

职工考勤管理信息系统

Employee Attendance Management Information System
(EAMIS)

高级数据库
专案项目

目录

阶段一.....	3
需求分析.....	3
背景.....	3
目标.....	3
功能分析.....	3
性能需求分析.....	3
应用范例.....	3
商业价值.....	4
概念设计.....	5
概念模型 (E-R 图)	5
流程图.....	6
阶段二.....	7
逻辑结构.....	7
实体与属性.....	7
关联.....	7
数据字典.....	7
关系型数据库设计.....	7
物理设计.....	8
功能设计.....	10
阶段三.....	14
数据库创建.....	14
创建表空间.....	14
创建用户.....	14
用户权限管理.....	14
创建表.....	15
数据库初始化.....	17
添加数据.....	17
增删改查样例.....	23
关于业务的 6 个查询语句.....	25
阶段四.....	29
技术选型.....	29
项目结构.....	30
如何启动.....	33
程序运行.....	34
GUI 逻辑结构	52

阶段一

阶段一：附录中有 16 个数据库应用范例，每位同学将选择一个应用范例作为本专案项目的题目。附录中的范例只有简单的数据描述，同学必须以这些数据描述为基础，针对该范例写一个 500 字以上的『故事』，简述范例情节及需要的实体与关联，并详述应用范例的操作程序与商业逻辑。

需求分析

背景

职工考勤管理信息系统，是反应公司员工的出勤情况，人员流动，工作状态，是管理职工的有效工具。本系统面向于职工，对人员基本情况进行管理。只能由公司内部操作和查看数据库中的数据。

目标

本系统对外部封闭，不允许外部人员访问公司考勤系统的数据库。程序实现数据库数据的直观显示，为职工提供便利的出勤情况登记，提供出勤记录以便职员之间相互监督，方便了管理层了解公司的人员情况。

功能分析

项目提供以下功能：职员登录及注册，职工信息管理，出勤记录管理，出差信息管理，请假信息管理，加班信息管理

- 职员登录及注册：职员有权限管理自己的基本信息，出勤打卡，填写出差、请假、加班信息。管理员有权利对所有职工进行管理，可以进行职员注册的操作。
- 职工信息管理：提供职工的基本信息，出勤信息，请假信息，出差信息，加班信息，可供全体成员查看。
- 出勤记录：提供所有员工签到签退信息。
- 出差信息：仅可填写当前已登录的用户的出差申请。提供当日出差人员信息。
- 请假信息：仅可填写当前已登录的用户的请假申请。提供当日请假人员信息。
- 加班信息：仅可填写当前已登录的用户的加班申请。提供当日加班人员信息。

性能需求分析

系统易操作性：项目管理系统应该做到操作简单，界面友好，使得用户可以快速上手使用，不受到专业知识的限制。

系统可维护性：由于系统涉及的信息比较多，数据库中的数据需定期修改，系统可利用的空间及性能也随之下降，为了使系统更好地运转，管理员可以对系统数据及一些简单的功能进行独立的维护及调整。

应用范例

张三是一家公司新来的职员，他需要通过管理员注册账户。管理员使用管理员账号登录了系统，进入了添加账号页面，输入了张三的名字，电话，职位等等的个人信息，完成了职员的添加操作。张三从管理员手中得到了他的公司编号。张三随后登录系统，进行了当日的签到打卡。在打卡页面中，张三看到隔壁工位的漂亮小姐姐李红是公司第一个来打卡的

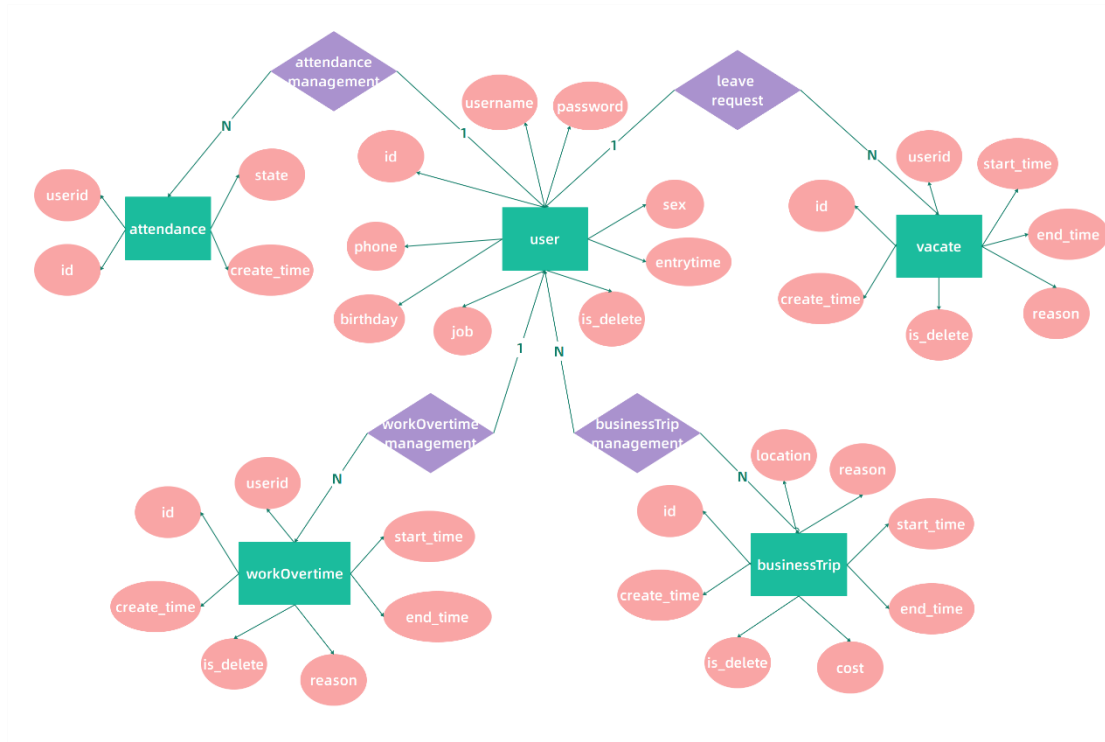
人。随后，张三的上级王五安排他下周准备去北京出差，张三就在系统上提交了为期一周的北京出差申请。在出差表中，他发现李红和他一起出差，张三乐坏了，点开了李红的个人资料，但是发现她已经四十多岁了，顿时心灰意冷。他点开了请假页面，准备请假明天一天带老婆出去散心。到了傍晚，张三准备签退，但是看到了他的上级王五申请了加班，但是他昨天旷工了一天，为了留下一个好印象，张三也申请了加班，并且偷偷的把王五旷工的事情反馈给了老板。一个月以后，老板看到在一个月里，李红每天都是最早来上班打卡的，便给她提高了工资，然后就规定了每天工作时间最久的人奖励一百。并且由于王五旷工一天，扣三百。老板发现，开发部是全年加班数量最多的部门，销售部是出差最多的部门，营销部是最闲的部门，都是迟到早退还不加班，就扣除了全年的奖金。

商业价值

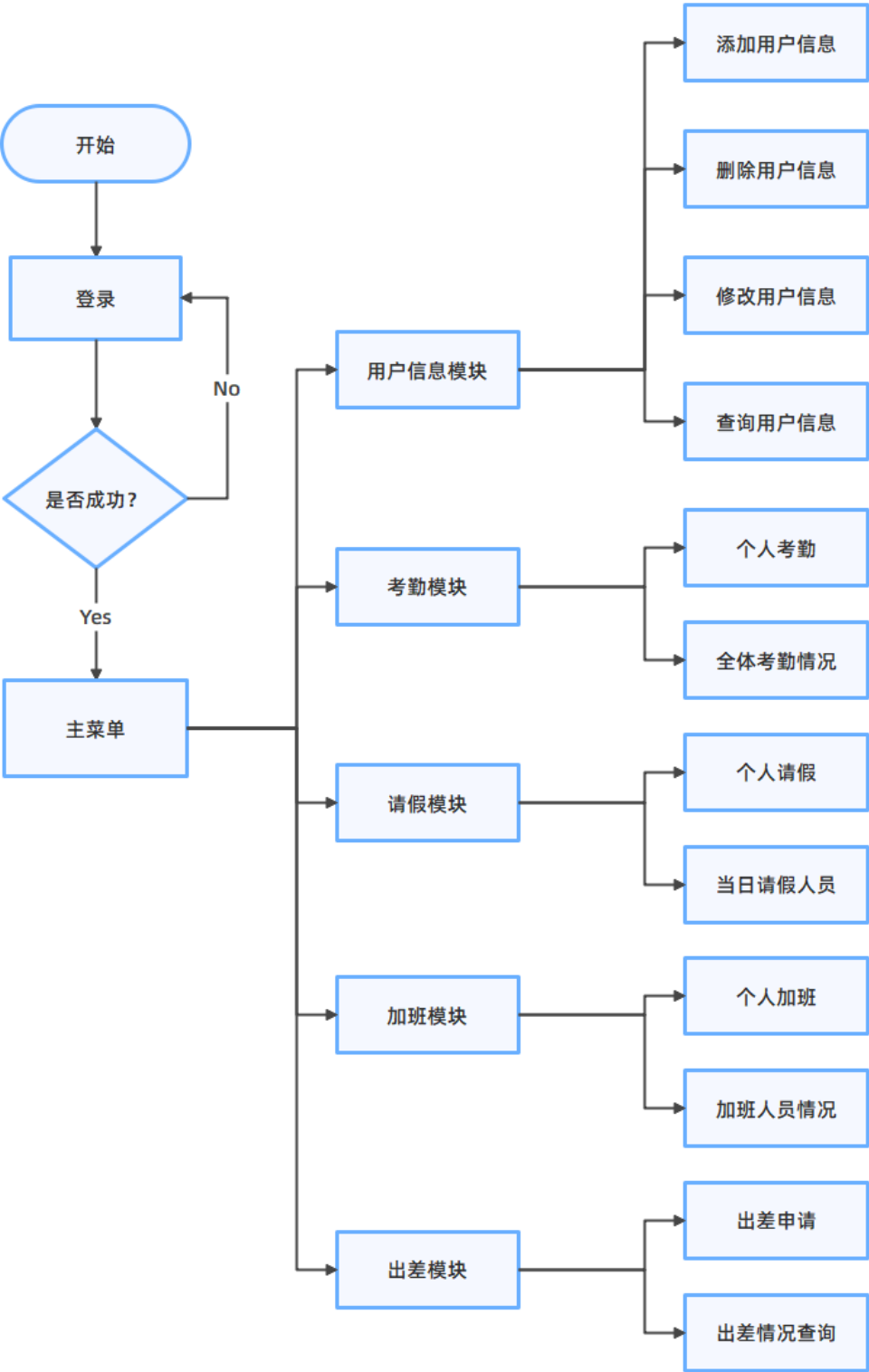
本系统极大的方便了管理者对于公司人员调度的管理，便于对员工的精神面貌和工作状态做出评估，提高了团队的积极性，自觉性，降低了团队管理的时间成本，操作成本，极大地提高了公司的整体效益。

概念设计

概念模型（E-R 图）



流程图



阶段二

依据阶段一的成果报告，将范例中的数据细化与深化，定义该范例的数据库模式；同时，解释应用情节与功能需求。基本的功能包含数据新增、删除、修改、查询等操作，请描述这些操作的情节与步骤。

