

测试品控

完备的测试和健壮的工作流支撑 Databend 半年以来的快速迭代，本次分享将会聚焦 Databend 的测试工作

演讲者：尚卓燃

演讲时间：2021.10.21

ABOUT US

Databend 的目标是打造一流的云数仓，与 Rust 的健壮生态共同构建新一代的数据云

An elastic and reliable Cloud Data Warehouse, offers Blazing Fast Query and combines Elasticity, Simplicity, Low cost of the Cloud, built to make the Data Cloud easy

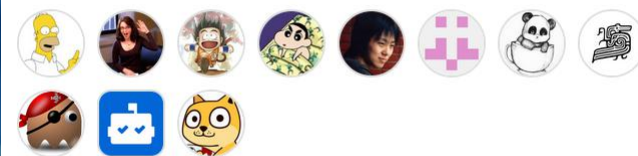
databend.rs

rust sql database storage olap
data-cloud query-processing cloud-warehouse

 Readme

 Apache-2.0 License

Contributors 47

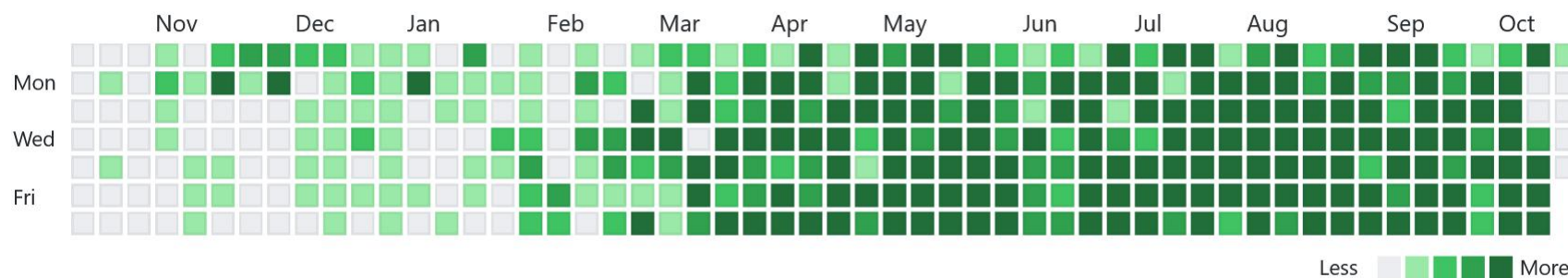


+ 36 contributors

Releases

 147 tags

[Create a new release](#)



PART.01

质 量 保 障

PART.02

测 试 概 况

PART.03

遗 珠 之 憾

PART.04

积 极 探 索

Databend
测试品控

PART.01

质量保障

Contributors & Ecosystem

感谢 **Databend** 贡献者们，他们的 **git.name** 将会保存在 **system.contributors** 表中

感谢 **Rust** 生态中的优秀上游项目，他们的版本和许可将会保存在 **system.credits** 表中

system.contributors

Contains information about contributors.

```
mysql> SELECT * FROM system.contributors LIMIT 20;
+-----+
| name                |
+-----+
| artorias1024        |
| BohuTANG             |
| dependabot[bot]     |
| dependabot-preview[bot] |
| drdr xp             |
| Eason                |
| ...
```

system.credits

Contains information about credits.

```
mysql> SELECT * FROM system.credits LIMIT 20;
+-----+-----+-----+
| name          | version | license                |
+-----+-----+-----+
| addr2line     | 0.16.0  | Apache-2.0 OR MIT     |
| adler         | 1.0.2   | 0BSD OR Apache-2.0 OR MIT |
| ahash         | 0.6.3   | Apache-2.0 OR MIT     |
| ahash         | 0.7.4   | Apache-2.0 OR MIT     |
| aho-corasick  | 0.7.18  | MIT OR Unlicense       |
| ansi_term     | 0.9.0   | MIT                    |
| ansi_term     | 0.11.0  | MIT                    |
```







