Terezie Schaller
Spring 2017 CS 340
June 7, 2017
Final Project

Signature Solar - Parts and Issues Tracking Database

Outline

Project miniworld description:

Signature Solar Technology is a group of mechanical engineers that design automated equipment for manufacturing solar panels. With each new design, a prototype must be built and tested before the new product can be made available for customers. Currently, the group of engineers is small (less than 10 people) so all group members contribute to a system of shared spreadsheets to track the tests and issues. However, as the group grows in size, there may be problems keeping the spreadsheets updated, avoiding redundancies, and securing the data. This database will track the testing and issue report/resolution process.

Examples of solar manufacturing equipment similar to what is being designed by the group here:

http://www.appliedmaterials.com/solar

The main function of the database is to ensure that high priority issues and critical path parts are given appropriate project management resources and that the engineering managers have an evenly distributed workload.

Description of entities

Owner - This is the Senior Design Engineer responsible for designing the part & guiding it through the testing. This is different than employee. The database is not intended to track employees, only the lead engineer for each part.

Part - This is an individual mechanical component such as a lid, lift, valve, chamber, etc. that is being tested.

Test - This is a test performed on a part.

Issue - This is a problem with a part or group of parts, the whole product.

Database Outline

Owners can have 0 to many parts.
Owner key attribute is owner ID.
Other owner attributes are last name and first name.

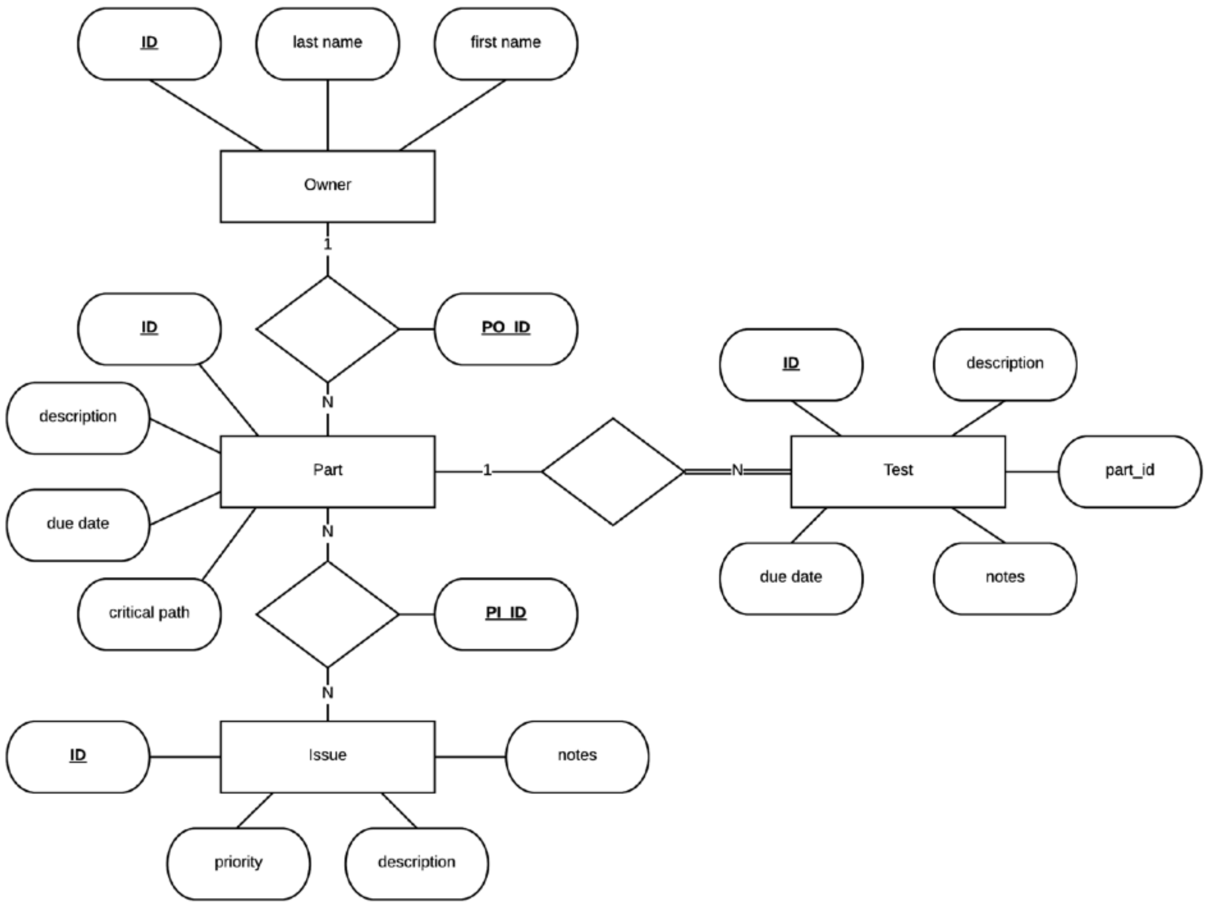
Parts can have 0 or 1 owners.
Parts key attribute is ID.
Other part attributes are due date, desc, and critical path (y/n).

