目 录

目录

[一、 需求分析 2](#_Toc185278551)

[二、 系统功能结构图 3](#_Toc185278552)

[三、 数据库逻辑结构 3](#_Toc185278553)

[四、 物理结构设计 5](#_Toc185278554)

[五、 数据库行为设计 7](#_Toc185278555)

[六、 功能测试 10](#_Toc185278556)

# 需求分析

**1. 用户需求**

* **求职者**
  + 注册并登录系统。
  + 浏览职位信息。
  + 申请感兴趣的职位。
  + 查看自己的申请记录。
* **招聘者**
  + 注册并登录系统。
  + 发布职位信息。
  + 查看职位的申请情况。
  + 管理（更新、删除）已发布的职位。

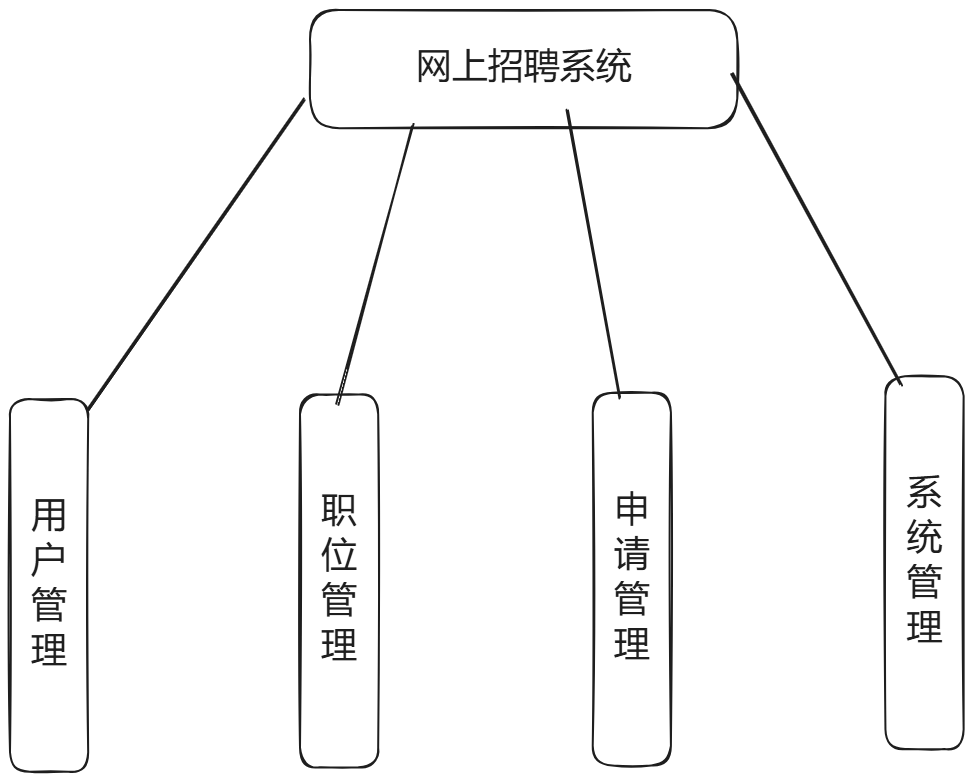
**2. 系统需求**

* **功能需求**
  + 用户注册与登录功能。
  + 职位信息的管理与查询功能。
  + 职位申请功能。
  + 查看申请历史与状态功能。
* **非功能需求**
  + 系统应具有良好的响应速度，支持高并发用户。
  + 数据库设计应保证数据完整性与一致性（如通过外键约束）。
  + 界面简洁易用，操作方便。
  + 数据隐私保护，如加密用户密码。

**3. 技术需求**

* 数据库：MySQL
* 软件：nacivat
* 操作系统：win10

# 系统功能结构图



# 数据库逻辑结构

关系模式：

**关系模式**

**表1：Users（用户信息表）**

* 描述用户的基本信息，包括求职者和招聘者。
* 主要属性：
  + user\_id：用户唯一标识，主键。
  + username：用户名。
  + password：密码。
  + role：用户角色（求职者或招聘者）。
  + email：邮箱。

