---------------------------------------------------

计划课程安排：

FUNDAMENTAL(7天)         该阶段已结束

OOP(11天)                 该阶段已结束

JAVASE01(10天)        该阶段已结束

JAVASE02(12天)        该阶段已结束

WEBBASIC(11天)       该阶段已结束

JQUERY(2天)            该阶段已结束

DATABASE(5天)      该阶段已结束

JDBC(2天) ←当前为该阶段第1天

SERVLETJSP(12天)

SPRINGMYBATIS01(8天)

AJAX(2天)

SPRINGMYBATIS02(12天)

LINUX(2天)

NGINX(2天)

REDIS(2天)

T-STORE(5天)

预计合计：105天

预计剩余：48天

注：上述信息随时可能发生更改，仅供参考。

---------------------------------------------------

**2019年5月23日**

**Java就业班-第70天**

**8:30实训室2-早自习**

今日晨讲主题——我没听懂这哥们讲的什么，根本听不见他在讲话，而且他写的代码我也不知道是什么意思，姑且抄下来了就是……

import java.util.Deque;

import java.util.LinkedList;

/\*\*

\* N:表示数到几则跳出

\* M:表示有几个人

\* @author Administrator

\*

\*/

public class Test03 {

public static void main(String[] args) {

test(4,100);

}

public static void test(int N,int M) {

Deque<Integer> list = new LinkedList<>();

for (int i = 0; i <= M; i++) {

list.add(i);

}

System.out.println(list);

int count = 0 ;

while(!list.isEmpty()) {

int i = list.pollFirst();

if(count %N == 0) {

System.out.println(i + ",");

}else {

list.addLast(i);

}

count++;

}

}

}

噢噢！我知道了，原来是约瑟夫环，因为我晚自习没上，所以项目经理讲到这一题的时候我没在。留个心眼了，的赶紧把这一题学习一下才行。

注：百度百科约瑟夫环有java和C的实现代码。

**9:00实训室2-正课**

开始上课。

——视图

-视图的分类：

1.简单视图：创建视图的子查询中不包含去重、分组、关联查询、聚合函数的视图称为简单视图，可以对视图中的数据进行增删改查操作

2.复杂视图：和简单视图相反，只能对视图中的数据进行查询操作

-创建查询每个部门平均工资，最高工资，最低工资，工资总和，部门人数的视图

Create view v\_emp\_info as (select deptno,avg(sal),max(sal),min(sal),sum(sal),count(\*) from emp group by deptno);

——对简单视图进行增删改查操作

-操作方式和table一样

Insert into v\_emp\_10 (empno,ename,deptno) values(10010,'Kamui',10);//插入成功

Insert into v\_emp\_10 (empno,ename,deptno) values(10011,'Subaru',20);//数据污染

-数据污染：当往视图中添加一条视图中不可见原表中却可见的数据成为数据污染，可以通过创建视图

时添加with check option解决。

Create view v\_emp\_20 as(select \* from emp where deptno=20)with check option;

insert into v\_emp\_20 (empno,ename,deptno)values(10012,'Kamui',10);//数据污染失败

insert into v\_emp\_20 (empno,ename,deptno)values(10012,'Kamui',20);//成功

-修改和删除只能操作视图中存在的数据

update v\_emp\_10 set ename='KKK' where ename='Kamui';//成功

update v\_emp\_10 set ename='SSS' where ename='Subaru';//成功

delete from v\_emp\_10 where empno=10010;成功

delete from v\_emp\_10 where empno=10011;失败

-创建或替换视图

create or replace view v\_emp\_10 as (select ename,sal from emp where deptno=10);

-删除视图

drop view v\_emp\_10;

-别名：创建视图时使用了别名，则对视图操作时只能使用别名

create view v\_emp\_30 as (select ename name from emp where deptno=30);

select ename from v\_emp\_30; 报错不认识ename 应该用name

**9:41实训室2-正课**

我有点忍不住了，这课上的好难受，我现在看到这个讲师的脸和听到他的声音就毛躁。

我还是休息一下得了。

**10:10实训室2-课间休息**

项目经理过来提到今明两天的课程和之后框架课程的事情，因为数据库的全部课程计划是在今明两天内结束，所以下周开始就是别的讲师讲课了，并且从下周开始往后就全都是java相关的课程内容了，再没有其他新内容了。（终于可以摆脱这个沙雕讲师了！）

