

「信息系统集成与管理」实验课

「信息系统集成与管理」实验课

实验一

实验代号

实验目的

实验准备

工具准备

数据准备

实验要求

输入与输出

实验步骤

实验结果评价

实验提交内容

实验打分依据

实验二

实验代号

实验目的

实验准备

技术准备

代码准备

实验要求

实验结果评价

实验结果提交内容

实验打分依据

实验一

实验代号

本实验代号为：**halibut**

实验目的

本实验考查学生对ETL知识的理解情况。Kettle 是一款开源的、元数据驱动的ETL工具集，是开源 ETL 工具里功能比较强大的一个。本实验利用Kettle提供的开源免费版本，完成一个具体的数据ETL过程。通过该过程，学生可以理解数据的抽取、转换、装载流程，从而加深对ETL知识的理解。

实验准备

工具准备

注意，学生需提前准备操作系统和相关的依赖工具。

我们已经将Kettle工具提前下载到自己的服务器，学生可以自行下载安装。我们下载的Kettle安装包均来自Pentaho官方网站（<https://sourceforge.net/projects/pentaho/files/>），请放心使用。安装包位置：

```
1 | {你的ftp根目录}/textbook/refs/
```

我们的ftp server中放置了1个最新的版本：

- pdi-ce-9.3.0.0-428.zip，该版本为2022年版本

同学们也可以从官网下载其他的版本使用。无论是哪种版本，均可以用于本次试验。区别在于所需的支持环境（JDK版本、MySQL Connect Driver版本）稍有差异，请同学自行查找相关资料。下面是一个相关的参考：

https://blog.csdn.net/qq_36135335/article/details/86538688

另外，我们ftp server中的版本均可以在Windows、Linux、MacOS操作系统中部署安装。

由于实验要求将结果输出到数据库中，因此，学生需提前准备一个关系数据库（推荐MySQL，可自行安装试用版）。

数据准备

本实验所使用的数据在理论课的支持服务器（ftp server）上，具体位置：

```
1 {你的ftp根目录}/halibut.log # 注意：每个人的文件内容是不一样的
```

该数据每条格式是一样的，以一条数据为例：

```
1 Tue Nov 16 15:41:44 2021 1 113.57.80.253 437423
  /zhuguobin/Maze.zip b _ i g profzhu ftp 0 * c
```

各项含义具体说明如下：

1	Tue Nov 16 15:41:44 2021	# 记录的时间
2	1	# 传输文件花费的时间（秒）
3	113.57.80.253	# 客户端ip地址
4	437423	# 传输的文件大小（byte）
5	/zhuguobin/Maze.zip	# 传输的文件
6	b	# 传输类型，b表示二进制传输，a表示
	ascii码传输	
7	_	# 特殊动作标记
8	i	# 传输方法，o表示从服务器下载，i表示
	向服务器上传	
9	g	# 访问模式，g表示虚拟用户，a表示匿名
	用户	
10	profzhu	# 用户名
11	ftp	# 服务名
12	0	# 授权方式
13	*	# 表示授权用户已认证IP
14	c	# 完成状态，c表示complete， i表示
	incomplete	

根据以上含义，在数据库中建立一个表（表名自定），字段为上述的14项（字段名自行拟定）。该表格将用于ETL的输出。

实验要求

输入与输出

输入：该实验要求在spoon中对ftp server中的halibut.log发起请求，以此为输入【即：不要将halibut.log下载到本机，不要以本地文件为输入】。

输出：该实验要求以数据库中的表为输出类型。

实验步骤

1. 创建数据库表，用于输出
2. 对ftp server中的halibut.log发起ftp请求，作为数据输入源
3. 定义转换规则
4. 定义输出（包括各种输出选项）
5. 执行转换过程
6. 检查数据库表，验证是否成功
7. 导出数据库表为sql文件