**表2：Jobs（职位信息表）**

* 描述招聘者发布的职位信息。
* 主要属性：
  + job\_id：职位唯一标识，主键。
  + job\_title：职位名称。
  + description：职位描述。
  + company\_name：公司名称。
  + user\_id：发布职位的招聘者，外键。

**表3：Applications（职位申请表）**

* 描述求职者申请的职位信息。
* 主要属性：
  + application\_id：申请唯一标识，主键。
  + user\_id：申请者的ID，外键。
  + job\_id：申请职位的ID，外键。
  + application\_date：申请日期。

数据库表结构：

**表1：Users**

| **字段名** | **数据类型** | **主键** | **外键** | **说明** |
| --- | --- | --- | --- | --- |
| user\_id | INT AUTO\_INCREMENT | 是 | 否 | 用户唯一标识 |
| username | VARCHAR(50) | 否 | 否 | 用户名 |
| password | VARCHAR(50) | 否 | 否 | 密码 |
| role | ENUM('求职者', '招聘者') | 否 | 否 | 用户角色 |
| email | VARCHAR(100) | 否 | 否 | 邮箱 |

**表2：Jobs**

| **字段名** | **数据类型** | **主键** | **外键** | **说明** |
| --- | --- | --- | --- | --- |
| job\_id | INT AUTO\_INCREMENT | 是 | 否 | 职位唯一标识 |
| job\_title | VARCHAR(100) | 否 | 否 | 职位名称 |
| description | TEXT | 否 | 否 | 职位描述 |
| company\_name | VARCHAR(100) | 否 | 否 | 公司名称 |
| user\_id | INT | 否 | 是 | 发布者ID，外键 |

**表3：Applications**

| **字段名** | **数据类型** | **主键** | **外键** | **说明** |
| --- | --- | --- | --- | --- |
| application\_id | INT AUTO\_INCREMENT | 是 | 否 | 申请唯一标识 |
| user\_id | INT | 否 | 是 | 申请者ID，外键 |
| job\_id | INT | 否 | 是 | 职位ID，外键 |
| application\_date | DATE | 否 | 否 | 申请日期 |

# 物理结构设计

-- 创建数据库

CREATE DATABASE OnlineRecruitmentSystem;

USE OnlineRecruitmentSystem;

-- 创建 Users 表

CREATE TABLE Users (

user\_id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(50) NOT NULL,

password VARCHAR(50) NOT NULL,

role ENUM('求职者', '招聘者') NOT NULL,

email VARCHAR(100) NOT NULL

);

-- 创建 Jobs 表

CREATE TABLE Jobs (

job\_id INT AUTO\_INCREMENT PRIMARY KEY,

job\_title VARCHAR(100) NOT NULL,

description TEXT NOT NULL,

company\_name VARCHAR(100) NOT NULL,

user\_id INT NOT NULL,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id)

);

-- 创建 Applications 表

CREATE TABLE Applications (

application\_id INT AUTO\_INCREMENT PRIMARY KEY,

user\_id INT NOT NULL,

job\_id INT NOT NULL,

application\_date DATE NOT NULL,

FOREIGN KEY (user\_id) REFERENCES Users(user\_id),

FOREIGN KEY (job\_id) REFERENCES Jobs(job\_id)

);

-- 插入测试数据

-- 插入用户

INSERT INTO Users (username, password, role, email)

VALUES

('alice', 'password123', '求职者', 'alice@example.com'),

('bob', 'password456', '招聘者', 'bob@example.com');

-- 插入职位

INSERT INTO Jobs (job\_title, description, company\_name, user\_id)

VALUES

('Software Engineer', 'Develop and maintain software solutions.', 'TechCorp', 2),

('Product Manager', 'Lead product development teams.', 'TechCorp', 2);

-- 插入职位申请

INSERT INTO Applications (user\_id, job\_id, application\_date)

