课程目标：

1. Scala基础

a) 熟悉scala 的隐式转换
b) 熟练运用scala 继承
c) 了解scala泛型的使用

2. Spark基础- Spark SQL

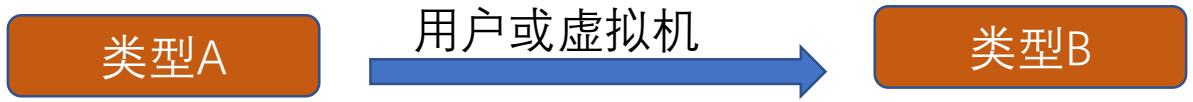
a) SparkSQL的背景
b) SparkSQL的基本使用
c) scala 中隐式转换机制在SparkSQL中的体现

3. 计算机基础补充：排序

注意：其中继承和泛型部分通过作业的讲解带大家学习

一、隐式转换

首先想想什么是类型转换？举例说明



类型转换：　　分为显式(强制)　　隐式

那什么是隐式转换？

重点在“隐”，　转换指类型转换

一、隐式与显式转换的比较

有隐也有显，Scala中的显式转换:

一、基本数据类型自常的toXX方法

```
val a = 3
val b= a.toFloat
val c = a.toDouble
val d = a.toByte
```

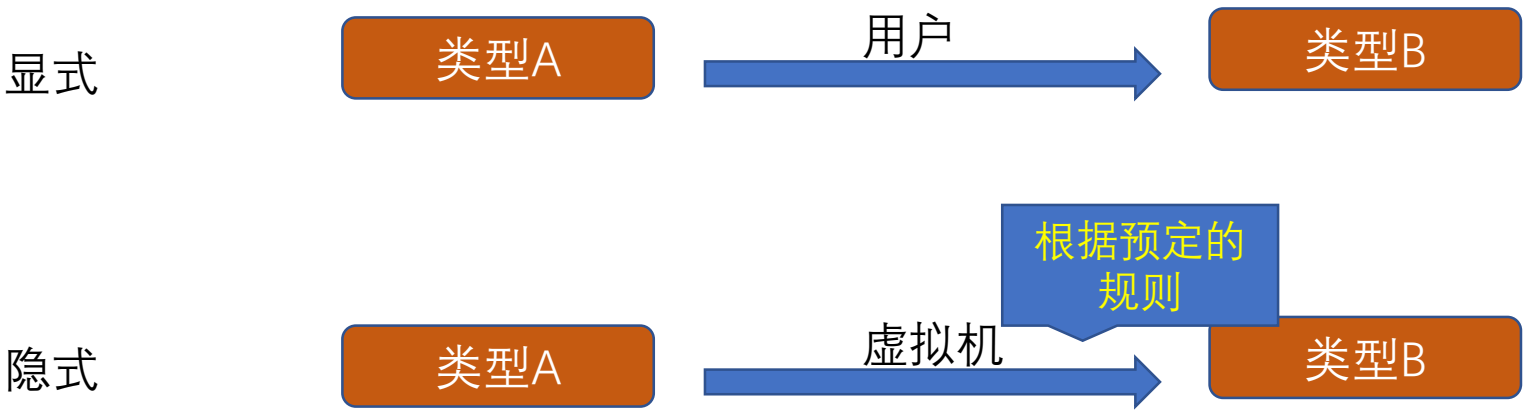
二、通用方法asInstanceOf

```
val a= 3
val b = a.asInstanceOf[Float]
```