Databend X GitHub Actions

GitHub 不仅提供了代码托管平台，还为开源项目提供免费的工作流时长。

Databend利用GitHub Actions 构建了一套完备的持续集成和测试机制，确保代码合入的可靠性。

✓ All checks have passed
11 successful checks

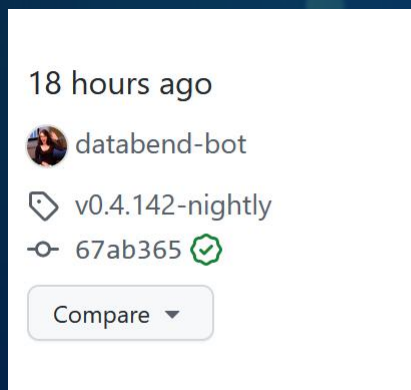
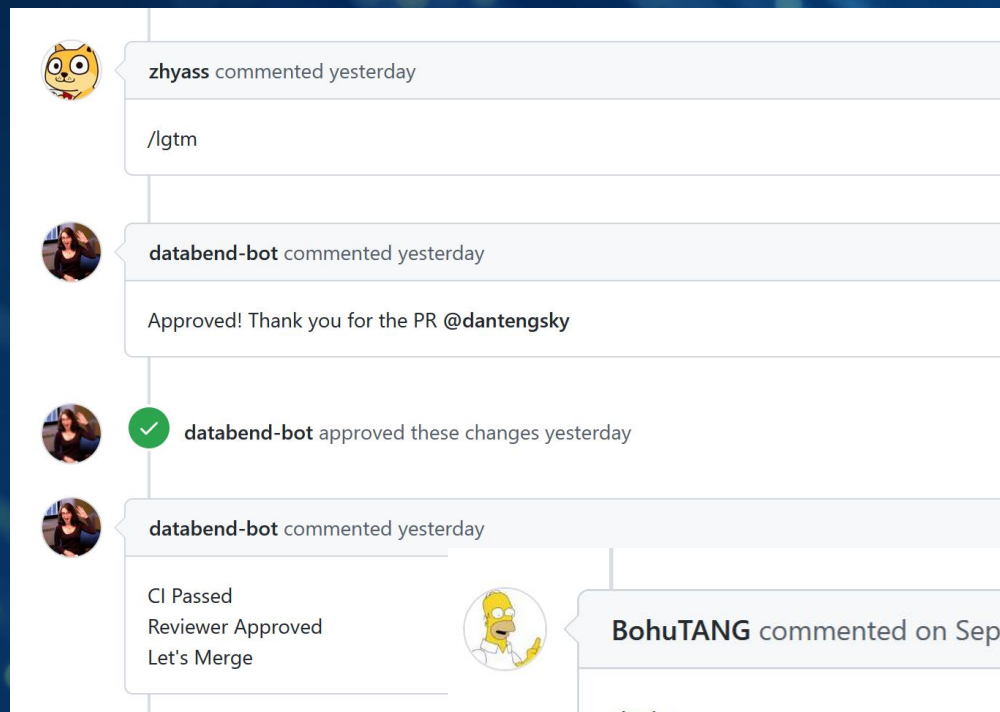
Hide all checks

✓	 Cargo / Lint (pull_request) Successful in 2m	Details	^
✓	 License checker / check-license (pull_request) Successful in 57s	Details	
✓	 Stateless(Cluster) / Tests (ubuntu-latest, stable, x86_64-unknown-linux-gnu, false) (pull_requ...	Details	
✓	 Stateless(Standalone) / Tests (ubuntu-latest, stable, x86_64-unknown-linux-gnu, false) (pull_r...	Details	
✓	 Unit Tests / Tests (ubuntu-latest, stable, x86_64-unknown-linux-gnu, false) (pull_request) S...	Details	
✓	 Build / build (macos-latest) (pull_request) Successful in 5m	Details	▼