VALUES

(1, 1, CURDATE()),

(1, 2, CURDATE());

-- 查询测试

-- 查看所有用户

SELECT \* FROM Users;

-- 查看所有职位

SELECT \* FROM Jobs;

-- 查看职位申请

SELECT \* FROM Applications;

# 数据库行为设计

-- 在 Users 表中添加索引

CREATE INDEX idx\_username ON Users(username);

CREATE INDEX idx\_email ON Users(email);

-- 在 Jobs 表中添加索引

CREATE INDEX idx\_job\_title ON Jobs(job\_title);

CREATE INDEX idx\_company\_name ON Jobs(company\_name);

-- 在 Applications 表中添加索引

CREATE INDEX idx\_application\_date ON Applications(application\_date);

CREATE INDEX idx\_user\_job ON Applications(user\_id, job\_id);

-- 求职者视图：查看申请记录及职位详情

CREATE VIEW ApplicantApplications AS

SELECT

a.application\_id,

u.username AS applicant\_name,

j.job\_title,

j.company\_name,

a.application\_date

FROM Applications a

JOIN Users u ON a.user\_id = u.user\_id

JOIN Jobs j ON a.job\_id = j.job\_id

WHERE u.role = '求职者';

-- 招聘者视图：查看职位及申请详情

CREATE VIEW RecruiterJobs AS

SELECT

j.job\_id,

j.job\_title,

j.company\_name,

COUNT(a.application\_id) AS total\_applications

FROM Jobs j

LEFT JOIN Applications a ON j.job\_id = a.job\_id

GROUP BY j.job\_id, j.job\_title, j.company\_name;

-- 用户视图：展示所有用户信息，隐藏密码

CREATE VIEW UserInfo AS

SELECT

user\_id,

username,

role,

email

FROM Users;

-- 触发器 1：在用户注册时自动记录时间

CREATE TRIGGER trg\_user\_registration

AFTER INSERT ON Users

FOR EACH ROW

BEGIN

INSERT INTO SystemLogs (log\_type, description, log\_date)

VALUES ('User Registration', CONCAT('New user registered: ', NEW.username), NOW());

END;

-- 触发器 2：记录职位申请操作

CREATE TRIGGER trg\_application\_creation

AFTER INSERT ON Applications

FOR EACH ROW

BEGIN

INSERT INTO SystemLogs (log\_type, description, log\_date)

VALUES ('Job Application',

CONCAT('User ', NEW.user\_id, ' applied for Job ', NEW.job\_id),

NOW());

END;

-- 触发器 3：禁止删除用户表中的管理员账号

CREATE TRIGGER trg\_prevent\_admin\_deletion

BEFORE DELETE ON Users

FOR EACH ROW

BEGIN

IF OLD.role = '管理员' THEN

SIGNAL SQLSTATE '45000'

SET MESSAGE\_TEXT = 'Admin account cannot be deleted.';

END IF;

END;

-- 存储过程 1：插入职位

DELIMITER $$

CREATE PROCEDURE AddJob(

IN job\_title VARCHAR(100),

IN description TEXT,

IN company\_name VARCHAR(100),

IN user\_id INT

)

BEGIN

INSERT INTO Jobs (job\_title, description, company\_name, user\_id)

VALUES (job\_title, description, company\_name, user\_id);

-- 添加到日志

INSERT INTO SystemLogs (log\_type, description, log\_date)

VALUES ('Job Creation', CONCAT('Job created by user ID: ', user\_id), NOW());

END $$

DELIMITER ;

-- 存储过程 2：统计求职者申请数量

DELIMITER $$

CREATE PROCEDURE GetApplicantStats(

IN applicant\_id INT

)

BEGIN

SELECT

u.username AS applicant\_name,

COUNT(a.application\_id) AS total\_applications

FROM Applications a

JOIN Users u ON a.user\_id = u.user\_id

WHERE u.user\_id = applicant\_id

GROUP BY u.username;

END $$

DELIMITER ;

