CA388 蓝洋智渔 BlueOcean AquaSmart 项目设计方案--灌槽养鱼管理系统（罗非）

CA388 BlueOcean AquaSmart Project - Raceway Tilapia Farming Management System

项目编号：CA388

目录

蓝洋智渔 BlueOcean AquaSmart 项目设计方案--灌槽养鱼管理系统（罗非）

一、系统总体功能概述

1.1. 用户与权限管理：

不同角色（管理员、技术员、饲养员、访客等）拥有不同的权限。

采用角色访问控制（RoleBased Access Control, RBAC），保证数据与操作安全。

1.2. 养殖环境管理：

灌槽（Raceway）区域设置与管理。

水质监控和预警功能，实时采集与分析水质数据。

1.3. 饲料与投喂管理：

不同罗非鱼群的饲喂计划制定与执行。

饲料类型、库存、用量及成本统计。

1.4. 鱼群管理：

鱼苗投放、日常巡检、成长记录与批次管理。

体重采集、成活率记录、饲料转化率统计等。

1.5. 健康与病害防治：

疾病诊断、治疗方案、预防措施记录。

日常巡查表、异常警报与处理流程。

1.6. 库存与销售管理：

鱼群出塘、销售记录、财务与报表功能。

供应链管理（饲料、药品等）及库存监控。

1.7. 数据统计与报表：

养殖过程关键指标可视化（如水质指标、成活率、饲料使用量等）。

导出报表、数据分析、成本利润估算。

1.8. 系统设置与运维：

日志管理、系统监控、参数配置。

备份恢复与系统安全策略。

二、主要模块设计

以下按照功能区分不同模块，并对各功能点进行详细描述。

2.1 用户与权限管理模块

2.1.1. 用户角色定义

管理员（Admin）：拥有最高权限，可管理系统所有功能与设置。

技术员（Technician）：负责水质监控、设备管理、数据分析。

饲养员（Operator）：负责饲料投放、日常巡检、登记鱼群信息。

访客（Guest）：只可浏览部分公开信息，无修改权限。

2.1.2. 用户管理

注册/添加用户：由管理员创建新用户并分配角色。

用户信息维护：可编辑用户信息（用户名、密码、联系方式等）。

密码重置：管理员或用户本人才可重置。

2.1.3. 权限控制

模块级权限：不同角色只能访问对应的功能模块。

操作级权限：如新增、编辑、删除、导出等。

2.1.4. 日志记录

用户操作日志：记录用户的增删改查操作，便于审计。

登录日志：跟踪用户登录时间、IP 地址等安全信息。

2.2. 灌槽（Raceway）环境管理模块

2.2.1. 灌槽/塘口设置

灌槽基本信息：名称、位置、尺寸、容积等。

环境参数：包括水温、水流速、增氧设备配置等。

维护计划：定期清理、养殖季节更换等相关信息记录。

2.2.2. 水质监控

数据采集：与传感器或手动输入相结合，采集 pH、溶解氧、氨氮、亚硝酸盐、温度等关键指标。

实时监控与预警：若指标超限，系统自动发出预警通知（短信或站内消息）。

数据可视化：用图表、仪表盘形式直观展示水质变化趋势。

2.2.3. 设备管理

增氧机、水泵、监测传感器等设备的登记与维护。

设备状态：在线/离线标记、定期检修提醒、故障报警。

使用记录：设备启停时间、能耗统计、维修记录等。

2.3. 鱼群管理模块

2.3.1. 鱼苗批次管理

投放批次记录：鱼苗数量、来源、投放日期、规格大小等。

分批管理：不同投放时间或不同年龄段的鱼群分开管理、统计。

2.3.2. 日常巡检与生长记录

巡检日志：日常检查水质、鱼群活力、投喂情况、疾病预兆等。

生长数据：定期测量鱼体重、规格，记录成活率。

饲料转化率（FCR）：帮助评估饲料使用效率。

2.3.3. 异常与警报

自动检测：当某个灌槽鱼群死亡率突然升高时触发预警。

手动上报：饲养员发现异常情况可提交工单或告警信息。

处置流程：针对警报事件的操作记录，形成闭环管理。

2.4. 饲料与投喂管理模块

2.4.1. 饲料档案

饲料类型：根据蛋白含量、营养配方等记录不同饲料品种。

供应商信息：可关联采购模块，记录饲料来源、到货批次。

2.4.2. 投喂计划

投喂规则：根据鱼群的生长阶段和数量，设置投喂频次、定时投喂量。