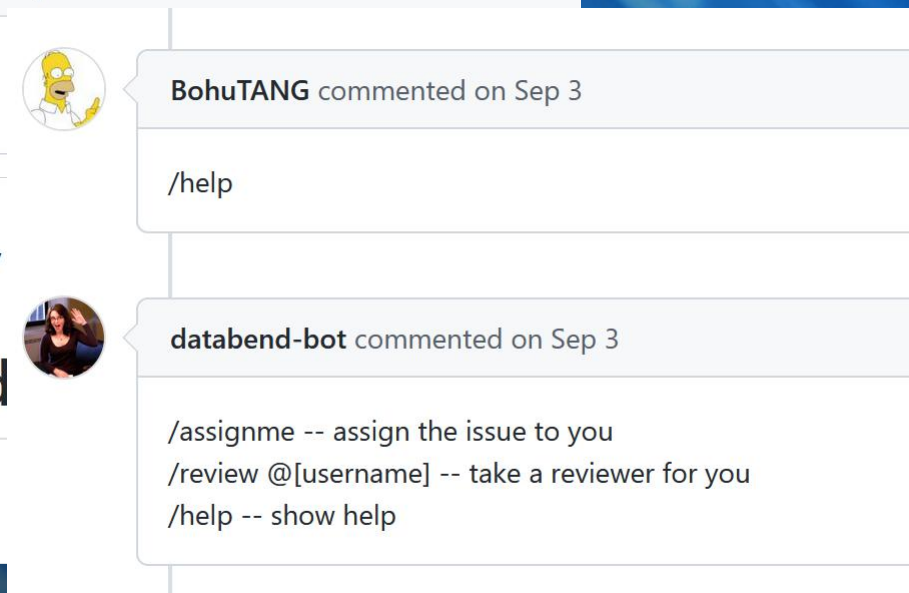
Databend Bot – She always loved you

对于代码贡献，除了必要的开发和审查人员之外，还有一个得力助手每天勤勤恳恳帮忙处理各种事务

变更日志生成，为 PR 打标签，分配问题/PR 给相关的成员，完成合并工作



v0.4.142-nightly
What's Changed
Bug Fixes





Databend

测试品控

PART.02

测试概况

测试什么

是否会导致功能不正常？

是否有跨平台编译问题？

是否会影响分布式执行？

是否会导致性能下降？

```
#[tokio::test(flavor = "multi_thread", worker_threads = 1)]
async fn test_credits_table() -> Result<()> {
    let ctx = crate::tests::try_create_context()?;

    let table: Arc<dyn Table> = Arc::new(CreditsTable::create(1));
    let io_ctx = ctx.get_single_node_table_io_context()?;
    let io_ctx = Arc::new(io_ctx);
    let source_plan = table.read_plan(
        io_ctx.clone(),
        None,
        Some(ctx.get_settings().get_max_threads()? as usize),
    );

    let stream = table.read(io_ctx, &source_plan.push_downs).await?;
    let result = stream.try_collect::(<Vec<_>>()).await?;
    let block = &result[0];
    assert_eq!(block.num_columns(), 3);
    Ok(())
}
```

```
target: x86_64-unknown-linux-gnu, cross: false}
target: aarch64-unknown-linux-gnu, cross: true}
target: arm-unknown-linux-gnueabi, cross: true}
target: armv7-unknown-linux-gnueabihf, cross: true}
target: x86_64-apple-darwin, cross: false}
```

执行测试 (stateless)

