设计文档

1.需求分析

题目：

抓取 腾讯课堂官网的数据： https://ke.qq.com/

统计每天的各个分类的课程数量。可以查看历史数据。

点击数量可以跳转页面查看每一门课的详情信息，包括课程id、课程标题、价格、老师等信息。

需要实现功能

- 爬取腾讯课堂官网的课程分类数据并保存入库
- 爬取分类的课程的详细信息，统计各分类的课程数量
- 编写前端页面（我的前端真的不咋，你凑合着看吧。。。。）
- 部署应用（用的内网穿透，不想买服务器，主要是没钱）

腾讯课堂官网分析

根据我观察 课程分类有多个层级且最多为三级

以上面的图为例

`https://ke.qq.com/course/list?mt=1001&st=2001&tt=3001`

- mt表示 为一级分类id
- st 表示 为二级分类id
- tt 表示 为三级分类id

我所观察的腾讯课堂官网

- 不存在 四级或者更多级分类
- 而且三级分类课程组成了 二级分类课程，二级分类课程组成了 一级分类课程

所以只需要统计所有三级分类课程即可统计出所有的课程

2.数据库设计

这里最少只需要2张表就可以实现功能

```
#课程表
CREATE TABLE `course` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT COMMENT 'id',
  `course_id` int(255) DEFAULT NULL COMMENT '课程id（腾讯学堂官方）',
  `title` varchar(255) DEFAULT NULL COMMENT '课程标题',
  `mt` int(255) DEFAULT NULL COMMENT '一级分类id （腾讯学堂官方）',
  `st` int(255) DEFAULT NULL COMMENT '二级分类id （腾讯学堂官方）',
  `tt` int(255) DEFAULT NULL COMMENT '三级分类id （腾讯学堂官方）',
  `teacher` varchar(255) DEFAULT NULL COMMENT '老师',
  `price` varchar(255) DEFAULT NULL COMMENT '价格',
  `is_package` bit(1) DEFAULT NULL COMMENT '是否是合集类型',
  `date` date NOT NULL COMMENT '爬取时间',
  PRIMARY KEY (`id`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=0 DEFAULT CHARSET=utf8mb4;

#课程分类表
CREATE TABLE `course_category` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT COMMENT 'id',
  `course_category_name` varchar(255) DEFAULT NULL COMMENT '课程分类名称',
  `mt` int(255) DEFAULT NULL COMMENT '一级分类id （腾讯学堂官方）',
  `st` int(255) DEFAULT NULL COMMENT '二级分类id （腾讯学堂官方）',
  `tt` int(255) DEFAULT NULL COMMENT '三级分类id （腾讯学堂官方）',
  `level` int(255) DEFAULT NULL COMMENT '层级',
  `course_number` int(255) DEFAULT NULL COMMENT '分类课程数',
  `date` date NOT NULL COMMENT '爬取时间',
  PRIMARY KEY (`id`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=1329 DEFAULT CHARSET=utf8mb4;
```

3.技术选型

2.1 前端

- vue+Element UI（主流配合 不解释）

2.1 后端

- Spirngboot+Mybatis +Druid+MySQL+jsoup（相比其他Java爬虫来 简单易用）

> jsoup 是一款Java 的HTML解析器，可直接解析某个URL地址、HTML文本内容。它提供了一套非常省力的API，可通过DOM，CSS以及类似于jQuery的操作方法来取出和操作数据。

