CREATE TABLE person\_first\_commit

SELECT MIN(created) as first\_commit\_date, owner as person\_id FROM `request\_detail` GROUP BY owner;

ALTER TABLE `person\_first\_commit` ADD PRIMARY KEY(`person\_id`);

CREATE TABLE regression\_dataset

SELECT request\_detail.request\_id, request\_detail.created, request\_detail.insertions, request\_detail.deletions,request\_detail.insertions+ request\_detail.deletions as total\_churn,

request\_detail.number\_patches, CASE WHEN status='MERGED' THEN 1 ELSE 0 END as is\_accepted,

request\_detail.owner,

ROUND(TIMESTAMPDIFF(SECOND, created,updated)/3600,2) as review\_interval\_hours

from request\_detail;

ALTER TABLE `regression\_dataset` ADD INDEX(`request\_id`);

ALTER TABLE `regression\_dataset` ADD INDEX(`owner`);

ALTER TABLE `regression\_dataset` ADD `author\_tenure\_months` INT NULL DEFAULT NULL AFTER `is\_accepted`;

UPDATE regression\_dataset INNER JOIN request\_detail on regression\_dataset.request\_id=request\_detail.request\_id

INNER JOIN person\_first\_commit on person\_first\_commit.person\_id=request\_detail.owner

SET regression\_dataset.author\_tenure\_months = FLOOR(TIMESTAMPDIFF(DAY, person\_first\_commit.first\_commit\_date,request\_detail.created)/30)+1;

ALTER TABLE `patch\_details` ADD `directory` VARCHAR(1024) NULL AFTER `file\_name`;

UPDATE patch\_details SET directory= substring(file\_name,1,locate(substring\_index(file\_name,'/',-1),file\_name)-2);

CREATE TABLE file\_count SELECT COUNT(file\_name) as file\_count, request\_detail.request\_id FROM `patch\_details`

INNER JOIN request\_detail on

request\_detail.request\_id=patch\_details.request\_id and

request\_detail.current\_patch\_id=patch\_details.patchset\_id

GROUP by request\_detail.request\_id;

ALTER TABLE `file\_count` ADD PRIMARY KEY(`request\_id`);

CREATE TABLE directory\_count SELECT COUNT( DISTINCT `directory`) as dir\_count, request\_detail.request\_id FROM `patch\_details`

INNER JOIN request\_detail on

request\_detail.request\_id=patch\_details.request\_id and

request\_detail.current\_patch\_id=patch\_details.patchset\_id

GROUP by request\_detail.request\_id;

ALTER TABLE `directory\_count` ADD INDEX(`request\_id`);

ALTER TABLE `regression\_dataset` ADD `file\_count` INT NOT NULL DEFAULT '1' AFTER `review\_interval\_hours`, ADD `directory\_count` INT NOT NULL DEFAULT '1' AFTER `file\_count`;

UPDATE regression\_dataset INNER JOIN file\_count on regression\_dataset.request\_id=file\_count.request\_id

SET regression\_dataset.file\_count=file\_count.file\_count;

UPDATE regression\_dataset INNER JOIN directory\_count on regression\_dataset.request\_id=directory\_count.request\_id

SET regression\_dataset.directory\_count=directory\_count.dir\_count;

DROP TABLE directory\_count;

DROP TABLE file\_count;

ALTER TABLE `regression\_dataset` ADD `is\_bugfix` INT NOT NULL DEFAULT '0' AFTER `is\_accepted`;

***– Following three queries must be executed together.***

*20. CREATE TEMPORARY TABLE bugfixes SELECT DISTINCT patches.request\_id FROM patches WHERE patches.message like '%[BUGFIX]%';*

*ALTER TABLE `bugfixes` ADD INDEX(`request\_id`);*

*UPDATE regression\_dataset INNER JOIN bugfixes on regression\_dataset.request\_id=bugfixes.request\_id SET regression\_dataset.is\_bugfix=1;*

ALTER TABLE `regression\_dataset` ADD `num\_new\_files` INT NOT NULL DEFAULT '0' AFTER `file\_count`;