自动/手动投喂：若有自动投喂设备，可联动控制；否则手动记录投喂量。

投喂提醒：在指定时间段或条件下，系统提醒饲养员操作。

2.4.3. 饲料库存与成本

库存管理：入库/出库记录，实时库存量查询，预警库存不足。

成本核算：根据饲料购买价格和使用量，估算养殖成本。

2.5. 健康与病害防治模块

2.5.1. 疾病诊断与治疗

病症分类：常见罗非鱼病害的诊断要点、图片资料、症状介绍。

用药方案：建议使用药物、剂量及注意事项；与投喂管理联动（如拌药饲料）。

2.5.2. 预防与记录

疫苗/预防投药记录：针对常见病害的定期预防措施。

诊断记录：发病时间、主要症状、处置方案、结果跟踪。

2.5.3. 健康巡查

巡查表：每日或每周巡查要点，例如观察鱼体表、食欲、活力等。

异常上报：若发现病症，及时上报并关联到具体鱼群。

2.6. 库存与销售管理模块

2.6.1. 出塘与存塘管理

出塘记录：出塘时间、数量、规格、去向（销售或加工）。

存塘统计：实时查看各灌槽当前的鱼群数量与平均规格。

2.6.2. 销售管理

订单管理：客户信息、订单数量、价格、出货时间、运输方式。

发票与结算：生成销售凭证、统计销售收入。

2.6.3. 仓库（饲料/药品）管理

库存档案：饲料、药品、工具等所有耗材的库存概况。

采购/入库：与饲料管理或病害防治用药相结合，记录入库批次和数量。

盘点与预警：定期或实时盘点库存，库存不足/过期预警。

2.7. 数据分析与报表模块

2.7.1. 统计分析

养殖指标：如成活率、平均体重增量、饲料转化率等。

成本与收益：饲料成本、药品成本、养殖周期收益估算。

销售趋势：查看不同时间段的销售量、收益、平均售价。

2.7.2. 可视化报表

仪表盘式总览：当日水质、鱼群健康状态、投喂情况等重要数据。

分模块图表：折线图、柱状图、饼图等展示不同维度的数据。

自定义报表：管理员可自定义条件筛选并导出 Excel/PDF。

2.7.3. 报表导出与分享

导出：支持 Excel、CSV、PDF 等格式的下载或打印。

分享：可针对特定角色或外部合作伙伴生成查看链接或发送邮件。

2.8. 系统设置与运维模块

2.8.1. 系统参数配置

水质指标阈值：可灵活设置 pH、溶解氧、温度等报警阈值。

通知方式：配置短信、邮件或系统消息推送。

界面语言、时区：支持多语言、不同时区设置。

2.8.2. 日志与审计

操作日志：记录各类系统操作，为系统审计提供依据。

错误日志：捕捉异常与错误信息，便于排查问题。

2.8.3. 备份与恢复

自动备份：设置每日/每周/每月定时数据库备份策略。

手动备份：管理员可随时发起备份。

恢复：支持基于备份文件的一键数据恢复。

2.8.4. 安全管理

数据加密：重要数据（如密码）采用哈希或加密存储。

访问控制：配合用户权限，设置白名单 IP、限制高危操作等。

防护措施：基础防护（SQL 注入、XSS）和必要的安全加固。

三、系统架构与技术要点（简述）

3.1. 前端

响应式设计，方便在桌面端和移动端访问。

主要功能通过 Web 界面实现，可使用如 Vue.js、Bootstrap 等前端框架。

3.2. 后端（PHP 7.1）

基于主流 MVC 开发框架（如 Laravel、CodeIgniter、Symfony 等）或自建框架。

处理业务逻辑、权限控制、数据检索、报告生成等。

3.3. 数据库（MySQL 8）

设计合理的数据表结构，重点关注：用户表、角色权限表、灌槽表、鱼群表、饲料表、库存表、水质监测表、日志表等。

使用触发器或存储过程自动统计部分关键数据，或做数据校验。

注意大规模数据的存储与性能优化。

3.4. 数据采集与集成

对接水质传感器设备（若有）：可通过 MQTT、HTTP API 或串口等方式与后端进行通信。

提供或接受第三方接口，对外交换数据（如销售平台或政府监管平台）。

在「蓝洋智渔项目设计方案——灌槽养鱼管理系统（罗非）」中，各个功能模块相互配合，实现了从水质监控、饲料投喂、鱼群健康管理，到库存销售、数据统计等一体化的养殖全流程管理。通过精细化的设计与权限控制，可以提高罗非鱼灌槽养殖效率，降低风险，提升产量和品质。

