helpful to see example queries and how you have used the subject URI. I agree that the {{concreteSubject}} *should* create Lucene queries that include a term of the form: {code} ... AND uri:http://example.org/data/resource/R0123 {code} Currently the code for {{concreteSubject}} collects results for all possible subjects and then after the results are returned selects just the ones corresponding to the provided {{subject}} and discards the rest of the results. Quite inefficient! This behavior is transparent to the user other than the performance; however, if there is some reason to keep this behavior then the _new_ behavior can be handled by adding a {{boolean}} {{TextIndex}} option in the configuration: {{text:useConcreteSubject}}. The implementation involves threading the subject into the {{TextIndex.query(...)}}, adding a new query method to {{TextIndex}}, {{TextIndexLucene}} and {{TextIndexES}}. It should be rather straightforward.
3. GitHub user DrBAXA opened a pull request: https://github.com/apache/jena/pull/503 JENA-1645: Use uri predicate in concrete subject query. Added URI predicate to the Lucene search in case of concrete subject search. Method added in TextIndex interface made default with a fallback to the previous implementation. You can merge this pull request into a Git repository by running: $ git pull https://github.com/DrBAXA/jena master Alternatively you can review and apply these changes as the patch at: https://github.com/apache/jena/pull/503.patch To close this pull request, make a commit to your master/trunk branch with (at least) the following in the commit message: This closes #503 ---- commit 52d959c7a654b03e525fad214b027b6ac6aba2b2 Author: vdanyliuk <vasyl.danyliuk@...> Date: 2018-12-05T09:10:49Z JENA-1645: Use uri predicate in concrete subject query. ----

4. The query is pretty straightforward: {code:java} PREFIX person: <http://person/> PREFIX email: <http://email/> PREFIX text: <http://jena.apache.org/text#> SELECT DISTINCT ?emailId ?content WHERE { ?person1Id person:name "Person One" . ?person2Id person:name "Second Person" . {?person1Id email:sent ?emailId . ?person2Id email:received ?emailId .} UNION {?person2Id email:sent ?emailId . ?person1Id email:received ?emailId .} (?emailId ?score ?content) text:query (email:indexedContent "ext to search" 1 "highlight:s:<em class='hiLite'> | e:</em>") . } {code} Such cases already covered by tests in jena-text module. Created pull request with code added to the Lucene index.
5. Github user rvesse commented on the issue: https://github.com/apache/jena/pull/503 cc @osma @xristy for review as main devs in this area
6. Github user xristy commented on the issue: https://github.com/apache/jena/pull/503 I'm looking at the PR and so far it looks good. I'm wanting to complete the PR for JENA-1646.
7. Github user osma commented on the issue: https://github.com/apache/jena/pull/503 Looks good to me, based on a quick look at the diff. It took a while to figure out the way you use UnaryOperator but in the end it made sense. I'm currently travelling and haven't found time to actually run the code, but if it passes the existing unit tests and you're sure that they trigger the `concreteSubject` case, I'm fine with that.
8. Commit 52d959c7a654b03e525fad214b027b6ac6aba2b2 in jena's branch refs/heads/master from vdanyliuk [https://git-wip-us.apache.org/repos/asf?p=jena.git;h=52d959c] JENA-1645: Use uri predicate in concrete subject query.
9. Github user xristy commented on the issue: https://github.com/apache/jena/pull/503 I merged PR 503 into local master and pushed to https://git-wip-us.apache.org/repos/asf/jena.git with message: JEAN-1645 Merge commit 'refs/pull/503/head' of https://github.com/apache/jena. This closes #503 owing to the misspelling of JENA as JEAN programmatic resolving of the issue and closing of the PR has not happened. The code changes are in the ASF jena repo as expected.
10. [PR #503|https://github.com/apache/jena/pull/503] is tested and merged via [501d9f68d|https://git-wip-us.apache.org/repos/asf?p=jena.git;a=commit;h=501d9f68d46e30ce5289b04dfba13f07bbf7d2d9] .
11. Commit 82121f9c52f4aac03d21219313267554d108938b2 in jena's branch refs/heads/master from [~code-ferret] [https://git-wip-us.apache.org/repos/asf?p=jena.git;h=82121f9] JENA-1645: This closes #503.
12. Github user asfgit closed the pull request at: https://github.com/apache/jena/pull/503