一、隐式与显式转换的比较

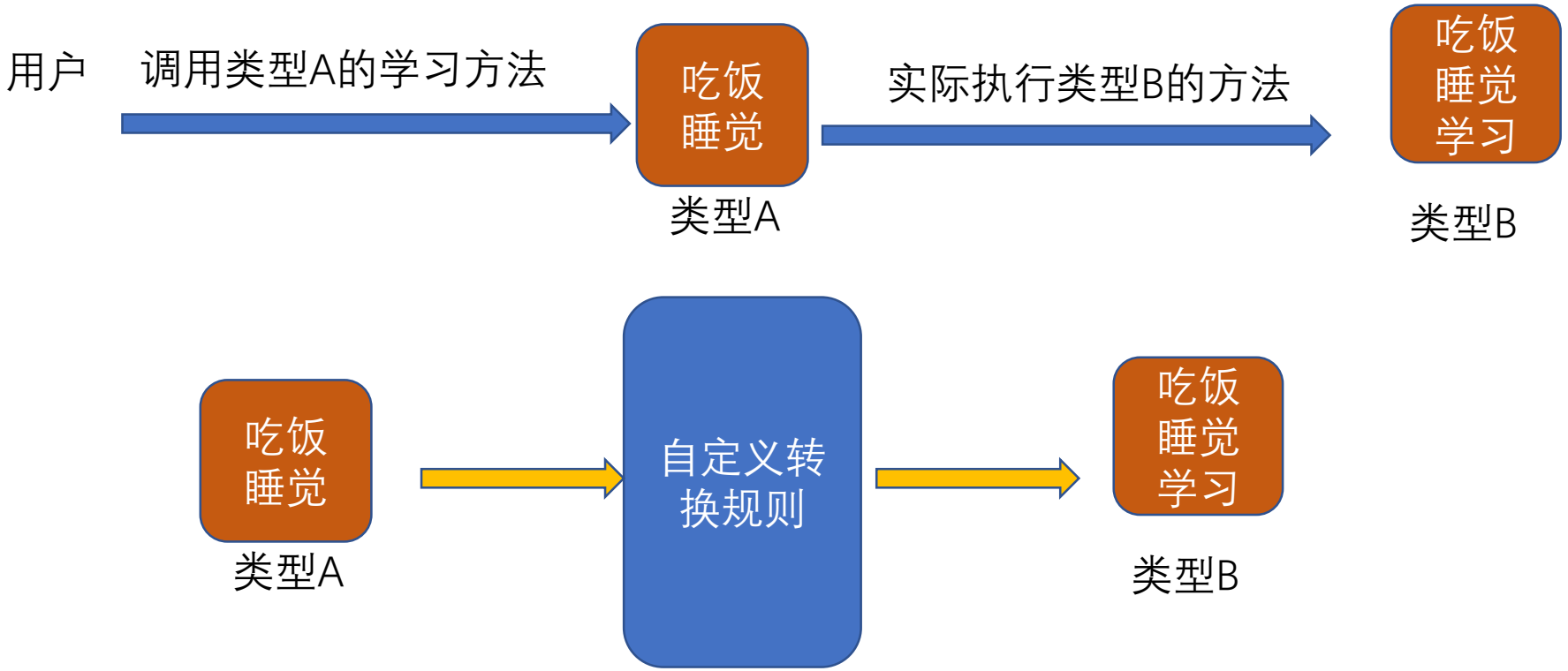
隐式与显式转换的比较

想想我们学过的其它编程的强制类型转换有哪些？？实现方式是什么？
强制类型转换可能有什么问题？



一、隐式转换

隐式转换举例



一、隐式转换

隐式转换举例

首先定义两个phone类，它们有不同的功能

```
class Iphone6{
  val screenSize = 5.1f
  val name = "苹果6"
  def playGame(): Unit ={
    println("我可以打游戏")
  }
}
```

```
class Iphone7(screenSize :Float = 5.5f, name :String){
  def VR(): Unit ={
    println("我是:" + this.name + " 我有牛逼的VR")
  }
}
```

一、隐式转换

隐式转换举例

定义规则：到底怎么转换

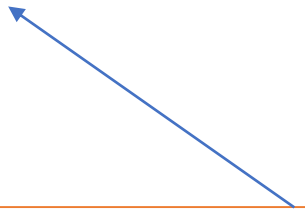
```
object ImplictConvertion {

  implicit def updateIphone(phone :Iphone6) = {
    new Iphone7( phone.screenSize, phone.name)
  }
}
```

语法点：implicit关键字

一、隐式转换

隐式转换的常用场景1

```
def main(args: Array[String]): Unit = {
    val iphone6 = new Iphone6()
    iphone6.playGame()
    iphone6.VR();
}
```

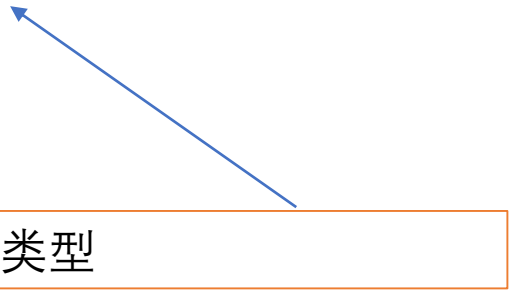


注意我们的类定义里面有没有VR方法

一、隐式转换

隐式转换的常用场景2

```
def main(args: Array[String]): Unit = {
  val iphone6 = new Iphone6()
  testParameterConvertion(iphone6)
}

def testParameterConvertion(iphone7 :Iphone7): Unit ={
  iphone7.VR()
}
```



