

# “植悟” 软件 数据库设计说明书

开发小组：成群结队

小组成员：余琪丽，蒋伊妍，钟扬，郑乐妍，陈芳玲，  
徐紫嫣，姜品伊，钟媛，方怡玥，洪蓉玫，黄伊萍

## 目录

“植悟”软件 数据库设计说明书 .....	1
一、总述 .....	3
(一) 编写目的 .....	3
二、外部设计 .....	3
(一) 环境说明 .....	3
(二) 指导原则 .....	3
(三) 配置说明 .....	4
三、物理实现 .....	4
(一) 物理结构 .....	4
(二) 安全设计 .....	4
四、表设计结构 .....	5
(一) 数据库表目录 .....	5
(二) 数据库表结构 .....	5
(三) 视图的设计 .....	8
五、存储过程、函数及触发器的设计 .....	10
(一) 存储过程 .....	10
(二) 触发器 .....	11
六、数据库标准 .....	12
(一) 表和视图命名规范 .....	12
(二) 字段命名规范 .....	12
(三) 表结构设计 .....	12
(四) 表字段设计 .....	13
七、数据元精度标准 .....	13
八、数据库账号及权限说明 .....	13

## 一、总述

### （一）编写目的

本文档旨在定义"植悟 (FloraMind)"系统的数据库结构、表关系、字段规范及数据管理策略，为后续数据库开发、维护与扩展提供技术依据，确保数据一致性、完整性及系统性能。本文档面向数据库管理员、开发人员、测试人员及相关技术决策者，与系统设计说明书保持一致，支撑系统分层架构的数据层实现。

## 二、外部设计

### （一）环境说明

开发环境（PHPStudy 本地环境）

集成环境：PHPStudy V8.3（Windows）

数据库版本：MySQL 8.0（PHPStudy 内置）

字符集配置：utf8mb4 / utf8mb4\_unicode\_ci

默认连接：

主机：localhost 或 127.0.0.1

端口：3306

用户名：root

密码：root

管理工具：phpMyAdmin（内置，访问 <http://localhost/phpmyadmin>）

### （二）指导原则

#### 1. 开发规范

环境统一：所有开发成员使用 PHPStudy 统一开发环境

ORM 优先：使用 SQLAlchemy ORM 进行数据库操作

数据初始化：通过 SQL 脚本在 phpMyAdmin 中初始化

版本控制：数据库变更脚本纳入 Git 版本管理

#### 2. 数据安全

开发环境：使用默认 root 账号，仅限于本地访问

敏感数据：用户手机号、邮箱等字段加密存储

操作审计：关键数据操作记录审计日志

#### 3. 性能优化

索引策略：为查询条件字段建立合适索引

查询优化：避免 N+1 查询，使用关联加载

数据归档：历史数据定期归档，维护表性能

## （三）配置说明

### 1. 环境搭建步骤

下载安装 PHPStudy V8.3，启动 MySQL 服务，通过 phpMyAdmin 创建 flora\_mind 数据库，执行数据库设计说明书中的 SQL 脚本

### 2. 日常使用流程

数据查看：使用 phpMyAdmin 浏览表数据

SQL 执行：在 phpMyAdmin 的 SQL 窗口执行调试语句

结构修改：通过 ALTER TABLE 语句更新表结构

数据备份：使用 phpMyAdmin 导出功能定期备份

## 三、物理实现

### （一）物理结构

数据库命名为：flora\_mind，与系统架构的数据层相对应，支撑四大功能模块的数据存储需求。

核心业务表（支撑用户中心和基础功能）：

1. users（用户表）：支持用户登录与认证
2. plants（植物表）：支持植物档案管理
3. plant\_types（植物种类表）：支撑智能提醒的知识库
4. care\_logs（养护日志表）：支持种植日记功能
5. reminders（提醒任务表）：支持智能养护提醒核心功能

