实验报告

江刻优 19307130124 & 杨之恒 19307130117

一、实验题目

Demons Go In FDU(一起来 FDU 捉妖) 数据管理系统设计与实现

二、开发环境

(1)操作系统 windows10

(2)数据库管理软件: PostgreSQL 12.7

(3)前端: html(.ejs)-ajax

(4)后端: Nodejs-express

三、数据库设计

1.ER 图

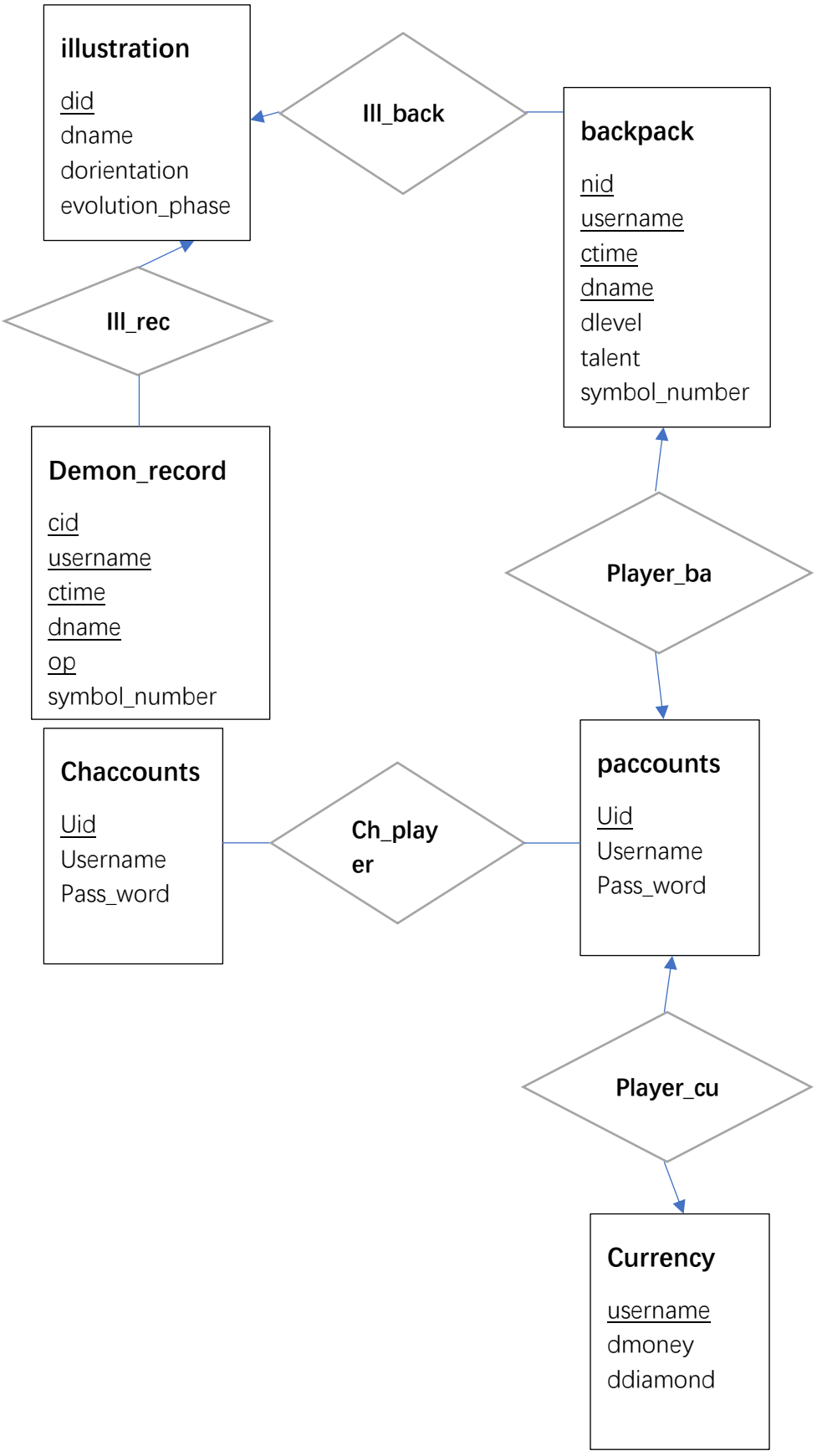
illustration
did
dname
dorientation
evolution_phase
Ill_back
backpack
nid
username
ctime
dname
dlevel
talent
symbol_number
Ill_rec
Demon_record
cid
username
ctime
dname
op
symbol_number
Player_ba
Chaccounts
Uid
Username
Pass_word
Ch_play
er
paccounts
Uid
Username
Pass_word
Player_cu
Currency
username
dmoney
ddiamond

2.