Issues can have 0 to many parts.
Parts can have 0 to many issues.
Issue key attribute is ID.
Other issue attributes are priority, description, and notes.

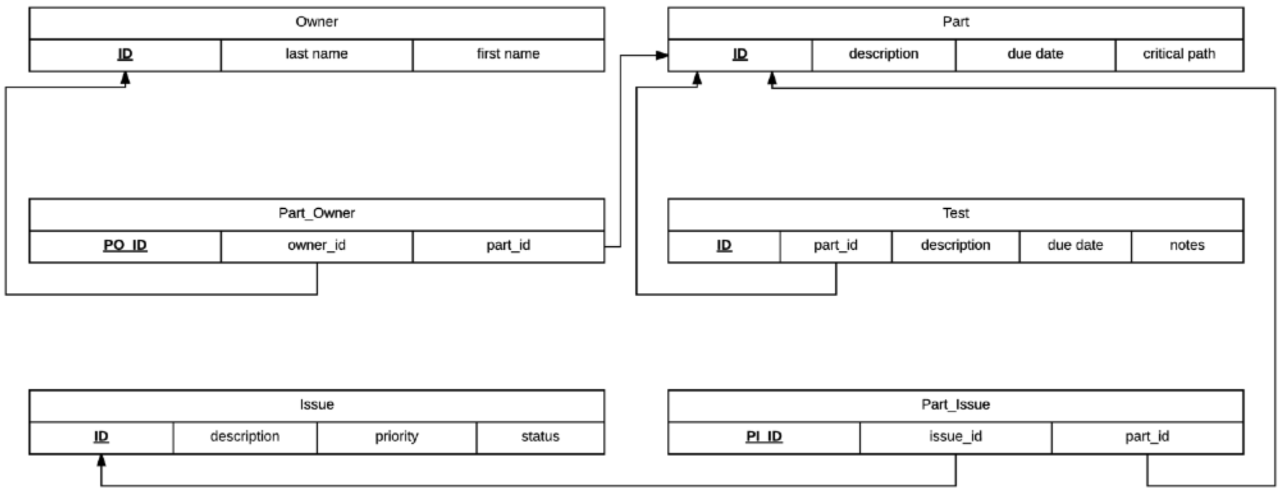
Parts can have many tests.
Tests must have 1 part, and only 1 part.
Test key attribute is ID.
Other test attributes are description, due_date, notes.

Part-owner relationship has a key attribute po_id.
Part-issue relationship has a key attribute pi_id.