# 功能测试

-- 添加一个招聘者

INSERT INTO Users (username, password, role, email)

VALUES ('david', 'securepwd', '招聘者', 'david@example.com');



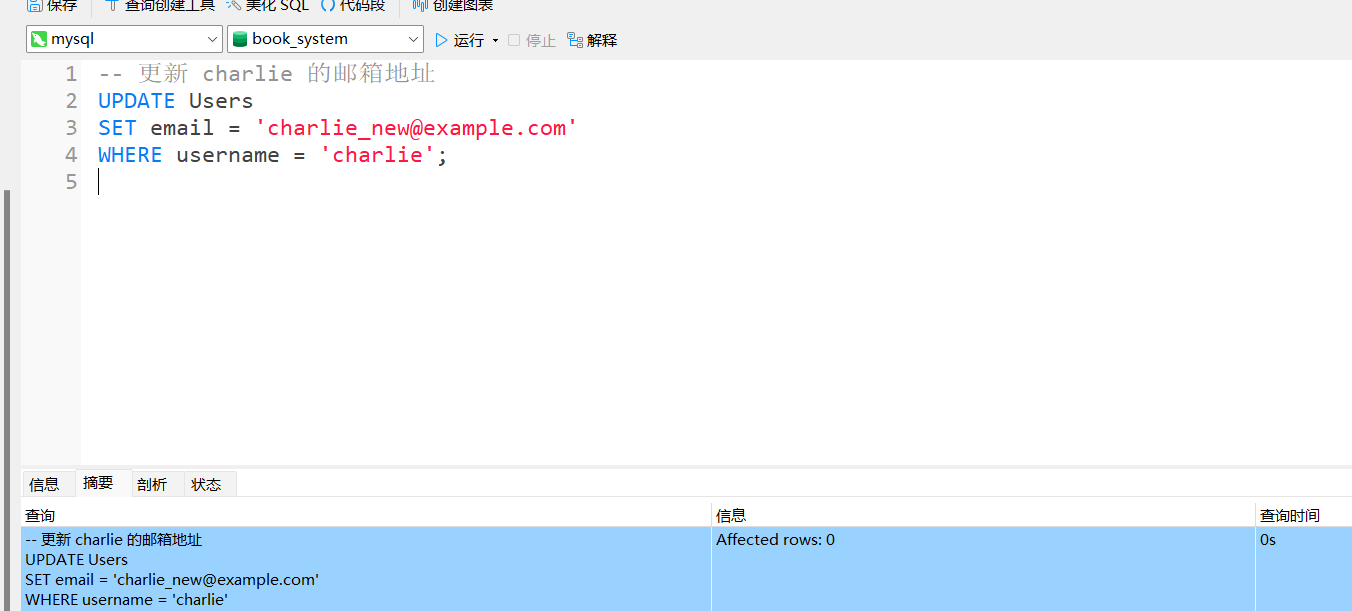
更新用户信息

-- 更新 charlie 的邮箱地址

UPDATE Users

SET email = 'charlie\_new@example.com'

WHERE username = 'charlie';