数据库物理设计

Backpack:

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	nid	integer			是	是
✎	🗑	username	character varying	20		是	是
✎	🗑	ctime	date			是	是
✎	🗑	dname	character varying	5		是	是
✎	🗑	dlevel	integer			否	否
✎	🗑	talent	integer			否	否
✎	🗑	symbol_number	integer			否	否

Chaccounts

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	uid	integer			是	是
✎	🗑	username	character varying	20		是	否
✎	🗑	pass_word	character varying	20		是	否

Currency:

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	username	character varying	20		是	是
✎	🗑	dmoney	integer			否	否
✎	🗑	ddiamond	integer			否	否

Demon_record:

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	username	character varying	20		是	是
✎	🗑	dmoney	integer			否	否
✎	🗑	ddiamond	integer			否	否

illustration:

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	dname	character varying	5		是	否
✎	🗑	evolution_phase	character varying	1		是	否
✎	🗑	dorientation	character varying	20		是	否
✎	🗑	did	integer			是	是

Paccounts:

列 +

		名称	数据类型	长度/精度	规模	不为 NULL?	主键?
✎	🗑	uid	integer			是	是
✎	🗑	username	character varying	20		是	否
✎	🗑	pass_word	character varying	40		是	否

3.数据库实现代码

创建 backpack

```
CREATE TABLE public.backpack
(
    nid integer not null,
    username character varying(20) not null,
    ctime date default current_date,
    dname character varying(5),
    dlevel integer,
    talent integer,
    symbol_number integer,
    primary key (nid, username, ctime, dname),
```

```
    check(dlevel >= 1 and dlevel <= 50 and talent >= 1 and talent <= 32)
);
```

创建 chaccounts

```
CREATE TABLE public.chaccounts
(
    uid integer primary key not null,
    username character varying(20) not null,
    pass_word character varying(20) not null,
);
```

创建 currency

```
CREATE TABLE public.currency
(
    username character varying(20) primary key NOT NULL,
    dmoney integer,
    ddiamond integer,
);
```

创建 demon_record

```
CREATE TABLE public.demon_record
(
    cid integer not null,
    username character varying(20) not null,
    ctime date default current_date,
    dname character varying(5),
    op character varying(10) not null,
    symbol_number integer,
    primary key (cid, username, ctime, dname, op),
    check(op in ('catch', 'abandon'))
);
```

创建 illustration:

```
CREATE TABLE public.illustration
(
    dname character varying(5) NOT NULL,
    evolution_phase character varying(1) NOT NULL,
    dorientation character varying(20) NOT NULL,
    did integer primary key NOT NULL
)
```

创建 paccounts

```
CREATE TABLE public.paccounts
(
```

```
    uid integer primary key not null,
    username character varying(20) not null,
    pass_word character varying(20) not null,
);
```

创建视图 op_of_users

```
create view op_of_users as(
    select cid, username, ctime, dname, op
    from demon_record);
```

函数 abandon_demon:

```
create function abandon_demon(nid1 integer, dn character varying(5), ct date,
usr character varying(20))
returns varchar(20) as $$
    declare stat varchar(20);
    declare sn integer;
    declare lev integer;
    declare did1 integer;
    declare dn2 varchar(5);
begin
    sn := (select symbol_number from backpack where nid = nid1);
    lev := (select dlevel from backpack where nid  = nid1);
    did1 := (select did from illustration where dname = dn);
    if(did1 % 2 = 0)
        then did1 := did1 - 1;
    else did1 := did1 + 1;
    end if;
    dn2 := (select dname from illustration where did = did1);

    delete from backpack
    where nid = nid1;

    update currency
    set dmoney = dmoney + lev * 300
    where username = usr;

    update backpack
    set nid = nid - 1
    where nid > nid1;

    if(dn in (select dname from backpack where username = usr) or dn2 in (select
dname from backpack where username = usr))
        then update backpack
                set symbol_number = symbol_number + lev
```

```
                where username = usr and dname in (dn, dn2);
             insert into demon_record
                values((select max(cid) + 1 from demon_record), usr,
current_date, dn, 'abandon', sn + lev);
    else update currency
            set dmoney = dmoney + sn * 200
            where username = usr;
          insert into demon_record
                values((select max(cid) + 1 from demon_record), usr,
current_date, dn, 'abandon', 0);
    end if;

    stat := 'abandon one!';
    return stat;

end
$$
language plpgsql;
```