一、隐式转换

课堂思考:
如果我们这里没有使用隐式转换， 而是强制转换ipone6 到iphone7,可不可行？为什么？

一、泛型

在作业中，大家已经不知不觉用到了 泛型

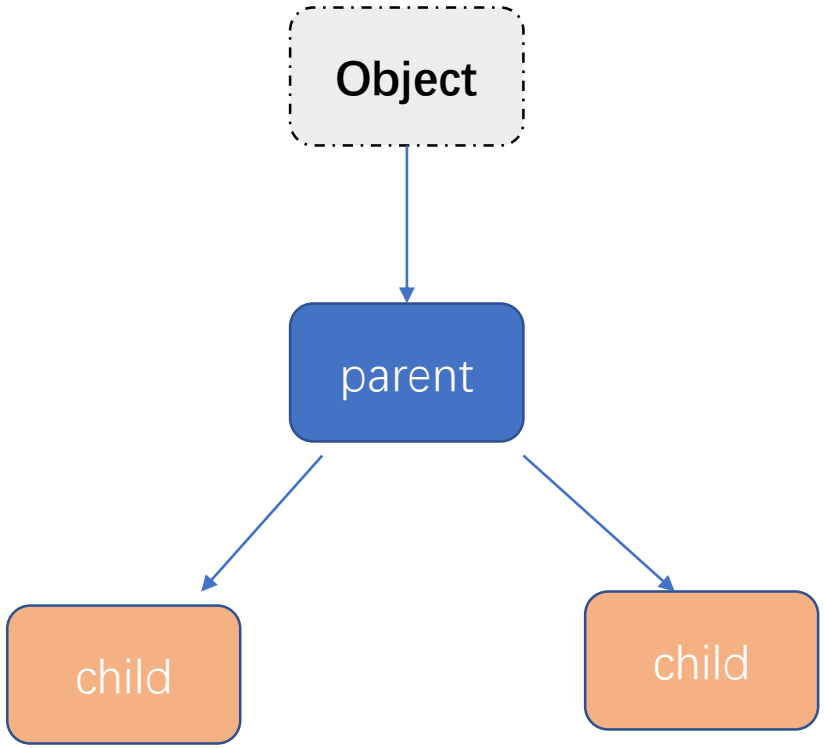
现在是时候了解一下泛型理论

Array[Int]
Array[String]
Array[Student]
.....
Array 支持存放各种数据类型

回想一下python 的 list

一、泛型

泛型是怎么做到的?



Array[Child1]
Array[Child2]

Array[Parent]

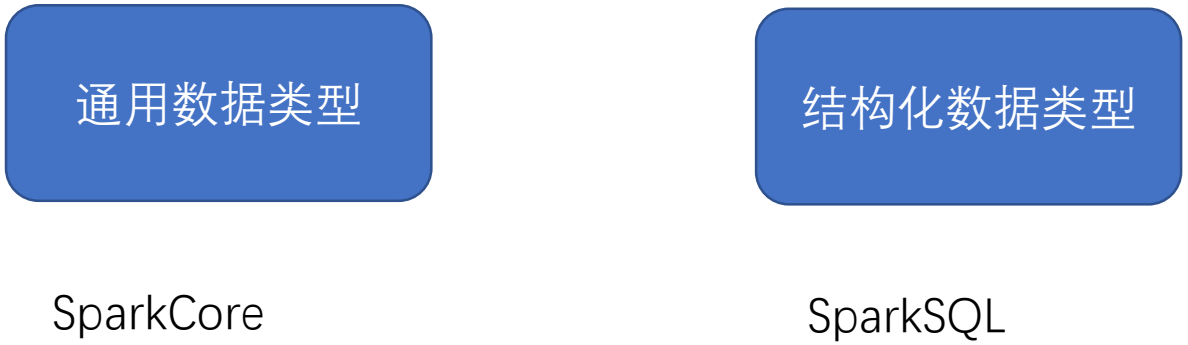
Array[Object]

二、SparkSQL

SPARK SQL是什么？对比SparkCore操纵的数据类型

SparkCore：将分布式数据抽象为弹性分布式数据集（RDD），实现了应用任务调度、RPC、序列化和压缩，并为运行在其上的上层组件提供API。

SparkSQL：Spark Sql 是Spark来操作**结构化数据**的程序包，可以让我使用SQL语句的方式来查询数据，Spark支持 多种数据源，包含Hive表，parquest以及JSON等内容。



二、SparkSQL

有了SparkCore，为什么还要有SparkSQL，通用不是已经包含了结构化数据类型吗?