逻辑结构

实体与属性

用户：用户 id，用户名，用户密码，性别，电话，生日，职位，入职时间

考勤记录：用户 id，考勤状态，考勤时间

出差记录：用户 id，出差地点，出差原因，出差开始时间，出差结束时间，出差花费

请假记录：用户 id，请假原因，请假时间，销假时间

加班记录：用户 id，加班理由，加班开始时间，加班结束时间

关联

用户与考勤记录的关系：一对多。一个用户可以有多条考勤记录，一个考勤记录只能有一个用户。

用户与出差记录的关系：多对多。一个用户可以有多条出差记录，一个出差记录可以有多个用户。

用户与请假记录的关系：一对多。一个用户可以有多条请假记录，一个请假记录只能有一个用户。

用户与加班记录的关系：一对多。一个用户可以有多条加班记录，一个加班记录只能有一个用户。

数据字典

关系型数据库设计

1) : 用户表

用户 id	用户名	密码	电话	性别	生日	职位	入职时间	是否离职
-------	-----	----	----	----	----	----	------	------

2): 出勤表

出勤 id	用户 id	状态	创建时间
-------	-------	----	------

3): 请销假表

请销假 id	用户 id	请假原因	请假开始时间	请假结束时间	是否删除	创建时间
--------	-------	------	--------	--------	------	------

4): 加班表

加班 id	用户 id	加班原因	加班开始时间	加班结束时间	是否删除	创建时间
-------	-------	------	--------	--------	------	------

5): 出差表

出差 id	出差原因	出差地点	出差花费	出差开始时间	出差结束时间	是否删除	创建时间
-------	------	------	------	--------	--------	------	------

6): 出差用户关系表

关系 id	出差 id	用户 id
-------	-------	-------

物理设计

数据字典包括的项目有数据项、数据结构、数据流、数据存储、加工逻辑和外部实体。可使用一些符号来表示数据结构、数据流和数据存储的组成。

1): 用户表

数据元素	数据类型	数据长度	数据描述
id	char	19	用户 id
username	varchar2	64	用户名
password	varchar2	64	密码
phone	char	11	电话
sex	int	1	性别
birthday	date		生日
job	varchar2	64	职位
entrytime	date		入职时间
state	int	1	状态 (0: 删除) (1: 在职) (2: 异常)

主键: id

2): 出勤表

数据元素	数据类型	数据长度	数据描述
id	char	19	出勤 id
user_id	char	19	用户 id
state	int	1	状态 (0: 提早) (1: 正常) (2: 迟到)
create_time	date		创建时间

主键: id

外键: user_id -> 用户表中的 id

3): 请销假表

数据元素	数据类型	数据长度	数据描述
------	------	------	------

id	char	19	请销假 id
user_id	char	19	用户 id
reason	clob		请假原因
start_time	date		请假开始时间
end_time	date		请假结束时间
状态	int	1	状态 (0: 删除) (1: 未开始) (2: 进行中) (3: 结束)
create_time	date		创建时间

主键: id

外键: user_id -> 用户表中的 id

4): 加班表

数据元素	数据类型	数据长度	数据描述
id	char	19	加班 id
user_id	char	19	用户 id
reason	varchar	255	加班原因
start_time	date		加班开始时间
end_time	date		加班结束时间
state	int	1	状态 (0: 删除) (1: 进行中) (2: 结束)
create_time	date		创建时间

主键: id

外键: user_id -> 用户表中的 id

5): 出差表

数据元素	数据类型	数据长度	数据描述
id	char	19	出差 id
reason	clob		出差原因
location	varchar	255	出差地点
cost	int		出差花费
start_time	date		出差开始时间
end_time	date		出差结束时间
state	date		状态 (0: 删除) (1: 未开始) (2: 进行中) (3: 结束)
create_time	date		创建时间

主键: id

6): 出差用户关系表

数据元素	数据类型	数据长度	数据描述
id	char	19	id

user_id	char	19	用户 id
trip_id	char	19	出差 id

主键: id

外键: user_id -> 用户表中的 id

外键: trip_id -> 出差表中的 id

功能设计

用户模块

- 查询单个用户所有信息

传入参数:

用户 id

返回结果:

用户 id 用户名 密码 电话 性别 生日 职位 入职时间
是否离职

- 根据电话查询用户信息

传入参数:

电话

返回结果:

用户 id 用户名 密码 电话 性别 生日 职位 入职时间
是否离职

- 新增用户

传入参数:

电话 密码 电话 性别 生日 职位

返回结果:

是否成功

- 修改用户

传入参数:

用户 id 用户名 密码 电话 性别 生日 职位

返回结果:

是否成功

- 删除用户

传入参数:

电话 密码

返回结果:

是否成功

出勤模块

- 打卡（新增出勤记录）

传入参数：

电话 密码

返回结果：

是否成功

- 所有人打卡情况

传入参数：

null

返回结果：

用户名 职位 打卡状态 打卡时间

- 单个人打卡情况

传入参数：

电话

返回结果：

用户名 职位 打卡状态 打卡时间

请假模块

- 请假（新增请假记录）

传入参数：

电话 密码 请假原因 请假开始时间 请假结束时间

返回结果：

是否成功 请销假 id

- 查询单人请假情况

传入参数：

电话

返回结果：

请销假 id 用户名 职位 请假原因 请假开始时间 请假结束时间 是否删除 创建时间

- 查询当日请假人员

传入参数：

日期

返回结果：

请销假 id 用户名 职位 请假原因 请假开始时间 请假结束时间 是否删除 创建时间

- 销假（更改请销假状态）

传入参数：

电话 密码 请销假 id

返回结果：
是否成功

加班模块

- 加班申请（新增加班记录）

传入参数：
电话 密码 加班原因 加班开始时间 加班结束时间
返回结果：
是否成功 加班 id

- 查询单人加班情况

传入参数：
电话
返回结果：
加班 id 用户名 职位 加班原因 加班开始时间 加班结束时间 是否删除 创建时间

- 查询当日加班人员

传入参数：
日期
返回结果：
加班 id 用户名 职位 加班原因 加班开始时间 加班结束时间 是否删除 创建时间

- 取消加班（删除加班信息）

传入参数：
电话 密码 加班 id
返回结果：
是否成功

出差模块

- 出差申请（新增出差记录）

传入参数：
电话 密码 出差原因 出差地点 出差花费 出差开始时间 出差结束时间
返回结果：
是否成功 出差 id

- 查询单人出差情况

传入参数：
电话

返回结果:

	出差 id	用户名	职位	出差原因	出差地点	出差花费	出差开始时间	出差结束时间	创建时间
--	-------	-----	----	------	------	------	--------	--------	------

- 查询当日出差人员

传入参数:

日期

返回结果:

	出差 id	用户名	职位	出差原因	出差地点	出差花费	出差开始时间	出差结束时间	创建时间
--	-------	-----	----	------	------	------	--------	--------	------

- 取消出差 (删除出差信息)

传入参数:

电话 密码 出差 id

返回结果:

是否成功

阶段三

依据阶段二的成果报告，设计及实现一个 Oracle 数据库，请自行命名数据库模式、表格、和行的名称，包含空值、外键、索引等设定；同时，将阶段二的每一个功能需求操作撰写相关的新增、删除、修改、查询的语句，每个表格至少要有 10 列的数据，且至少要设计 6 个查询操作。使用 Navicat 测试你的数据库和新增、删除、修改、查询操作。阶段三的成果报告主要内容是数据库设计实现的心得与结果，并包含数据库表格设计与数据内容的 Navicat 截图，以及新增、删除、修改、查询操作的 Navicat 运行截图（若运行结果超过一个页面，截取第一页即可），每个操作必须有文字说明。

数据库创建

创建表空间

创建表空间 t_table，将其文件放置在 c:\HW\jeck.dbf 中，设置大小为 100M。

```
create tablespace t_table datafile 'c:\HW\jeck.dbf' size
100M;
```

```
SQL> create tablespace t_table datafile 'c:\HW\jeck.dbf' size 100M;
表空间已创建。
```

创建用户

创建用户 u_jeck，设置密码为 123456，设置默认的表空间为 t_table

```
create user u_jeck identified by 123456 default tablespace
t_table;
```

```
SQL> create user u_jeck identified by 123456 default tablespace t_table;
用户已创建。
```

用户权限管理

为用户 u_jeck 赋予正式用户和临时用户权限

```
grant connect,resource to u_jeck;
```

```
SQL> grant connect,resource to u_jeck;
```

授权成功。

创建表

脚本

```
create table TB_USER
(
    ID          CHAR(19)          not null,
    USERNAME    VARCHAR2(64)      not null,
    PASSWORD    VARCHAR2(64)      not null,
    PHONE       CHAR(11)          not null,
    SEX         NUMBER default 1,
    BIRTHDAY    DATE,
    JOB         VARCHAR2(255),
    ENTRYTIME   DATE,
    STATE       NUMBER default 1 not null,
    constraint USER_PK
        primary key (ID)
);

comment on table TB_USER is '用户表';

create unique index USER_PHONE_UINDEX
on TB_USER (PHONE);

create table TB_ATTENDANCE
(
    ID          CHAR(19) not null,
    USER_ID     CHAR(19),
    STATE       NUMBER default 1,
    CREATE_TIME DATE,
    constraint ATTENDANCE_PK
        primary key (ID),
    constraint ATTENDANCE_USER_ID_FK
        foreign key (USER_ID) references TB_USER
);

comment on table TB_ATTENDANCE is '出勤表';

create table "tb_workOvertime"
(
    ID          CHAR(19) not null,
    USER_ID     CHAR(19),
    REASON      VARCHAR2(255),
    START_TIME  DATE,
    END_TIME    DATE,
    STATE       NUMBER default 1,
    CREATE_TIME DATE,
    constraint TB_WORKOVERTIME_PK
        primary key (ID),
    constraint TB_WORKOVERTIME_TB_USER_ID_FK
        foreign key (USER_ID) references TB_USER
);
```

```

);

comment on table "tb_workOvertime" is '加班表';

create table TB_VACATE
(
    ID          CHAR(19) not null,
    USER_ID     CHAR(19),
    REASON       CLOB,
    START_TIME  DATE,
    END_TIME    DATE,
    STATE        NUMBER default 1,
    CREATE_TIME DATE,
    constraint TB_VACATE_PK
        primary key (ID),
    constraint TB_VACATE_TB_USER_ID_FK
        foreign key (USER_ID) references TB_USER
);

comment on table TB_VACATE is '请销假表';

create table "tb_businessTrip"
(
    ID          CHAR(19) not null,
    REASON       CLOB,
    LOCATION     VARCHAR2(255),
    COST         NUMBER(10, 2),
    START_TIME  DATE,
    END_TIME    DATE,
    STATE        NUMBER default 1,
    CREATE_TIME DATE,
    constraint TB_BUSINESSTRIP_PK
        primary key (ID)
);

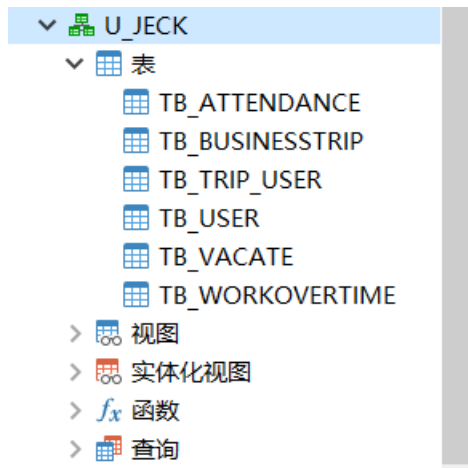
comment on table "tb_businessTrip" is '出差表';

create table "tb_trip_user"
(
    ID          CHAR(19) not null,
    USER_ID     CHAR(19),
    TRIP_ID     CHAR(19),
    constraint TB_TRIP_USER_PK
        primary key (ID),
    constraint TB_TRIP_USER_TB_TRIP_ID_FK
        foreign key (TRIP_ID) references "tb_businessTrip",
    constraint TB_TRIP_USER_TB_USER_ID_FK
        foreign key (USER_ID) references TB_USER
);

comment on table "tb_trip_user" is '加班_用户_关系表';