实验结果评价

实验提交内容

本实验需提交以下内容作为成果【实验结果提交到ftp server的个人目录下】：

1. 将最终输出的数据库表导出为SQL文件，命名为**halibut.sql**，提交到ftp server。授课老师可以根据此文件检查结果是否正确。该文件样式类似于如下（仅供参考，不同的数据库平台输出的样式有差异）：

```

1  -- -----
2  -- Table structure for user
3  -- -----
4  DROP TABLE IF EXISTS `user`;
5  CREATE TABLE `user` (
6      `id` int NOT NULL AUTO_INCREMENT,
7      `username` varchar(255) DEFAULT NULL,
8      `password` varchar(255) DEFAULT NULL,
9      `addr` varchar(255) CHARACTER SET utf8 COLLATE
10     utf8_general_ci DEFAULT NULL,
11     PRIMARY KEY (`id`)

```

```

11 ) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb3;
12
13 -----
14 -- Records of user
15 -----
16 BEGIN;
17 INSERT INTO `user` VALUES (3, 'zhuguobin', '987654321',
    '武汉大学遥感信息工程学院');
18 INSERT INTO `user` VALUES (4, 'zhuguobin', '12345678',
    '武汉大学');
19 COMMIT;
20
21 SET FOREIGN_KEY_CHECKS = 1;

```

2. 实验过程描述文档，命名为**halibut.doc**【word文档】，提交到个人ftp目录。文档内容包括：
 - 对上述实验操作的7个步骤，每步至少一个截图。
 - 实验环境描述，包括：所使用的halibut.log文件的行数（每个人不同）、所使用的数据库名称（包括版本号），等

实验打分依据

1. SQL文件【**halibut.sql**】
 - 该文件可导入，且与原始halibut.log内容一致，满分100分；
 - 该文件可导入，但与原始halibut.log不一致，根据实际情况，得60~90分；
 - 该文件不可导入，根据实际情况，得30~60分；
 - 未提交，得0分
2. 实验描述文档【**halibut.doc**】
 - 根据内容情况，得60~100分；
 - 未提交，得0分
3. 以上两项内容，加权平均，总分以百分制给定。
4. 授课老师根据具体情况，决定是否现场演示。

实验二

实验代号

实验二代号为：**dace**

实验目的

本实验考察学生对Web Service理论掌握的情况，通过一个具体的Web应用的前后端编写，实现相应的REST风格Web API，服务于前端网页、以及各种API调用。通过该实验过程，学生可以了解REST架构风格的特点、集合类资源的Web API设计方式、以JSON为格式的表述方法，从而加深对Web Service理论的理解。

实验准备

技术准备

本实验要求学生在掌握Web Service基础理论的前提下，深入理解REST架构风格的Web API设计方法。

首先，针对业务需求，设计资源模版。模版的设计请参考理论课所讲的collection+json样式。其次，安装业务功能，设计网页端所需的各种REST服务，以及下面所要求的API服务。最后，根据实验要求，实现前后端的代码，并进行相关的测试。

具体技术准备包括（但不限于）：

- 操作系统。学生根据自己的情况，在Windows或Linux中完成实验
- 服务端编程语言。尽管实验为大家准备了node.js基础代码，但学生可以根据自己偏好，选择不同的语言完成实验。
- API测试环境。在curl或postman两种中选择一个。Windows环境中需要自行安装curl程序包，postman可以从官网下载试用版（Windows或Linux均可）
- 客户端App编程语言。客户端对自定义的API编程，以完成本实验指定的业务目标。客户端编程语言可以从下面的语言中任选一个：
 - python

- node.js
- C++
- Java

代码准备

为便于学生在规定时间内完成实验，本实验准备了服务端基础代码。代码以Node.js方式提供，源码请在理论课程的ftp server中自行下载。

```
1 | {你的ftp根目录}/textbook/refs/type.zip
```