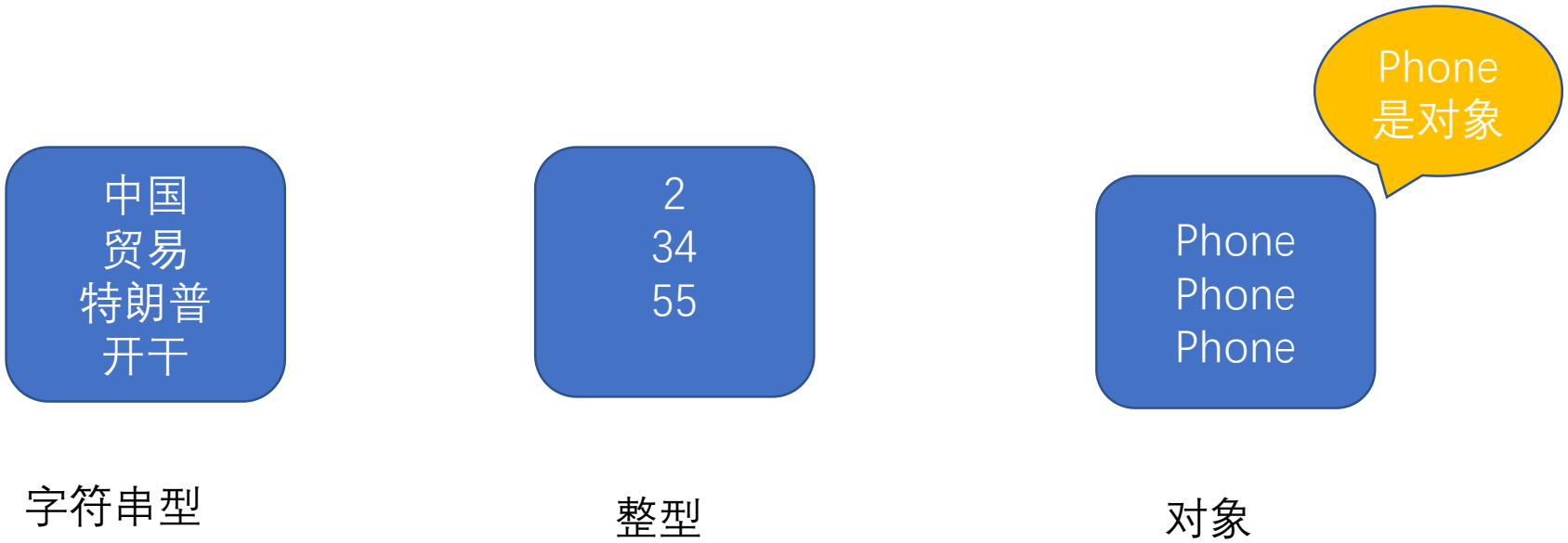
SparkCore　　SparkSQL

二、SparkSQL

SparkSQL的要主数据结构 –DataFrame (类比pandas 中的DataFrame)

DataFrame 相比RDD有啥不一样的?