```

执行结果



数据库初始化

添加数据

TIP: 由于初始数据过多，本文档仅为部分语句进行展示，全部数据请查看附件！

脚本

```
--
--
-- Records of TB_ATTENDANCE
--
INSERT INTO "U_JECK"."TB_ATTENDANCE" VALUES
('3955539060662367744', '7663049381084454912', '1',
TO_DATE('2019-12-31 00:28:22', 'SYYYY-MM-DD HH24:MI:SS'));
INSERT INTO "U_JECK"."TB_ATTENDANCE" VALUES
('7493627565608359936', '1469144004281016832', '1',
TO_DATE('2004-05-17 08:19:41', 'SYYYY-MM-DD HH24:MI:SS'));
INSERT INTO "U_JECK"."TB_ATTENDANCE" VALUES
('7770473862336182272', '1538254444394360064', '2',
TO_DATE('2019-12-31 18:10:51', 'SYYYY-MM-DD HH24:MI:SS'));
--
--
-- Records of TB_BUSINESSTRIP
--
INSERT INTO "U_JECK"."TB_BUSINESSTRIP" VALUES
('9636950117401733120', '帮助东莞的客户完成软件的部署', '东莞',
'217235.26', TO_DATE('2001-11-18 19:12:13', 'SYYYY-MM-DD
HH24:MI:SS'), TO_DATE('2002-02-03 06:39:06', 'SYYYY-MM-DD
HH24:MI:SS'), '3', TO_DATE('2014-01-04 16:16:59', 'SYYYY-MM-
DD HH24:MI:SS'));
```

```

-- -----
-- Records of TB_TRIP_USER
-- -----

INSERT INTO "U_JECK"."TB_TRIP_USER" VALUES
('5766196852949350400', '1608981710249993216',
'6429157426592350208');
INSERT INTO "U_JECK"."TB_TRIP_USER" VALUES
('7246332460174764032', '8209856577979705344',
'3163650588528361472');
INSERT INTO "U_JECK"."TB_TRIP_USER" VALUES
('7139875782140471296', '1469144004281016832',
'7245998640939827200');
-- -----

-- Records of TB_USER
-- -----

INSERT INTO "U_JECK"."TB_USER" VALUES ('1376489856438086144',
'丁震南', 'SIv8EeeuGe', '18721266716', '0', TO_DATE('2016-05-
31 03:50:45', 'SYYYY-MM-DD HH24:MI:SS'), '生物化学家',
TO_DATE('2000-05-14 21:06:38', 'SYYYY-MM-DD HH24:MI:SS'),
'1');
INSERT INTO "U_JECK"."TB_USER" VALUES ('5877921081410105344',
'谭璐', 'Oyv1Pjzrei', '14544993416', '1', TO_DATE('2008-07-21
03:19:07', 'SYYYY-MM-DD HH24:MI:SS'), '导师', TO_DATE('2020-
08-04 04:27:42', 'SYYYY-MM-DD HH24:MI:SS'), '1');
INSERT INTO "U_JECK"."TB_USER" VALUES ('1469144004281016832',
'罗詩涵', 'sANJCZVBUC', '15676660959', '2', TO_DATE('2010-07-
10 14:11:39', 'SYYYY-MM-DD HH24:MI:SS'), '客戶協調員',
TO_DATE('2020-01-22 17:20:08', 'SYYYY-MM-DD HH24:MI:SS'),
'1');
-- -----

-- Records of TB_VACATE
-- -----

INSERT INTO "U_JECK"."TB_VACATE" VALUES
('1358623858828564224', '7864123933601216512', '前女友要结婚, 前
往庆祝', TO_DATE('2017-07-07 05:52:07', 'SYYYY-MM-DD
HH24:MI:SS'), TO_DATE('2017-07-12 21:49:43', 'SYYYY-MM-DD
HH24:MI:SS'), '0', TO_DATE('2008-02-07 09:38:25', 'SYYYY-MM-
DD HH24:MI:SS'));

```

-- Records of TB_WORKOVERTIME

INSERT INTO "U_JECK"."TB_WORKOVERTIME" VALUES

('6511637171867447000', '1511637171867447000', '老板逼得',
TO_DATE('2022-05-16 00:11:45', 'SYYYY-MM-DD HH24:MI:SS'),
TO_DATE('2022-05-16 23:48:43', 'SYYYY-MM-DD HH24:MI:SS'),
'1', TO_DATE('2022-05-16 00:11:53', 'SYYYY-MM-DD
HH24:MI:SS'));

INSERT INTO "U_JECK"."TB_WORKOVERTIME" VALUES

('4105671405702732288', '6186245436122126336', 'In a Telnet
session, all communications, including username and password,
are transmitted in plain-text, allowing anyone to listen-in
on your session and steal passwords and other information.',
TO_DATE('2015-11-29 14:17:03', 'SYYYY-MM-DD HH24:MI:SS'),
TO_DATE('2015-11-29 15:14:39', 'SYYYY-MM-DD HH24:MI:SS'),
'1', TO_DATE('2004-04-09 16:56:50', 'SYYYY-MM-DD
HH24:MI:SS'));

执行结果

ID	USER_ID	STATE	CREATE_TIME
3955539060662367	7663049381084454	1	2019-12-31 00:28:22
7493627565608359	1469144004281016	1	2004-05-17 08:19:41
7770473862336182	1538254444394360	2	2019-12-31 18:10:51
580581110236496216666271	544283918	1	2017-02-17 19:17:11
450591141860654731	6179570999337089	1	2002-12-22 20:04:07
62496368556906611	1608981710249993	1	2003-08-10 23:39:40
2350305705189051	5764129231047518	2	2013-10-31 12:22:44
6058104267285899	4316654561738088	0	2003-10-08 18:56:44
1422817461585220	3792635825490380	1	2016-06-18 07:43:45
3823834355546671	4563915641856457	0	2020-04-12 16:50:49
4472674071438936	5733577358595024	2	2012-01-16 09:03:50
8443038006755823	2347351397880948	0	2004-06-29 17:26:53
4914862442192373	5844351399880570	0	2014-12-07 23:39:25
5470655883889433	3310494741052283	1	2017-07-02 12:10:43
6836800335122493	7611951658302097	2	2017-05-11 19:27:19
4925122566595565	4494676014822126	1	2001-11-30 05:20:45
5767643993827567	6478122388807311	1	2016-02-23 00:13:23
9127234752764055	8020908491647868	0	2004-01-24 13:53:17
3014584726402643	1158394635511522	0	2021-03-29 20:11:23
2748159947869964	4945884100313235	1	2006-12-23 13:42:13
9266349545163673	5260279417279801	0	2018-04-19 08:14:38
1723817129661976	5764129231047518	1	2014-11-14 20:43:36
4667970814182285	4563915641856457	2	2020-06-25 03:10:24
1801895666492132	6179570999337089	1	2013-11-24 06:01:31
8040247265689909	4403841455540668	1	2005-11-09 03:32:42
4280681389657081	18270084395076231	2	2005-11-06 07:41:07
8912762501230899	3840906144574457	2	2006-08-27 23:09:48

TB_BUSINESSTRIP @U_JECK (jeck) - 表 - Navicat Premium

文件 编辑 查看 表 收藏夹 工具 窗口 帮助

连接 新建查询 表 视图 实体化视图 函数 用户 其它 数据泵 查询 自动运行 模型 图表

对象 TB_ATTENDANCE @... TB_BUSINESSTRIP @... TB_TRIP_USER @U_J... TB_USER @U_JECK (j... TB_VACATE @U_JEC... TB_WORKOVERTIME...

开始事务 文本 筛选 排序 导入 导出

ID	REASON	LOCATION	COST	START_TIME	END_TIME	STATE	CREATE_TIME
9636950117401733	A man is not	东莞	217235.26	2001-11-18 19:12:13	2002-02-03 06:39:06	3	2014-01-04 16:16:59
8121017451650791	Anyone who	深圳	423355.57	2014-04-06 14:32:27	2014-04-24 18:34:22	3	2006-03-05 07:22:36
2188581836694953	If your Interr	东莞	884740.24	2017-04-24 02:15:02	2017-06-06 09:42:52	0	2002-08-27 22:16:26
3828490323065325	Navicat Mor	上海市	762809.7	2008-08-28 22:21:31	2008-09-10 10:37:21	0	2002-05-29 01:15:51
5377130555571648	A man is not	广州市	896893.67	2018-05-23 20:53:53	2018-07-14 11:07:48	3	2007-05-18 12:05:50
9340626594699266	Navicat Clou	东莞	849884.62	2007-12-24 14:54:52	2008-02-20 04:31:21	3	2007-03-17 05:07:04
1742241619134272	Instead of w	广州市	453200.26	2010-10-22 17:56:36	2010-11-29 09:22:31	0	2007-09-09 20:17:02
3934382330570660	To connect t	上海市	901653.23	2008-03-10 12:17:01	2008-05-12 20:26:37	3	2014-02-07 20:29:34
2969154074863527	Navicat prov	深圳	299483.2	2000-09-05 00:05:08	2000-11-05 06:25:16	3	2008-12-01 22:26:05
4765163315787106	The reason v	东莞	104127.37	2017-01-15 21:50:05	2017-02-07 04:18:53	3	2006-08-14 09:03:37
5245424761969392	It is used wh	上海市	589765.18	2018-02-13 11:04:44	2018-03-10 20:13:22	3	2020-09-05 19:13:05
6787994407185025	Navicat Data	广州市	580823.14	2003-03-25 18:08:34	2003-04-06 10:37:51	3	2016-06-02 09:42:16
7726966771723534	The repositc	北京市	687141.89	2009-09-26 04:39:26	2009-11-21 10:00:33	3	2015-04-03 05:33:49
7951155033960253	The repositc	东莞	730266.28	2010-06-16 17:31:40	2010-08-18 13:06:42	3	2018-01-13 10:09:58
5012004546932212	It wasn't rain	上海市	36851.31	2021-01-08 19:12:18	2021-03-22 19:02:13	3	2015-04-20 23:04:49
438195446663929	After compa	东莞	649204.07	2022-03-04 01:12:28	2022-04-03 10:12:26	3	2003-08-25 13:32:35
2604926685103524	Always keep	北京市	325030.07	2018-08-05 05:04:25	2018-10-16 15:13:32	3	2013-08-28 19:29:50
4554532038552528	Remember t	广州市	459293.04	2006-04-26 09:05:55	2006-07-07 21:20:19	3	2018-12-07 21:36:09
3065853206191946	To successfu	上海市	344762.98	2020-08-27 18:34:15	2020-11-28 07:26:05	3	2017-10-19 11:34:43
5878872248330178	The Navigat	广州市	843163.56	2007-01-09 06:18:49	2007-04-06 00:46:11	3	2007-10-03 20:46:54
6643713827718219	The Synchro	中山	950489.5	2013-08-14 11:05:27	2013-11-18 19:23:41	3	2008-11-08 12:24:01
1420919680384467	A man is not	上海市	772349.33	2013-09-29 11:22:06	2013-11-30 06:57:08	3	2006-07-08 18:47:41
5014260365381769	Remember t	上海市	404248.79	2016-09-13 07:41:05	2016-11-25 23:30:03	3	2009-02-14 14:48:04
5941251468471775	If opportuni	北京市	764809.81	2004-02-05 08:49:41	2004-02-21 01:47:46	3	2000-12-07 19:26:18
1463208039834608	Typically, i	东莞	346754.46	2004-04-05 17:10:59	2004-04-11 04:46:30	3	2014-01-30 02:29:32
8135034157079988	It can also m	中山	524371.61	2021-04-11 14:24:43	2021-06-17 19:18:29	3	2008-07-07 23:51:24
1652704036336487	I destroy my	东莞	248284.41	2010-08-10 03:01:18	2010-11-17 06:11:23	3	2001-08-06 02:10:30

SELECT * FROM (SELECT "NAVICAT_TABLE".*, ROWNUM "NAVICAT_ROWNUM" FROM (SELECT "U_JECK".*

第 1 条记录 (共 300 条) 于第 1 页

TB_TRIP_USER @U_JECK (jeck) - 表 - Navicat Premium

文件 编辑 查看 表 收藏夹 工具 窗口 帮助

连接 新建查询 表 视图 实体化视图 函数 用户 其它 数据泵 查询 自动运行 模型 图表

对象 TB_ATTENDANCE @... TB_BUSINESSTRIP @... TB_TRIP_USER @U_J... TB_USER @U_JECK (j... TB_VACATE @U_JEC... TB_WORKOVERTIME...