ER Diagram



Schema



Data Definition Queries

```
DROP TABLE IF EXISTS `part_issue`;
DROP TABLE IF EXISTS `issue`;
DROP TABLE IF EXISTS `test`;
DROP TABLE IF EXISTS `part_owner`;
DROP TABLE IF EXISTS `part`;
DROP TABLE IF EXISTS `owner`;

# owner
CREATE TABLE `owner` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `last_name` varchar(255) NOT NULL,
  `first_name` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

# part
CREATE TABLE `part` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `desc` varchar(255) NOT NULL,
  `due_date` date,
  `critical_path` ENUM ('y','n'),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
# part_owner
CREATE TABLE `part_owner` (
  `po_id` int(11) AUTO_INCREMENT,
  `part_id` int(11),
  `owner_id` int(11),
  PRIMARY KEY(`po_id`),
  FOREIGN KEY (`part_id`) REFERENCES `part` (`id`) ON DELETE CASCADE,
  FOREIGN KEY (`owner_id`) REFERENCES `owner` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

# test
CREATE TABLE `test`(
  `id` int(11) AUTO_INCREMENT,
  `part_id` int(11) NOT NULL,
  `desc` varchar(255) NOT NULL,
  `due_date` date,
  `note` text,
  PRIMARY KEY (`id`),
  FOREIGN KEY (`part_id`) REFERENCES `part` (`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

# issue
CREATE TABLE `issue` (
  `id` int(11) AUTO_INCREMENT,
  `desc` varchar(255) NOT NULL,
  `priority` ENUM('1','2','3','4','5'),
  `status` varchar(255),
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

# part_issue
/* maybe this should cascade?? Do we want  */
CREATE TABLE `part_issue` (
  `pi_id` int(11) AUTO_INCREMENT,
  `part_id` int(11),
  `issue_id` int(11),
  PRIMARY KEY(`pi_id`),
  FOREIGN KEY (`part_id`) REFERENCES `part`(`id`) ON DELETE CASCADE,
  FOREIGN KEY (`issue_id`) REFERENCES `issue`(`id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/* seeding tables with sample data */
# owner data
INSERT INTO `owner` (`last_name`, `first_name`) VALUES
("Arendelle","Elsa"),("Beauty","Belle"),
("Lightyear","Buzz"),("McQueen","Lightning");
```

```
# part data
INSERT INTO `part` (`desc`, `due_date`, `critical_path`) VALUES
("gripper", 20170701, "Y"),("shutter", 20170630, "Y"),
("load lock", 20170615, "Y"),("lift", 20170913, "N"),
("end affector", 20171210, "N"),("deceleration lens", 20180215, "Y");

# part_owner data
INSERT INTO `part_owner` (`part_id`, `owner_id`) VALUES
(1,1),(2,2),(3,2),(4,4),(5,3),(6,1);

# test data
INSERT INTO `test` (`desc`, `part_id`, `due_date`, `note`) VALUES
("cluster tool processing", 1, 20170722, "in progress"),
("suction system test", 2, 20171010, "waiting for vacuume module"),
("robot optimization", 5, 20171112, "don arigato mr roboto"),
("wafer handling analysis", 1, 20180111, "due next year"),
("load lock calibration", 3, 20170606, "lock and load!"),
("dynamic lift balancing", 6, 20171013, "contact supplier: 512-222-7876"),
("pitch adjustment", 6, 20170923, "n/a"),
("robot calibration", 4, 20170820, "waiting on software group"),
("linkage evaluation", 3, 20170516, "you are a rock star!!");

# issue data
INSERT INTO `issue` (`desc`,`priority`,`status`) VALUES
("unbalanced load", 1, "in progress"),
("no charge", 2, "waiting"),
("out of calibration", 3, "on hold"),
("squeaking noise", 1, "is there a mouse?"),
("no seal", 2, "checking lid gasket"),
("breaking wafers", 4, "in progress"),
("out of alignment", 5, "waiting on balancing tool"),
("excessive vibration", 3, "dampener on order"),
("overheating", 4, "new coolant pump on order");

# part_issue
INSERT INTO `part_issue` (`part_id`,`issue_id`) VALUES
(1,1),(1,6),(1,7),(2,8),(3,5),
(4,4),(4,7),(5,1),(5,4),(5,8),
(6,9),(6,7),(6,2),(6,3);
```

Data Manipulation Queries

```
/* show tables helper function */
SELECT * from [userTable]

/* add */
#add owner
INSERT INTO owner (`first_name`, `last_name`)
VALUES ([userFname], [userLname])

#add part
INSERT INTO part (`desc`, `due_date`, `critical_path`)
VALUES ([userDesc], [userDate], [userYN])
INSERT INTO part_owner (`part_id`, `owner_id`)
VALUES ((SELECT MAX(id) AS p_id FROM part), [owner_id])

#add test
INSERT INTO test (`part_id`,`desc`,`due_date`,`note`)
VALUES ([userPID], [userDesc], [userDD], [userNote])

#add issue
INSERT INTO issue (`desc`, `priority`, `status`)
VALUES ([userDesc], [userPriority], [userStatus])


/* delete */
# del owner
DELETE FROM owner WHERE id=[UserOID];

# del part
DELETE FROM part WHERE id=[UserPID];

# del test
DELETE FROM test WHERE id=[UserTID];

# del issue
DELETE FROM issue WHERE id=[UserIID];

/* edit */
#edit owner
/* gets database values */
SELECT last_name, first_name FROM owner
WHERE id=[ownerID]
/* if user does not supply fields then current database value is used */
UPDATE owner SET last_name=[lname], first_name=[fname]
WHERE id=[ownerID]

#edit part-owner
DELETE FROM part_owner
```

```
WHERE part_id=[userPID]
INSERT INTO part_owner (part_id, owner_id)
VALUES ([userPID], [userOID])

#edit part
/* gets database values */
SELECT `desc`, `due_date`, `critical_path`
FROM part WHERE id=[userPID]
/* if user does not supply fields then current database value is used */
UPDATE part SET `desc`=[userDesc], `due_date`=[userDD], `critical_path`=[userCP]
WHERE id=[userPID]

#edit test
/* gets database values */
SELECT `part_id`,`desc`, `due_date`, `note` FROM test
WHERE id=[userTID]
/* if user does not supply fields then
current database value is used */
UPDATE test SET `part_id`=[userPID], `desc`=[userdesc], `due_date`=[userDD],
`note`=[userNote]
WHERE id=[userTID]

#edit part-issue
INSERT INTO part_issue (part_id, issue_id)
VALUES ([userPID], [userIID])

#edit issue
/* gets database values */
SELECT `desc`, `priority`, `status` FROM issue
WHERE id=[userIID]
/* if user does not supply fields then
current database value is used */
UPDATE issue SET `desc`=[userDesc], `priority`=[userPriority], `status`=[userStatus]
WHERE id=[userIID]

/* complex queries, searches */

/* find Owners and part info with critical
path parts sorted by due date */
SELECT o.last_name, o.first_name, p.desc, p.due_date
FROM owner o
INNER JOIN part_owner po ON o.id = po.owner_id
INNER JOIN part p ON po.part_id = p.id
WHERE p.critical_path = 'y'
ORDER BY p.due_date
```

```
/* find owners with no parts */
SELECT id, last_name, first_name FROM owner WHERE id NOT IN
  ((SELECT id FROM owner o
  INNER JOIN part_owner po ON o.id = po.owner_id))

/* find parts and issues, sort by due date then priority */
SELECT p.id, p.desc, p.due_date, i.desc, i.priority
FROM part p
INNER JOIN part_issue p_i ON p.id = p_i.part_id
INNER JOIN issue i ON p_i.issue_id = i.id
ORDER BY p.due_date, i.priority
```

