**Salesforce Developers**

**Basic Knowledge Test: 1**

**Name:**

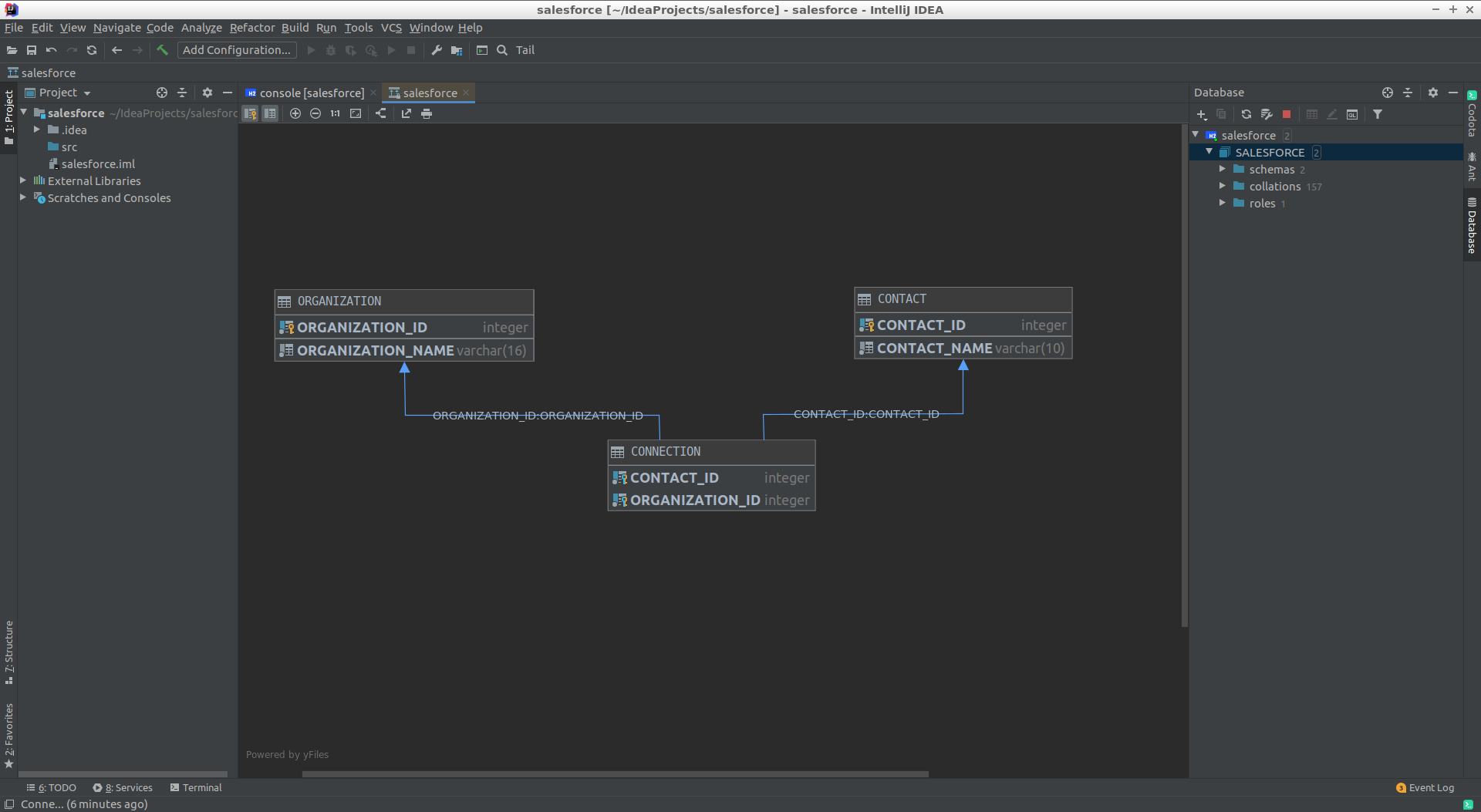
**Skype:**

**Phone:**

**Note:** All test tasks should be completed in well-structured code/ scripts applying Java, Vanilla JS, SQL best practices and Code Conventions.

1. Your DB needs to hold two main objects – **Contacts** and **Organizations**. Each Contact can work in several organizations and each Organization has a lot of contacts.   
     
   - What type of relationship would you use?   
   - Draw the scheme of all tables with relationships.

Relationship Name:   
Many-to-many,с использованием таблицы связей для приведения отношения один к многим  
  
Schema:



drop table if exists Contact, Organization, Connection;

create table Contact

(

contact\_ID integer not null auto\_increment,

contact\_name varchar(10) not null,

primary key (contact\_ID)

);

create table Organization

(

organization\_ID integer not null auto\_increment,

organization\_name varchar(16) not null,

primary key (organization\_ID)

);

create table Connection

(

contact\_ID integer not null,

organization\_ID integer not null,

primary key (contact\_ID, organization\_ID),

foreign key (contact\_ID) references Contact,

foreign key (organization\_ID) references Organization,

);

show tables;

1. Now we have DB with each **Contact** has the only one related **Organization**.   
     
   - How can you find all contacts with **Name** starts with *“A”* in organization named *“Cats”*?   
   - How can you count number of contacts for each organization?   
   - Write SQL query for each question above.