扩展功能表（支撑 AI 服务和外部集成）：

6. weather\_data（天气数据表）：支持 MCP 天气服务数据缓存
7. plant\_images（植物成长图片表）：支持种植日记的图文记录
8. community\_posts（社区帖子表）：预留社区功能扩展（暂不实现）
9. sensors（传感器设备表）：支持未来 IoT 设备接入（暂不实现）
10. sensor\_data（传感器监测数据表）：支撑环境数据采集

系统支持表（支撑系统运维和 AI 服务）：

11. ai\_conversations（AI 对话记录表）：支持 AI 养护助手的多轮对话
12. operation\_logs（操作日志表）：支持系统审计和故障排查
13. system\_config（系统配置表）：存储系统运行参数

### （二）安全设计

身份认证：用户密码使用 BCrypt 加密存储

权限控制：基于角色的访问控制（RBAC）

数据加密：

敏感字段（手机号、邮箱）加密存储

数据传输全程 HTTPS 加密

审计日志：所有敏感操作记录详细日志

备份恢复：

开发环境：手动备份

数据保留策略：业务数据 3 年，日志数据 1 年

## 四、表设计结构

### （一）数据库表目录

表名	说明	增长频率	对应系统功能
users	用户基本信息	低	用户登录与认证
plants	用户种植的植物	中	植物档案管理
plant_types	植物种类库	极低	提醒规则引擎
care_logs	养护操作记录	高	种植日记功能
reminders	智能提醒任务	中	智能养护提醒
weather_data	天气数据缓存	高	MCP 天气服务集成
plant_images	植物成长图片	高	种植打卡
sensors	传感器设备信息	低	未来 IoT 扩展
sensor_data	传感器监测数据	极高	环境数据采集
ai_conversations	AI 对话记录	高	AI 养护助手
operation_logs	系统操作日志	高	系统运维监控

### （二）数据库表结构

#### 1. users（用户表）

字段名	类型	长度	主键	允许空	默认值	说明
user_id	bigint	20	是	否	AUTO_INCREMENT	用户 ID
username	varchar	50	否	否		用户名
email	varchar	100	否	否		邮箱
phone	varchar	20	否	是	NULL	手机号
password	varchar	255	否	否		密码
avatar_url	varchar	255	否	是	NULL	头像 URL
location_city	varchar	100	否	是	NULL	所在城市
notification_preferences	json		否	是	NULL	通知偏好
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间
updated_at	datetime		否	否	CURRENT_TIMESTAMP	更新时间
is_deleted	tinyint	1	否	否	0	删除标记

主键：user\_id

唯一索引：uk\_email (email)

普通索引：idx\_username (username)

# 植 悟

## 2. plants（植物表）

字段名	类型	长度	主键	允许空	默认值	说明
plant_id	bigint	20	是	否	AUTO_INCREMENT	植物 ID
user_id	bigint	20	否	否		用户 ID
plant_type_id	int	11	否	否		植物种类 ID
nickname	varchar	100	否	否		植物昵称
personality_signature	varchar	255	否	是	NULL	个性签名
plant_date	date		否	否		种植日期
initial_photo_url	varchar	255	否	是	NULL	初始照片
location	varchar	100	否	是	NULL	摆放位置
status	varchar	20	否	否	'healthy'	生长状态
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间
updated_at	datetime		否	否	CURRENT_TIMESTAMP	更新时间

主键：plant\_id

外键：fk\_plants\_user (user\_id) REFERENCES users(user\_id)

外键：fk\_plants\_plant\_type` (plant\_type\_id) REFERENCES plant\_types(plant\_type\_id)

普通索引：idx\_user\_status (user\_id, status)

