案例中所用到的数据

```json
{
  "took": 45,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 1,
    "hits": [
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "2",
        "_score": 1,
        "_source": {
          "name": "xiaomi 6",
          "desc": "xiaomi6 jiushi kuai!!",
          "price": 2399,
          "tag": [
            "xiaomi",
            "shouji"
          ]
        }
      },
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "1",
        "_score": 1,
        "_source": {
          "name": "xiaomi 7",
          "desc": "xiaomi shouji jiushi kuai! quanmianping",
          "price": 2699,
          "tag": [
            "xiaomi",
            "shouji",
            "quanmianping"
          ]
        }
      },
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "3",
        "_score": 1,
        "_source": {
          "name": "huawei mate 10",
          "desc": "huawei shouji niandu niandu shangwu shouji qijian",
          "price": 4399,
          "tag": [
            "huawei",
            "shouji"
          ]
        }
      }
    ]
  }
}
```

实战案例分析：

案例需求一： 计算每个 tag 下，商品数量

```
GET /ecommerce/product/_search
{
  "size": 0,        // 展示 doc 的数量 ，如果只是做徐鹤分析，设置 size: 0,  可以不
展示数据。
  "aggs": {        //开始聚合
    "group_by_tag": {        //聚合的名称（可以自己随便取）
      "terms": {             // 聚合函数的名字，这里是 terms 分类聚合函数
        "field": "tag"       // 按照 tag field 进行分类
      }
    }
  }
}

```

返回结果：

```
{
  "took": 307,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 1,
    "hits": [
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "2",
        "_score": 1,
        "_source": {
          "name": "xiaomi 6",
          "desc": "xiaomi6 jiushi kuai!!",
          "price": 2399,
          "tag": [
            "xiaomi",
            "shouji"
          ]
        }
      },
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "1",
        "_score": 1,
        "_source": {
          "name": "xiaomi 7",
          "desc": "xiaomi shouji jiushi kuai! quanmianping",
          "price": 2699,
          "tag": [
            "xiaomi",
            "shouji",
            "quanmianping"
          ]
        }
      },
      {
        "_index": "ecommerce",
        "_type": "product",
        "_id": "3",
        "_score": 1,
        "_source": {
          "name": "huawei mate 10",
          "desc": "huawei shouji niandu niandu shangwu shouji qijian",
          "price": 4399,
          "tag": [
            "huawei",
            "shouji"
          ]
        }
      }
    ]
  },
  "aggregations": {
    "group_by_tag": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [        // 分类结果 buckets 结果桶
        {
          "key": "shouji",
          "doc_count": 3
        },
        {
          "key": "xiaomi",
          "doc_count": 2
        },
        {
          "key": "huawei",
          "doc_count": 1
        },
        {
          "key": "quanmianping",
          "doc_count": 1
        }
      ]
    }
  }
}
```

案例需求二：对名称中包含 “xiaomi” 商品， 计算每个 tag 下的商品的数量

```

```

```
GET /ecommerce/product/_search
{
  "size": 0,
  "query": {         // 在聚合分析之前，先进行条件查询
    "match": {
      "name": "xiaomi"
    }
  },
  "aggs": {
    "group_by_tags": {
      "terms": {
        "field": "tag"
      }
    }
  }
}
```

返回结果：

```
{
  "took": 293,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 2,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "group_by_tags": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [
        {
          "key": "shouji",
          "doc_count": 2
        },
        {
          "key": "xiaomi",
          "doc_count": 2
        },
        {
          "key": "quanmianping",
          "doc_count": 1
        }
      ]
    }
  }
}
```

案例需求三：先分组，再算每组的平均值，计算每个 tag 下的商品的平均价格

```
GET /ecommerce/product/_search
{
  "size": 0,
  "aggs": {
    "group_by_tag": {
      "terms": {
        "field": "tag"
      },
      "aggs": {   //在前面进行了 按照tag 分组的聚合分析之后， 在 terms 并列层级添加第二个聚合分析，avg，在每组中进行 avg 聚合分析
        "price_avg": {
          "avg": {
            "field": "price"
          }
        }
      }
    }
  }
}
```

返回结果：

```
{
  "took": 86,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "group_by_tag": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [
        {
          "key": "shouji",        //第一个tag “shouji” 平均值 3165.666
          "doc_count": 3,
          "price_avg": {
            "value": 3165.6666666666665
          }
        },
        {
          "key": "xiaomi",        //第二个tag “xiaomi” 平均值 2549
          "doc_count": 2,
          "price_avg": {
            "value": 2549
          }
        },
        {
          "key": "huawei",        //第三个tag “huawei” 平均值 4399
          "doc_count": 1,
          "price_avg": {
            "value": 4399
          }
        },
        {
          "key": "quanmianping",        //第三个 tag “quanmianping” 平均值 2699
          "doc_count": 1,
          "price_avg": {
            "value": 2699
          }
        }
      ]
    }
  }
}
```