- 在单机或者集群模式下进行
- 类似 clickhouse
- 通过对比预期结果和实际结果完成
- **make stateless-test**

✓ Run Stateless Tests with Standalone mode

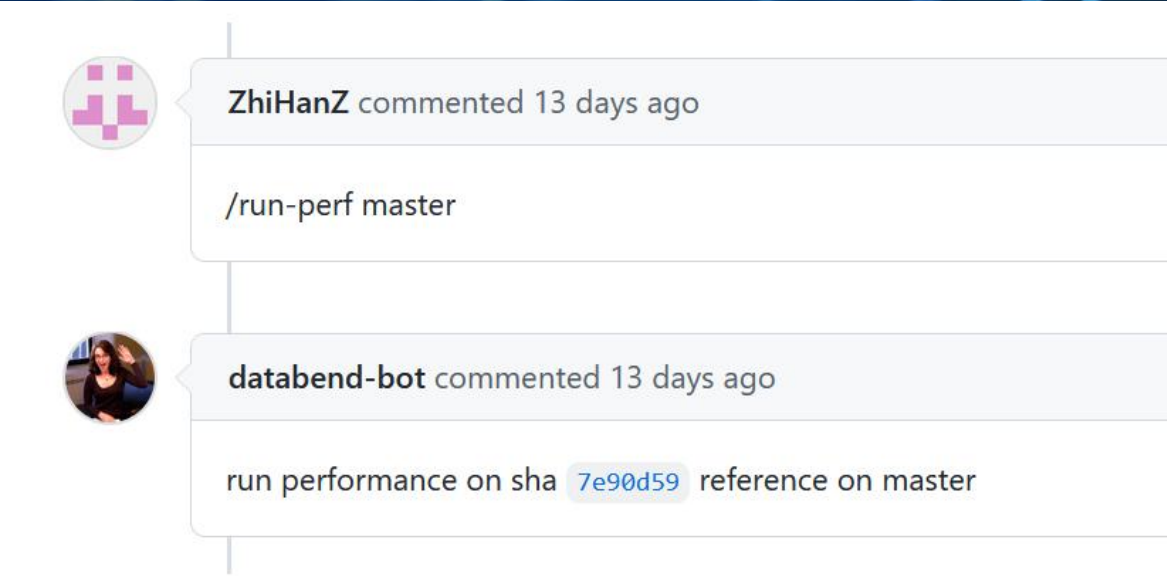
```
65 00_0000_dummy_select_1: [ OK ]
66 01_0000_system_numbers: [ OK ]
67 01_0001_system_tables: [ OK ]
68 02_0000_function_arithmetic: [ OK ]
69 02_0001_function_to_type_name: [ OK ]
70 02_0002_function_cast: [ OK ]
71 02_0003_function_database: [ OK ]
72 02_0004_function_name_display: [ OK ]
73 02_0005_function_compare: [ OK ]
```

```
1 SELECT 1;
2 SELECT x; -- {ErrorCode 6}
3 SELECT 'a';
4 SELECT NOT(1=1);
5 SELECT NOT(1);
6 SELECT NOT(1=1) from numbers(3);
```

```
1 1
2 a
3 0
4 0
5 0
6 0
```


性能测试

- 多次执行同一条语句
- 对比 Master 和 PR
- 由 Databend Bot 和 Test Infra 提供支持



Changes in Performance

current_score , s	ref_score , s	Ratio of speedup (-) or slowdown (+)	Relative difference (after_score – before_score) / before_score	Test	Status	Query
2166.457	2130.997	-0.984x	-0.016	Filter-Gt	stable	SELECT count() FROM numbers_mt(1000000000) where number > 10000
39245.857	35194.542	-0.897x	-0.103	Aggregation-avg	slower	SELECT avg(number) from numbers_mt(1000000000)
2435.639	2505.466	+1.029x	0.029	Filter-Neq	stable	SELECT count() FROM numbers_mt(1000000000) where number != 10000
160.339	159.634	-0.996x	-0.004	Substring	stable	SELECT substring(cast(number as text) from 3) from numbers_mt(100000000) where number > 100 order by number desc limit 10
793.456	695.878	-0.877x	-0.123	Group-By-Multiple-Aggr	slower	SELECT max(number) as max,sum(number) as sum FROM numbers_mt(1000000000) GROUP BY number % 3, number % 4, number % 5
67937.618	63168.860	-0.930x	-0.070	Aggregation-count	slower	SELECT count(number) from numbers_mt(1000000000)
8047.885	8902.444	+1.106x	0.106	Sort	faster	SELECT number FROM numbers_mt(100000000) ORDER BY number DESC LIMIT 10