开始事务 文本 筛选 排序 导入 导出

ID	USER_ID	TRIP_ID
5766196852949350	1608981710249993	6429157426592350
724632460174764	8209856577979705	3163650588528361
7139875782140471	1469144004281016	7245998640939827
3289162370366474	209161650833341	3227856674079518
3566709584342196	2347351397880948	8262280021559336
3465408693557721	7455622026174580	7991039602446860
8429932433852262	1923035266161908	7479680679500060
2463072691692952	6937184156619642	7367447253291030
9405274722187143	7864123933601216	2602566501472257
9956796241002336	8020908491647868	3570648829417118
3731005614751927	4563915641856457	5940099047533332
7436368893342697	1923035266161908	2602566501472257
7773815082172848	2712542177330311	3843099246549280
6214080459561381	5844351399880570	3613087045151906
7265401018248933	2347351397880948	5101388432698696
8294469535012466	5260279417279801	3160714679524944
8626155133526945	3310494741052283	1840335335452863
6036194980651640	2347351397880948	4726561963326597
9789667625568018	2347351397880948	6331307950879303
9672053508311945	7251692689381707	9511513945236926
7852593497572235	8175270811524118	8372902383271356
1974590550883189	7611951658302097	3198188045279213
4286865962587200	3310494741052283	4956979216896311
8630126574244777	1608981710249993	8262280021559336
8153689189367326	3149260141931764	3647119017150686
1134040184417668	8270084395076231	6479384040816107
7083558446049528	7663049381084454	3160714679524944

SELECT * FROM (SELECT "NAVICAT_TABLE".*, ROWNUM "NAVICAT_ROWNUM" FROM (SELECT "U_JECK".*

第 1 条记录 (共 998 条) 于第 1 页

TB_USER @U_JECK (jeck) - 表 - Navicat Premium

文件 编辑 查看 表 收藏夹 工具 窗口 帮助

连接 新建查询 表 视图 实体化视图 函数 用户 其它 数据泵 查询 自动运行 模型 图表

对象 TB_ATTENDANCE @... TB_BUSINESSTRIP @... TB_TRIP_USER @U_J... TB_USER @U_JECK (j... TB_VACATE @U_JEC... TB_WORKOVERTIME...

开始事务 文本 筛选 排序 导入 导出

ID	USERNAME	PASSWORD	PHONE	SEX	BIRTHDAY	JOB	ENTRYTIME	STATE
1376489856438086	丁黄南	Slv8EeeuGe	18721266716	0	2016-05-31 03:50:45	生物化学家	2000-05-14 21:06:38	1
5877921081410105	谭璐	Oyv1Pjzrei	14544993416	1	2008-07-21 03:19:07	导师	2020-08-04 04:27:42	1
1469144004281016	罗特涵	sANJCZVBUC	15676660959	2	2010-07-10 14:11:39	客户协调员	2020-01-22 17:20:08	1
7185286713752215	滕子昇	tTUUnHmJOA	14597098032	2	2020-08-02 23:43:38	团体领导	2002-06-21 15:08:58	1
2347351397880948	胡安琪	He7f6ZtC0y	19887326044	0	2007-05-23 20:50:56	活动经理	2017-02-22 06:05:23	0
4316654561738088	韩子斌	cXY4luTsn1	16629894551	1	2010-10-09 03:17:16	工程师	2017-03-16 10:28:07	1
8209856577979705	韩晓明	t6Y4nYgm7b	13484330239	0	2000-05-12 13:22:06	牙医	2022-05-10 04:16:43	1
8166266253289617	陈璐	OyavMJE0hN	16680720881	1	2021-05-20 13:32:24	多媒体动画师	2016-02-02 13:10:59	1
4945884100313235	姜云熙	DUJTCl8p6H	13821365148	0	2008-10-17 02:04:48	多媒体动画师	2000-04-29 13:21:59	1
8175270811524118	阎秀英	aum0sOzyGq	14982688474	1	2022-01-31 12:42:46	歌手	2021-09-29 17:15:21	1
3310494741052283	程璐	hDoKBktIE5	17597373353	2	2017-01-08 16:52:28	水疗经理	2017-08-03 18:50:14	0
7799701429963675	徐杰宏	uY8aiqujir	19883613299	1	2006-08-24 03:02:37	牙医	2003-06-12 21:28:31	1
7251692689381707	汤安琪	EMVfl3QNpb	16672385749	2	2021-10-11 21:56:01	社交媒体协调员	2012-04-23 06:07:17	0
1158394635511522	李宇宁	eeWzdFbssy	16654262941	0	2001-08-30 15:08:47	市场总监	2007-10-15 12:16:03	0
7624963756112478	熊璐	gRjuMh1sOn	17302702886	1	2020-01-05 22:13:06	零售助理	2012-09-28 17:29:08	0
8259051727504646	尹子昇	ISd3piAUMY	16696452297	2	2019-08-05 23:18:32	图书馆管理员	2008-08-13 18:35:50	1
1923035266161908	段黄南	dukLnlNvew	16691193841	1	2017-07-21 14:53:17	化妆师	2004-08-05 07:49:46	1
8249142621977708	卢嘉伦	uGW6H5QEstb	16610621312	2	2007-04-12 09:40:57	护士	2013-03-14 06:46:20	0
4262301815617200	邹宇宁	aF8p4rq3Hr	16626147038	0	2016-04-22 13:58:05	人力资源经理	2004-11-07 18:14:40	1
5501140371026518	任宇宁	QZ56WFAXd4	16645548950	0	2016-07-18 03:25:20	牙齿矫正医生	2004-03-10 03:34:23	1
8029716071084597	谭岚	jrY5BjCtUr	16669152866	1	2002-10-20 16:04:12	办公室主管	2000-05-15 07:27:53	1
5764129231047518	黄嘉伦	1mjnl24J6m	18659800118	2	2016-01-15 13:18:58	移动应用程序开发人员	2008-10-13 09:18:02	0
6987639974177691	邵特涵	K7kfsulzU	19850936445	1	2007-06-09 19:10:41	食品科学家	2021-06-06 05:59:15	1
6666271544283918	于睿	HGeyepVsu1	16651712485	1	2012-11-22 02:56:58	软件开发员	2009-06-28 03:56:59	0
4246226675423747	龙特涵	SlYcrhY3wd	13529954755	2	2002-02-21 14:40:58	园艺家	2015-07-01 18:42:36	2
4403841455540668	韩安琪	NdUnvidyh6	14927457339	2	2000-02-27 03:09:44	食品科学家	2002-02-07 01:30:11	1
6478122388807311	侯睿	rFTauVQ21y	16698476890	1	2013-10-18 11:22:11	审计师	2016-12-10 16:55:03	1

SELECT * FROM (SELECT "NAVICAT_TABLE".*, ROWNUM "NAVICAT_ROWNUM" FROM (SELECT "U_JECK".*)

第 1 条记录 (共 51 条) 于第 1 页

TB_VACATE @U_JECK (jeck) - 表 - Navicat Premium

文件 编辑 查看 表 收藏夹 工具 窗口 帮助

连接 新建查询 表 视图 实体化视图 函数 用户 其它 数据泵 查询 自动运行 模型 图表

对象 TB_ATTENDANCE @... TB_BUSINESSTRIP @... TB_TRIP_USER @U_J... TB_USER @U_JECK (j... TB_VACATE @U_JEC... TB_WORKOVERTIME...