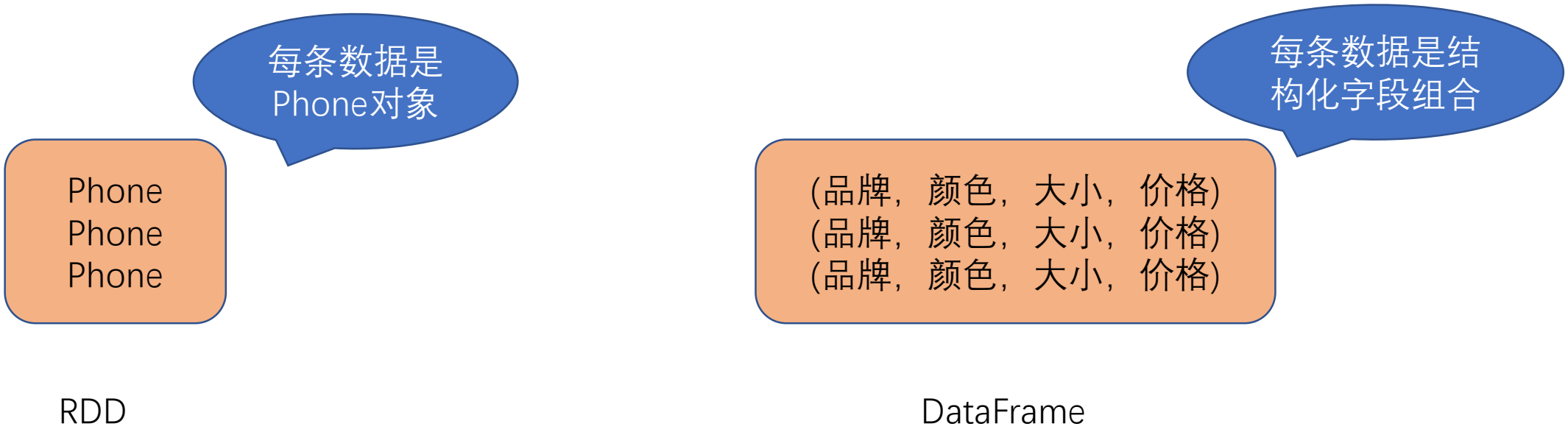
RDD中的数据类型有哪些?



字符串型　　整型　　对象

二、SparkSQL

重点对比下当数据类型为对象时，RDD与DataFrame的表现形式



RDD

DataFrame

二、SparkSQL

如何使用SparkSQl

构建DataFrame，来自二维表格文件

数据文件来自高德POI

city_id	gov_id	shop_title	type_name	address	category_	category_name	latitude	location	longitude	shop_id	update_da	version
2269	锦江区	四川邮电职业技	高等院校	静康路536号	596	科教文化服务;学	30.61978	01010000	104.1241	B001C7XV	2019/8/8	2019/8/8
2269	锦江区	四川师范大学狮	高等院校	静安路5号	596	科教文化服务;学	30.6102	01010000	104.1232	B0FFGJR0	2019/8/8	2019/8/8
2269	锦江区	成都市职工大学(	高等院校	南府街72号	596	科教文化服务;学	30.64879	01010000	104.0694	B001C0AF	2019/8/8	2019/8/8
2269	锦江区	四川师范大学教	高等院校	北辰路与顺庆路交叉口	596	科教文化服务;学	30.61005	01010000	104.1212	B0FFGXL5	2019/8/8	2019/8/8
2269	锦江区	四川省成都市七中	中学	双槐树街54号成都七中	597	科教文化服务;学	30.64453	01010000	104.088	B001C060	2019/8/8	2019/8/8
2269	锦江区	七中育才(银杏校	中学	红砂联合四组401	597	科教文化服务;学	30.58846	01010000	104.1472	B001C8OD	2019/8/8	2019/8/8
2269	锦江区	成都市七中育才学	中学	东南里街5号	597	科教文化服务;学	30.64681	01010000	104.0802	B001C7WI	2019/8/8	2019/8/8
2269	锦江区	四小师大附中	中学	二环高架路入口与莲桂	597	科教文化服务;学	30.63372	01010000	104.1002	B0FFITUO	2019/8/8	2019/8/8
2269	锦江区	成都十七中	中学	较场坝东街48号	597	科教文化服务;学	30.64816	01010000	104.0915	B001C09Q	2019/8/8	2019/8/8
2269	锦江区	成都三中	中学	水杉街300号	597	科教文化服务;学	30.6037	01010000	104.1287	B001C8NF	2019/8/8	2019/8/8
2269	锦江区	成都十九中	中学	工农院街14号附1号	597	科教文化服务;学	30.63184	01010000	104.0961	B001C09Q	2019/8/8	2019/8/8
2269	锦江区	成都市盐道街中学	中学	琉璃场琉璃场街1号	597	科教文化服务;学	30.59626	01010000	104.0964	B001C80P	2019/8/8	2019/8/8
2269	锦江区	四川省成都市第一	中学	红星路2段-83号	597	科教文化服务;学	30.65902	01010000	104.0836	B001C92G	2019/8/8	2019/8/8
2269	锦江区	四川省成都市盐道	中学	横丁字街22号	597	科教文化服务;学	30.64902	01010000	104.0715	B0FFFW6L	2019/8/8	2019/8/8
2269	锦江区	成都七中育才学校	中学	银木街38号	597	科教文化服务;学	30.58862	01010000	104.1476	B0FFGGZ3	2019/8/8	2019/8/8
2269	锦江区	北京师范大学成都	中学	红星路一段37号	597	科教文化服务;学	30.66494	01010000	104.0877	B001C7X6	2019/8/8	2019/8/8
2269	锦江区	成都七中嘉祥外国	中学	晨晖北路6号(近晨辉北路	597	科教文化服务;学	30.60915	01010000	104.1036	B001C7WI	2019/8/8	2019/8/8
2269	锦江区	四川师大附中(东	中学	劼人路与北辰路交叉口	597	科教文化服务;学	30.61449	01010000	104.1215	B0FFHGM	2019/8/8	2019/8/8
2269	锦江区	四川师范大学附属	中学	静安路288号	597	科教文化服务;学	30.60759	01010000	104.1123	B0FFFYN7	2019/8/8	2019/8/8
2269	锦江区	四川师大附中初中	中学	海椒市街103号	597	科教文化服务;学	30.6343	01010000	104.1003	B001C03E	2019/8/8	2019/8/8
2269	锦江区	成都市田家炳中学	中学	顺江路369号	597	科教文化服务;学	30.63104	01010000	104.0959	B001C04C	2019/8/8	2019/8/8
2269	锦江区	博骏公学	中学	静安路290号附近	597	科教文化服务;学	30.60867	01010000	104.1151	B0FFIXF8X	2019/8/8	2019/8/8
2269	锦江区	成都市锦江实验学	中学	碟花街577号	597	科教文化服务;学	30.61092	01010000	104.1336	B001C7WI	2019/8/8	2019/8/8
2269	锦江区	树德中学	中学	百日红西路398	597	科教文化服务;学	30.60588	01010000	104.1417	B0FFIB908	2019/8/8	2019/8/8