Issues/Resolution

(This was not required in the assignment description, but I like to keep track of it when working on a large project. I thought I would include it in this report.)

1) I had a really hard time getting started. I had no idea what this was supposed to be or where to attack it. Resolution: I made a simple 1 table database with node route handlers that would add, delete, and edit. Wrote out diagrams, schema. Made more complicated multi table database. Expanded simple node template to fit more complicated database

2) While writing my node.js code, flip3 was not working. I thought the problem was me so spent a huge amount of time trouble shooting my own code before posting on Piazza to find out that it wasn't just me. Other people were having similar problems. Continued to have problems intermittently with flip3 & flip2. Resolution: downloaded node and worked on project locally.

3) I had a hard time working with foreign keys and delete. At first I thought there was a problem with my node.js code or my MySQL query. It turns out I needed to use on delete cascade.

4) I kept losing part-owner relationships on update when I wasn't supposed to. Had to add po_id and make that the primary key instead of using part_id + owner_id as a composite primary key. Also added pi_id to avoid the same problem.

5) I am having a hard time with front end in general. I should have partnered with someone with some front end experience. If I have time this summer, I hope to learn some jQuery and improve on the front end of this project a little.

6) I apologize to whoever ends up grading this for the clunky user interface. I could not get drop down menus or tables working well at all, despite trying a bunch of things. Resolution: I got the data to display in string form along with messages about what was happening. I decided to work on the database, queries, and node.js, and leave the front end part for last. However, I could easily spend 1-2 weeks messing with it and not get it right. I began working on this as soon as the project description was released, and I estimate that I have already worked on it for 90-100 hrs. My elementary school aged kids are already out of school for summer, and I still need to study for the final exam. Since I am now pressed for time, I decided to turn in the working test version and hope that I won't loose too many points for style.

References

Books

Brown, Ethan. Web Development with Node & Express. O'reilly, 2014.

Cantelon, Mike, et al. Node.js in Action. Manning, 2014.

Churcher, Clare. Beginning Database Design. Apress, 2007.

DuBois, Paul. MySQL Cookbook. O'reilly, 2014.

Webpages

- https://www.youtube.com/watch?v=hGZX_SA7IYg
- https://www.terlici.com/2015/08/13/mysql-node-express.html
- https://webapplog.com/handlebars/
- https://druckit.wordpress.com/2014/05/28/node-js-101-querying-a-database-and-outputting-a-dynamic-html-table-with-express-and-jade/
- https://www.youtube.com/watch?v=wSNa5b1mS5Y
- https://www3.ntu.edu.sg/home/ehchua/programming/sql/SampleDatabases.html
- http://www.mysamplecode.com/2012/04/generate-html-table-using-javascript.html
- http://www.encodedna.com/javascript/populate-json-data-to-html-table-using-javascript.htm
- https://stackoverflow.com/questions/26218243/dynamic-dropdown-in-node-js
- https://dev.mysql.com/doc/refman/5.7/en/create-table-foreign-keys.html
- https://stackoverflow.com/questions/838354/mysql-removing-some-foreign-keys
- https://dev.mysql.com/doc/refman/5.6/en/create-table-foreign-keys.html
- https://stackoverflow.com/questions/2973558/select-a-value-where-it-doesnt-exist-in-another-table