4.实现细节

4.1 定时任务

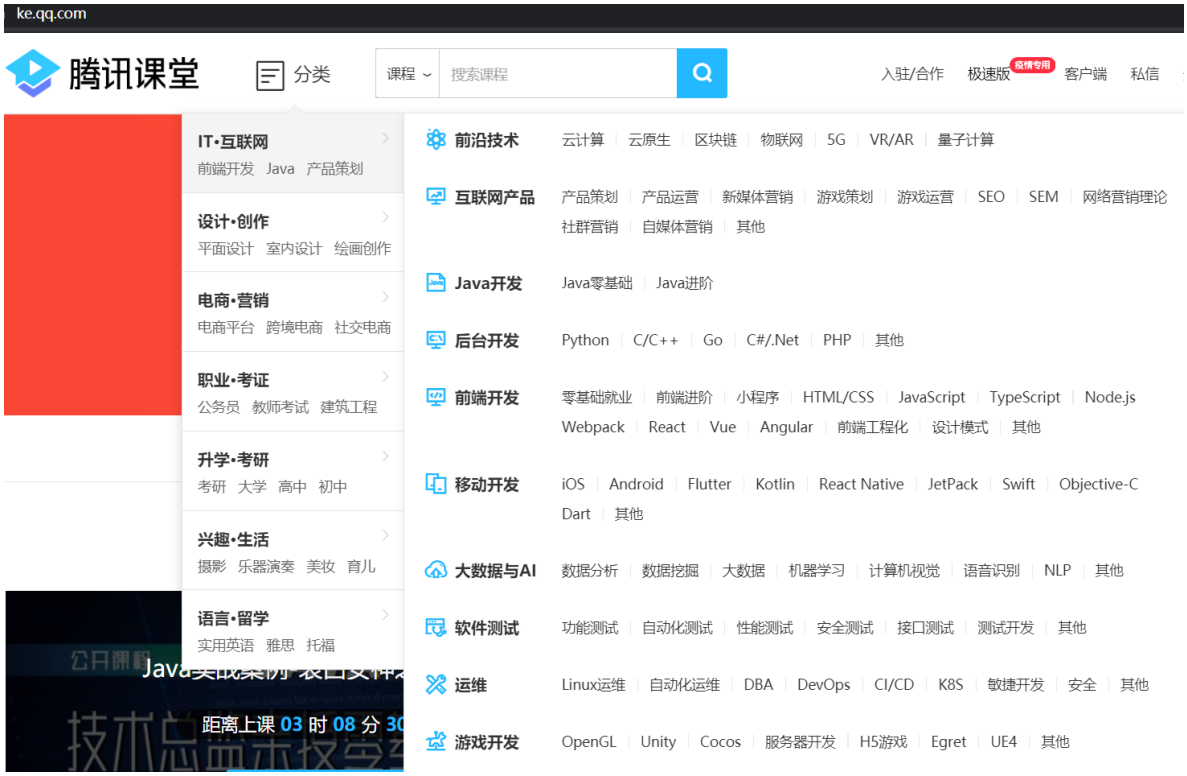
选用Spring自带@Scheduled注解实现 每天凌晨爬取

```
package com.wp.webcrawlerdemo.task;

import com.wp.webcrawlerdemo.service.WebCrawlerService;
import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.scheduling.annotation.Scheduled;
import org.springframework.stereotype.Component;

@Component
@Slf4j
public class WebCrawlerTask {
    @Autowired
    private WebCrawlerService webCrawlerService;

    @Scheduled(cron = "0 0 0 * * ?")
    public void getInfoFromKeQQ() {
        try {
            long start = System.currentTimeMillis();
            webCrawlerService.getInfoFromKeQQ();
            long end = System.currentTimeMillis();
            log.info("爬取过程总耗时为" + (end - start) + "毫秒");
        } catch (Exception e) {
            log.error("爬取过程中出现异常: ", e);
        }
    }
}
```

4.2爬取过程

```
    @Override
    public void getInfoFromKeQQ() throws Exception {
        Date now = new Date();
        List<CourseCategory> allLevel = getAllLevel(now);
        if (!CollectionUtils.isEmpty(allLevel)) {
            for (CourseCategory courseCategory : allLevel) {
                // 三级分类下面的课程信息 使用多线程异步拉取
                executor.submit(() -> {
                    try {
                        getCourseInfo(courseCategory, now);
                    } catch (Exception e) {
                        log.error("爬取课程信息出现错误! ",e);
                    }
                });
            }
        }
        // 汇总结果
        summary(now);
    }
```

4.2.1 爬取所有分类



在主页的分类框中的a标签链接 即可爬取所有分类，并且获取所有的三分类

```
/**
 * 爬取主页的所有分类并保存入库，返回所有的三级分类
 *
 * @return
 * @throws Exception
 */
@Override
public List<CourseCategory> getAllLevel(Date date) throws Exception {
    // 保存所有的三级分类
    List<CourseCategory> courseCategoriesLevel = new ArrayList<>();
    Document document = Jsoup.connect(INDEX_URL).get();
    Elements allA = document.select("header").select("a");
    for (Element element : allA) {
        String href = element.attr("href");
        Map<String, String> urlParam = getUrlParam(href);
        Integer categoryLevel = getCategoryLevel(urlParam);
        // 如果链接是一个课程详情链接
        if (!categoryLevel.equals(-1)) {
            CourseCategory courseCategory = new CourseCategory();
            courseCategory.setCourseCategoryName(element.text());
            courseCategory.setLevel(categoryLevel);
            courseCategory.setDate(date);
            switch (categoryLevel) {
                case 1:
                    // 1级分类
                    courseCategory.setMt(Integer.valueOf(urlParam.get("mt")));
                    break;
                case 2:
                    // 2级分类
                    courseCategory.setMt(Integer.valueOf(urlParam.get("mt")));
                    courseCategory.setSt(Integer.valueOf(urlParam.get("st")));
                    break;
```

```
                case 3:
                    //3级分类
                    courseCategory.setMt(Integer.valueOf(urlParam.get("mt")));
                    courseCategory.setSt(Integer.valueOf(urlParam.get("st")));
                    courseCategory.setTt(Integer.valueOf(urlParam.get("tt")));
                    courseCategoriesLevel.add(courseCategory);
                    break;

            }
            //保存入库
            courseCategoryMapper.insert(courseCategory);

        }
    }
    return courseCategoriesLevel;
}
```