二、SparkSQL

如何使用SparkSQl
构建DataFrame，来自二维表格文件

```
val dataframe2 = spark.read.option("header", "true")
  .csv("data/cbd-chengdu.csv")
```

来自json文件

```
{"name": "jim", "age": 23}
{"name": "tick", "age": 33}
{"name": "刘强", "age": 13}
```

```
val dataframe1 = spark.read.json("data/cbd-chengdu.json")
```

二、SparkSQL

如何使用SparkSQl
构建DataFrame，常用操作

查看schema定义

什么是Schema

```
dataframe2.printSchema()
```

查看部分数据

```
dataframe2.show()
```

city_id	gov_id	shop_title	type_name	address	category_code	category_name
2269	2270	四川邮电职业技术学院	高等院校	静康路536号		
2269	2270	四川师范大学狮子山校区	高等院校	静安路5号		
2269	2270	成都市职工大学(南府街校区)	高等院校	南府街72号		
2269	2270	四川师范大学教师教育与心理学院学院	高等院校	北辰路与顺庆路交叉口西南		
2269	2271	四川旅游学院青羊校区	高等院校	一环路西二段17号		
2269	2271	四川商务职业学院(文家东校区)	高等院校	文家场文家正街227号		
2269	2271	四川大学(华西青羊校区)	高等院校	黄田坝高坎村310号		

二、SparkSQL

如何使用SparkSQl， 常用操作

列选择

```
dataframe.select("gov_id").show()
dataframe.select("gov_id", "shop_title").show()
```

条件选择

写法1

```
dataframe.filter($“gov_id” === “锦江区”).show()
dataframe.filter($“gov_id” === “锦江区” && $“type_name” === “中学”).show()
```

写法2

```
dataframe.filter("gov_id = ’锦江区’ and type_name=’中学’").show()
```

二、SparkSQL

如何使用SparkSQl， 常用操作

列选择

```
dataframe.select("gov_id").show()
dataframe.select("gov_id", "shop_title").show()
```

条件选择

为什么3个等号

写法1

```
dataframe.filter($“gov_id” === “锦江区”).show()
dataframe.filter($“gov_id” === “锦江区” && $“type_name” === “中学”).show()
```