## 3. plant\_types（植物种类表）

字段名	类型	长度	主键	允许空	默认值	说明
plant_type_id	int	11	是	否	AUTO_INCREMENT	种类 ID
name	varchar	100	否	否		植物名称
scientific_name	varchar	100	否	是	NULL	学名
water_freq_days	int	11	否	否	7	浇水频率
light_need	varchar	50	否	否		光照需求
optimal_temperature	varchar	50	否	是	NULL	适宜温度
description	text		否	是	NULL	养护说明
care_instructions	json		否	是	NULL	详细指南

主键：plant\_type\_id

唯一索引：uk\_name (name)

## 4. care\_logs（养护日志表）

字段名	类型	长度	主键	允许空	默认值	说明
log_id	bigint	20	是	否	AUTO_INCREMENT	日志 ID
plant_id	bigint	20	否	否		植物 ID
care_type	varchar	50	否	否		养护类型
care_date	datetime		否	否	CURRENT_TIMESTAMP	养护时间
notes	text		否	是	NULL	备注
image_url	varchar	255	否	是	NULL	图片链接

## 植 悟

weather_condition	varchar	100	否	是	NULL	天气状况
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间

主键: log\_id

外键: fk\_care\_logs\_plant (plant\_id) REFERENCES plants(plant\_id)

复合索引: idx\_plant\_care\_date (plant\_id, care\_date)

### 5. reminders (提醒任务表)

字段名	类型	长度	主键	允许空	默认值	说明
reminder_id	bigint	20	是	否	AUTO_INCREMENT	提醒 ID
user_id	bigint	20	否	否		用户 ID
plant_id	bigint	20	否	否		植物 ID
reminder_type	varchar	50	否	否		提醒类型
reminder_content	text		否	是	NULL	提醒内容
scheduled_at	datetime		否	否		计划时间
is_completed	tinyint	1	否	否	0	是否完成
is_urgent	tinyint	1	否	否	0	是否紧急
trigger_condition	varchar	100	否	是	NULL	触发条件
completed_at	datetime		否	是	NULL	完成时间
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间

主键: reminder\_id

外键: fk\_reminders\_user (user\_id) REFERENCES users(user\_id)

外键: fk\_reminders\_plant (plant\_id) REFERENCES plants(plant\_id)

复合索引: idx\_user\_scheduled (user\_id, scheduled\_at)

索引: idx\_is\_completed (is\_completed)

### 6. sensors (传感器设备表)

字段名	类型	长度	主键	允许空	默认值	说明
sensor_id	bigint	20	是	否	AUTO_INCREMENT	传感器 ID
user_id	bigint	20	否	否		用户 ID
plant_id	bigint	20	否	是	NULL	植物 ID
sensor_type	varchar	50	否	否		传感器类型
device_name	varchar	100	否	否		设备名称
mac_address	varchar	50	否	否		MAC 地址
connection_status	tinyint	1	否	否	0	连接状态
battery_level	int	11	否	是	NULL	电池电量
last_data_update	datetime		否	是	NULL	最后更新
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间
updated_at	datetime		否	否	CURRENT_TIMESTAMP	更新时间

主键: sensor\_id

外键: fk\_sensors\_user (user\_id) REFERENCES users(user\_id)  
外键: fk\_sensors\_plant (plant\_id) REFERENCES plants(plant\_id)  
唯一索引: uk\_mac\_address (mac\_address)

7. sensor\_data (传感器数据表)

字段名	类型	长度	主键	允许空	默认值	说明
data_id	bigint	20	是	否	AUTO_INCREMENT	数据 ID
sensor_id	bigint	20	否	否		传感器 ID
soil_moisture	decimal	5, 2	否	是	NULL	土壤湿度
temperature	decimal	5, 2	否	是	NULL	环境温度
humidity	decimal	5, 2	否	是	NULL	环境湿度
light_intensity	decimal	8, 2	否	是	NULL	光照强度
recorded_at	datetime		否	否	CURRENT_TIMESTAMP	记录时间
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间

主键: data\_id  
外键: fk\_sensor\_data\_sensor (sensor\_id) REFERENCES sensors(sensor\_id)  
复合索引: idx\_sensor\_recorded (sensor\_id, recorded\_at)