Databend

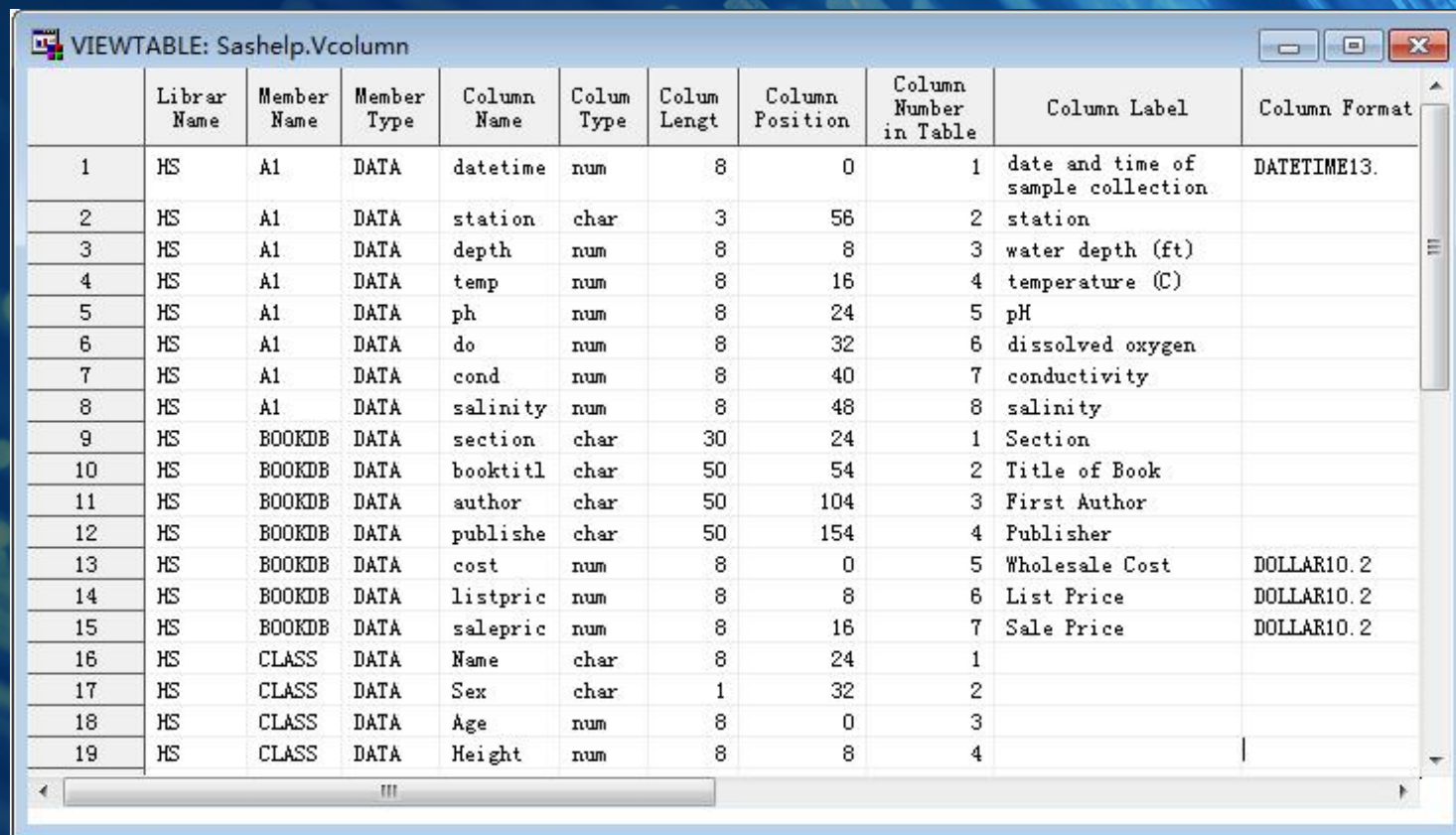
测试品控

PART.03

遗珠之憾

集成测试（需完善）

- 需要补充当前的 SQL 测试集，考虑一些可能的边界案例
- 提供一套 **stateful** 测试，覆盖大部分常用场景和功能
- 与 **cli** 系统结合进行测试，同时测试 **cli** 的能力
- 测试不同驱动的兼容能力