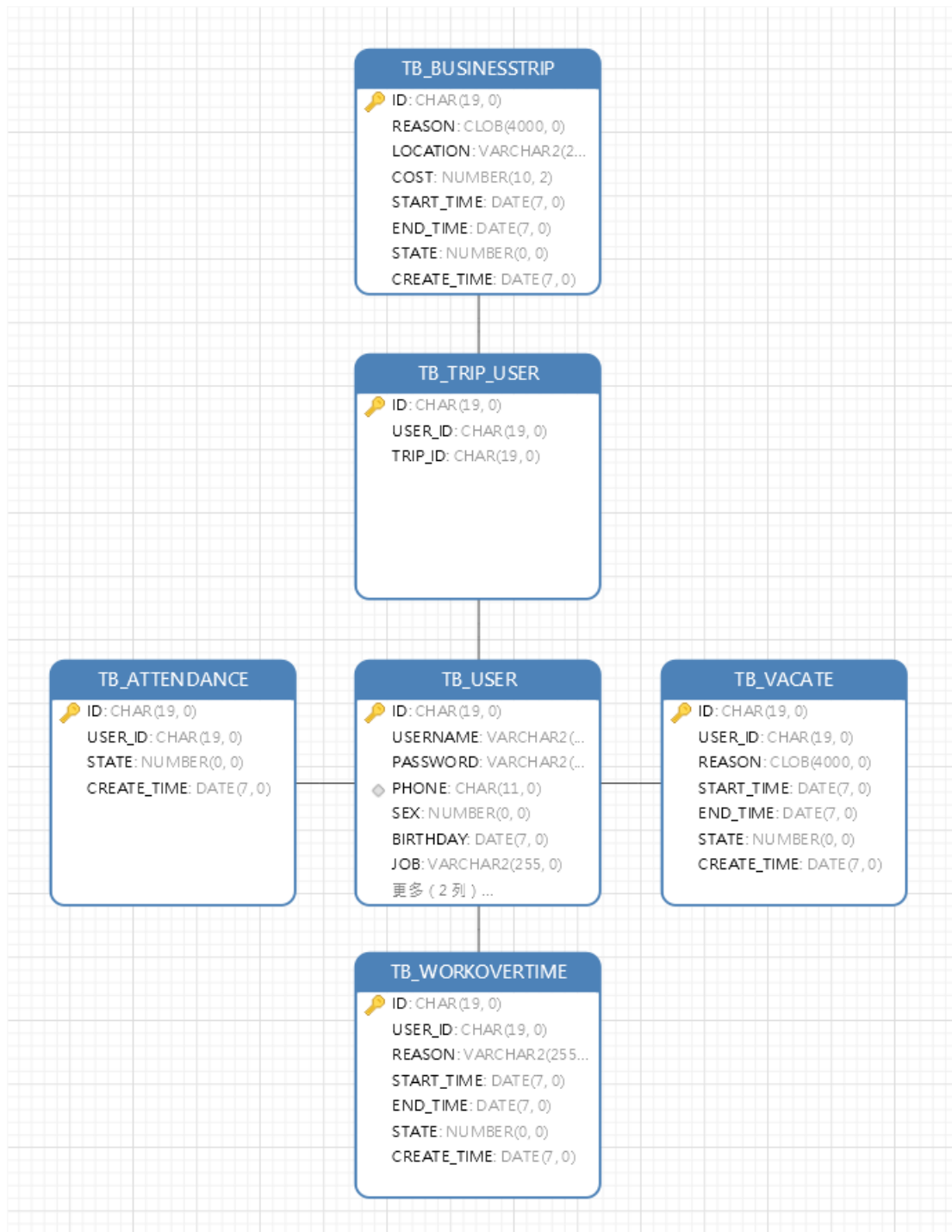
开始事务 文本 筛选 排序 导入 导出

ID	USER_ID	REASON	START_TIME	END_TIME	STATE	CREATE_TIME
1358623858828564	7864123933601216	In other wor	2017-07-07 05:52:07	2017-07-12 21:49:43	0	2008-02-07 09:38:25
3639018437532185	3310494741052283	SQL Editor a	2016-01-10 13:49:26	2016-01-31 10:14:52	2	2020-04-29 08:11:51
2879852444420655	8209856577979705	Champions k	2003-12-30 01:47:10	2004-01-01 01:24:08	0	2009-07-11 13:17:48
8887648364490100	8029716071084597	The Synchro	2015-11-06 17:27:56	2015-11-18 15:19:46	2	2004-06-19 07:47:49
8375494763695079	3792635825490380	Navicat Clou	2008-07-02 13:40:14	2008-07-21 20:19:07	2	2003-12-05 15:15:46
3302527706048271	6134983509223247	With its well	2020-07-16 05:19:45	2020-08-12 01:30:51	2	2003-01-17 02:55:16
4228802331407532	7624963756112478	Navicat Clou	2016-02-08 21:02:35	2016-02-26 03:01:09	0	2019-10-30 20:37:09
733550364498485	8029716071084597	You can sele	2017-03-07 02:01:18	2017-03-25 02:05:37	3	2013-05-28 04:29:12
8510579435186653	5733577358595024	It is used wh	2018-09-15 19:37:09	2018-10-15 18:36:40	2	2011-02-08 16:26:47
2380753978841382	6186245436122126	Sometimes y	2001-02-10 20:16:06	2001-02-17 17:31:56	0	2017-01-23 03:03:41
3034155892417337	8175270811524118	If the plan d	2018-06-14 09:13:16	2018-06-21 01:29:35	0	2018-11-15 16:47:48
737212568322992	1158394635511522	You cannot i	2003-04-19 23:31:02	2003-05-14 12:22:52	1	2003-05-25 03:59:55
5919804878688034	7251692689381707	Champions k	2007-05-27 21:30:14	2007-06-20 04:09:07	1	2004-11-12 13:51:47
2110443470854313	7799701429963675	Difficult circ	2018-08-25 12:59:11	2018-09-10 06:43:21	2	2018-07-17 18:32:52
1169167858956979	7185286713752215	The reason v	2002-11-29 15:32:30	2002-12-24 22:24:20	1	2021-11-09 17:00:15
3303385821142807	1538254444394360	Navicat Clou	2021-07-11 01:45:27	2021-07-31 03:37:46	1	2019-01-30 00:28:07
2298041082242571	7185286713752215	Navicat Data	2015-10-30 07:14:46	2015-11-13 16:57:58	0	2013-06-21 10:54:17
2573788008882412	39373499480621736	How we spe	2001-04-02 08:05:21	2001-04-10 19:09:11	1	2006-02-07 20:03:39
2874868049922593	6937184156619642	The On Start	2003-02-16 11:24:33	2003-03-03 13:21:11	1	2001-07-27 09:15:22
3161513480204731	7799701429963675	Import Wiza	2012-03-29 05:22:13	2012-04-12 09:37:06	1	2020-01-14 01:13:29
4959158935872018	7185286713752215	A comfort zc	2005-06-02 11:45:12	2005-06-24 15:57:12	1	2005-02-04 13:20:48
5897484581422022	4246226675423747	Navicat auth	2013-09-19 04:25:19	2013-10-02 23:53:09	0	2019-11-06 02:57:20
8266512765493164	6937184156619642	Navicat 15 h	2010-01-29 17:30:06	2010-02-19 17:35:52	1	2015-07-21 01:27:21
8759502975377453	3840906144574457	To get a sec	2000-12-15 01:50:21	2000-12-23 16:53:14	1	2001-02-28 15:50:07
3770372419226959	2347351397880948	If the Show c	2010-05-25 18:52:08	2010-06-15 15:23:20	1	2015-03-04 21:50:04
1693629934437788	2091616508333341	If your Interr	2001-12-25 07:07:10	2002-01-04 01:41:44	3	2008-04-08 05:57:41
5078518922979872	6937184156619642	To clear or n	2013-04-03 00:47:35	2013-04-20 16:32:13	2	2006-04-15 14:26:50

SELECT * FROM (SELECT "NAVICAT_TABLE".*, ROWNUM "NAVICAT_ROWNUM" FROM (SELECT "U_JECK".*)

第 1 条记录 (共 1000 条) 于第 1 页

TB_WORKOVERTIME @U_JECK (jeck) - 表 - Navicat Premium						
文件 编辑 查看 表 收藏夹 工具 窗口 帮助						
连接 新建查询 表 视图 实体化视图 函数 用户 其它 数据泵 查询 自动运行 模型 图表						
对象 TB_ATTENDANCE @... TB_BUSINESSTRIP @... TB_TRIP_USER @U_J... TB_USER @U_JECK (j... TB_VACATE @U_JEC... TB_WORKOVERTIME...						
开始事务 文本 筛选 排序 导入 导出						
ID	USER_ID	REASON	START_TIME	END_TIME	STATE	CREATE_TIME
6511637171867447	1511637171867447	老板逼得	2022-05-16 00:11:45	2022-05-16 23:48:43	1	2022-05-16 00:11:53
4105671405702732	6186245436122126	In a Telnet session, a	2015-11-29 14:17:03	2015-11-29 15:14:39	1	2004-04-09 16:56:50
3049682548800485	6666271544283918	Navicat Data Model	2019-11-06 07:30:50	2019-11-06 11:12:36	1	2021-03-25 08:04:56
8492617886266331	4563915641856457	It wasn't raining whe	2014-07-08 23:05:43	2014-07-09 16:21:05	1	2002-02-23 09:28:46
7472963310197528	2091616508333341	Creativity is intelli	2019-12-02 22:09:21	2019-12-03 14:25:40	0	2001-04-15 06:24:46
3250879209253358	6134983509223247	To start working wtl	2011-06-13 13:24:46	2011-06-13 20:16:36	1	2012-04-01 00:35:12
9251584350500673	4494676014822126	Navicat Monitor is a	2005-06-07 03:01:54	2005-06-07 19:49:54	1	2018-05-27 07:47:22
8855355464368435	1376489856438086	What you get by ach	2015-12-04 15:58:49	2015-12-04 23:52:35	2	2003-04-11 03:36:46
9027868598942652	6186245436122126	Instead of wonderin	2021-03-26 08:59:19	2021-03-27 00:05:05	0	2005-11-28 05:02:44
4574029660845865	2712542177330311	To start working wtl	2021-09-11 09:18:20	2021-09-12 03:38:30	1	2010-07-16 05:46:15
5475969415282508	7455622026174580	Navicat Cloud provi	2007-02-03 04:00:24	2007-02-03 13:24:53	2	2019-02-13 13:43:23
5192590130199595	2712542177330311	The Synchronize to	2002-09-01 03:17:29	2002-09-01 23:48:41	1	2016-10-31 22:56:55
4066159742553093	7799701429963675	To connect to a data	2015-11-05 14:54:12	2015-11-06 08:51:19	0	2020-03-25 12:55:54
8806159001734500	8020908491647868	SSH serves to preve	2020-12-25 03:30:14	2020-12-25 14:18:14	1	2013-12-27 02:02:04
9067207291161630	1511637171867447	In the middle of wint	2016-11-13 07:17:20	2016-11-13 21:06:46	2	2013-07-11 00:05:56
5782361912674676	6134983509223247	It is used while your	2016-06-25 06:38:21	2016-06-25 16:21:33	1	2018-10-20 21:02:39
5166678983008032	3973499480621736	SQL Editor allows yo	2004-06-27 17:31:15	2004-06-28 05:44:13	1	2022-04-08 09:56:35
5016486222604169	5844351399880570	All the Navicat Clou	2013-05-01 20:56:24	2013-05-02 10:25:41	0	2012-07-10 19:27:47
4666616328966055	6186245436122126	SSH serves to preve	2012-03-06 01:02:26	2012-03-06 03:09:09	0	2003-09-30 10:00:30
5016747954618225	6478122388807311	Anyone who has eve	2010-12-15 14:48:26	2010-12-16 07:24:57	1	2005-10-02 22:23:24
9754233148454266	1608981710249993	I destroy my enemie	2014-03-08 20:10:37	2014-03-09 10:40:23	2	2015-07-30 07:54:25
8647003741620350	4246226675423747	In a Telnet session, a	2004-10-18 23:29:43	2004-10-19 11:23:57	0	2011-08-26 06:24:28
8282955482607642	2347351397880948	You cannot save pec	2010-10-30 09:38:01	2010-10-31 07:14:01	1	2013-07-11 17:12:34
3298777873220091	1158394635511522	The Navigation pane	2009-03-04 02:47:08	2009-03-04 02:51:27	0	2003-09-04 11:10:33
6821402107780141	1469144004281016	The Main Window cc	2004-10-04 05:31:05	2004-10-04 08:44:03	2	2015-10-30 19:48:59
2409305182389214	7611951658302097	Navicat Cloud provi	2017-10-07 13:22:46	2017-10-08 05:54:56	0	2020-03-25 06:26:17
2530991338920168	6987639974177691	You can select any c	2003-03-01 06:05:41	2003-03-01 14:32:34	1	2013-01-27 17:39:51
+ - ✓ ✕ ☐						
SELECT * FROM (SELECT "NAVICAT_TABLE".*, ROWNUM "NAVICAT_ROWNUM" FROM (SELECT "U_JECK".*)						
第 1 条记录 (共 1000 条) 于第 1 页						



增删改查样例

--分别插入 TB_ATTENDANCE、TB_BUSINESSTRIP、TB_TRIP_USER、TB_USER、TB_VACATE、TB_WORKOVERTIME 数据

```

INSERT INTO "U_JECK"."TB_ATTENDANCE" VALUES ('12312312312',
'7663049381084454912', '1', TO_DATE('2019-12-31 00:28:22',
'SYYYY-MM-DD HH24:MI:SS'));
INSERT INTO "U_JECK"."TB_BUSINESSTRIP" VALUES
  