函数 catch_demon

```
create function catch_demon(nid1,integer, num integer, usr character varying(20))
returns varchar(20) as $$
declare stat varchar(20);
    declare nam character varying(5);
    declare nam1 character varying(5);
    declare n integer;
begin
        select dname into nam from illustration where did = num;
        select dname into nam1 from illustration where did = num + 1;

        if(nam in (select dname from backpack where username = usr) or nam1 in
(select dname from backpack where username = usr))
            then select distinct symbol_number into n from backpack where dname
in (nam, nam1) and username = usr;
        else  n := 10;
        end if;

        insert into backpack
            values((select max(nid) + 1 from backpack), usr, current_date, nam,
1, random()*31 + 1,n + 1);
        insert into demon_record
            values((select max(cid) + 1 from demon_record), usr, current_date,
nam, 'catch', n + 1);
        update backpack
```

```
        set symbol_number = n + 1
        where dname in (nam, nam1) and username = usr;

        update currency
        set dmoney = dmoney + 400
        where username = usr;

        stat := 'catch one !';
        return stat;
end
$$
language plpgsql;
```

函数 dmoney_ddiamond

```
create function dmoney_ddiamond(usr character varying(20), money integer)
returns varchar(20) as $$
    declare stat varchar(40);
begin
    if((select dmoney from currency where username = usr) < money)
    then stat := 'no sufficient dmoney!';
    else update currency
            set dmoney = dmoney - money, ddiamond = ddiamond + money / 10
            where username = usr;
            stat := 'success!';
    end if;

    return stat;
end
$$
Language plpgsql;
```

函数 sign_up

```
create function sign_up(usr character varying(20), pa character varying(20))
returns varchar(20) as $$
    declare stat varchar(40);
    declare uid1 integer;
    /*declare did1 integer;*/
begin
    uid1 := (select max(uid) from paccounts) + 1;

    insert into paccounts
        values(uid1, usr, pa);

    if (uid1 in (select uid from paccounts))
```

```
        then stat := 'Welcome to demonsgo!';
    else stat := 'username already existed!';
    end if;
    return stat;
end
$$
Language plpgsql;
```

函数 upgrade_demon

```
create function upgrade_demon(dn1 character varying(5), nid1 integer, usr
character varying(20))
returns varchar(20) as $$
    declare stat varchar(40);
    declare lv integer;
    declare did1 integer;
    declare did2 integer;
    declare dn2 varchar(5);
    declare syn integer;
    declare dm integer;
    declare dd integer;
begin
    did1 := (select did from illustration where dname = dn1);
    if did1 % 2 = 0
        then did2 := did1 - 1;
    else did2 := did1 + 1;
    end if;
    dn2 := (select dname from illustration where did = did2);

    syn := (select symbol_number from backpack where nid = nid1);
    lv := (select dlevel from backpack where nid = nid1);
    dm := (select dmoney from currency where username = usr);
    dd := (select ddiamond from currency where username = usr);

    if lv < 29
        then if syn < 3
                    then stat := 'no sufficient dsymbol!';
             elseif dm < 500
                    then stat := 'no suffcient dmoney!';
             else update currency
                        set dmoney = dmoney - 500
                        where username = usr;
                  update backpack
                        set dlevel = dlevel + 1
                        where nid = nid1;
```

```
                    update backpack
                        set symbol_number = symbol_number - 3
                        where dname in (dn1, dn2) and username = usr;
                    stat := 'upgrade success!';
                    end if;
    elseif lv = 29
        then if syn < 30
                    then stat := 'no sufficient dsymbol!';
             elseif dm < 500
                    then stat := 'no sufficient dmoney!';
             else update currency
                        set dmoney = dmoney - 500
                        where username = usr;
                    update backpack
                        set dlevel = dlevel + 1, dname = dn2
                        where nid = nid1;
                    update backpack
                        set symbol_number = symbol_number - 30
                        where dname in (dn1, dn2) and username = usr;
                    stat := 'upgrade success!';
                    end if;
    else if syn < 5
                    then stat := 'no sufficient dsymbol!';
             elseif dm < 1000
                    then stat := 'no sufficient dmoney!';
             elseif dd < 10
                    then stat := 'no sufficient ddiamond!';
             else update currency
                        set dmoney = dmoney - 1000, ddiamond = ddiamond - 10
                        where username = usr;
                    update backpack
                        set dlevel = dlevel + 1
                        where nid = nid1;
                    update backpack
                        set symbol_number = symbol_number - 5
                        where dname in (dn1, dn2) and username = usr;
                    stat := 'upgrade success!';
                    end if;
    end if;

    return stat;
end
$$
Language plpgsql;
```