然后，关于后续框架课程的问题，因为之前也提到过我们这一个班的人都要提前毕业，所以框架的课程可能讲不完。项目经理表示这一点可以不用担心，因为来这里招人的企业都很清楚我们当前的水平，并且各公司单位基本上都有公司自己的框架，所以在工作上还是要接着学习的。

…………

我怎么听着觉得很不靠谱的感觉……有种说不上来的违和感。

**10:30实训室2-正课**

——约束

- 约束： 创建表时给字段添加的限制条件

1. 主键约束 primary key

-限制主键的值唯一且非空

2. 非空约束 not null

-限制字段的值不能为null

create table t1(id int,age int not null);

insert into t1 values(1,20); //成功

insert into t1 values(2,null);//失败

3. 唯一约束 unique

-限制字段的值不能重复

create table t2(id int,age int unique);

insert into t2 values(1,20);//成功

insert into t2 values(2,20);//失败

4. 默认约束 default

- 给字段添加默认值

create table t3(id int,age int default 20);

insert into t3 values(1,10);

insert into t3 values(2,null);

insert into t3 (id) values (3);//默认值生效

5. 外键约束

-外键：用于建立关系的字段称为外键

-添加外键约束的字段值可以为null，可以重复，但是不能是关联表中不存在的数据，如果建立了关系被关联的数据不能先删除，被关联的表不能先删除

-测试：

create table mydept(id int primary key auto\_increment,name varchar(10));

create table myemp(id int primary key

auto\_increment,name varchar(10),dept\_id int,

constraint fk\_dept foreign key(dept\_id) references mydept(id));

- 格式介绍：constraint 约束名 foreign key(外键字段名) references 被依赖的表名(被依赖的字段名)

insert into mydept values(null,'DPT'),(null,'ACT');

insert into myemp values(null,'Kamui',1);//成功

insert into myemp values(null,'Subaru',3);//失败

delete from mydept where id=1; //失败 不能先删除

delete from myemp where id=1; //成功

delete from mydept where id=1; //成功

drop table mydept; //失败 被依赖的表不能先删除

drop table myemp; //成功

drop table mydept;//成功

**10:50实训室2-正课**

——索引

索引是数据库中提高查询效率的技术，类似于字典的目录。

-索引的作用：由于数据是零散的保存在每一个磁盘中的，如果不使用索引，查询数据时会挨个遍历每一个磁盘块查找数据，如果使用了索引，磁盘块会以树桩结构保存，查找数据时能够大大降低访问的磁盘块的数量，从而提高查询效率。

-索引是否必须：如果数据量小，反而会降低查询效率，所以仅针对大量数据访问时才使用索引技术。

-索引是否越多越好：因为索引会占用磁盘空间，只针对查询时频繁使用的字段创建索引

-创建索引的格式：create index 索引名 on 表名(字段名(?长度));

1.创建索引前先查询一次 看耗费时间 1.14sec

Select \* from item2 where title='100';

2.创建索引

create index i\_item\_title on item2(title);

3.验证 0.18sec

select \* from item2 where title='100';

-查看索引

如果给表添加主键约束，会自动创建主键的索引

1.格式：show index from 表名;

Show index from item2;

-删除索引

Drop index 索引名 on 表名;

Drop index i\_item\_title on item2;

-复合索引：

通过多个字段创建索引

Create index i\_item\_title on item2(title,price);

Show index from item2;//创建了2个索引

- 总结：

1. 索引是数据库中提高查询效率的技术类似于目录

2. 因为索引会占用磁盘空间不是越多越好，只针对查询时频繁使用的字段创建索引。

3. 数据量越大索引效果越明显，如果数据很少可能会降低查询效率。

**14:02实训室2-正课**

——事务

-事务是数据库中执行同一业务多条SQL语句的工作单元，保证多条SQL语句全部执行成功或全部执行失败。

-相关SQL：

开启事务begin

提交事务commit

回滚事务rollback

保存回滚点savepoint xxx

回滚到某个回滚点rollback to xxx;