写法2

```
dataframe.filter("gov_id = ’锦江区’ and type_name=’中学’").show()
```

二、SparkSQL

如何使用SparkSQl， 常用操作

改变列结构

重命名

```
dataframe.withColumnRenamed("gov_id", "gov_name").show()
```

添加新一列

```
val addNewCol = udf{(value :String) => "成都市" + value}
dataframe.withColumn("full_name", addNewCol($"gov_id")).show()
```

删除列

```
dataframe.drop("gov_id").show()
```

udf 关键字解释

二、SparkSQL

如何使用SparkSQl， 常用操作

聚合操作

groupBy 方法

统计成都各区POI数量

```
dataframe.groupBy("gov_id").count().show()
```

按数量倒序排序

```
dataframe.groupBy("gov_id").count().orderBy($"count" desc).show()
```

二、SparkSQL

如何使用SparkSQl， 常用操作

聚合操作 groupBy 方法

统计成都各区的每种POI类型各自的数量

```
dataframe.groupBy("gov_id", "type_name").count()
  .orderBy($"gov_id", $"count").show()
```

```
+------+---------+-----+
|gov_id|type_name|count|
+------+---------+-----+
|  成华区|   急救中心|    1|
|  成华区| 疾病预防机构|    1|
|  成华区|   高等院校|    6|
|  成华区|   产业园区|   20|
|  成华区|    钟表店|   24|
|  成华区|     中学|   27|
|  成华区|     小学|   31|
|  成华区|    甜品店|   52|
|  成华区| 动物医疗场所|   56|
|  成华区|   综合医院|   84|
|  成华区|  商务写字楼|  108|
```

二、SparkSQL

课堂练习:

找出成都甜品店数量排名前三的是哪些区

二、SparkSQL

SparkSQL 对SQL语句的支持， 绝大部分语法支持

```
将 dataframe 注册成为临时表
dataframe.createOrReplaceTempView("CBD")

调用sql方法执行 sql语句
spark.sql("select * from CBD").show()
spark.sql("select gov_id, shop_title, type_name from CBD").show()
spark.sql("select gov_id, shop_title, type_name from CBD where gov_id ='成华区'").show()

聚合操作
spark.sql("select gov_id,type_name,  count(shop_title) count from CBD " +
  "group by gov_id, type_name").show()
```

二、SparkSQL

既然已经有了DataFrame API 为什么还要有 sql()方法

方式1

```
dataframe.groupBy(“gov_id”, “type_name”).count().show()
```

方式2

```
spark.sql("select gov_id,type_name,  count(shop_title) count from CBD " +
  "group by gov_id, type_name").
```

三、计算机基础---排序

基本排序：冒泡排序 （假设后面的排序都是指升序）

主要思想：每次从无序数据中选中最小的，加入到有序数据

步骤：

① 从无序数据中取出两个数(从前往后挨个取出)，比较大小，
若不符合要求，交换它们的位置

② 循环执行第一步，直到所有无序数据都参与了比较

③ 当第二步执行完毕，如果发生了数据交换：
循环执行第一步、第二步
否则排序结束

三、计算机基础---排序

基本排序：冒泡排序 举例

原始数据：13，21，5，2，18，9，8，7，23，22

① 比较13，21，符合要求，不交换，

② 13，21，5，2，18，9，8，7，23，22

③ 比较21，5 不合要求， 交换 21，5

④ 13，5，21，2，18，9，8，7，23，22

⑤ 比较21，2 不合要求， 交换 21，2

⑥ 13，5，2，21，18，9，8，7，23，22

……

最后一步：
比较23，22， 交换 23，22
13，5，2，18，9，8，7，21，22，23

循环一次完毕后，发现最大值到了该去的位置

三、计算机基础---排序

基本排序：冒泡排序 代码实现要点

1. 假设待排数据是一个数组，初始是无序的
2. 从前往后，从无序数组中开始，两两比较元素大小，将较大的往后面移，直到比较完所有无序元素
3. 重复第2步，使整个数组有序

```
for(i <- Range(data.length - 1, 0, -1)){
  for(j <- 0 until i){
    if(data(j) > data(j + 1)){
      val temp = data(j)
      data(j) = data(j + 1)
      data(j + 1) = temp
    }
  }
}
```