VIEWTABLE: Sashelp.Vcolumn

	Librar Name	Member Name	Member Type	Column Name	Column Type	Column Length	Column Position	Column Number in Table	Column Label	Column Format
1	HS	A1	DATA	datetime	num	8	0	1	date and time of sample collection	DATETIME13.
2	HS	A1	DATA	station	char	3	56	2	station	
3	HS	A1	DATA	depth	num	8	8	3	water depth (ft)	
4	HS	A1	DATA	temp	num	8	16	4	temperature (C)	
5	HS	A1	DATA	ph	num	8	24	5	pH	
6	HS	A1	DATA	do	num	8	32	6	dissolved oxygen	
7	HS	A1	DATA	cond	num	8	40	7	conductivity	
8	HS	A1	DATA	salinity	num	8	48	8	salinity	
9	HS	BOOKDB	DATA	section	char	30	24	1	Section	
10	HS	BOOKDB	DATA	booktitl	char	50	54	2	Title of Book	
11	HS	BOOKDB	DATA	author	char	50	104	3	First Author	
12	HS	BOOKDB	DATA	publishe	char	50	154	4	Publisher	
13	HS	BOOKDB	DATA	cost	num	8	0	5	Wholesale Cost	DOLLAR10.2
14	HS	BOOKDB	DATA	listpric	num	8	8	6	List Price	DOLLAR10.2
15	HS	BOOKDB	DATA	salepric	num	8	16	7	Sale Price	DOLLAR10.2
16	HS	CLASS	DATA	Name	char	8	24	1		
17	HS	CLASS	DATA	Sex	char	1	32	2		
18	HS	CLASS	DATA	Age	num	8	0	3		
19	HS	CLASS	DATA	Height	num	8	8	4		

Fuzz Tests for Sqlparser (需修复)

- 测试工具通过随机或是半随机的方式生成大量数据;
- 测试工具将生成的数据发送给被测试的系统 (输入) ;
- 测试工具检测被测系统的状态 (如是否能够响应, 响应是否正确等) ;
- 根据被测系统的状态判断是否存在潜在的安全漏洞。

```
-----[ 0 days 00 hrs 00 mins 12 secs ]-----
Iterations : 310,996 [311.00k]
Mode [3/3] : Feedback Driven Mode
  Target : hfuzz_target/x86_64-unknown-linux-gnu/release/fuzz_parse_sql
  Threads : 4, CPUs: 8, CPU%: 548% [68%/CPU]
  Speed : 103/sec [avg: 25,916]
  Crashes : 210 [unique: 1, blocklist: 0, verified: 0]
  Timeouts : 0 [1 sec]
Corpus Size : 3,689, max: 8,192 bytes, init: 1,187 files
Cov Update : 0 days 00 hrs 00 mins 00 secs ago
Coverage : edge: 1,395/17,270 [8%] pc: 10 cmp: 92,539
----- [ LOGS ] -----/ honggfuzz 2.4 /-
5214.CODE.-6.ADDR.0.INSTR.mov___0x108(%rsp),%rax.fuzz' already exists, skipping
Signal 2 (Interrupt) received, terminating
Terminating thread no. #0, left: 3
Terminating thread no. #3, left: 2
Terminating thread no. #1, left: 1
Crash (dup): 'hfuzz_workspace/fuzz_parse_sql/SIGABRT.PC.7ffff7dca2a2.STACK.c27b5
5214.CODE.-6.ADDR.0.INSTR.mov___0x108(%rsp),%rax.fuzz' already exists, skipping
Terminating thread no. #2, left: 0
Summary iterations:310996 time:12 speed:25916 crashes_count:211 timeout_count:0
new_units_added:3149 slowest_unit_ms:222 guard_nb:17270 branch_coverage_percent:
8 peak_rss_mb:147
```


MIRI 未定义行为分析工具（需调查）

- 运行 Rust 二进制文件，对其进行测试，可以检查出某些未定义的行为。当前已经在 **setup** 包含了 **miri** 工具，可以执行 **make miri** 启动分析
- 目前涉及上下游依赖的 **UB** 比较多，需要逐个进行甄别和 **ignore**，虽然艰苦，但是仍然很有必要

```
#[test]
#[cfg_attr(miri, ignore)]
fn does_not_work_on_miri() {
    tokio::run(futures::future::ok::<_, ()>(()));
}
```

There is no way to list all the infinite things Miri cannot do, but the interpreter will explicitly tell you when it finds something unsupported:

```
error: unsupported operation: can't call foreign function: bind
...
= help: this is likely not a bug in the program; it indicates that the program \
        performed an operation that the interpreter does not support
```