**– Following three queries must be executed together.**

CREATE TEMPORARY TABLE new\_files

SELECT COUNT(file\_name) as new\_file\_count, request\_detail.request\_id FROM `patch\_details` INNER JOIN request\_detail on

request\_detail.request\_id=patch\_details.request\_id

and request\_detail.current\_patch\_id=patch\_details.patchset\_id and change\_type='A'

GROUP BY request\_detail.request\_id;

ALTER TABLE `new\_files` ADD INDEX(`request\_id`);

UPDATE regression\_dataset INNER JOIN new\_files on regression\_dataset.request\_id=new\_files.request\_id

SET regression\_dataset.num\_new\_files=new\_files.new\_file\_count;

ALTER TABLE `regression\_dataset` ADD `code\_commit\_exp` INT NOT NULL DEFAULT '0' AFTER `owner`, ADD `code\_review\_exp` INT NULL DEFAULT '0' AFTER `code\_commit\_exp`;

CREATE TABLE prior\_commit\_count

SELECT regression\_dataset.request\_id, COUNT(DISTINCT(request\_detail.request\_id)) as commit\_count FROM `regression\_dataset`

INNER JOIN request\_detail on request\_detail.owner=regression\_dataset.owner and request\_detail.request\_id<regression\_dataset.request\_id

GROUP BY regression\_dataset.request\_id;

ALTER TABLE `prior\_commit\_count` ADD INDEX(`request\_id`);

UPDATE regression\_dataset INNER JOIN prior\_commit\_count on regression\_dataset.request\_id=prior\_commit\_count.request\_id

SET regression\_dataset.code\_commit\_exp=prior\_commit\_count.commit\_count;

CREATE TABLE prior\_review\_count

SELECT regression\_dataset.request\_id, COUNT(DISTINCT(review\_comments.request\_id)) as review\_count FROM regression\_dataset

INNER JOIN review\_comments on regression\_dataset.owner=review\_comments.author

INNER JOIN request\_detail ON request\_detail.request\_id=review\_comments.request\_id and request\_detail.owner<>regression\_dataset.owner

and request\_detail.request\_id<regression\_dataset.request\_id

GROUP by regression\_dataset.request\_id;

INSERT INTO prior\_review\_count

SELECT regression\_dataset.request\_id, COUNT(DISTINCT(review\_comments.request\_id)) as review\_count FROM regression\_dataset

INNER JOIN review\_comments on regression\_dataset.owner=review\_comments.author and review\_comments.is\_response=0 and regression\_dataset.request\_id<1000000 and regression\_dataset.request\_id>=300000

and review\_comments.request\_id<regression\_dataset.request\_id

GROUP by regression\_dataset.request\_id;

ALTER TABLE `prior\_review\_count` ADD INDEX(`request\_id`);

UPDATE regression\_dataset INNER JOIN prior\_review\_count on regression\_dataset.request\_id=prior\_review\_count.request\_id

SET regression\_dataset.code\_review\_exp=prior\_review\_count.review\_count;

DROP TABLE prior\_review\_count;

DROP TABLE prior\_commit\_count;

ALTER TABLE `file\_details` ADD `comment\_volume` FLOAT NOT NULL DEFAULT '0' AFTER `SRC\_count`;

UPDATE file\_details SET comment\_volume = SRC\_count/LOC\_count where LOC\_count <>0;

ALTER TABLE `regression\_dataset` ADD `avg\_comment\_volume` FLOAT NOT NULL DEFAULT '0' AFTER `directory\_count`;

ALTER TABLE `regression\_dataset` ADD `avg\_complexity` FLOAT NOT NULL DEFAULT '0' AFTER `directory\_count`;

– Next three together

CREATE TEMPORARY TABLE temp\_metrics SELECT request\_id, AVG(comment\_volume) as avg\_comment\_volume, AVG(complexity) as avg\_complexity FROM `file\_details` GROUP BY request\_id;

ALTER TABLE `temp\_metrics` ADD INDEX(`request\_id`);

UPDATE regression\_dataset INNER JOIN temp\_metrics on regression\_dataset.request\_id=temp\_metrics.request\_id SET regression\_dataset.avg\_complexity=temp\_metrics.avg\_complexity, regression\_dataset.avg\_comment\_volume=temp\_metrics.avg\_comment\_volume;