

今回取得するメトリクス

- 対象：2016年2月 時点で Rockstar である 516人
 - Rockstar：GitHubでのフォロワーが1000人以上
- 月毎に以下のデータを取得

メトリクス名	内容
フォロワー数	他の開発者にフォローされた数
プルリクエスト数	自分のものではないリポジトリに投稿したプルリクエストの回数
課題報告数	自分のものではないリポジトリに投稿した課題報告の回数

- プルリクエスト
 - 機能追加や改修など、作業内容をレビュー・マージ担当者やその他関係者に通知する機能

今後追加してもいいと思うメトリクス

- ・掘り下げる感じで…

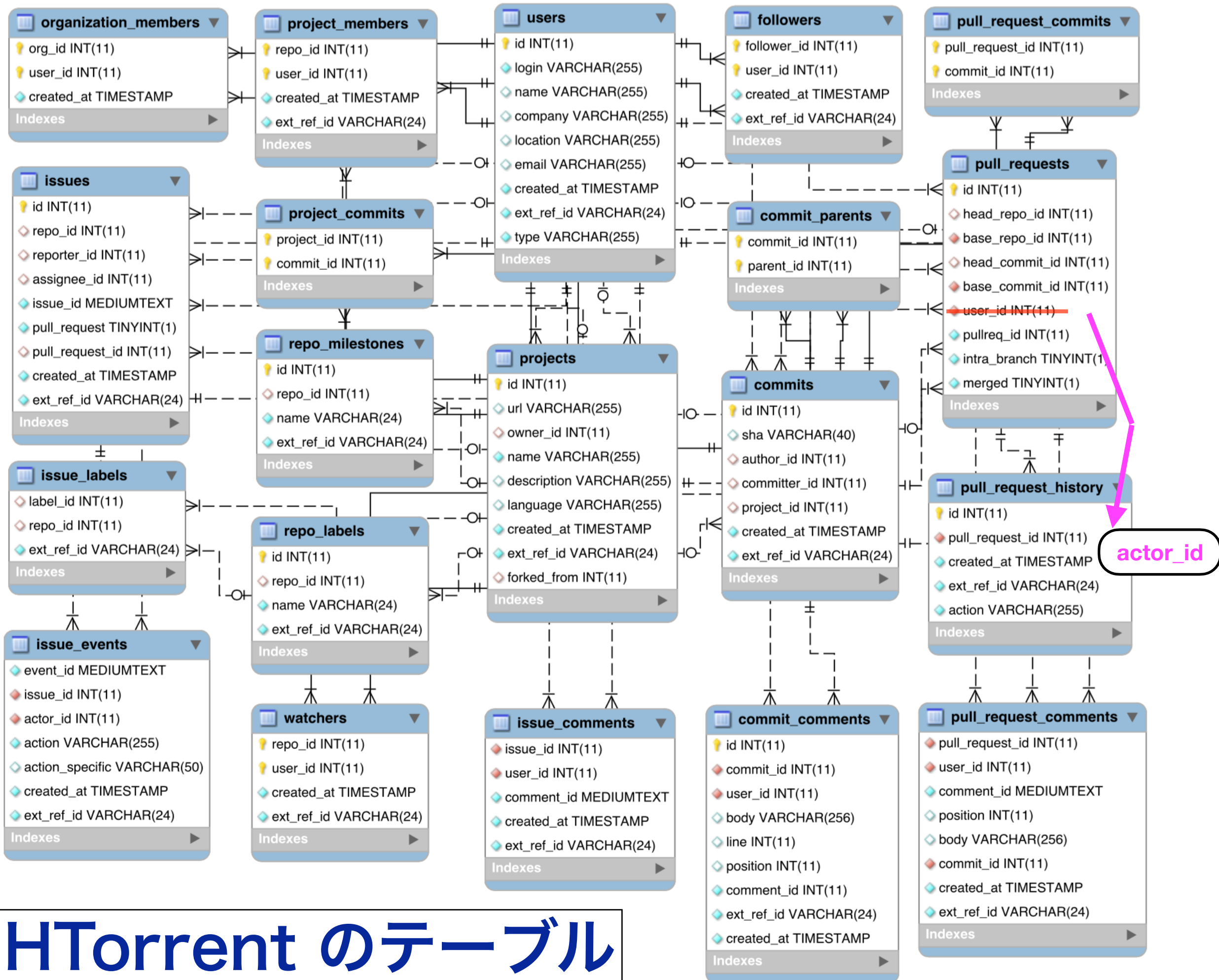
メトリクス名	内容
フォロワー数	他の開発者にフォローされた数 (誰にフォローされているか)
プルリクエスト数	自分のものではないリポジトリに投稿したPRの回数 どれくらい採用されているか？ どのリポジトリにしたか？ どのリポジトリにどれくらい採用されているか？ コメントの有無
課題報告数	自分のものではないリポジトリに投稿した課題報告の回数 どのリポジトリにしたか？ すぐにclosedされてないか？ (ちゃんとしたもの)