-事务的四大特性，ACID特性，ACID特性是保证事务正确执行的四大基本要素

1.Atomicity :原子性，最小不可拆分，保证全部成功或全部失败

2.Consistency :一致性，从一个一致状态到另一个一致状态

3.Isolation :隔离性，多个事物之间互不影响

4.Durability :持久性，事务提交后数据保存到磁盘中持久生效。

※事务的四大特性必须要背记下来。

——group\_concat()方法 组连接

1.查询每个部门的员工姓名和工资,要求每个部门的数据显示在一行。

Select deptno,ename,sal from emp group by deptno;

↓

Select deptno,group\_concat(ename,':',sal) from emp group by deptno;

2.查询每种工作的员工姓名，要求工作与员工姓名显示在同一行。

Select job,group\_concat(ename) from emp group by job ;

**14:22实训室2-正课**

——JDBC

（总算是开始讲JDBC了，虽然之前的日记中在自学时写了一部分笔记，

但这里就当是从头开始学，笔记也再写一遍吧。）

-Java DataBase Connectivity:Java数据库连接，JDBC实际上就时Sun公司提供的一套和数据库交换的API(Application Program Interfae应用程序编程接口)

-为什么使用JDBC:因为Java开发语言需要连接多种数据库，为了避免Java程序员每种数据库都需要学习一套新的API，sun公司提出了一个JDBC接口，让各个数据库厂商根据JDBC接口写实现类。

**15:30实训室2-正课**

代码:

package jdbc;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.Statement;

public class Demo01 {

public static void main(String[] args) throws Exception {

//1.注册驱动

Class.forName("com.mysql.jdbc.Driver");

//2.获取连接对象

Connection conn = DriverManager.getConnection(

"jdbc:mysql://127.0.0.1:3306/newdb3",

"root", "123456");

System.out.println(conn);

//3.创建SQL执行对象

Statement stat = conn.createStatement();

//4.执行SQL

String sql = "create table jdbct1"

+ "(id int,name varchar(10));";

stat.execute(sql);

//5.关闭连接

conn.close();

System.out.println("执行完成！");

}

}

**15:48实训室2-正课**

——Statement执行SQL语句对象

-execute() 可以执行任意SQL语句，但是推荐执行DDL(数据定义语言)

-executeUpdate() 推荐执行增删改SQL

-executeQuery() 执行select查询SQL语句

测试execute()代码：

package jdbc;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.Statement;

public class Demo02 {

public static void main(String[] args) throws Exception {

Class.forName("com.mysql.jdbc.Driver");

Connection conn = DriverManager.getConnection(

"jdbc:mysql://127.0.0.1:3306/newdb3", "root", "123456");

Statement st = conn.createStatement();

String sql = "insert into jdbct1 values(1,'Tom')";

st.execute(sql);

conn.close();

System.out.println("执行完毕！");

}

}

**16:20实训室2-正课**

测试executeQuery()代码：

package jdbc;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

public class Demo03 {

public static void main(String[] args) throws Exception {

//注册取得

Class.forName("com.mysql.jdbc.Driver");

//获取连接对象

Connection conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/newdb3", "root", "123456");

//SQL执行对象

Statement stat = conn.createStatement();

String sql = "select \* from emp";

ResultSet rs = stat.executeQuery(sql);

//遍历结果集

while(rs.next()) {

String name = rs.getString("ename");

double sal = rs.getDouble("sal");

System.out.println(name + ":" + sal);

}

conn.close();//关闭连接

}

}

**16:42实训室2-正课**

封装一个JDBC工具类：

package jdbc;

import java.sql.Connection;

import java.sql.DriverManager;

/\*\*

\* JDBC工具类

\* @author Administrator

\*

\*/

public class DBUtils {

/\*

\* getConn()连接数据库，返回Connection对象conn;

\*/

public static Connection getConn() throws Exception {

//注册驱动

Class.forName("com.mysql.jdbc.Driver");

//获取连接对象

Connection conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/newdb3", "root", "123456");

return conn;

}

}

**16:46实训室2-正课**

-注解@Test的作用

@Test的使用 是该方法可以不用main方法调用就可以测试出运行结果，是一种测试方法

一般方法都需要有main方法调用才能执行，注意被测试的方法必须是public修饰的

测试代码：

package jdbc;

import java.sql.Connection;

import java.sql.ResultSet;

import java.sql.Statement;

import org.junit.Test;

public class Demo04 {

@Test

public void test01() {

System.out.println("方法1执行了");

try(Connection conn = DBUtils.getConn()) {

//创建SQL执行对象

Statement stat = conn.createStatement();

