Work days and times: M - F(8AM-8PM) or Sat (9AM - 1PM)

|  |  |  |  |
| --- | --- | --- | --- |
| Entity | Attribute | Data or Information |  |
| VehicleManf | vehicleManfId(pk, number, 3), manufacturerName char(50), | Honda, Nissan, Toyota, Lexus, and Infiniti | {VehicleManf} -> {name} |
| Services | serviceId(pk, number(3)), name char(50), laborHours number(3)  Repairs: 6 categories   1. Engin Services  * Belt Replacement * Engine Repair  1. Exhaust Services  * Catalytic Converter Repair * Muffler Repair  1. Electrical Services  * Alternator Repair * Power Lock Repair  1. Transmission Services  * Axle Repair * Transmission Flush  1. Tire Services  * Tire Balancing * Wheel Alignment  1. Health and Air Conditioner Services  * Compressor Repair   Maintenance:   * Oil Changes * Brake Repair * Check Engine Light Diagnostics | | These services are loosely categorized into two broad categories: a *repair service* or a *maintenance service*.  {serviceId}-> {name}  {name}->{laborHours} |
| RepairServices | serviceId(pk, fk), category char(50), | | Repair services are categorized into 6 main subcategories  {serviceId}->{ category } |
| MaintenanceServices | serviceId(pk, fk), |  |  |
| Roles | roleType(pk, char(20)10 | manager, receptionist, mechanic |  |
| ServiceCenter | CenterId(pk, number, 5),  managerId(fk)  address(varchar2, 100), phone(char, 10),  satOpen(char, 1) |  | each center has a manager who manages all employees, a receptionist, and several mechanics.  {centerId}->{managerId, address, phone}  {centerId}->{managerId}  {address}->{centerId}  {phone}->{centerId} |
|  |  |  |  |
| Employees | employeeId(9 digits, pk),  centerId(fk)  userName, password, firstname, lastName, address, email, phone, roleId(fk, not null)  startDate, endDate |  | Each employee is associated with only one service center.  Each employee can only play one role at a time (for example, a mechanic cannot also be a receptionist).  {employeeId}->{userName}  {userName}->{password}  {employeeId}->{roleId}  {employeeId}->{firstName, lastName, address, email, phone, startDate, endDate} |
| ContractEmployees | employeeId(pk, fk),  annualSalary |  | While the manager and receptionist are contract employees with fixed annual salary  {employeeId}->{salary} |
| HourlyEmployees | employeeId(pk, fk), wage | Each mechanic works no more than (50hours a week).  Check Minimum wage. | mechanics are hourly paid workers. Each center has its own hourly rate for mechanics.  {employeeId}->{wage} |
| Vacations | vacationId(pk),employeeId( fk) from, to |  | Scheduling will need to avoid double-booking or overbooking a mechanic i.e. > 50hrs, or booking them while they are on vacation.  {vacationId}->{employeeId}  {vacationId}->{from, to} |
|  |  | |  |
| ServicePricedByManf | VehicleManf(fk), serviceId( fk), centerId(fk), ratePrice |  | Each center offers a list of car services.  Each service also has a price and a time estimated for the service job, which is based on the car and the specific auto center.  {VehicleManf, serviceId, centerId}->{ratePrice} |
|  |  |  |  |
| Schedules | scheduleId(pk), scheduleType (char(1)), centerId(fk), price |  | Maintenance services are usually provided in “bundles'' called schedules. Schedules include a set of individual services in a bundle that are completed together and are priced as a bundle.  {type, centerId}->{price} |
|  |  |  |  |
| ScheduledServices | scheduleId(fk) serviceId(fk) |  | Each schedule has a specific subset of services that are covered and there is a downward inclusion relationship so that Schedule B contains all the services in A and some extra ones and likewise Schedule C contains everything in B and some additional ones. As with the individual services, the pricing estimate is dependent on the type of car |
|  |  |  |  |
| ServiceEvents | eventId(pk), vehicleId(fk), mechanicId(fk),  totalPrice, totalPaid  completed  scheduleType | there are always at least 3 mechanics present at any given time. | For each customer service event, the services rendered, by which mechanics and total amount charged and total amount paid are recorded.  Each service event (all services scheduled by a customer in a single visit to the interface) is handled by a single mechanic.  {eventId} -> {vehicleid}  {eventId}-> {mechanicId}  {eventId}->{totalprice, totalPaid} |
| EventOnServices, not sure | Id(pk)  eventId(fk),  serviceId(fk)  serviceType |  | For each service event/visit, a customer selects what services they desire (it can be a combination of repair services and maintenance services). |
|  |  |  |  |
| Customers | customerId(pk),centerId( fk, not null), firstName, lastName, address, active, status |  | Every customer is associated with one specific service store.  {customerId}-{firstName, lastName, address, active, status}  {customerId}->{centerId} |
| CustomerVehicles(Owned) | vin(pk, 8chars)  customerId (fk), vehicleManfId(fk), year, mileage, class | , the last scheduled maintenance service class performed denoted by a single character ‘A’, ‘B’, or ‘C’. | A customer is associated with at least one vehicle which is identified by globally unique *vin number* (8 alphanumeric characters), *car manufacture*r e.g. Honda, and *current mileage* (integer), *year*  *{vin}->{cuatomeId, vehicleManfId}, {vin}->{year, class, mileage, class}* |
| Invoices | invoiceId(pk),  serviceEventId(fk) serviceDate, totalPrice, status(boolean), totalPaid | An invoice also has a *status* attribute recorded as a boolean field (*0* - unpaid, *1* - paid). By default when an invoice is created the *status* is unpaid (*0*).  Invoice entity is finalized information. It needs to store additional information to prevent the possibility for the entity values to be updated at some point. | For each service event, an invoice with unique invoice id, customer id, vin of car serviced, date of service, services provided, cost for each service, mechanics that provided the service and total bill (sum of costs) is stored.  {invoiceId}->{serviceDate, totalPrice, status}  {invoiceId}->{serviceEventId} |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Serviced | servicedId(pk)  invoiceId(pk, fk),  serviceName, category, cost,  on delete cascade |  |  |