```



```

('9636950117401733120', '帮助东莞的客户完成软件的部署', '东莞',
'217235.26', TO_DATE('2001-11-18 19:12:13', 'YYYY-MM-DD
HH24:MI:SS'), TO_DATE('2002-02-03 06:39:06', 'YYYY-MM-DD
HH24:MI:SS'), '3', TO_DATE('2014-01-04 16:16:59', 'YYYY-MM-DD
HH24:MI:SS'));
INSERT INTO "U_JECK"."TB_TRIP_USER" VALUES
('5766196852949350400', '1608981710249993216',
'6429157426592350208');
INSERT INTO "U_JECK"."TB_USER" VALUES ('1376489856438086144',
'丁震南', 'SIv8EeeuGe', '18721266716', '0', TO_DATE('2016-05-31
03:50:45', 'YYYY-MM-DD HH24:MI:SS'), '生物化学家',
TO_DATE('2000-05-14 21:06:38', 'YYYY-MM-DD HH24:MI:SS'),
'1');
INSERT INTO "U_JECK"."TB_VACATE" VALUES
('1358623858828564224', '7864123933601216512', '前女友要结婚, 前
往庆祝', TO_DATE('2017-07-07 05:52:07', 'YYYY-MM-DD
HH24:MI:SS'), TO_DATE('2017-07-12 21:49:43', 'YYYY-MM-DD
HH24:MI:SS'), '0', TO_DATE('2008-02-07 09:38:25', 'YYYY-MM-DD
HH24:MI:SS'));
INSERT INTO "U_JECK"."TB_WORKOVERTIME" VALUES
('6511637171867447000', '1511637171867447000', '老板逼得',
TO_DATE('2022-05-16 00:11:45', 'YYYY-MM-DD HH24:MI:SS'),
TO_DATE('2022-05-16 23:48:43', 'YYYY-MM-DD HH24:MI:SS'), '1',
TO_DATE('2022-05-16 00:11:53', 'YYYY-MM-DD HH24:MI:SS'));
--分别删除 TB_ATTENDANCE、TB_BUSINESSTRIP、TB_TRIP_USER、TB_USER、
TB_VACATE、TB_WORKOVERTIME 数据
DELETE FROM "U_JECK"."TB_ATTENDANCE" WHERE ID=12312312312;
DELETE FROM "U_JECK"."TB_BUSINESSTRIP" WHERE ID=12312312312;
DELETE FROM "U_JECK"."TB_TRIP_USER" WHERE ID=12312312312;
DELETE FROM "U_JECK"."TB_USER" WHERE ID=12312312312;
DELETE FROM "U_JECK"."TB_VACATE" WHERE ID=12312312312;
DELETE FROM "U_JECK"."TB_WORKOVERTIME" WHERE ID=12312312312;
--分别更新 TB_ATTENDANCE、TB_BUSINESSTRIP、TB_TRIP_USER、TB_USER、
TB_VACATE、TB_WORKOVERTIME 数据
UPDATE "U_JECK"."TB_ATTENDANCE" SET STATE=1 WHERE
ID=12312312312;
UPDATE "U_JECK"."TB_BUSINESSTRIP" SET STATE=1 WHERE
ID=12312312312;
UPDATE "U_JECK"."TB_TRIP_USER" SET ID=11111111111 WHERE
ID=12312312312;
UPDATE "U_JECK"."TB_USER" SET STATE=1 WHERE ID=12312312312;
UPDATE "U_JECK"."TB_VACATE" SET STATE=1 WHERE ID=12312312312;
UPDATE "U_JECK"."TB_WORKOVERTIME" SET STATE=1 WHERE
ID=12312312312;

```



```
--分别查询 TB_ATTENDANCE、TB_BUSINESSTRIP、TB_TRIP_USER、TB_USER、
TB_VACATE、TB_WORKOVERTIME 数据
SELECT * FROM "U_JECK"."TB_ATTENDANCE";
SELECT * FROM "U_JECK"."TB_BUSINESSTRIP";
SELECT * FROM "U_JECK"."TB_TRIP_USER";
SELECT * FROM "U_JECK"."TB_USER";
SELECT * FROM "U_JECK"."TB_VACATE";
SELECT * FROM "U_JECK"."TB_WORKOVERTIME";
```

关于业务的 6 个查询语句

查找正在进行的出差行程

```
SELECT U.USERNAME AS 姓名,U.PHONE AS 电话,U.JOB AS 职位,T.LOCATION AS 地点,T.REASON AS 原因,T.START_TIME AS 开始时间,T.END_TIME AS 结束时间,T.STATE AS 状态码
FROM TB_BUSINESSTRIP T
LEFT JOIN TB_TRIP_USER TU on T.ID = TU.TRIP_ID
LEFT JOIN TB_USER U on U.ID = TU.USER_ID
WHERE T.END_TIME>SYSDATE
ORDER BY T.ID;
```

```
SELECT U.USERNAME AS 姓名,U.PHONE AS 电话,U.JOB AS 职位,T.LOCATION AS 地点,T.REASON AS 原因,T.START_TIME AS 开始时间,T.END_TIME AS 结束时间,T.STATE AS 状态码
FROM TB_BUSINESSTRIP T
LEFT JOIN TB_TRIP_USER TU on T.ID = TU.TRIP_ID
LEFT JOIN TB_USER U on U.ID = TU.USER_ID
WHERE T.END_TIME>SYSDATE
ORDER BY T.ID;
```

姓名	电话	职位	地点	原因	开始时间	结束时间	状态码
程璐	17597373353	水疗经理	中山	Difficult	2022-04-19 04:07:23	2022-06-02 21:50:06	2
尹子异	19933669666	建筑师	中山	Difficult	2022-04-19 04:07:23	2022-06-02 21:50:06	2
于杰宏	16623840286	保险销售代理	中山	Difficult	2022-04-19 04:07:23	2022-06-02 21:50:06	2
陈睿	16626204115	护士	广州市	Navicat	2022-04-19 03:54:53	2022-06-29 13:07:51	2
卢宇宁	19809117529	首席运营官	广州市	Navicat	2022-04-19 03:54:53	2022-06-29 13:07:51	2
尹子异	19933669666	建筑师	广州市	Navicat	2022-04-19 03:54:53	2022-06-29 13:07:51	2

查询出差地点最多的城市

```
SELECT DISTINCT LOCATION,COUNT(*) over (PARTITION BY LOCATION)
AS NUM
FROM TB_BUSINESSTRIP ORDER BY NUM DESC ;
```

```
SELECT DISTINCT LOCATION,COUNT(*) over (PARTITION BY LOCATION) AS NUM FROM TB_BUSINESSTRIP ORDER BY NUM DESC ;
```

LOCATION	NUM
▶ 东莞市	52
上海市	45
成都市	44
北京市	43
深圳	41
广州市	38
中山	37

查找在职员工中工作时间最长的

```
SELECT DISTINCT TU.USERNAME, COUNT(*) over (PARTITION BY
USER_ID) AS NUM
FROM TB_ATTENDANCE LEFT JOIN TB_USER TU on TU.ID =
TB_ATTENDANCE.USER_ID
WHERE TU.STATE=1
ORDER BY NUM DESC;
```

```
SELECT DISTINCT TU.USERNAME, COUNT(*) over (PARTITION BY USER_ID) AS NUM
FROM TB_ATTENDANCE LEFT JOIN TB_USER TU on TU.ID = TB_ATTENDANCE.USER_ID
WHERE TU.STATE=1
ORDER BY NUM DESC;
```

USERNAME	NUM
▶ 尹子异	632
段震南	622
罗诗涵	621
邹宇宁	619
邓嘉伦	617
韩子韬	617
侯睿	617
吕安琪	615
邹诗涵	611

查询在职员工在 2022 年 4 月期间的请假信息

```
SELECT
TU.USERNAME, TU.PHONE, TU.JOB, TV.REASON, TV.START_TIME, TV.END_TIM
E
FROM TB_VACATE TV LEFT JOIN TB_USER TU on TU.ID = TV.USER_ID
WHERE TV.START_TIME>TO_DATE('2022-04-01', 'yyyy-mm-dd') AND
TV.STATE=3 AND TU.STATE=1;
```

```
SELECT TU.USERNAME,TU.PHONE,TU.JOB,TV.REASON,TV.START_TIME,TV.END_TIME
FROM TB_VACATE TV LEFT JOIN TB_USER TU on TU.ID = TV.USER_ID
WHERE TV.START_TIME>TO_DATE('2022-04-01','yyyy-mm-dd') AND TV.STATE=3 AND TU.STATE=1;
```

USERNAME	PHONE	JOB	REASON	START_TIME	END_TIME
谭璐	14544993416	导师	HTTP Tunnel	2022-04-09 21:30:05	2022-04-22 16:26:15
雷云熙	18942068802	水疗经理	Remember t	2022-04-28 05:28:40	2022-05-19 08:30:06

在职工中 2022 年度加班时间的总小时数

```
SELECT DISTINCT TU.USERNAME, COUNT(END_TIME-START_TIME) over
(PARTITION BY USER_ID) AS HOURSE
FROM TB_WORKOVERTIME LEFT OUTER JOIN TB_USER TU on TU.ID =
TB_WORKOVERTIME.USER_ID
WHERE START_TIME>TO_DATE('2022-01-01','yyyy-mm-dd') AND
TU.STATE=1
ORDER BY HOURSE DESC;
```

```
SELECT DISTINCT TU.USERNAME,COUNT(END_TIME-START_TIME) over (PARTITION BY USER_ID) AS HOURSE
FROM TB_WORKOVERTIME LEFT OUTER JOIN TB_USER TU on TU.ID = TB_WORKOVERTIME.USER_ID
WHERE START_TIME>TO_DATE('2022-01-01','yyyy-mm-dd') AND TU.STATE=1
ORDER BY HOURSE DESC;
```

USERNAME	HOUSE
陈睿	2
王昭君	2
雷云熙	1
徐杰宏	1
阎秀英	1
尹子异	1
邹宇宁	1
邹诗涵	1

查询 2021 年度在各个地区花费的总出差费

```
SELECT DISTINCT LOCATION,SUM(COST) over (PARTITION BY
LOCATION) AS TOTAL_COST
FROM TB_BUSINESSTRIP
WHERE START_TIME>TO_DATE('2021-01-01','yyyy-mm-dd') AND
END_TIME<TO_DATE('2022-01-01','yyyy-mm-dd') AND STATE=3;
```

```
SELECT DISTINCT LOCATION,SUM(COST) over (PARTITION BY LOCATION) AS TOTAL_COST
FROM TB_BUSINESSTRIP
WHERE START_TIME>TO_DATE('2021-01-01','yyyy-mm-dd') AND END_TIME<TO_DATE('2022-01-01','yyyy-mm-dd') AND STATE=3;
```

LOCATION	TOTAL_COST
成都市	1245685.01
中山	2287495.59
广州市	6482.39
东莞	751607.36
北京市	508266.56
▶上海市	1019332.54

阶段四

依据阶段三的成果报告，使用 *Java Swing* 及 *Oracle JDBC* 设计与实现数据库系统的视窗介面与应用程序编程。阶段四的成果报告是数据库应用系统的心得与结果，并包含视窗操作的截图，每个操作必须具有文字说明。