删除用户

-- 删除用户 charlie

DELETE FROM Users

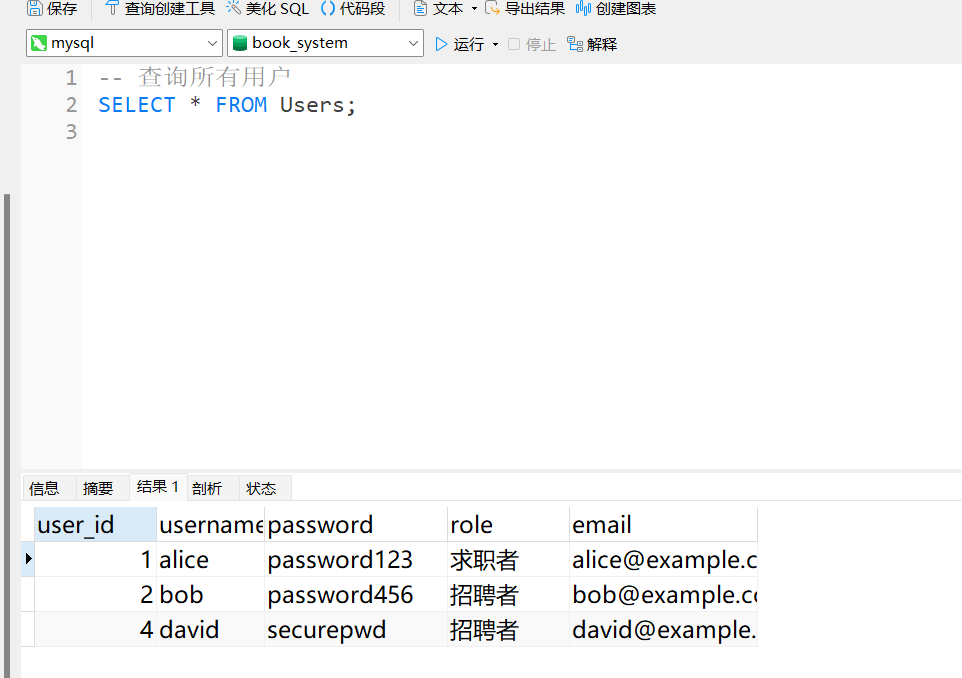
WHERE username = 'charlie';



查询用户

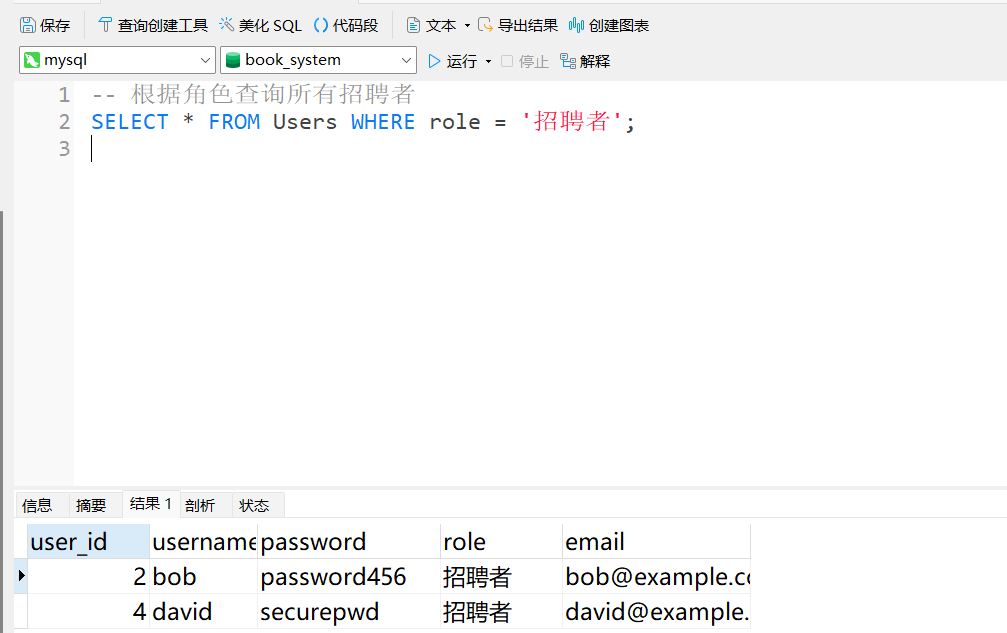
-- 查询所有用户

SELECT \* FROM Users;



-- 根据角色查询所有招聘者

SELECT \* FROM Users WHERE role = '招聘者';



测试用户注册触发器

-- 添加新用户，触发日志记录

INSERT INTO Users (username, password, role, email)

VALUES ('emma', 'passwordxyz', '求职者', 'emma@example.com');

-- 查看触发器记录的日志

SELECT \* FROM SystemLogs WHERE log\_type = 'User Registration';