8. ai\_conversations (AI 对话记录表)

字段名	类型	长度	主键	允许空	默认值	说明
conversation_id	bigint	20	是	否	AUTO_INCREMENT	对话 ID
user_id	bigint	20	否	否		用户 ID
plant_id	bigint	20	否	是	NULL	植物 ID
user_message	text		否	否		用户消息
ai_response	text		否	否		AI 回复
message_type	varchar	50	否	否	general	问题类型
conversation_context	json		否	是	NULL	对话上下文
created_at	datetime		否	否	CURRENT_TIMESTAMP	创建时间

主键: conversation\_id  
外键: fk\_ai\_conversations\_user (user\_id) REFERENCES users(user\_id)  
外键: fk\_ai\_conversations\_plant (plant\_id) REFERENCES plants(plant\_id)  
索引: idx\_user\_created (user\_id, created\_at)

(三) 视图的设计

1. view\_plant\_care\_summary (植物养护概览视图)

支撑种植日记的成长时间轴功能:

```
01 CREATE VIEW view_plant_care_summary AS
02 SELECT
03     p.plant_id,
```



```
04     p.nickname,
05     p.personality_signature,
06     pt.name AS plant_type,
07     COUNT(cl.log_id) AS care_count,
08     MAX(cl.care_date) AS last_care_date,
09     COUNT(CASE WHEN r.is_completed = 0 THEN 1 END) AS
10 pending_reminders
11 FROM plants p
12 LEFT JOIN plant_types pt ON p.plant_type_id = pt.plant_type_id
13 LEFT JOIN care_logs cl ON p.plant_id = cl.plant_id
14 LEFT JOIN reminders r ON p.plant_id = r.plant_id
    GROUP BY p.plant_id;
```

## 2. view\_user\_plant\_summary (用户植物概览视图)

支撑用户中心的数据展示需求:

```
01 CREATE VIEW view_user_plant_summary AS
02 SELECT
03     u.user_id,
04     u.username,
05     COUNT(p.plant_id) as total_plants,
06     COUNT(CASE WHEN p.status = 'healthy' THEN 1 END) as
07 healthy_plants,
08     COUNT(CASE WHEN r.is_completed = 0 THEN 1 END) as
09 pending_reminders,
10     MAX(cl.care_date) as last_care_date
11 FROM users u
12 LEFT JOIN plants p ON u.user_id = p.user_id
13 LEFT JOIN reminders r ON p.plant_id = r.plant_id
    LEFT JOIN care_logs cl ON p.plant_id = cl.plant_id
    GROUP BY u.user_id, u.username;
```

## 3. view\_sensor\_monitoring (传感器监控视图)

支撑未来 IoT 设备监控功能:

```
01 CREATE VIEW view_sensor_monitoring AS
02 SELECT
03     s.sensor_id,
04     s.device_name,
05     s.connection_status,
06     s.battery_level,
07     p.nickname as plant_nickname,
08     sd.soil_moisture,
09     sd.temperature,
10     sd.light_intensity,
```

```
11     sd.recorded_at
12 FROM sensors s
13 LEFT JOIN plants p ON s.plant_id = p.plant_id
14 LEFT JOIN sensor_data sd ON s.sensor_id = sd.sensor_id
15 WHERE sd.recorded_at = (SELECT MAX(recorded_at) FROM sensor_data
16 WHERE sensor_id = s.sensor_id);
```

