Postgresql GSoC 2014 Proposal

This proposal expands on Simon Riggs's original proposal here[1])

Background

When Postgresql must store a variable-length datum that is too large to fit into one page, it uses TOAST (The Oversized-Attribute Storage Technique), which compresses and/or stores the datum in a separate TOAST table, slicing it into smaller parts if necessary. When the user accesses the datum in the future, Postgresql de-toasts it transparently by choosing the correct slices to retrieve.

There are four toasting strategies: `PLAIN` (no TOAST), `MAIN` (compressed, in-line), `EXTERNAL` (uncompressed, out-of-line), and `EXTENDED` (compressed, out-of-line). In particular, `text` and `bytea` are `EXTERNAL` by default, so that substring operations can seek straight to the exact slice (which is `O(1)`) instead of de-toasting the whole datum (which is `O(file size)`). Specifically, `varlena.c`'s `text_substring(…)` and `bytea_substring(…)` call `DatumGetTextPSlice(…)`, which retrieves only the slice(s) at an easily-computed offset.

Proposal

As a GSoC student, I will implement a similar optimization for substring operations of other predictably structured data types, in two phases:

1. First, I will optimize array element retrieval and UTF-8 substring retrieval. Both are straightforward, as they involve calculating slice numbers and using similar code to above.
2. Second, I will implement a `SPLITTER` clause for the `CREATE TYPE` statement. As [1] proposes, one would define a type, for example:

```
CREATE TYPE my_xml
  LIKE xml
  SPLITTER  my_xml_splitter;
```

with a SPLITTER function "that gets called iteratively on a column value until it returns no further slices" ([1]). Here is my mockup for such a function, technical aspects to be corrected later:

```
static Datum[] my_xml_splitter(PG_FUNCTION_ARGS) {
        Datum **results = (Datum**) palloc(2 * TOAST_TUPLE_TARGET);
        xmltype *chunk = PG_GETARG_XML_P(0);
        int size = VARSIZE(chunk);
        if (size > TOAST_TUPLE_TARGET) {
                // Use code similar to tuptoaster.c's toast_save_datum()'s
                // "Split up the item into chunks" section.
                results[0] = a small slice ready for toasting
                results[1] = the remaining part of the datum after slice
        }
        else {
                results[0] = chunk;
                results[1] = NULL;
        }
        return results;
}
```

[1] www.postgresql.org/message-id/CA+U5nMJGgJNt5VXqkR=crtDqXFmuyzwEF23-fD5NuSns+6N5dA@mail.gmail.com

Then, user-defined `my_xml` functions can optimize seeking by determining the correct slice number, perhaps as a best guess before retrieving the entire datum as usual. For example, the first element might always be in the first slice. Or, certain elements might usually be in a certain slice because the previous elements are of predictable lengths (as real-world preconditions or asspecified by the XML schema).

Deliverables

- Optimized `text_substring` for UTF-8 text
- Optimized array element retrieval
- `SPLITTER` clause in `CREATE TYPE` statement
- Primary documentation. As far as I can tell, in these sections of the official documentation:
  - TOAST: explanation of splitting, which is to remain optional
  - User-Defined Types: where TOAST is mentioned, mention splitting option
  - `CREATE TYPE` and `ALTER TYPE` descriptions: explain the `SPLITTER` clause and how it requires `STORAGE/SET STORAGE` to be `EXTERNAL`.
- Secondary documentation. Here, I recommend splitting as a work-around to existing Postgresql problems. For example, but not limited to,
  - XML Type: Currently, one must search for a specific XML element by serializing the document and doing a textual search. The documentation here[2] suggests that XML should have built-in text search one day. But in the meantime, mention the splitting option.
  - `BLOB` and `bytea` storage are better for different applications; for example, `BLOB`s support streaming but `bytea` does not. Here[3], Pavel Stehule supports the idea of `bytea` streaming functionality; and here[4], Pavel Golub connects Stehule's idea to this splitting project. Since `bytea` streaming does not exist yet, wherever the pros and cons between `BLOB` and `bytea` are mentioned in the Postgresql or JDBC driver documentation, I will mention splitting as an alternative.

Schedule

You are reading a first draft of this proposal. I'll devise a schedule closer to the proposal submission deadline.

Bio

I am a third year computer science and math student at California State University, Long Beach, USA. After graduating, I would like to work with databases as an architect, analyst, admin, etc. In the meantime (this summer + my fourth year), I would like to produce a public portfolio of work with Postgresql starting with, but not ending with, this Google Summer of Code project.

[2] www.postgresql.org/docs/9.3/static/datatype-xml.html

[3] www.postgresql.org/message-id/BANLkTini+ChGKfnyjkF1rsHSQ2kMktSDjg@mail.gmail.com

[4] www.postgresql.org/message-id/1886757050.20130515120151@gf.microolap.com