4.2.1 爬取所有课程信息

具体过程分为三步

- 拼接课程详情链接
- 获取最大页码，遍历该分类下所有页
- 爬取具体的课程信息 ，并保存入库

下面只展示关键代码

```
    public void getCourseInfo(CourseCategory courseCategory, Date date) throws
Exception {
        // 接课程详情链接
        StringBuilder sb = new StringBuilder(COURSE_URL);
        if (courseCategory != null) {
            sb.append("?");
            if (!Objects.isNull(courseCategory.getMt())) {
                sb.append("mt=").append(courseCategory.getMt());
            }
            if (!Objects.isNull(courseCategory.getSt())) {
                sb.append("&st=").append(courseCategory.getSt());
            }
            if (!Objects.isNull(courseCategory.getTt())) {
                sb.append("&tt=").append(courseCategory.getTt());
            }
            sb.append("&page=");
        }
        String requestUrl = sb.toString();
        int page = 1;
        Integer pageCount = null;
        // 遍历该分类下所有页
        do {
            Document document = Jsoup.connect(requestUrl + page).get();
            Elements courseList = document.getElementsByClass("course-card-
list").select("li");
            if (pageCount == null) {
                pageCount = getPageSize(document);
            }
            for (Element element : courseList) {
```

```
                Elements priceElements = element.getElementsByClass("line-cell item-
price  custom-string");
                // 没有价格信息就不是要爬取的课程信息
                if (priceElements == null ||
StringUtils.isEmpty(priceElements.text())) {
                    continue;
                }
                Elements itemElements = element.getElementsByClass("item-tt");
                Course course = new Course();
                course.setDate(date);
                course.setSt(courseCategory.getSt());
                course.setMt(courseCategory.getMt());
                course.setTt(courseCategory.getTt());

 course.setCourseId(getCourseId(itemElements.select("a").attr("href")));
                course.setTitle(itemElements.text());
                course.setPrice(priceElements.text());

 course.setIsPackage(itemElements.select("a").attr("href").contains("package"));
                course.setTeacher(element.getElementsByClass("item-line item-line--
middle").select("a").text());
                // 保存入库
                courseMapper.insert(course);
            }
            page++;
        } while (page <= pageCount);
    }
```

4.2.3 统计课程数

统计所有分类的课程数

```
    @Override
    public void summary(Date date) {
        // 一级分类
        List<CourseCategory> courseCategories01 =
courseCategoryMapper.queryByLevel(date, 1);
        // 二级分类
        List<CourseCategory> courseCategories02 =
courseCategoryMapper.queryByLevel(date, 2);
        // 三件分类
        List<CourseCategory> courseCategories03 =
courseCategoryMapper.queryByLevel(date, 3);
        summaryCourseCategory(courseCategories01);
        summaryCourseCategory(courseCategories02);
        summaryCourseCategory(courseCategories03);
    }

    // 统计课程数
    private void summaryCourseCategory(List<CourseCategory> list) {
        if (!CollectionUtils.isEmpty(list)) {
            for (CourseCategory courseCategory : list) {
                Integer courseNumber = courseMapper.getCourseNumber(courseCategory);
                courseCategory.setCourseNumber(courseNumber);
                courseCategoryMapper.update(courseCategory);
            }
        }
    }
```

4.3 前端展示

4.3.1 分类树展示

以树结构展示 比较清晰直观

4.3.2 课程详情展示

点击课程数量就可以看改分类下 课程详情

总结

首先说一句抱歉，我最近忙着离职的事情，没有投入太多时间完成这个练习，项目比较简陋。

我一直都是写后端的，从来没写过爬虫相关的需求，这一次也学到了一些爬虫相关的知识，而且幸运的是我在爬腾讯课堂也没有遇到反爬虫处理。

最后感谢给我这次机会，期待有机会与您共事。