代码说明：

```

1 | — server                                # 网站服务器目录
2 |   | — Procfile
3 |   | — app.js                          # 主应用代码
4 |   | — data                            # 客户提交数据的目录
5 |     | — 2579674512460237
6 |     | — 5762090880192063
7 |     | — 5923644971070143
8 |     | — 6036179874679288
9 |     | — 6108741329096414
10 |    | — 613856331910938
11 |   | — messages.js                    # 读、写信息的脚本
12 |   | — package.json
13 |   | — templates                      # 模版目录
14 |     | — about.html
15 |     | — collection.js
16 |     | — home.html
17 |     | — item.html
18 |     | — list.html
19 |     | — script.js
20 |   | — templates.js                  # 模版文件
21 | — test                              # API测试所用的各种数据，可自行修
    | 改
22 |   | — README.md
23 |   | — item-response.json
24 |   | — item-response2.json
25 |   | — post.json
26 |   | — put.json
27 |   | — sidt.json

```

本实验所提供的Node.js代码仅供参考使用。学生可以自行选择采用何种语言作为编程工具。

实验要求

本实验需实现以下要求：

1. **编写网页前端**，实现对人员信息的基本采集，包括：姓名、学号、邮箱、手机号码、个人兴趣，共5项内容。其中，对学号、邮箱、手机号码应采取正则匹配验证（服务端或客户端验证，自行选择），对姓名、个人兴趣采取字符长度限制（姓名不超过8个中文字符，个人兴趣不超过32个中文字符，自行选择采取服务端或客户端验证）
2. **网页端页面流转逻辑**参照课堂上所讲的YOUTYITWEPOSTIT网站逻辑
3. **编写客户端App代码**，以独立应用程序的方式完成相应的功能：
 1. 添加一条新的条目
 2. 删除一条已存在的条目。条目不存在时，给予错误提示
 3. 修改一条已存在的条目的内容，条目不存在时，给予错误提示
 4. 对所有的条目列表输出，并可以指定按照条目中的时间，升序或降序列表
 5. 其它自定义的功能（可选，如：可采用模版文件的方式增加、修改服务端的资源）

该App可自行设计，这里给出范例的样式：

```
1 | App [options]
```

a) 该App是独立应用程序，采用命令行方式运行，如有输出，输出到标准终端。

b) 命令行以及释义如下：

-h | --help: 输出对该App的用法帮助

-a | --add: 增加一条新的条目，如果成功，输出成功的消息；如果失败，输出失败的警告。该选项需要配合下列选项(需4个同时联合使用)：

-n | --name={增加的姓名}:

-i | --id={增加的学号}:

-m | --mobile={增加的手机号}

-b | --hobby={增加的兴趣爱好}

-d | --delete={id}: 对编号id的条目删除操作。如果成功, 输出成功的消息; 如果失败, 输出失败的警告

-u | --update={id}: 对编号id的条目修改内容。如果成功, 输出成功的消息; 如果失败, 输出失败的警告。该选项需要配合下列选项(可独立, 也可组合使用):

-n | --name={更新后的姓名}:

-i | --id={更新后的学号}:

-m | --mobile={更新后的手机号}

-b | --hobby={更新后的兴趣爱好}

-l | --list={mode}: 当mode是"descend"时【默认】, 降序排列; 当mode是"ascend"时, 升序排列。输出到终端显示。

c) 以上选项, 均为非强制选项 (即: 如果运行时无选项, 则按照「-list=descend」操作 (因为这是一个安全操作); 如果有选项, 则按照选项操作

d) 互斥选项: -a | -d | -u | -l, 即该4个选项只能同时选择一个。如果选择多个, 警告提示

e) 联合选项: -u id {-n | -i | -m | -b}, 即-u选项必须附带最少后面的四个选项之一才能工作, 否则警告提示

-a {-n -i -m -b}, 即-a选项必须同时附带后面的4个选项, 才能工作, 否则警告提示

f) 长、短项: 单「-」代表短选项, 双「--」代表长选项。长、短项表示的效果相同, 可随意选择使用

g) 调用示例:

```

1  #输出用法帮助:
2  App -h
3  #增加一条新的纪录, 演示了短选项的使用方式:
4  App -a -n "张三" -i "202012345678" -m "13355555555"
   -b "宅家"
5  #输出一条记录, 演示了长选项的使用方式:
6  App --delete=6462773664664
7  #按照时间升序方式输出全部的纪录:
8  App -l ascend