具体实现时，可根据实际养殖规模、设备投入程度、人员配备以及预算等情况，对功能进行裁剪或扩展，以实现最适合自身生产流程的数字化管理系统。

四、PHP 语言程序设计文件架构

下面给出一个示例性的文件/目录结构，按照每个功能模块在独立文件夹中进行划分，并列出了常用的 PHP 文件。此结构可用于基于 PHP7.1 + MySQL 的 MVC 或类似分层模式。根据实际需求，你可以灵活增删文件或文件夹。

4.1、项目总体结构示例

```
CA388_BlueOcean_AquaSmart/
├── public/
│   ├── index.php
│   └── .htaccess                    // 如果需要 Apache 重写规则
├── config/
│   ├── config.php                   // 系统主配置文件（数据库配置等）
│   ├── database.php                 // 数据库连接逻辑
│   └── routes.php                   // URL 路由配置
├── core/
│   ├── BaseController.php           // 控制器父类，公共方法
│   ├── BaseModel.php                // 模型父类，封装数据库操作
│   ├── SessionManager.php           // 会话管理，可选
│   └── Helpers.php                  // 通用函数
├── modules/
│   ├── UserAuth/                    // 用户与权限管理模块
│   │   ├── UserController.php
│   │   ├── RoleController.php
│   │   ├── UserModel.php
│   │   ├── RoleModel.php
│   │   └── views/                   // 该模块对应的视图文件夹（MVC模式下）
│   ├── RacewayManagement/           // 灌槽环境管理模块
│   │   ├── RacewayController.php
│   │   ├── WaterQualityController.php
│   │   ├── EquipmentController.php
│   │   ├── RacewayModel.php
│   │   ├── WaterQualityModel.php
```

```
│   │   ├── EquipmentModel.php
│   │   └── views/
│   ├── FishManagement/              // 鱼群管理模块
│   │   ├── FishBatchController.php
│   │   ├── GrowthController.php
│   │   ├── FishBatchModel.php
│   │   ├── GrowthModel.php
│   │   └── views/
│   ├── FeedManagement/              // 饲料与投喂管理模块
│   │   ├── FeedController.php
│   │   ├── FeedingPlanController.php
│   │   ├── FeedModel.php
│   │   ├── FeedingPlanModel.php
│   │   └── views/
│   ├── HealthDisease/               // 健康与病害防治模块
│   │   ├── HealthController.php
│   │   ├── DiseaseController.php
│   │   ├── HealthModel.php
│   │   ├── DiseaseModel.php
│   │   └── views/
│   ├── InventorySales/              // 库存与销售管理模块
│   │   ├── InventoryController.php
│   │   ├── SalesController.php
│   │   ├── InventoryModel.php
│   │   ├── SalesModel.php
│   │   └── views/
│   ├── AnalysisReports/             // 数据分析与报表模块
│   │   ├── AnalysisController.php
│   │   ├── ReportController.php
```

```
│   │   ├── AnalysisModel.php
│   │   ├── ReportModel.php
│   │   └── views/
│   └── SystemSettings/              // 系统设置与运维模块
│       ├── SettingsController.php
│       ├── BackupController.php
│       ├── LogController.php
│       ├── SettingsModel.php
│       ├── BackupModel.php
│       ├── LogModel.php
│       └── views/
├── vendor/                          // 如果使用 Composer 管理第三方库
│   └── autoload.php
└── composer.json                    // 若使用 Composer，可选
```

> 说明：

> 以上仅为参考示例，不同团队或框架的目录结构可能略有差异。

> 若使用 Laravel、Symfony 等框架，它们自带约定的目录结构，可以在此基础上进行模块划分。

4.2、各模块文件说明

以下针对每个功能模块，列出核心文件示例及其作用；视图（views）可根据实际页面需求再细分（如 `indexView.php`, `editView.php` 等）。