技术选型

技术选型

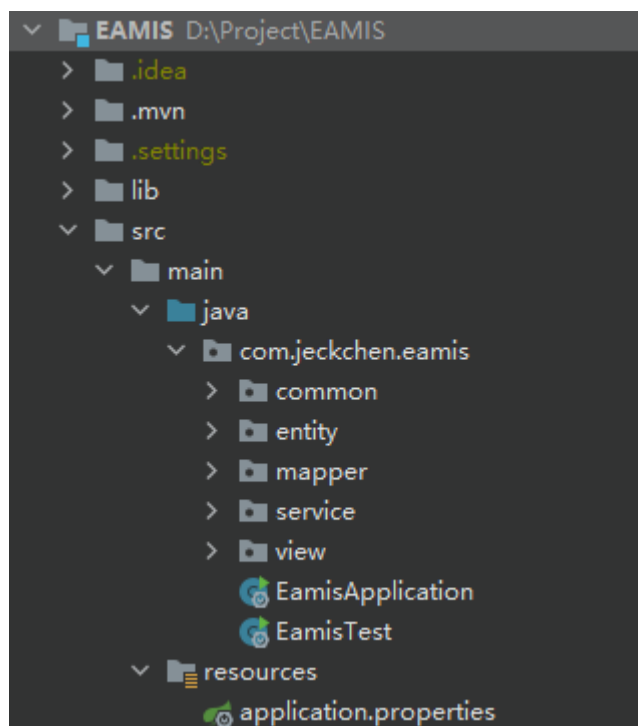
- 1、开发语言：Java（JDK1.8）
- 2、数据库：Oracle 11c
- 3、开发工具：Eclipse、idea、Navicat（数据库可视化工具）
- 4、主要技术栈：
 - 1、Java Swing
 - 2、SpringBoot 2.5.13
 - 3、MybatisPlus 3.0.5
 - 4、Maven
 - 5、Hutool 5.8.1

选型说明

选用 springboot 主要是利用其控制反转的特性（IoC）以及对于 ORM 框架的良好支持。使用 spring 的 IoC 可以在一定程度上提高程序的健壮性，便于扩展。ORM 框架使用了 MyBatisPlus，是目前流行的主要 ORM 框架之一，它对于常用的数据库操作进行了封装，便于开发者更快更好的对数据进行操作。Hutool 是一款非侵入性的工具，里面包含了常用的功能，包括但不限于对于日期计算，ID 生成等操作。

项目结构

本项目层次主要分为 common, entity, mapper, service, view 这五个。



Common

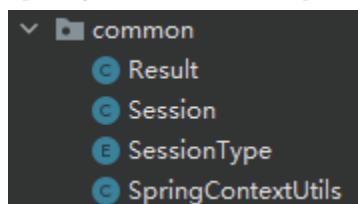
Common: 存放了通用的部件。其中

Result: 返回结果包装。

Session: 类似于浏览器的 Session, 在本项目中用于保存登录用户的信息, 保存模块信息。

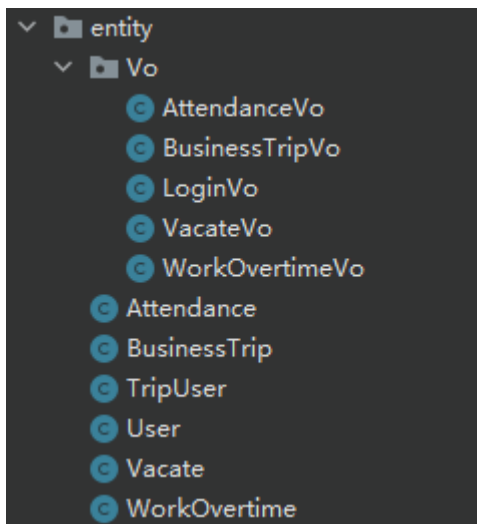
SessionType: 作为 Session 类的枚举, 规范格式。

SpringContextUtils: spring 上下文工具, 用于获取 bean



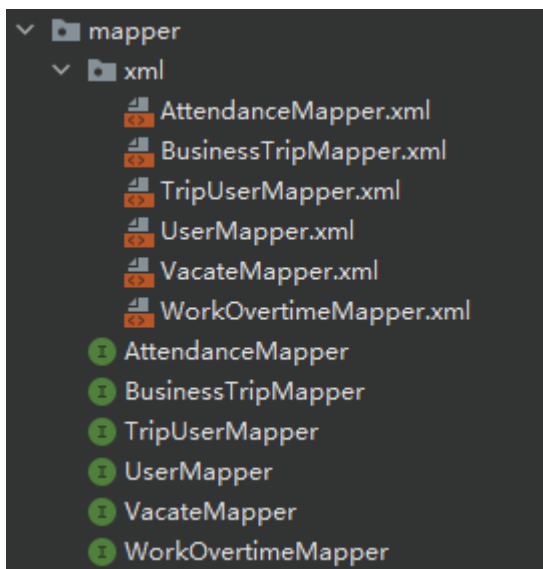
Entity

Entity: 存放了与数据对应的实体类, 和用于显示的 Vo 类。



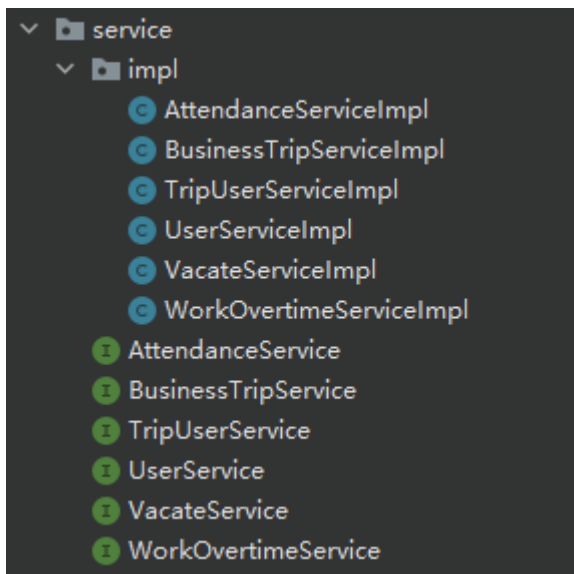
Mapper

Mapper: 用于基本操作数据库，可以自定义编写 SQL 语句。



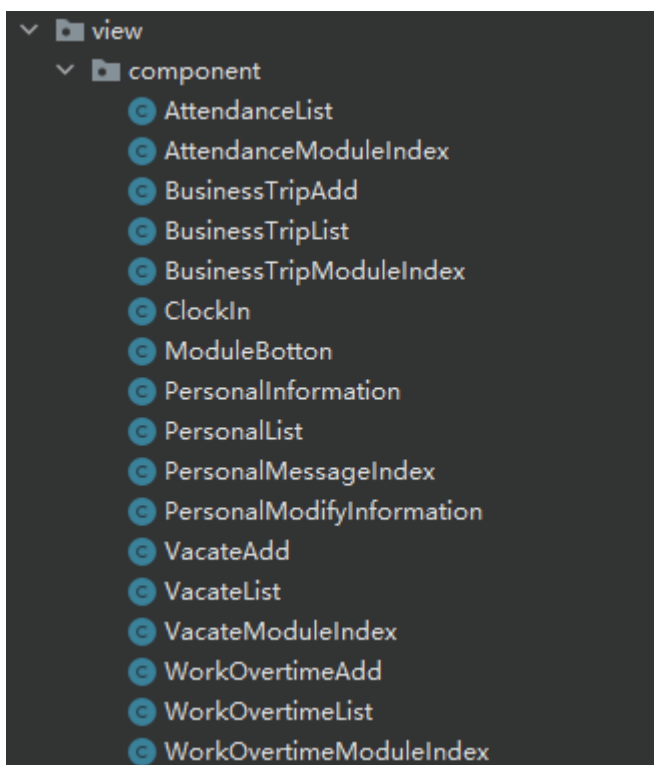
Service

Service: 用于提供业务方面的服务，会调用 Mapper 完成对于数据库数据的操作。



View

View: 使用 swing 编写的 GUI 页面，便于进行人机交互。

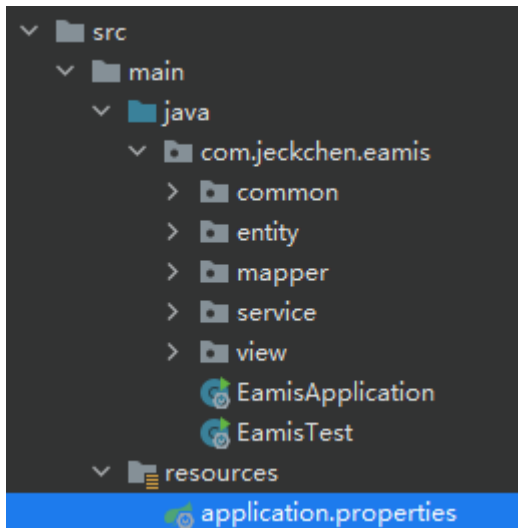


如何启动

在经历过第三阶段的数据库配置后，便可以在改 Java 中进行链接。（数据库配置，数据库创建，数据导入等，详见第三阶段）

配置 properties

打开项目中的 src/main/resources/application.properties 文件。



编辑其中的配置为你的数据库

```
1 spring.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver  驱动名，通常不需要改
2 spring.datasource.url=jdbc:oracle:thin:@192.168.180.130:1521:orcl  数据库url地址
3 spring.datasource.username=u_jeck  你所使用的Oracle数据库用户名
4 spring.datasource.password=123456  用户名对应的密码
```

使用启动类

在确认 Maven 导包结束后，运行

src/main/java/com/jeckchen/eamis/EamisApplication.java

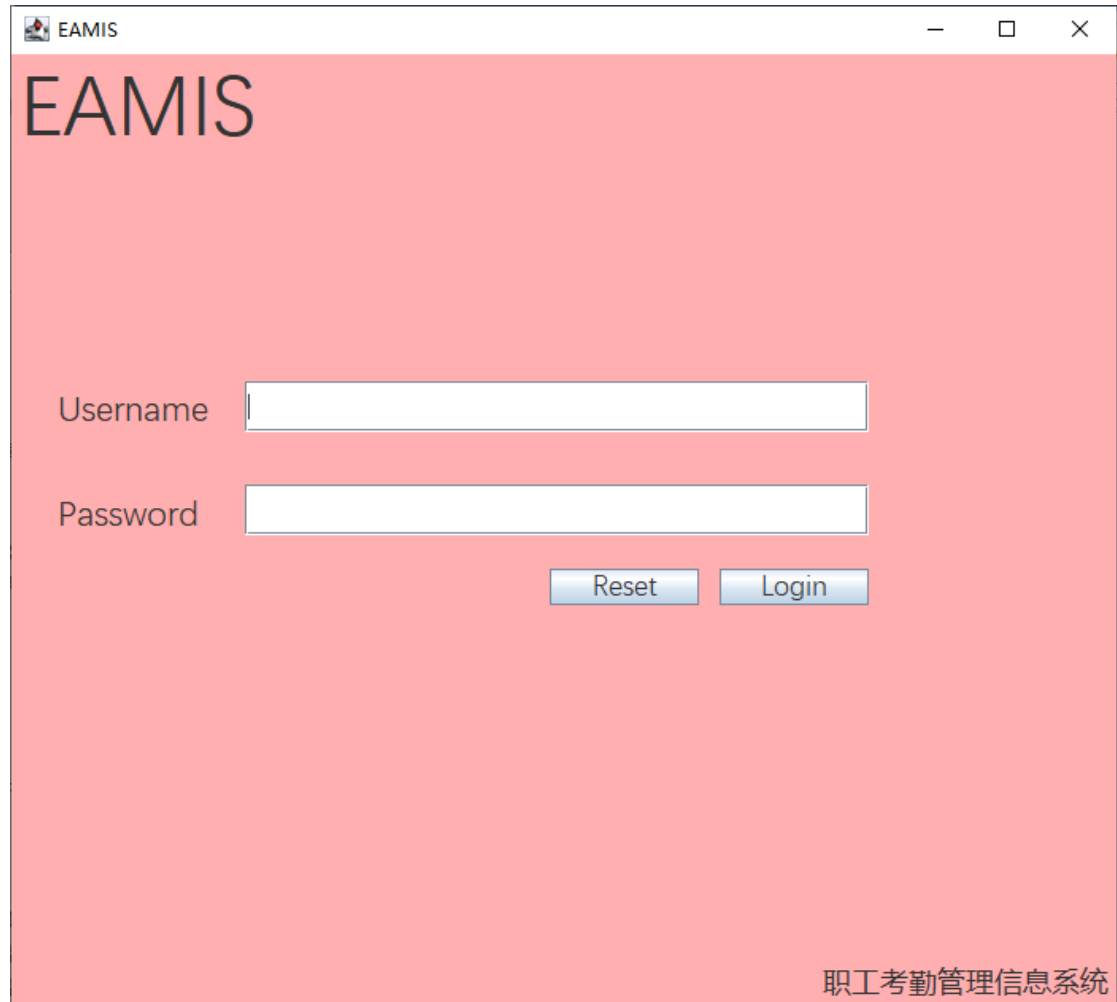
```
1 package com.jeckchen.eamis;
2
3 import ...
4
5
6
7
8 @SpringBootApplication
9 @ComponentScan(basePackages = {"com.jeckchen"})
10 public class EamisApplication {
11
12     public static void main(String[] args) {
13         SpringApplication.run(EamisApplication.class, args);
14         Start.run();
15         // 如果报java.awt.HeadlessException, 请在VM options中追加 -Djava.awt.headless=false
16         // https://blog.csdn.net/yhj_911/article/details/104097594
17     }
18
19 }
20
```