```

```

9  #对某条目修改“手机号”内容，演示了长、短项的混合使用：
10 App -u 6462773664664 --mobile="1592222222222"
11 #错误示例1：-u后缺少条目id，不知道对谁更新
12 App -u -m "13972635342"
13 #错误示例2：不可以同时使用-a、-d选项
14 App -a -n "张三" -i "202012345678" -m "13355555555"
   -b "宅家" -d 6462773664664
15 #错误示例3：添加条目时，缺少-b选项
16 App -a -n "张三" -i "202012345678" -m "13355555555"

```

4. 编写服务端代码，以响应前端的请求。服务端除响应前端的网页请求外，还以API方式提供相应的服务。API服务包括：

- 对条目的新增 (add)
- 对条目的删除 (deleteById)
- 对条目的修改 (updateById)
- 对所有条目的列表 (list)
- 其它自定义的功能 (可选，如：支持模版文件的增加、修改等)

注意：在设计上述api时，需要指定调用的http方法 (/GET /POST /PUT /DELETE)，并充分考虑各种http方法的幂等性要求，并在实现时验证。

友情提示：我们给出的服务端示例代码中，/PUT方法是故意错误的，学生在使用时需改正。

5. 信息资源以JSON方式组织，对信息条目的新增、删除、修改支持模版方式（参照 collection+json，自行定义资源模版）
6. 本实验不限定所采用的编程语言，学生可以根据个人的偏好，自行选择。实验所准备的代码仅供参考

下图为提交信息网页页面的样式（仅供参考，学生可以按照自己的方式完成页面布局和验证错误提示）。

个人信息采集

请在下面输入你的个人信息：

姓名	<input type="text"/>	
学号	<input type="text"/>	
手机	<input type="text" value="777-22545879552"/>	这不是一个正确的手机号码
兴趣	<input type="text"/>	
<input type="button" value="提交"/>		

实验结果评价

实验结果提交内容

本实验需提交以下内容作为成果【实验结果提交到ftp server的个人目录下】：

1. 满足实验要求的完整代码，以压缩包的方式提交【命名为：**dace.zip**】，内含：源代码、测试所采用的数据、测试所获得的结果，文件目录结构与准备材料一致
2. 实验结果说明，命名为：**dace.doc**【word格式】，在压缩包之外独立提交，内容包括：
 - 前端网页操作的截图，每个功能至少一张
 - API测试的语句命令（以curl命令操作，或者postman操作），以及相应的测试结果截图
 - 自定义的资源模版【JSON格式】
 - 其它需要说明的内容（可选）

实验打分依据

1. 代码完整程度。以代码是否能够运行为依据：
 - 能够运行，且结果与dace.doc中描述一致，100分
 - 能够运行，但结果与dace.doc中描述不一致，根据情况给：60~90分
 - 无法运行，0分
 - 未提交，0分
2. 满足实验要求程度：

- 前端测试满足要求程度，按实际情况，60~100分；未提交代码，0分
- API满足要求程度，按实际情况，60~100分；未提交代码，0分
- 资源模版定义的完整性、可行性，按实际情况，60~100分；未提交，0分
- dace.doc的内容完整性，按实际情况，60~100分；；未提交，0分
- 以上4项平均得到第二大项分数

3. 以上两项内容，加权平均，总分以百分制给定。

4. 授课老师根据具体情况，决定是否现场演示。