PingCAP

Code Comment Style

Caitin Chen | I18N | PingCAP

Agenda

PingCAP

- Why

  Why does a good comment matter?

- Where/When

  Where/When to comment?

- How

  How to comment?

- Examples

  Comment examples

PingCAP

Why does a good comment matter?

A good comment matters

PingCAP

- To speed up the reviewing process
- To help maintain the code
- To improve the API document readability
- To improve the development efficiency of the whole team

PingCAP

Where/When to comment?

Write a comment where/when:

PingCAP

- For important code
- For obscure code
- For tricky or interesting code
- For a complex code block
- If a bug exists in the code but you cannot fix it or you just want to ignore it for the moment
- If the code is not optimal but you don’t have a smarter way now
- To remind yourself or others of missing functionality or upcoming requirements not present in the code

Example: comment for important code

PingCAP

Chunk is the core data structure of SQL engine

```
// Chunk stores multiple rows of data in Apache Arrow format.
// See https://arrow.apache.org/docs/memory_layout.html
// Values are appended in compact format and can be directly accessed without decoding.
// When the chunk is done processing, we can reuse the allocated memory by resetting it.
type Chunk struct {
    columns []*column
    // numVirtualRows indicates the number of virtual rows, which have zero column.
    // It is used only when this Chunk doesn't hold any data, i.e. "len(columns)==0".
    numVirtualRows int
}
```

Example: comment for complex code

PingCAP

Parallel Aggregation

```
// HashAggExec deals with all the aggregate functions.
// It is built from the Aggregate Plan. When Next() is called, it reads all the data from Src
// and updates all the items in PartialAggFuncs.
// The parallel execution flow is as the following graph shows:
//
//                          +-------------+
//                          | Main Thread |
//                          +------+------+
//                                 ^
//                                 |
//                                 +
//                            +-+-            +-+
//                            | |    ......   | |  finalOutputCh
//                            +++-            +-+
//                             ^
//                             |
//                             +---------------+
//                             |               |
//                  +----------+---+           +---------------+
//                  | final worker |    ......  | final worker |
//                  +----------+-+-+            +-+-------------+
//                             ^                 ^
//                             |                 |
//                            +-+  +-+  ......  +-+
//                            | |  | |          | |
//                            ...  ...          ...    partialOutputChs
//                            | |  | |          | |
//                            +++  +++          +++
//                             ^    ^            ^
//                             |    |            |
//          +-+                |    |            |
//          | |        +-------+----+            |
// inputCh  +-+        |                +--------+---+
//          | |        |                             |
//          ...    +---+----------+           +------+-----+
//          | |    | partial worker |   ......  | partial worker |
//          +++    +--------------+-+           +-+--------------+
//           |                    ^               ^
//           |                    |               |
//      +----v---------+         +++ +-+         +++
//      | data fetcher | +------> | | | |  ......  | |   partialInputChs
//      +--------------+         +-+ +-+          +-+
type HashAggExec struct {
```

Example: comment for tricky code

PingCAP

A corner case

```
if (e.groupMap.Len() == 0) && len(e.GroupByItems) == 0 {
    // If no groupby and no data, we should add an empty group.
    // For example:
    // "select count(c) from t;" should return one row [0]
    // "select count(c) from t group by c1;" should return empty result set.
    e.groupMap.Put([]byte{}, []byte{})
}
e.prepared = true
```

Example: TODO

PingCAP

Have no idea about how to handle it, so add a TODO comment.

```
// TODO: Need to do something when err is not nil.
err := statsOwner.CampaignOwner(cancelCtx)
if err != nil {
    log.Warnf("[stats] campaign owner fail: %s", errors.ErrorStack(err))
}
return statsOwner
```

A comment is generally required for:

PingCAP

- Package (Go)
- File
- Type
- Constant
- Function
- Method
- Variable
- Typical algorithm
- Exported name
- Test case
- TODO
- FIXME

PingCAP

How to comment?

Write a good comment

PingCAP

- Format
- Language
- Tips

Format of a good comment (1/3)

PingCAP

- Go
  - Use // for a single-line comment and trailing comment
  - Use /* **...** */ for a block comment (used only when needed)
  - Use **gofmt** to format your code

Format of a good comment (2/3)

PingCAP

- Rust
  - Non-doc comment
    - Used to document implementation details
    - Use // for a line comment
    - Use /* ... */ for a block comment (used only when needed)
  - Doc comment
    - Used to document the interface of code (structures, fields, macros, etc.)
    - Use /// for item documentation (functions, attributes, structures, etc.)
    - Use **//!** for module level documentation
    - Use **rustfmt** to format your code

Format of a good comment (3/3)

PingCAP

- Place the single-line and block comment above the code it's annotating
- Fold long lines of comments
- The maximum width for a line is 100 characters
- For a comment containing a URL
  - Use a **relative URL** if the text is linked to a file within the same GitHub repository
  - Use an **absolute URL** in **docs and docs-cn repositories** if the code with this comment is copied from another repository

Language for a good comment (1/4)