Databend

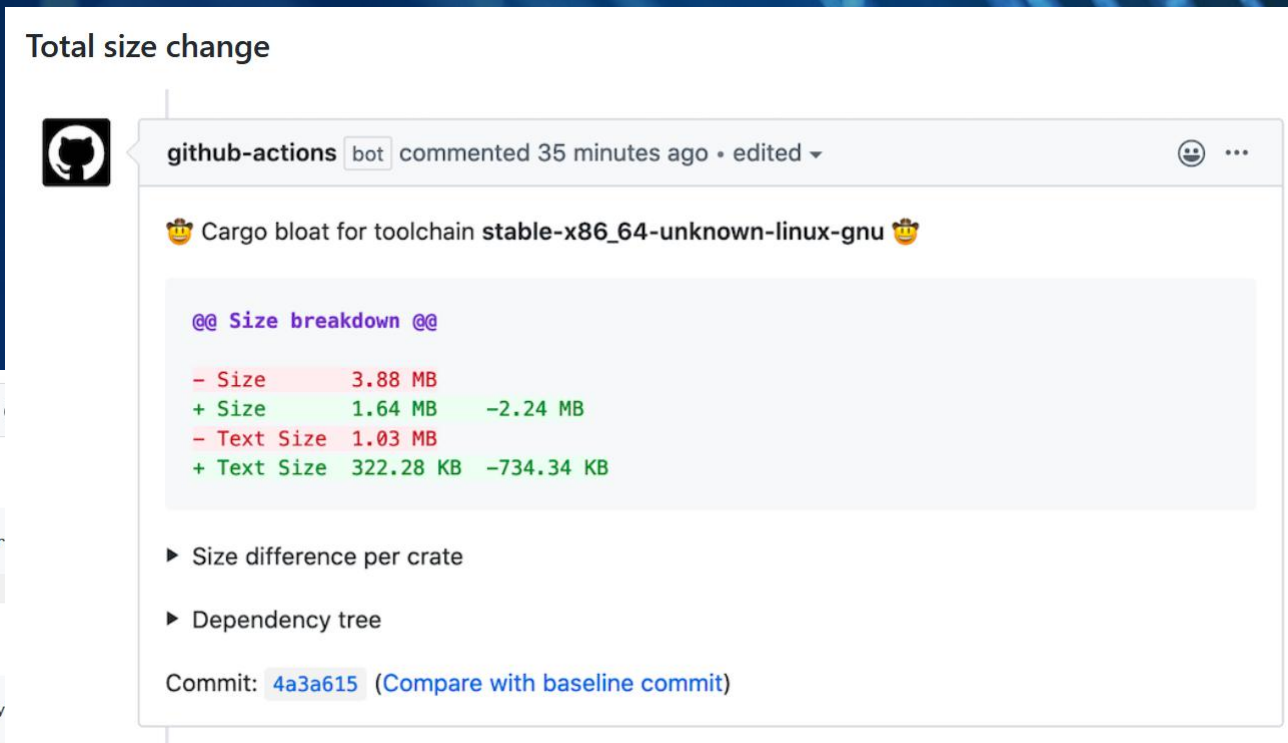
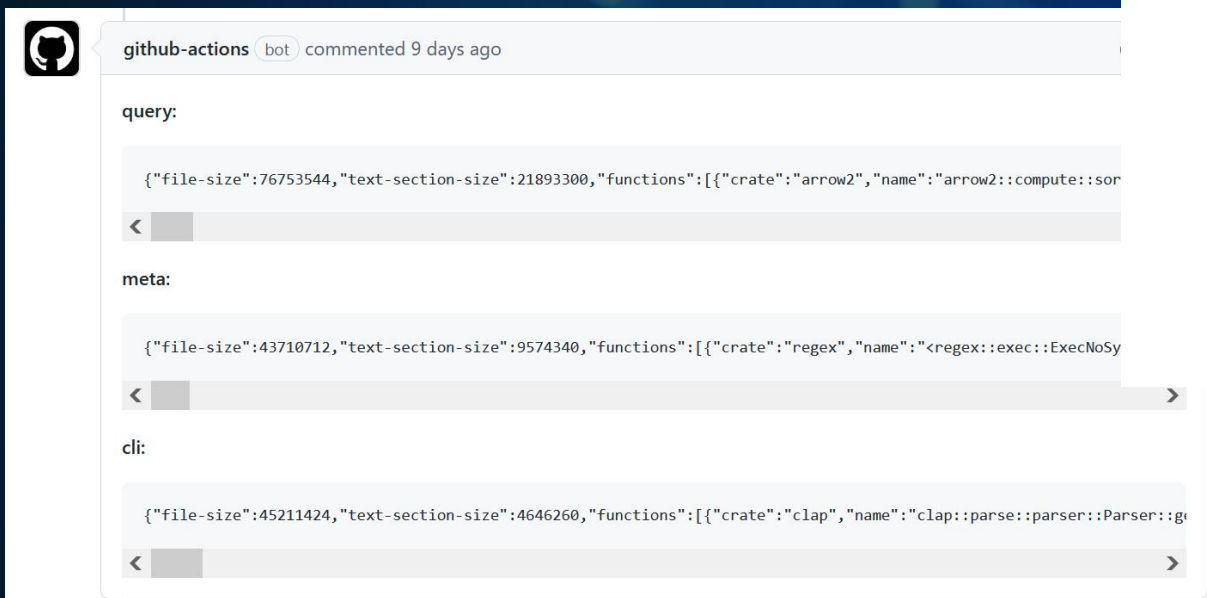
测试品控