程序运行

登录界面

推荐使用 账户 1812090111 密码 123456

在正常启动后，您会看到这个页面



EAMIS

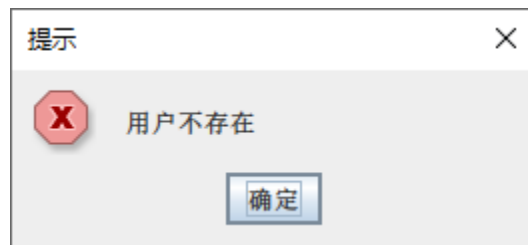
Username

Password

Reset Login

职工考勤管理信息系统

如果您输入的账号不存在，会有提示



如果密码输入错误



首页

当您成功登录后，会看到以下界面
选择对应的模块就可以进入



用户信息模块
个人信息

EAMIS

个人信息

修改信息

所有职工

用户名:项羽

电话: 18120901111

性别: 女

生日: 2022年05月31日

职位: boss

入职时间: 2022年05月16日

状态: 2.0

用户id: 1511637171867447000

状态提示: 0->注销 1->在职 2->管理员

2022年06月15日 19时53分

Home

修改信息

当前为管理员用户下，所有信息均可以编辑

EAMIS

个人信息

修改信息

所有职工

ID

1511637171867447000

查询

用户名

项羽

电话

18120901111

性别

0

生日

2022年05月31日

职位

boss

入职时间

2022年05月16日

状态

2. 0

状态提示： 0->注销 1->在职 2->管理员

输入时间格式为：1970-01-01

提交

2022年06月15日 19时53分

Home

不仅如此，还可以编辑其他人的信息

EAMIS

—

□

×

个人信息

修改信息

所有职工

ID

1376489856438086144

查询

用户名

丁震南

电话

18721266716

性别

0

生日

2016年05月31日

职位

生物化学家

入职时间

2000年05月14日

状态

1.0

状态提示：0->注销 1->在职 2->管理员

输入时间格式为：1970-01-01

提交

2022年06月15日 20时31分

Home

当用户没有权限的时候，编辑框将不能编辑

EAMIS

个人信息

修改信息

所有职工

ID

1376489856438086144

查询

用户名

丁震南

电话

18721266716

性别

0

生日

2016年05月31日

职位

生物化学家

入职时间

2000年05月14日

状态

1. 0

状态提示： 0->注销 1->在职 2->管理员

输入时间格式为：1970-01-01

提交

2022年06月15日 20时29分

Home

还可以通过改页面添加新的用户

个人信息

修改信息

所有职工

ID

5877921081410105344

查询

用户名

谭璐

电话

14544993416

性别

1

生日

2008年07月21日

职位

导师

入职时间

2020年08月04日

状态

1.0

状态提示： 0->注销 1->在职 2->管理员

输入时间格式为：1970-01-01

提交

2022年06月15日 20时31分

Home

查询所有用户

EAMIS							
<div> <div>个人信息</div> <div>修改信息</div> <div>所有职工</div> </div>							
状态提示: 0->删除 1->在职 2->异常							
id	username	phone	sex	birthday	job	entrytime	state
1376489856...	丁震南	18721266716	0.0	2016年05月...	生物化学家	2000年05月...	1.0
5877921081...	譚璐	14544993416	1.0	2008年07月...	导师	2020年08月...	1.0
1469144004...	罗詩涵	15676660959	2.0	2010年07月...	客戶協調員	2020年01月...	1.0
7185286713...	顾子昇	14597098032	2.0	2020年08月...	团体领导	2002年06月...	1.0
2347351397...	胡安琪	19887326044	0.0	2007年05月...	活动经理	2017年02月...	0.0
4316654561...	韩子韬	16629894551	1.0	2010年10月...	工程师	2017年03月...	1.0
8209856577...	韩晓明	13484330239	0.0	2000年05月...	牙医	2022年05月...	1.0
8166266253...	陈璐	16680720881	1.0	2021年05月...	多媒体动画师	2016年02月...	1.0
4945884100...	姜云熙	13821365148	0.0	2008年10月...	多媒体动画师	2000年04月...	1.0
8175270811...	阎秀英	14982688474	1.0	2022年01月...	歌手	2021年09月...	1.0
3310494741...	程璐	17597373353	2.0	2017年01月...	水疗经理	2017年08月...	0.0
7799701429...	徐杰宏	19883613299	1.0	2006年08月...	牙医	2003年06月...	1.0
7251692689...	汤安琪	16672385749	2.0	2021年10月...	社交媒体协...	2012年04月...	0.0
1158394635...	李宇宁	16654262941	0.0	2001年08月...	市场总监	2007年10月...	0.0
2022年06月15日 20时31分							
							Home

考勤模块

未打卡

当没有打卡的时候，将出现打卡的按钮



已打卡

打卡会有提示



已打卡后页面会是这样



打卡记录

EAMIS

个人打卡

打卡记录

状态提示：0->提早 1->正常 2->迟到

username	create_time	state
项羽	2022年06月15日 20:35	2.0
项羽	2022年06月14日 14:53	2.0
项羽	2022年06月13日 23:31	2.0
项羽	2022年05月19日 10:16	2.0
项羽	2022年05月16日 00:06	1.0
项羽	2022年05月11日 08:01	2.0
项羽	2022年05月05日 10:40	2.0
项羽	2022年04月29日 08:55	2.0
项羽	2022年04月13日 10:40	2.0
项羽	2022年03月30日 08:42	2.0
项羽	2022年03月23日 01:38	2.0
项羽	2022年03月04日 10:11	2.0
项羽	2022年01月14日 09:23	2.0
项羽	2022年01月13日 10:42	2.0

2022年06月15日 20时35分

Home

请销假模块

请假表

可以查看到过去的请销假情况，可以对请假信息进行开始，销假和删除的操作。

EAMIS

请假表

新增请假

状态提示: 0->删除 1->未开始 2->进行中 3->结束

id	username	reason	start_time	end_time	state
153690700319811...	项羽	饿了	2022年06月15日 ...	2022年06月15日 ...	1. 0
633958996580953...	项羽	To open a query...	2021年09月10日 ...	2021年09月25日 ...	0. 0
338177792943264...	项羽	It provides str...	2019年06月29日 ...	2019年07月06日 ...	0. 0
671207444391810...	项羽	In the Objects ...	2018年11月04日 ...	2018年11月17日 ...	2. 0

Start

ReportBack

Delete

2022年06月15日 20时36分

Home

新增请假

EAMIS

请假表

新增请假

原因

泡妞

开始时间

2022-6-15 16:00

结束时间

2022-6-16 07:00

Tip: 时间格式1970-01-01 14:15 默认状态为1, 既未开始


Submit

2022年06月15日 20时36分

Home

提示

×



是否成功: true

确定

EAMIS

请假表

新增请假

状态提示: 0->删除 1->未开始 2->进行中 3->结束

id	username	reason	start_time	end_time	state
153705196611645...	项羽	泡妞	2022年06月15日 ...	2022年06月16日 ...	1.0
153690700319811...	项羽	饿了	2022年06月15日 ...	2022年06月15日 ...	1.0
633958996580953...	项羽	To open a query...	2021年09月10日 ...	2021年09月25日 ...	0.0
338177792943264...	项羽	It provides str...	2019年06月29日 ...	2019年07月06日 ...	0.0
671207444391810...	项羽	In the Objects ...	2018年11月04日 ...	2018年11月17日 ...	2.0

Start

ReportBack

Delete

2022年06月15日 20时36分

Home

加班模块
加班信息

EAMIS

加班

加班申请

状态提示：0->删除 1->进行中 2->结束

id	username	reason	start_time	end_time	state
153690986720842...	项羽	作业做不完	2022年06月15日 ...	2022年06月15日 ...	1. 0
651163717186744...	项羽	老板逼得	2022年05月16日 ...	2022年05月16日 ...	2. 0
478793302944309...	项羽	Creativity is i...	2022年04月09日 ...	2022年04月09日 ...	2. 0
551649930634570...	项羽	The reason why ...	2019年05月12日 ...	2019年05月13日 ...	0. 0

Start

End

Delete

2022年06月15日 20时39分

Home

加班申请

EAMIS

加班

加班申请

原因

开始时间

结束时间

Tip: 时间格式1970-01-01 14:15 默认状态为1, 既进行中

Submit

2022年06月15日 20时39分

Home

出差模块

出差信息

EAMIS

出差

出差申请

状态提示: 0->删除 1->未开始 2->进行中 3->结束

id	username	reason	location	cost	start_time	end_time	state
1536981166...	项羽	赚大钱	福州	10.0	2022年06月...	2022年06月...	1.0
8869947337...	项羽	How we spe...	广州市	620670.08	2019年08月...	2019年09月...	0.0
9511513945...	项羽	How we spe...	深圳	835513.73	2018年08月...	2018年08月...	3.0
8088117601...	项羽	If you wai...	成都市	666313.55	2017年11月...	2017年11月...	3.0

Start

End

Delete

2022年06月15日 20时43分

Home

出差申请

EAMIS

出差

出差申请

原因

开始时间

结束时间

位置

花费

Tip: 时间格式1970-01-01 14:15 默认状态为1, 既未开始

Submit

2022年06月15日 20时43分

Home

GUI 逻辑结构

GUI 部分采用模块化设计，在 SpringBoot 领导这个项目启动的时候，会带起 Start.class 进而带动整个 GUI 部分的启动，启动后会直接跳转至 Login 界面，通过 Login 界面进入到 Home 界面，就可以选择对应的模块进行使用。模块只用一个 JFrame，模块中的内容全部都是以 JPanel 的形式进行载入，保证了其后期的拓展性。

Swing图形化界面启动流程

