CREATE TABLE pull\_request\_comments\_short

SELECT A.pull\_request\_id, A.user\_id, A.comment\_id, A.position, A.body, A.commit\_id, A.created\_at

FROM `pull\_request\_comments` A

INNER JOIN ghtorrent\_short.pull\_requests B ON A.pull\_request\_id=B.pullreq\_id;

CREATE TABLE regression\_dataset

SELECT pull\_id, project\_id, commit\_count, num\_changed\_files, additions, deletions, (additions+ deletions) AS total\_churn,

created\_at AS created, total\_comments, CASE WHEN merged\_at IS NULL THEN 0 ELSE 1 END AS is\_accepted

FROM `pull\_details`;

ALTER TABLE `regression\_dataset` ADD `review\_interval\_hours` FLOAT DEFAULT '0' AFTER `is\_accepted`

UPDATE regression\_dataset INNER JOIN pull\_details on

regression\_dataset.pull\_id=pull\_details.pull\_id

SET regression\_dataset.review\_interval\_hours = ROUND(TIMESTAMPDIFF(SECOND, pull\_details.created\_at,pull\_details.closed\_at)/3600,2);

CREATE TABLE pull\_stats

SELECT pull\_id, AVG(comment\_volume) as avg\_comment\_volume,

AVG(complexity) AS avg\_complexity FROM `pull\_files` GROUP BY pull\_id;

ALTER TABLE `regression\_dataset` ADD `avg\_comment\_volume` FLOAT NOT NULL DEFAULT '0' AFTER `review\_interval\_hours`, ADD `avg\_complexity` FLOAT NOT NULL DEFAULT '0' AFTER `avg\_comment\_volume`;

UPDATE regression\_dataset INNER JOIN pull\_stats on regression\_dataset.pull\_id=pull\_stats.pull\_id

SET regression\_dataset.avg\_comment\_volume=pull\_stats.avg\_comment\_volume, regression\_dataset.avg\_complexity=pull\_stats.avg\_complexity;

ALTER TABLE `regression\_dataset` ADD `owner` INT NULL DEFAULT NULL AFTER `project\_id`;

UPDATE regression\_dataset INNER JOIN pull\_request\_history on regression\_dataset.pull\_id=pull\_request\_history.pull\_request\_id and pull\_request\_history.action='opened'

SET regression\_dataset.owner=pull\_request\_history.actor\_id;

DELETE FROM regression\_dataset WHERE owner IS NULL;

CREATE TABLE first\_commits

SELECT project\_id, committer\_id, MIN(created\_at) as first\_commit

FROM `commits`

WHERE commits.created\_at>'2000-01-01' and commits.project\_id IS NOT NULL AND commits.project\_id<>0

GROUP BY project\_id, committer\_id;

CREATE TABLE first\_pull\_by\_person

SELECT pull\_request\_history.actor\_id, pull\_requests.base\_repo\_id, MIN(pull\_request\_history.created\_at) as fist\_pull\_date

FROM `pull\_request\_history` INNER JOIN pull\_requests on

pull\_requests.id=pull\_request\_history.pull\_request\_id and pull\_request\_history.action='opened'

GROUP BY pull\_request\_history.actor\_id, pull\_requests.base\_repo\_id;

UPDATE regression\_dataset INNER JOIN first\_commits on regression\_dataset.project\_id=first\_commits.project\_id

and regression\_dataset.owner=first\_commits.committer\_id

SET regression\_dataset.author\_tenure\_months=FLOOR(TIMESTAMPDIFF(DAY, first\_commits.first\_commit,regression\_dataset.created)/30)+1

UPDATE regression\_dataset INNER JOIN issues on regression\_dataset.pull\_id=issues.pull\_request\_id

inner JOIN issue\_labels on issues.id=issue\_labels.issue\_id

INNER JOIN repo\_labels on issue\_labels.label\_id=repo\_labels.id and repo\_labels.is\_bug=1

SET regression\_dataset.is\_bugfix=1

CREATE TABLE issue\_events\_short

SELECT issue\_events.\* FROM `issue\_events` INNER JOIN ghtorrent\_short.issues on issue\_events.issue\_id=ghtorrent\_short.issues.id;