mechanics need databases too

Prepared by Group 14: Heather Fillerup- Software Developer, Chris Nelson- Software Developer

CS 340-400: Spr 2020

Project- Step 7 Final

June 2, 2020

http://flip3.engr.oregonstate.edu:5455/home

## Executive Summary

* The TA recommended that we carefully considered the implementation of having a DELETE function for all tables. We decided to leave the delete function for all entities since we have logic on server side to catch any MySQL errors and notify the user if they cannot delete a record because of a foreign key constraint or any other reason.
* Reviewers were initially confused by what we were trying to accomplish. This helped us realize that we needed to update our repairs table to repair\_orders, which is the main tracking mechanism for the overall reason the car is in the shop, then we can add work orders (tracked in the composite table work\_orders) that records the various tasks being done to the cars, the mechanic assigned to each task and start and end dates of the task.
  + Made work\_orders a composite table with attributes moved from work\_tasks (mechanic\_id, start\_date and end\_date). This was to satisfy the requirement that when we delete our M:M task and repair orders relationship record, we cannot delete any record from the repair\_orders or work\_tasks tables. We also gave work\_orders a PK of id, instead of using the FKs as the PK
  + Changed mechanics relationship with work\_tasks (statuses). Mechanics has a M:M relationship with both repair\_orders and work\_tasks, which are both nullable.
* Removed Parts table since we only need to implement one M:M relationship

## Project Outline

Mahinui auto shop, a single location, has seen record business in the last decade, repairing 50 or more cars on any given day. With more customers coming in by the day, keeping track of records has become a nightmare. The owner, Brad, has finally decided to upgrade his repair order workflow from pen and paper being passed between his 10 mechanics to a website database. Brad is looking to create a system for his mechanics to track the tasks involved with a car's repair, from diagnosis to customer pick up, and be able to view a display of the progress on the homepage. The website will allow users to:

1. Add Customer
2. Add Car
3. Add repair order to a car
4. Add work orders to repair order
   1. Add work task to work order
      1. Diagnosis, customer approval, order parts, repair, test drive and finally contact customer
   2. Add Mechanic to work order
5. Add end date to work order to complete
6. Add new work order
7. View on the website homepage the following display of all the cars currently being repaired at the shop and the current task being performed. **This display is not intended to be part of the grading requirements, but more to show the usefulness of the database.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mahinui Auto Shop Dashboard** | | | | |
| **Customer Name** | **Car Description** | **Current Task** | **Start Date** | **Mechanic** |
| Jason Bateman | 2015 Honda Accord | Diagnosis | 03/1/2020 | Johnny |
| Charlize Theron | 2014 Toyota Civic | Customer Approval | 3/2/2020 | Ben |
| Ryan Reynolds | 2011 Honda Ridgeline | Order Parts | 3/3/2020 | Cameron |
| Scarlet Johansen | 2009 Toyota Front Runner | Repair | 3/4/2020 | Peter |
| Jeff Bridges | 2014 Fiat 500 | Test Drive | 3/5/2020 | Frank |

## Database Outline

**customers**: records details about the customers who own the cars being repaired (Heather)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* f\_name: VARCHAR, NOT NULL
* l\_name: VARCHAR, NOT NULL
* contact\_no: VARCHAR, NOT NULL
* email\_address: VARCHAR, NOT NULL
* relationship: a 1:M relationship between customers and cars is implemented with customer\_id as a FK inside of cars, where a customer can have 0 to many cars, and a car can only have one customer.

**cars:** records details about the car being repaired (Chris)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* customer\_id: INT, FK
* license\_plate: VARCHAR, NOT NULL
* model\_year: YEAR, NOT NULL
* make: VARCHAR, NOT NULL
* model\_name: VARCHAR, NOT NULL
* relationship: a 1:M relationship between cars and repair\_orders is implemented with car\_id as a FK inside of repair\_orders, were a car can have 0 or more repair orders and a repair order can have only one car ; a 1:M relationship between customers and cars is implemented with customer\_id as a FK inside of cars, where a car requires zero or one customer and a customer can have 0 or more cars

**repair\_orders:** records details about the repair order being done on a car (Heather and Chris)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* car\_id: INT, FK NOT NULL
* date\_received: DATE NOT NULL
* date\_completed: DATE
* relationship: a M:M relationship between repair\_orders and work\_tasks and a M:M relationship between repair\_orders and mechanics are both implemented with a composite table work\_orders; a 1:M relationship between cars and repair\_orders is implemented with car\_id as a FK inside of repair\_orders, where a repair order must have only 1 car, but a car can have 0 or more repairs

**work\_tasks:** records the types of tasks that can be added to repair orders, these tasks are associated to repair orders through work orders (Heather)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* name: VARCHAR, NOT NULL (Diagnosis, Customer Approval, Order Parts, Repair, Test Drive, Contact Customer)
* relationship: a M:M relationship between repair\_orders and work\_tasks and a M:M relationship between mechanics and work\_tasks are both implemented with a composite table work\_orders

**work\_orders**: composite table that records the tasks that have been added to the repair\_orders and also tracks the mechanic responsible for the work order (Heather and Chris)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* repair\_order\_id, NOT NULL FK
* order\_task\_id, NOT NULL FK
* mechanic\_id: INT, NOT FK
* start\_date: DATE NOT NULL
* end\_date: DATE

**mechanics:** records details of the mechanic responsible for the work orders (Chris)

* id: INT, AUTO\_INCREMENT, UNIQUE, NOT NULL, PK
* f\_name: VARCHAR, NOT NULL
* l\_name: VARCHAR, NOT NULL
* relationship: a 1:M relationship between mechanics and work order is implemented with mechanic\_id as a FK inside of work\_orders, where a mechanic can have 0 or more work\_orders but a work order can only have one mechanic; a M:M relationship between repair\_orders and mechanics and a M:M relationship between mechanics and work\_tasks are both implemented with a composite table work\_orders;

## Schema

**Customers** (

id,

f\_name,

l\_name,

contact\_no,

email\_address)

**cars** (

id,

customer\_id,

license\_plate,

model\_year,

make,

model\_name)

**repair\_orders** (

id,

car\_id,

date\_received,

date\_completed)

**work\_orders** (

id,

repair\_order\_id,

work\_task\_id,

mechanic\_id,

start\_date,

end\_date)

**work\_tasks** (

id,

name)

**mechanics** (

id,

f\_name,

l\_name)