- ・他は, この3つを取って結果を見てから考えます
 - 掘り下げるのかどうかも踏まえて

GHTorrentのテーブル

GitHubについて理解できていないことも多いので、
共有させておいてください。

データの取得方法, テーブルの意味の理解など
間違いに気づいたら適宜教えてください。



GHTorrent のテーブル

フォロワー数取得方法（今回取得するメトリクス）

- ・ フォロワーは取得済み

- followers

```
mysql> select * from followers limit 1;
```

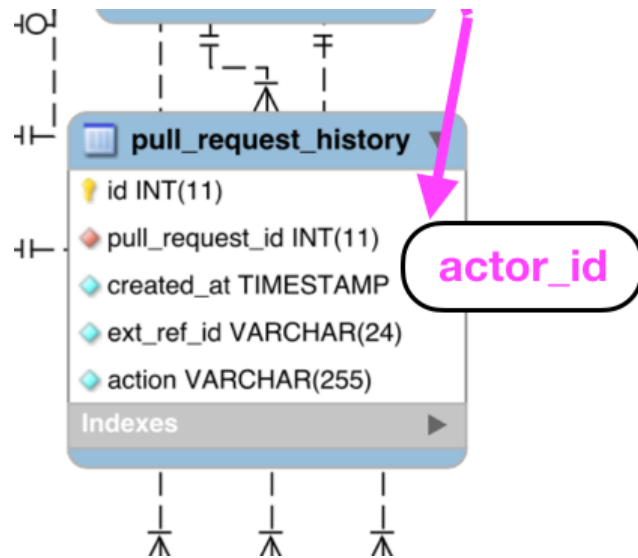
+	-----	+	-----	+	-----	+
	follower_id		user_id		created_at	
+	-----	+	-----	+	-----	+
	1		2		2008-12-15 12:28:33	
+	-----	+	-----	+	-----	+

中身も割と単純

- ・ follower_id：フォローしている
- ・ user_id：フォロされている

プルリクエスト取得方法（今回取得するメトリクス）

- ・ プルリクエスト数
 - テーブル：pull_request_history



[sql文]

```
select count(*) from pull_request_history
where action = opened
and actor_id = ?
and created_at like ? ;
```

actionをopenedしているactor_idのプル
リクエスト回数を取得（月毎）

・ pull_request_history

テーブルの中身はこんな感じ

```
mysql> select * from pull_request_history limit 1;
```

id	pull_request_id	created_at	action	actor_id
15017368	352984	2012-09-04 14:17:50	opened	974

外部の開発者（今回はこれに注目）

内部（権限有）の開発者

pull_request_history

An event in the pull request lifetime

The **action** field can take the following values

- **opened**: When the pull request has been opened
- **closed**: When the pull request has been closed
- **merged**: When Github detected that the pull request has been merged. No merges outside Github (i.e. Git based) are reported
- **reopened**: When a pull request is opened after being closed
- **synchronize**: When new commits are added/removed to the head repository

action
opened
closed
merged
reopened
synchronize

課題報告数取得方法（今回取得するメトリクス）

- 課題報告数
 - テーブル：issues

- issues

```
mysql> select * from issues limit 1;
```

id	repo_id	reporter_id	assignee_id	pull_request	pull_request_id	created_at	issue_id
2	1334	345886	NULL	0	NULL	2012-01-19 05:24:55	4535

- 詳しく見たい場合
 - テーブル：issue_events

- issue_events

```
mysql> select * from issue_events limit 1;
```

event_id	issue_id	actor_id	action	action_specific	created_at
8131201	2	345886	subscribed	NULL	2012-01-19 05:24:55

どんな課題だったか、経緯など
詳しい内容がわかる
(すぐにclosedされていないか)

PRコメントの有無 / 採用されたかどうか（今後取得するメトリクス）

- ・ プルリクエストコメント数
 - テーブル：pull_request_comments

- pull_request_comments

```
mysql> select * from pull_request_comments limit 1;
```

```
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
| pull_request_id | user_id | comment_id | position | body
| commit_id | created_at          |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
|          16311 |    66821 | 1343794    |         6 | I think auto-submitting to cause a full page refresh would be a pretty bad
experience, so I'd probably prefer they had to trigger the submit manually. |    3189885 | 2012-08-09 14:02:33 |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

プルリクエストコメント
あったかどうかも見れそう

- ・ 採用されたかどうか
 - テーブル：pull_request_commits

- pull_request_commits

```
mysql> select * from pull_request_commits limit 1;
```

```
+-----+-----+
| pull_request_id | commit_id |
+-----+-----+
|          6350   | 1113212   |
+-----+-----+
```

採用されたかどうかも
確認出来そう

まとめ

取得するデータの表

メトリクス名	内容
フォロワー数	他の開発者にフォローされた数
プルリクエスト数	自分のものではないリポジトリに投稿したプルリクエストの回数
課題報告数	自分のものではないリポジトリに投稿した課題報告の回数