触发器函数 if_exist()的主体代码

```
declare did1 integer;
declare did2 integer;
begin
    did1 := ceil((random()*24) + 0);
    did2 := 2 * did1 + 1;
    if((select count(username) from paccounts where username = new.username) =
2)
        then delete from paccounts
                where uid = new.uid;
    else insert into currency
            values(new.username, 10000, 1000);
          insert into backpack
            values((select   max(nid)   from   backpack)   +   1,new.username,
current_date, (select dname from illustration where did = did2), 1, random()*32
+ 1, 11);
          insert into demon_record
            values((select  max(cid)  +  1  from  demon_record),  new.username,
current_date, (select dname from illustration where did = did2), 'catch', 11);
    end if;
    return new;
end
```

四、系统设计（相应的 sql 代码）

1.展示所有用户

```
select *  from paccounts
```

2.展示妖灵背包（illustration）

```
select did, dname, evolution_phase, dorientation
    from illustration
```

3.展示该用户所有妖灵

```
select dname, dlevel, talent, symbol_number, ctime count(all dname)
as damount
    from backpack
    where username = $1
```

4.查看新捕捉的妖灵

```
select ctime, dname, dlevel, talent
    from backpack
    where nid = (
```

```
        select max(nid)
        from backpack
    )
```

5.捕捉妖灵

```
select catch_demon($1, $2)
```

6.用户注册

```
select sign_up($1, $2)
```

7.放生妖灵

```
select abandon_demon($1, $2, $3, $4)
```

8.升级妖灵

```
select upgrade_demon($1, $2, $3)
```

9.dmoney 和 ddiamond 的转换

```
select dmoney_to_ddiamond($1, $2)
```

10.添加新妖灵到妖灵背包

```
insert into illustration
        values($1, $2, $3, $4);
```

11.更新妖灵的属性（dorientation）

```
update illustration
        set dorientation = $1
        where did = $2;
```

12.查看该用户的记录

```
select cid, username, ctime, dname, op
    from demon_record
    where username = $
```

13.查看时间内该用户的记录

```
select cid, username, ctime, dname, op
    from demon_record
    where (ctime between $1 and $2) and (username = $3)
```

14.展示该用户的 currency

```
select dmoney, ddiamond
from currency
where username = $1
```

五、特色和创新点

前后端: 使用了大量 Nodejs 的异步执行特性，提高了服务器效率

SQL: 使用了大量函数包装，方便了 API 调用与返回信息获取

六、实验分工

江刻优: SQL 库设计与创立、SQL 操作编写、提供 SQL 与 web 的 API

杨之恒: 前端 html-ajax 编写、后端 Nodejs-express 服务器框架搭建

共同完成: 项目需求设计

七、提交文件说明

文件夹 src {

主要包含前后端源代码

README.md: Nodejs 环境配置指导

文件夹 node_modules: Nodejs-express 环境配置

package-lock.json: Nodejs-express 环境配置

package.json: Nodejs-express 环境配置

文件夹 public {

主要包含服务器运行所需静态文件

文件夹 pic: html(.ejs)引用图片

safe_md5.js: 密码加密方法 hex_hmac_md5()、区间随机数生成方法 randomNUM()、日期格式转换方法 getNowFormatDate()

sql.js: SQL 提供 API，为服务器运行所需调用 SQL 语句

}

文件夹 views: 渲染 web 页面的 html 文件(.ejs)

}

文件夹 SQL {

项目需求: 项目设计方案

文件夹 version: SQL 项目阶段性版本

demonsgo: SQL 项目最终使用版本

}

八、实验总结

熟悉了 SQL 的综合运用，了解了为后端提供接口的方法与后端连接数据库的方法，以及学习了前端的简单制作。