PART.04

积极探索

Bloat 二进制分析工具（需改进）

- 找出什么占用了可执行文件中的大部分空间。
- <https://github.com/PsiACE/databend/pull/2>
- <https://github.com/orf/cargo-bloat-action>



重构测试套件（调查中）

- New testing style in Rust crate #1866
- <https://matklad.github.io/2021/02/27/delete-cargo-integration-tests.html>
- 目前的测试很多都停留在 go style，一方面是采用 Rust style，另一方面调查能否改善编译和测试的时间

```
1  awesomeness-rs/  
2    Cargo.toml  
3    src/          # unit tests go here  
4      lib.rs  
5      submodule.rs  
6      submodule/  
7        tests.rs  
8  
9    tests/        # integration tests go here  
10     is_awesome.rs
```

```
1  tests/  
2    it.rs  
3  
4  # Or, for larger crates  
5  
6  tests/  
7    it/  
8      main.rs  
9      foo.rs  
10     bar.rs
```


SQL Logic Tests (构想中/待实施)

- 模糊测试虽然可以帮忙发现问题，但毕竟会脱离真实的情况
- 根据语法树生成 SQL 语句并执行解析/尝试运行可能是更好的办法
- AFL -> 字典，Gen data by grammer, SQL/WASM Smith
- SQL Logic Tests In Materialize SQLancer



SQLancer (Synthesized Query Lancer) is a tool to automatically test Database Management Systems (DBMS) in order to find logic bugs in their implementation. We refer to logic bugs as those bugs that cause the DBMS to fetch an incorrect result set (e.g., by omitting a record).

SQLancer operates in the following two phases:

1. Database generation: The goal of this phase is to create a populated database, and stress the DBMS to increase the probability of causing an inconsistent database state that could be detected subsequently. First, random tables are created. Then, randomly SQL statements are chosen to generate, modify, and delete data. Also other statements, such as those to create indexes as well as views and to set DBMS-specific options are sent to the DBMS.
2. Testing: The goal of this phase is to detect the logic bugs based on the generated database. See Testing Approaches below.

分布式测试（实施中 / 需调查）

- 分布式系统的测试是整个系统开发过程中重要的部分，我们需要对系统进行完整的正确性测试才能确保系统符合我们的设计
- **Jepsen** 分布式系统测试框架，正确性验证。性能和可用性图表。
- **Rust** 分布式模型检查器 <https://github.com/stateright/stateright>

Here are some related materials:

- FoundationDB
 - [Simulation and Testing](#)
 - [FoundationDB or: How I Learned to Stop Worrying and Trust the Database \(Markus Pilman, Snowflake\)](#)
- [sled simulation guide \(jepsen-proof engineering\)](#)
- <https://github.com/tokio-rs/simulation>
- <https://github.com/madsys-dev/madsim>

Unorganized materials:

- <https://github.com/osrg/namazu>
- <https://github.com/rr-debugger/rr>
- <https://github.com/stateright/stateright>

讨论环节

完备的测试和健壮的工作流支撑 Databend 半年以来的快速迭代，本次分享将会聚焦 Databend 的测试工作

演讲者：尚卓燃

演讲时间：2021.10.21