## 五、存储过程、函数及触发器的设计

### （一）存储过程

#### 1. sp\_create\_reminder\_for\_plant（创建植物提醒）

支撑提醒规则引擎的自动化任务生成：

```
01 DELIMITER $$
02 CREATE PROCEDURE sp_create_reminder_for_plant(
03     IN p_plant_id BIGINT,
04     IN p_reminder_type VARCHAR(50)
05 )
06 BEGIN
07     INSERT INTO reminders (plant_id, user_id, reminder_type,
08 scheduled_at, is_completed)
09     SELECT
10         p_plant_id,
11         p.user_id,
12         p_reminder_type,
13         DATE_ADD(NOW(), INTERVAL pt.water_freq_days DAY),
14         0
15 FROM plants p
16 JOIN plant_types pt ON p.plant_type_id = pt.plant_type_id
17 WHERE p.plant_id = p_plant_id;
18 END$$
DELIMITER ;
```

#### 2. sp\_generate\_personalized\_reminder（生成个性化提醒）

支撑 AI 个性化呈现的拟人化内容生成：

```
01 DELIMITER $$
02 CREATE PROCEDURE sp_generate_personalized_reminder(
03     IN p_plant_id BIGINT,
04     IN p_reminder_type VARCHAR(50)
05 )
06 BEGIN
07     DECLARE v_plant_nickname VARCHAR(100);
08     DECLARE v_user_id BIGINT;
```

```
09 DECLARE v_reminder_content TEXT;
10 SELECT nickname, user_id INTO v_plant_nickname, v_user_id
11 FROM plants WHERE plant_id = p_plant_id;
12 (生成拟人化提醒内容, 支撑 AI 个性化功能)
13 CASE p_reminder_type
14     WHEN 'watering' THEN
15         SET v_reminder_content = CONCAT('嗨主人, 我是',
16 v_plant_nickname, ' ! 我嗓子快冒烟啦, 求喂水 ! ');
17     WHEN 'temperature' THEN
18         SET v_reminder_content = CONCAT('救命啊 ! 我是',
19 v_plant_nickname, ', 外面好冷啊, 快把我搬进屋里吧 ! ');
20     ELSE
21         SET v_reminder_content = CONCAT('主人, 我是',
22 v_plant_nickname, ', 我需要你的照顾哦 ! ');
23 END CASE;
24 INSERT INTO reminders (user_id, plant_id, reminder_type,
reminder_content, scheduled_at, is_completed)
VALUES (v_user_id, p_plant_id, p_reminder_type,
v_reminder_content, NOW(), 0);
END$$
DELIMITER ;
```

## (二) 触发器

### 1. tri\_after\_care\_log\_insert (养护日志插入触发器)

支撑种植日记的自动数据关联:

```
01 DELIMITER $$
02 CREATE TRIGGER tri_after_care_log_insert
03 AFTER INSERT ON care_logs
04 FOR EACH ROW
05 BEGIN
06     UPDATE plants SET updated_at = NOW() WHERE plant_id =
07 NEW.plant_id;
08
09     INSERT INTO operation_logs (user_id, operation_type,
10 operation_detail)
11     SELECT
12         p.user_id,
13         'care_log',
14         CONCAT('Added care log for plant: ', p.nickname)
15     FROM plants p WHERE p.plant_id = NEW.plant_id;
16 END$$
17 DELIMITER ;
```

## 2. tri\_after\_sensor\_data\_insert (传感器数据插入触发器)

支撑未来 IoT 设备集成的智能提醒：

```
01 DELIMITER $$
02 CREATE TRIGGER tri_after_sensor_data_insert
03 AFTER INSERT ON sensor_data
04 FOR EACH ROW
05 BEGIN
06     DECLARE v_plant_id BIGINT;
07     SELECT plant_id INTO v_plant_id
08     FROM sensors WHERE sensor_id = NEW.sensor_id;
09
10     IF NEW.soil_moisture < 20 THEN
11         CALL sp_generate_personalized_reminder(v_plant_id,
12 'watering');
13     END IF;
14     IF NEW.temperature < 5 THEN
15         CALL sp_generate_personalized_reminder(v_plant_id,
16 'temperature');
17     END IF;
18 END$$
19 DELIMITER ;
```