1:

select Contacts.contact\_name

from Connection

inner join Contacts on Connection.contact\_ID = Contacts.contact\_ID

inner join Organization on Connection.organization\_ID = Organization.organization\_ID

where organization\_name = 'Cat'

and contact\_name like 'A%';

2:

select Organization.organization\_name, *count*(Organization.organization\_name)

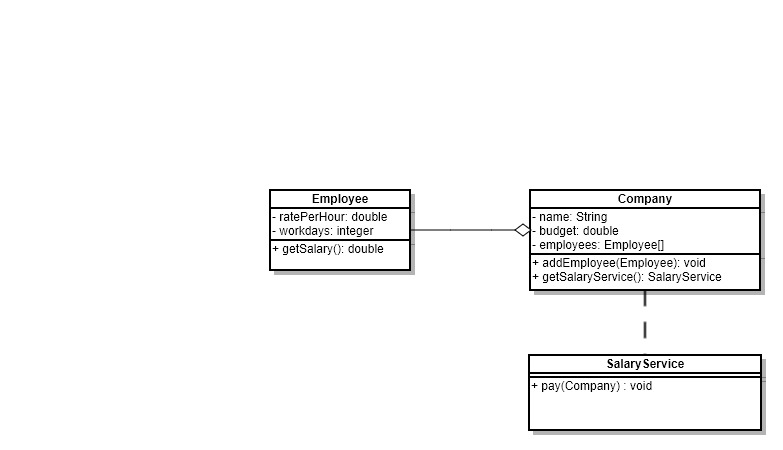
from Connection

join Organization on Connection.organization\_ID = Organization.organization\_ID

group by Organization.organization\_name;

1. As a part of project, you need to implement functionality of calculating salaries for employees in different companies. Each company have its own salary budget. Every employee has his own hourly rate and work hours. Basic formula for salary calculation is ratePerHour \* workdays \* 8

Each company should have ability to pay salaries to its workers. We should take into consideration that total salary payments can’t be greater than company wage budget. UML diagram listed below.



1) Write classes that implement this diagram.   
2) You should have at least 3 types of employees  
e.g.

Manager

Developer

Scrum master

Developer may have bonuses that should be add to the basic salary, it should be taken into consideration during calculation of his wage.

1. You have a list of numbers to fill the table of prices in the shown way

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Jan** | **Feb** | **Mar** | **Apr** | **May** | **Jun** | **Jul** | **Aug** | **Sep** | **Oct** | **Nov** | **Dec** |
| Prod. 1 | Num1 | Num2 | Num3 | Num4 | Num5 | Num6 | Num7 | Num8 | Num9 | Num10 | Num11 | Num12 |
| Prod. 2 | Num13 | Num14 | … |  |  |  |  |  |  |  |  |  |
| Prod. 3 |  |  |  |  |  |  |  |  |  |  |  |  |

- Implement Java method which receive this list and return the object which will be convenient to find price using Month and Product keys.

package Main;

import java.time.Month;

import java.util.ArrayList;

import java.util.LinkedHashMap;

import java.util.List;

import java.util.Map;

public class ProductList {

//variables

private List<Product> list = new <Product>ArrayList();

private class Product {

String productName;

Map<Month, Double> price = new <Month, Double>LinkedHashMap();

public Product(String productName, Month month, double value) {

this.productName = productName;

this.price = new LinkedHashMap<>();

this.price.put(month, value);

}

public double getPrice(Month month) {

return price.get(month);

}

public void setPrice(Month month, double value) {

price.put(month, value);

}

}

*/\*\**

*\* add Product to the ProductList, if it not already there, or set the price of the currently added product*

*\**

*\** ***@param*** *productName product Name*

*\** ***@param*** *month month*

*\** ***@param*** *price price*

*\*/*

public void addValue(String productName, Month month, double price) {

Product product = list.stream().filter(x -> x.productName.equals(productName)).findAny().orElse(null);

if (null == product) {

product = new Product(productName, month, price);

list.add(product);

return;

}

product.setPrice(month, price);

}

*/\*\**

*\* get Price value for selected product.It will throw exceptions if something wrong*

*\**

*\** ***@param*** *productName product Name*

*\** ***@param*** *month month*

*\** ***@return*** *price of the product*

*\** ***@throws*** *NullPointerException if there is no expected product. or its value for a given month is not set*

*\*/*

public double getValue(String productName, Month month) throws NullPointerException {

Product product = list.stream().filter(x -> x.productName.equals(productName)).findAny().orElse(null);

if (null == product)

throw new NullPointerException(String.*format*("no %s product in current list", productName));

try {

return product.getPrice(month);

} catch (NullPointerException e) {

// e.printStackTrace();

throw new NullPointerException(String.*format*("no price for %s product at %s", productName, month));

}

}

public static void main(String[] args) {

ProductList productList = new ProductList();

// ProductList.Product product = productList.new Product("beer", Month.NOVEMBER, 20.2);

productList.addValue("beer", Month.*NOVEMBER*, 20.2);

System.*out*.println(productList.getValue("beer", Month.*NOVEMBER*));

try {

System.*out*.println(productList.getValue("beer", Month.*DECEMBER*));

} catch (NullPointerException e) {

System.*out*.println(e.getMessage());

}

}

}

1. Given todo list in file ‘Test\_Backend\_4’.



Implement following tasks using javaSript:

* 1. creation of the toDo list item.

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