ResultSet rs = stat.executeQuery("select \* from emp");

//遍历

while(rs.next()) {

String name = rs.getString("ename");

System.out.println(name);

}

} catch (Exception e) {

e.printStackTrace();

}

}

@Test

public void test02() {

System.out.println("方法2执行了");

}

@Test

public void test03() {

System.out.println("方法3执行了");

}

}

**17:20实训室2-正课**

在项目目录下的resoures中新建一个file文件：jdbc.properties

内容：

name=\u5F20\u4E09

age=18

driver=com.mysql.jdbc.Driver

url=jdbc:mysql://127.0.0.1:3306/newdb3

username=root

password=123456

※在文件中写的中文汉字会自动转为ISO码。

测试Properties读取文件：

package jdbc;

import java.io.IOException;

import java.io.InputStream;

import java.util.Properties;

public class Demo05 {

public static void main(String[] args) throws IOException {

Properties prop = new Properties();

//获取文件输入流

InputStream ips = Demo05.class.getClassLoader().getResourceAsStream("jdbc.properties");

//加载文件流

prop.load(ips);

//获取数据

String name = prop.getProperty("name");

String age = prop.getProperty("age");

System.out.println(name + ":" + age);

}

}

**17:36实训室2-正课**

更新工具类代码：

package jdbc;

import java.io.InputStream;

import java.sql.Connection;

import java.sql.DriverManager;

import java.util.Properties;

/\*\*

\* JDBC工具类

\* @author Administrator

\*

\*/

public class DBUtils {

/\*

\* getConn()连接数据库，返回Connection对象conn;

\*/

public static Connection getConn() throws Exception {

Properties prop = new Properties();

//获取文件输入流

InputStream ips = DBUtils.class.getClassLoader().getResourceAsStream("jdbc.properties");

prop.load(ips);

String username = prop.getProperty("username");

String password = prop.getProperty("password");

String driver = prop.getProperty("driver");

String url = prop.getProperty("url");

//注册驱动

Class.forName(driver);

//获取连接对象(读取配置文件)

Connection conn = DriverManager.getConnection(url, username, password);

return conn;

}

}

**17:52实训室2-下课时间**

今天的课程已结束。这个沙雕讲师还有明天最后一天课。

回宿舍。

**19:10宿舍-休息**

终于是熬到头了，明天再熬过去就不用再受这个讲师的折磨了。

关于今天上午课间休息的时候，项目经理到教室来说的事情，之前是只和我单独提到过，这次是公开在班里说明了部分详细情况。

在这周一晚上搞得模拟考试，目的就是为了挑选成绩较好的学员，把测试的结果发给招人企业，好让学员赶上下个月底的第一波企业面试。难怪我无论怎么推算课程计划时间都觉得时间安排不合理，原来是机构根本就没有打算让我们这批2月班的学员学完全部课程，准备让我们跟着1月班的学员一起推出去就业，提前完成签订的合约。这就有些操蛋了，明摆着赶人走啊！

根据我的调查，推测出机构要这么做的理由很简单，就是因为我们班和1月班人数较多，占用了几间大教室，导致后来的4-6月班招的人没有足够的座位，这一点直接地导致了机构的收益下降，所以直接无视课程进度要提前将我们赶走。而与机构所签订的合同中没有明确说明机构要保证推荐的就业单位待遇标准如何，所以在明知我们当前水平如何的情况下还愿意来机构招人的企业，只可能是来寻求廉价劳动力的黑心企业了……

对此我决定继续观察，目前我对收入的需求还没有到走投无路的地步，所以我不准备参加下个月底的第一波面试。即便是企业来招人，也不可能一次性把一个班30-40多号人一次性招走吧。那么留下来的人必定是需要继续上课的，我就不信机构敢让还没就业也没上完课的学员回去呆着。

我也正好可以利用那一段时间巩固一下基础知识，顺便看看机构还有什么花招没有使出来的。

不过我当前的经济条件仅允许我撑到7月底，在那之前收入的问题也是必须要解决的了。否则等到贷款压下来，可就把自己也给搭进去了。

今日感想总结：只要是写Java代码，什么都好说！

今日评分：7.5/10分

今天就到这里，明天继续投稿。

感谢支持！

================================================================

**Acer程序员群：790482850**

**up主的java学习日记word文件及全部代码（随日记同步更新）：**

Adress：github.com/wt62635