PingCAP

- Capitalize the first letter of sentences and end them with periods
  - If a lower-case identifier comes at the beginning of a sentence, don't capitalize it

```
// enterGame causes Players to enter the
// video game, which is about a romantic
// story in ancient China.
func enterGame() os.Error {
   ...
}
```

Language for a good comment (2/4)

PingCAP

- Word
  - Use **American English** rather than British English
    - color, canceling, synchronize (Recommended)
    - colour, cancelling, synchronise (Not recommended)
  - Use correct spelling
    - grammar ✓
    - grammer ✗
  - Use **standard or official capitalization**
    - TiKV, TiDB-Binlog, Region, gRPC, RocksDB, GC, k8s, mydumper, Prometheus Pushgateway ✓
    - Tikv, TiDB Binlog, region, grpc, rocksdb, gc, K8S, MyDumper, Prometheus PushGateway ✗

Language for a good comment (3/4)

PingCAP

- Word
  - Use words and expressions consistently
    - “dead link” vs. “broken link”
  - Do not use lengthy compound words
  - Do not abbreviate unless it’s absolutely necessary
  - “we” should be used only when it means the code writer and the reader
- Sentence
  - Use standard grammar and correct punctuation
  - Use relatively short sentences

Language for a good comment (4/4)

PingCAP

- When used for description, comments should be **descriptive** rather than **imperative**
  - Opens the file ✓
  - Open the file ✗
- Use “this” instead of “the” to refer to the current thing
  - Gets the toolkit for this component (Recommended)
  - Gets the toolkit for the component (Not recommended)
- Markdown format is allowed
  - Opens the `log` file ✓

Tips for a good comment

PingCAP

- Comment code while writing it
- Do not assume the code is self-evident
- Avoid unnecessary comments for simple code
- Write comments as if they were for you
- Make sure the comment is up-to-date
- Let the code speak for itself

PingCAP

Comment examples

Comment examples

PingCAP

- Package comment (Go)
- Type comment
- Function comment
- TODO comment
- FIXME comment
- Test case comment

Package comment (Go) (1/2)

PingCAP

- Put the comment immediately preceding the package statement in one of the files in the package (It only needs to appear in one file)

```
// Package superman implements methods for saving
// the world.
//
// Experience has shown that a small number of
// procedures can prove helpful when attempting to //
save the world.
package superman
```

Package comment (Go) (2/2)

PingCAP

- Comment contains:
  - The first sentence: “Package *packagename*” + a concise summary of the package functionality
  - Subsequent sentences give more details

```
/*
Package regexp implements a simple library for regular expressions.

The syntax of the regular expressions accepted is:

    regexp:
        concatenation { '|' concatenation }
    concatenation:
        { closure }
    closure:
        term [ '*' | '+' | '?' ]
    term:
        '^'
        '$'
        '.'
        character
        '[' [ '^' ] character-ranges ']'
        '(' regexp ')'
*/
package regexp
```

Type comment

PingCAP

- Comment non-obvious type declaration
- Comment may contain:
  - What this type is for
  - How/When to use this type
  - Example

```
// Iterates over the contents of a GargantuanTable.
// Example:
//    GargantuanTableIterator* iter = table->NewIterator();
//    for (iter->Seek("foo"); !iter->done(); iter->Next()) {
//        process(iter->key(), iter->value());
//    }
//    delete iter;
type GargantuanTableIterator {
    ...
}
```

Function comment

PingCAP

- Comment every function unless the function is simple and obvious
- Comment may contain:
  - What the inputs and outputs are
  - Whether any of the arguments can be a null pointer or how zero values are treated
  - Whether there are any performance implications of how a function is used
  - How and why the function might panic or report an error

TODO comment

PingCAP

- Use TODO comments for code that is temporary, a short-term solution, or good-enough but not perfect
- A TODO comment should include the string TODO in all caps, followed by the issue to resolve

```
// TODO: Use a "*" here for concatenation operator.
// TODO: Change this to use relations.
// TODO: Remove the "Last visitors" feature.
```

FIXME comment

PingCAP

- Use FIXME comments for code that is broken but you do not want to worry about for the moment
- A FIXME comment should include the string FIXME in all caps, followed by the issue description

```
// FIXME: This won't work if the file is missing.
```

```
// FIXME: It should return a decimal value, but
// now it returns a double value.
```

Test case comment

PingCAP

- Comment may contain:
  - Test purpose
  - Test method
  - Test data
  - The expected result

```
+       // The ODBC syntax for time/date/timestamp literal.
+       // See: https://dev.mysql.com/doc/refman/5.7/en/date-and-time-literals.html
+       {"select {ts '1989-09-10 11:11:11'}", true},
+       {"select {d '1989-09-10'}", true},
+       {"select {t '00:00:00.111'}", true},
+       // If the identifier is not in (t, d, ts), we just ignore it and consider the following expression as a
string literal.
+       // This is the same behavior with MySQL.
+       {"select {ts123 '1989-09-10 11:11:11'}", true},
```

PingCAP

Thanks for your cooperation!

Embrace elegant comments