## 六、数据库标准

### (一) 表和视图命名规范

表名：小写 + 下划线，名词复数形式，如 care\_logs，与系统架构的数据层对应

视图：view\_前缀 + 功能描述，如 view\_plant\_summary，支撑业务查询

存储过程：sp\_ 前缀 + 功能描述，如 sp\_create\_reminder，实现业务逻辑

触发器：tri\_ 前缀 + 时机 + 表名，如 tri\_after\_care\_log\_insert，保障数据一致性

索引：idx\_ 前缀（普通），uk\_ 前缀（唯一），fk\_ 前缀（外键），优化查询性能

### (二) 字段命名规范

主键：表名\_id，如 plant\_id，保持一致性

外键：与引用表主键同名，维护关系完整性

时间字段：created\_at, updated\_at, deleted\_at，支持审计和同步

布尔字段：is\_前缀，如 is\_deleted，提高可读性

状态字段：status，使用枚举类型，规范状态管理

数量字段：\_count 后缀，如 view\_count，明确字段含义

### (三) 表结构设计

每表必须包含自增主键，确保唯一标识；所有表使用 utf8mb4 字符集，支持多语言；时间字

段统一使用 `datetime` 类型，包含时区信息；金额、温度等精确数值使用 `decimal` 类型，保证计算精度；文本字段根据实际需求设置合理长度，平衡存储和性能。

（四）表字段设计

- ID 字段：`bigint` 类型，自增主键，支持大规模数据
- 名称字段：`varchar` 类型，长度 50-100，适应业务需求
- 描述字段：`text` 类型或 `varchar(255)`，支持详细说明
- 状态字段：`varchar(20)` 或 `enum` 类型，规范状态流转
- 时间字段：`datetime` 类型，包含时区信息，支持全球化
- 数值字段：根据精度需求选择 `int` 或 `decimal`，保证数据准确性

七、数据元精度标准

数据项	类型	精度	范围	说明	对应系统需求
温度	<code>decimal(5, 2)</code>	0.01° C	-99.99~999.99	环境温度	提醒规则引擎
湿度	<code>decimal(5, 2)</code>	0.01%	0.00-100.00	环境湿度	天气服务集成
土壤湿度	<code>decimal(5, 2)</code>	0.01%	0.00-100.00	土壤含水量	IoT 设备数据
光照强度	<code>decimal(8, 2)</code>	0.01 lux	0-999999.99	光照度	环境因素监测
经纬度	<code>decimal(10, 7)</code>	0.0001°	-180~180	地理位置	天气服务定位
设备电量	<code>int</code>	1%	0-100	电池电量	设备状态监控
图片大小	<code>int</code>	1 字节	0-10MB	图片文件大小	图文记录
时间精度	<code>datetime</code>	1 秒	100-999 年	时间记录	全系统时间标准

八、数据库账号及权限说明

账号角色	权限范围	连接限制	说明	对应系统架构
<code>app_user</code>	<code>SELECT, INSERT, UPDATE</code>	最大 100 连接	应用运行时账号	服务层访问
<code>admin_user</code>	<code>ALL PRIVILEGES</code>	最大 10 连接	数据库管理员	系统运维
<code>read_only</code>	<code>SELECT</code>	最大 50 连接	报表分析账号	数据分析
<code>backup_user</code>	<code>SELECT, LOCK TABLES</code>	最大 5 连接	备份专用账号	数据备份

- 权限细化：
- `app_user`：业务表 `CRUD` 权限，无 `DDL` 权限，保障生产安全
  - `read_only`：只读权限，用于数据分析和报表，支持业务决策
  - `backup_user`：查询和锁表权限，用于数据备份，支持容灾恢复

- 安全要求：
- 1. 密码保护  
所有账号密码长度至少 8 位，包含字母和数字；
  - 2. 安全监控  
定期检查数据库日志，了解系统运行情况；

### 3. 安全防护

删除重要数据前提示用户确认。