案例需求四：计算每个 tag 下的商品平均价格，并按照降序排列

```
GET /ecommerce/product/_search
{
  "size": 0,
  "aggs": {
    "group_by_tag": {
      "terms": {
        "field": "tag",
        "order": {                //进行排序， 按照下面聚合分析的结果，这里是按照
下面 avg_price 求价格平均数的 结果降序
          "avg_price": "desc"
        }
      },
      "aggs": {
        "avg_price": {
          "avg": {
            "field": "price"
          }
        }
      }
    }
  }
}

```

返回结果：

```
{
  "took": 22,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "group_by_tag": {
      "doc_count_error_upper_bound": 0,
      "sum_other_doc_count": 0,
      "buckets": [
        {
          "key": "huawei",
          "doc_count": 1,
          "avg_price": {
            "value": 4399
          }
        },
        {
          "key": "shouji",
          "doc_count": 3,
          "avg_price": {
            "value": 3165.6666666666665
          }
        },
        {
          "key": "quanmianping",
          "doc_count": 1,
          "avg_price": {
            "value": 2699
          }
        },
        {
          "key": "xiaomi",
          "doc_count": 2,
          "avg_price": {
            "value": 2549
          }
        }
      ]
    }
  }
}
```

案例需求五：先按照价格范围区间进行分组， 然后在每组内再按照 tag 进行分组，最后再计算每组的平均价格

```
GET /ecommerce/product/_search
{
  "size": 0,
  "aggs": {
    "range_by_price": {
      "range": {                // 按照 price 进行范围分组
        "field": "price",
        "ranges": [
          {
            "from": 1000,
            "to": 2000
          },
          {
            "from": 2000,
            "to": 3000
          },
          {
            "from": 3000,
            "to": 4000
          },
          {
            "from": 4000,
            "to": 5000
          }
        ]
      },
      "aggs": {                   // 按照 tag 进行分组
        "group_by_tag": {
          "terms": {
            "field": "tag"
          },
          "aggs": {               // 在计算 price 的平均值
            "price_avg": {
              "avg": {
                "field": "price"
              }
            }
          }
        }
      }
    }
  }
}
```

结果：

```
{
  "took": 25,
  "timed_out": false,
  "_shards": {
    "total": 5,
    "successful": 5,
    "skipped": 0,
    "failed": 0
  },
  "hits": {
    "total": 3,
    "max_score": 0,
    "hits": []
  },
  "aggregations": {
    "range_by_price": {
      "buckets": [                    // 按照范围分组桶
        {
          "key": "1000.0-2000.0",
          "from": 1000,
          "to": 2000,
          "doc_count": 0,
          "group_by_tag": {
            "doc_count_error_upper_bound": 0,
            "sum_other_doc_count": 0,
            "buckets": []
          }
        },
        {
          "key": "2000.0-3000.0",
          "from": 2000,
          "to": 3000,
          "doc_count": 2,
          "group_by_tag": {
            "doc_count_error_upper_bound": 0,
            "sum_other_doc_count": 0,
            "buckets": [                    //每个分组中按照 tag 分组桶
              {
                "key": "shouji",
                "doc_count": 2,
                "price_avg": {
                  "value": 2549            //计算平均值
                }
              },
              {
                "key": "xiaomi",
                "doc_count": 2,
                "price_avg": {
                  "value": 2549
                }
              },
              {
                "key": "quanmianping",
                "doc_count": 1,
                "price_avg": {
                  "value": 2699
                }
              }
            ]
          }
        },
        {
          "key": "3000.0-4000.0",
          "from": 3000,
          "to": 4000,
          "doc_count": 0,
          "group_by_tag": {
            "doc_count_error_upper_bound": 0,
            "sum_other_doc_count": 0,
            "buckets": []
          }
        },
        {
          "key": "4000.0-5000.0",
          "from": 4000,
          "to": 5000,
          "doc_count": 1,
          "group_by_tag": {
            "doc_count_error_upper_bound": 0,
            "sum_other_doc_count": 0,
            "buckets": [
              {
                "key": "huawei",
                "doc_count": 1,
                "price_avg": {
                  "value": 4399
                }
              },
              {
                "key": "shouji",
                "doc_count": 1,
                "price_avg": {
                  "value": 4399
                }
              }
            ]
          }
        }
      ]
    }
  }
}
```

注：如果第一次使用聚合函数 是要设置正排索引 所以要将 文本的 field 的 fielddata 属性设置为 true

```
PUT /ecommerce/_mapping/product
{
  "properties":{
    "tag":{
      "type": "text",
      "fielddata": "true"
    }
  }
}

```