Recruitment Module – JavaScript

# 1. Schema

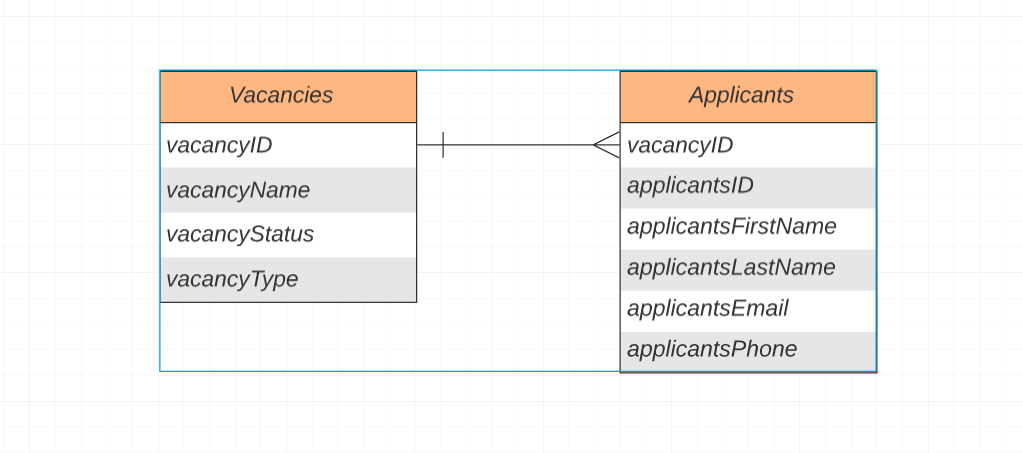
## 1.1 Recruitment Database

Design a database scheme of Recruitment module and attach an Entity Relationship Diagram.

The database should provide:

1. A possibility to create vacancies and applicants
2. There should be an option to create applicants related to multiple vacancies
3. There should be an ability to store a vacancy Name, Status, Type
4. There should be an ability to store applicant FirstName, LastName, Email, Phone

Solution:



CREATE TABLE IF NOT EXISTS VACANCIES(

vacancyID INT NOT NULL AUTO\_INCREMENT,

vacancyName varchar(40),

vacancyStatus boolean,

vacancyType varchar (60),

PRIMARY KEY vacancyID

);

CREATE TABLE IF NOT EXISTS APPLICANTS(

vacancyID int is not null,

applicantID INT NOT NULL AUTO\_INCREMENT,

applicantFirstName varchar(40),

applicantLastName varchar(60),

applicantEmail(255),

applicantPhone int

);

INSERT INTO VACANCIES (vacancyName,vacancyStatus,vacancyType)

VALUES ("Pupkin", true, "HR");

INSERT INTO APPLICANTS(vacanyID,applicantFirstName,applicantLastName,applicantEmail,applicantPhone)

VALUES(1,"Kolia", "Pupiskin", "kpupiskin@gmail.com", 096123456);

## 1.2 SQL

Based on the previously built database schema create SQL queries to retrieve records:

1. Select vacancies with "Active" Status and "HR" Type
2. Select the number of vacancies Types
3. Select applicants that applied for more than 5 vacancies

Solution:

1.

SELECT \* vacancyName

FROM VACANCIES

WHERE vacancyStatus ="ACTIVE" AND vacancyType = "HR";

2.

SELECT DISTINCT vacancyType COUNT(\*)

FROM VACANCIES

GROUP BY vacancyType;

3.

# SELECT T2.vacancyID, T1.applicantFirstName

# FROM APPLICANTS T1

# LEFT JOIN VACANCIES T2 ON T1.vacancyID = T2.vacancyID

# GROUP BY T2.vacancyID, T1.applicantFirstName

# HAVING COUNT(vacancyID)>5

# 2. Object Oriented Design

## 2.1 Theory

1. What is polymorphism?
2. What Is the Prototype in JavaScript? Describe an interesting use-case from your experience when you were modifying the Prototype of an object; what was the reason? If you weren’t, where would you do it and why?
3. Have you ever used "Static Factory" OO pattern? If yes, what was the reason? If no, where would you use it and why?

Solution:

1..

Polymorphism is one of the principles of Object-oriented programming (OOP). It is an ability to create a property, a function, or an object that has more than one realization and substitute classes that have common functionality in sense of methods and data. Polymorphism takes advantage of inheritance in order to make this happen.

2.

All objects in JavaScript are descendants of Object. All objects inherit methods and properties from the prototype of Object.prototype, although they can be overridden (except null). For example, prototypes of other constructors override the constructor property and provide their own toString () methods.

3.

## Yes, I used the "Static Factory", because it has some features such as:

## - Created with named "constructors"

## - If appropriate, may return null

## - If need, can return an instance of a derived class

## Another reason that comes to mind is that we can implement interesting logic to avoid persistent new object

## 2.2 Practical Tasks

**Vacancy Approval Process**

**Business Overview**

The application recruitment module requires an additional service to execute actions that the head office users perform during a vacancy approval process. There is a defined list of action types that a product owner requested to implement. Potentially, there will be more requested in future.

**Action Types**

* Submit for Approval
* Approve
* Reject

**Functional requirements**

* Implement a service (JavaScript class) that is able to execute actions (Submit for Approval, Approve, Reject) applied to a record
* Keep in mind that, potentially, in future the service should be able to execute other type of actions that are not related to approval process or vacancy module. Moreover, it can happen that based on a Record type a behaviour of an action may differ.
* An action should log a message to the console (replace record\_id placeholder with an actual record Id)

Submit for Approval: Submitted for approval Vacancy: {record\_id}

Approve: Approved Record: {record\_id}

Reject: Reject Record: {record\_id}

**Testing solution**

The solution should provide RecruitmentSolution.js file with the logic to test a service. Do not create new files, defined your classes inside the received template. As a result of the RecruitmentSolution.js file execution the console should contain next lines:

Expected console output:

Submitted for approval Vacancy: SoftwareEngineer0001  
Approved Record: SoftwareEngineer0001  
Submitted for approval Vacancy: FrontEndEngineer0020  
Rejected Record: FrontEndEngineer0020

Solution:

<https://github.com/Anna-Shy/Salesforce_task>



# 3. Algorithms

## 3.1 Difference between list of Integers

Find difference between two lists of Integers [1, 3, 3, 4, 6, 5, 4] & [6, 3, 5, 2, 2]

**Functional Requirements**

Implement a logic that finds difference between "first" and "second" lists and prints the result to the console.

Expected console output:

[1, 2, 4]

Solution:

<https://github.com/Anna-Shy/Salesforce_task>



# 4. Front-End

## 4.1 Todo App

Given the TodoSolution.html page with a basic design of the Todo application.

**Functional Requirements**

Implement a logic inside the HTML file

*Required:*

* The user should be able to add todo items with an entered text title
* The user should be able to remove a particular todo item
* The user should see an alert message if they try to create a todo item without entering a text
* The user should be able to change the title of a particular todo item

*Optional:*

* The user should be able to see previously created todo items even after refreshing a page

Solution:

<https://github.com/Anna-Shy/Salesforce_task>



# 5. Summary

Why have you chosen to be a software developer? Describe your main objectives and hopes about your future carrier.

Answer:

Software developer is a growing industry with endless progression, opportunities a challenging task and rewarding career option. One reason is constantly education. There are always new tools to acquire and better ways of developing software. The other reason to many career options available. A lot of difficult problems, which software development can solve it. And, of course, a creative outlet, including in the way to solve a problem through coding. Software developer can create practically any type of software product you want on a computer.