THIS QUIZ CONTRIBUTES MORE THAN 80% WHEN DECIDING WHETHER YOU WILL ENTER THE NEXT ROUND OF INTERVIEW

这份笔试题的答卷将以80%以上的权重决定您是否能晋级下一轮面试，请不要满足于“差不多”、“还不错”、“就这样”的答案哦！

About us: http://www.v2ex.com/t/166173

A video version: https://youtu.be/uQO62wCE8co

Want to know more about us? hai@orgrimmar.io

Not sure what the question is? jh@orgrimmar.io

Notes before you begin:

1. Feel free to use search engine or anything you find helpful, great hackers steal
2. Please list the reference websites
3. Please copy paste and answer in a SEPARATE doc or email

## #0

What’s your GitHub or Stackoverflow account? If you have blog, please show a recent technical article.

## #1 Programming Skills

#1.0 Write a function which takes three numbers as arguments and return the sum of two larger numbers. (Write as *elegant* as possible, DO NOT use predefined method)

### # 1.1

```
const originalData = [
  { name: 'jch', age: 30, score: 90, sex: 1, lesson: 'math' },
  { name: 'oh', age: 31, score: 80, sex: 1, lesson: 'math' },
  { name: 'jia', age: 27, score: 70, sex: 0, lesson: 'math' },
  { name: 'jch', age: 30, score: 80, sex: 1, lesson: 'english' }
```

```
];
```

Please write functions to do following actions for this data.

1. return Array contains all data that age more than given certain value.
2. return an object keyed by ‘name’ and ‘lesson’ based on result of Q1.
3. return an object shows the average scores group by ‘lesson’ and ‘sex’;

## #2 Git

#2.1 Describe your git workflow (May include how to resolve conflicts with others)

#2.2 **Branch split** Suppose you have been working on a branch `foo` for several days, in the middle of break you reformat some code style in a series of commit, then you continue working for another days. After all you raise a pull request and your deadhead reviewer tell you to split your branch into one containing the necessary commits and the other containing ONLY the code style reformat. What should you do in git?

#2.3 **Remote is god** Suppose you have been working on a branch `foo` tracking the remote branch `origin/foo`, you make a mess in the working directory, do some unnecessary commits and finally you feel like starting from remote branch again. The remote branch also have some commits ahead. What should you do in git?

## #3 Node.js

#3.1 What’s the difference between promises below.

```
doSomething().then(function () {
  return doSomethingElse();
});

doSomething().then(function () {
  doSomethingElse();
});

doSomething().then(doSomethingElse());

doSomething().then(doSomethingElse);
```

#3.2 Have you used `co`? If so, please describe how `co` works.

## # 4 ECMAScript

# 4.1 explain why `+[] === 0` is true

# 4.2 What's your favorite ES2015 features?

## # 5 Database

# 5.1 What is `transaction` in database? How to use different isolation levels of transaction?

Please add some examples to the following scenario. Which isolation level should you use?

You are a seller, you use the system with database to sell some items (such as ticket with seat number) shared by many other sellers.

Answer Example:
[This is one kinds of transaction]
I am selling a ticket to somebody. I haven't finished the sell process, but I might find if this ticket had already been sold to other people.

# 5.2 using one SQL statement (not function/procedure) to do insert into multiple tables with relationship.

The relationship is:  teacher  <- n:m -> student <- n:1 -> class

Insert 2 of  t1, 3 of t2, 2 of t3, using junction table t1_t2 to make some relations between inserted rows of t1 & t2, such as first row of t1 link 2nd&3rd t2 rows, etc.

Table student

| id | student_name |
|---|---|
| 4 | 张娜 |
| 5 | 陈盼 |
| 6 | 吴瞳勃 |

Table teacher

| id | teacher_name |
|---|---|
| 1 | 裴坚 |
| 2 | 卞祖强 |

Table class

| id | class_name |
|---|---|
| 7 | 甲班 |
| 8 | 丁班 |

# 5.3 Please give a table definition to save tree structure data. For example, a person should have his/her father, and the father is a person which also has his father…. then it become an family tree.
It will be better if you can also tell how to access all `descendants` from one node in the tree.

## # 6 Abstraction

### # 6.1 If you were to build an app that helps your friends manage their financial transactions, e.g. lending/borrowing/repaying, how would you model the relationship in an object-oriented fashion? Which objects will you define, what properties and types will you choose?

*EXAMPLE ANSWER:*
You can use sequelize to define the tables. Object definition of “Series” and “Video” is like this.

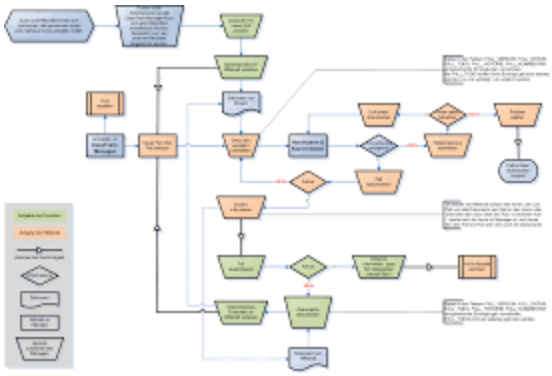
```
Series = sequelize.define('series', {
  title:        DataTypes.STRING,
  sub_title:    DataTypes.STRING,
  description:  DataTypes.TEXT
```

```
})

// Video has a series_id=Series.id foreign reference key after we call Series.hasOne(Video)...
Video = sequelize.define('video', {
  title:        DataTypes.STRING,
  description:  DataTypes.TEXT,

  // set relationship (hasOne) with `Series`
  series_id: {
    type: DataTypes.INTEGER,
    references: {
      model: Series, // Can be both a string representing the table name, or a reference to the model
      key:   "id"
    }
  }
});
```

Reference: http://docs.sequelizejs.com/en/latest/docs/models-definition/

# 6.2. Can you ***briefly*** and ***precisely*** describe the MOST COMPLEX logic you've ever implemented? It can be in one of the following forms that you find most comprehensible: ***flow-diagrams, Pseudo-code,*** *or* ***plain text***.

*EXAMPLE ANSWER:*



OR

```
function TARJAN(Node* node)
    node.visited ← true
    node.index ← indexCounter
    s.push(node)
    for all successor in node.successors do
        if !node.visited then TARJAN(successor)
        end if
        node.lowlink ← MIN(node.lowlink, successor.lowlink)
    end for
    if node.lowlink == node.index then
        repeat
            successor ← stack.pop()
        until successor == node
    end if
end function
```

OR

Lorem ipsum dolor sit amet, ligula suspendisse nulla pretium, rhoncus tempor fermentum, enim integer ad vestibulum volutpat. Nisl rhoncus turpis est, vel elit, congue wisi enim nunc ultricies sit, magna tincidunt. Maecenas aliquam maecenas ligula nostra, accumsan taciti. Sociis mauris in integer, a dolor netus non dui aliquet, sagittis felis sodales, dolor sociis mauris, vel eu libero cras. Faucibus at. Arcu habitasse elementum est, ipsum purus pede porttitor class, ut adipiscing, aliquet sed auctor, imperdiet arcu per diam dapibus libero duis. Enim eros in vel, volutpat nec pellentesque leo, temporibus scelerisque nec.