4.2.1. UserAuth（用户与权限管理模块）

UserController.php

处理用户增删改查、注册登录、密码重置等操作。

RoleController.php

处理角色管理及权限分配逻辑（RBAC）。

UserModel.php

用户数据操作，如增删改查用户信息、验证密码等。

RoleModel.php

角色与权限的数据库操作。

4.2.2. RacewayManagement（灌槽环境管理模块）

RacewayController.php

灌槽（Raceway）的基本信息管理，例如新建、编辑、维护计划等。

WaterQualityController.php

处理水质监控和数据采集业务逻辑，触发预警或通知。

EquipmentController.php

设备管理（增氧机、水泵、传感器等）的登记、维修、状态监控等。

RacewayModel.php

灌槽数据表对应的增删改查。

WaterQualityModel.php

水质采集记录表、预警参数等数据库操作。

EquipmentModel.php

设备表的数据库操作，如状态更新、维修记录等。

4.2.3. FishManagement（鱼群管理模块）

FishBatchController.php

鱼苗投放、批次管理；鱼群日常巡检及分批统计等。

GrowthController.php

管理鱼群生长数据（如体重、成活率、FCR 计算）与巡检日志。

FishBatchModel.php

鱼群投放批次、数量、来源等数据库操作。

GrowthModel.php

生长记录、巡检记录、成活率等统计数据操作。

4.2.4. FeedManagement（饲料与投喂管理模块）

FeedController.php

饲料类型、供应商、库存管理的业务逻辑。

FeedingPlanController.php

生成和管理投喂计划（定时投喂、投喂量）、自动或手动投喂记录。

FeedModel.php

饲料数据表、库存表的增删改查。

FeedingPlanModel.php

投喂记录、投喂计划等数据库操作。

4.2.5. HealthDisease（健康与病害防治模块）

HealthController.php

处理鱼群健康巡查、异常上报、预防计划等。

DiseaseController.php

处理常见病害诊断与治疗方案、用药记录的业务逻辑。

HealthModel.php

健康巡查记录、健康档案等数据库操作。

DiseaseModel.php

病害数据表、用药记录、治疗日志等数据库操作。

4.2.6. InventorySales（库存与销售管理模块）

InventoryController.php

管理饲料、药品、工具等综合库存的进销存；盘点与预警。

SalesController.php

鱼群出塘、销售订单、发票与结算等业务逻辑。

InventoryModel.php

库存数据表的增删改查。

SalesModel.php

销售订单、客户信息、财务记录的数据库操作。

4.2.7. AnalysisReports（数据分析与报表模块）

AnalysisController.php

养殖关键指标的统计与分析（如成活率、饲料转化率、成本收益等）。

ReportController.php

生成各类报表（日报、周报、月报、可视化图表）；导出功能 (Excel、PDF)。

AnalysisModel.php

数据分析计算所需的查询与汇总。

ReportModel.php

报表模板、生成记录、导出记录的数据库操作。

4.2.8. SystemSettings（系统设置与运维模块）

SettingsController.php

水质指标阈值、通知方式、多语言等系统全局配置管理。

BackupController.php

数据库备份与恢复功能、手动/自动备份逻辑。

LogController.php

系统日志查看、操作审计、错误日志处理。

SettingsModel.php

读写系统配置表（如全局阈值、短信/Email 配置等）。

BackupModel.php

备份记录、备份文件存储与恢复流程数据库操作。

LogModel.php

日志数据表的增删改查。

4.4. 核心文件/文件夹

4.4.1. public/index.php

系统入口文件，初始化加载配置、路由或自动加载类库 (Composer)。

4.4.2. config/config.php

全局配置文件，定义数据库连接信息、常量、错误级别等。

4.4.3. core/BaseModel.php

模型基类，封装数据库连接、通用的增删改查方法。

4.4.4. core/BaseController.php

控制器基类，提供通用的请求处理、视图渲染、权限验证等。

4.4.5. core/Helpers.php

通用工具函数，如格式化日期、生成唯一 ID、发送邮件/短信等辅助方法。

4.4.6. config/database.php

专门用于数据库连接初始化，也可结合 PDO 进行封装。

4.4.7. config/routes.php

如果使用路由机制进行 URL 到控制器方法的映射，则在此定义路由规则。

4.4.8. composer.json / vendor/

如果使用 Composer 安装第三方库（如 PHPMailer、Guzzle 等），则此处存放相关文件。

备注与建议

若使用 MVC 框架（如 Laravel、Symfony、CodeIgniter 等），可根据框架自带的目录结构进行“模块化”划分。上面只是一个方便理解与管理的通用文件夹方案。

视图（`views/`）部分可根据前端技术或模板引擎（Blade、Twig 等）进一步细分。

可在 `core/` 或 `libs/` 中放置常用的第三方库、工具类、单例模式的数据库封装等。

在实际开发中，可以将“Controller”和“Model”命名更具象一些（如 `UserController.php`、`UserService.php`、`UserRepository.php` 等），以反映系统的分层结构。

通过以上文件及目录规划，每个模块都在各自文件夹中保持相对独立，方便后期的维护和扩展，也能让团队成员快速了解系统功能的归属和代码位置。你可以根据实际开发和项目规模，对此结构进行进一步的裁剪或完善。

五、数据库设计--MYSQL8

以下给出一个示例性的 MySQL 8 数据库设计方案，涵盖「CA388 蓝洋智渔--灌槽养鱼管理系统（罗非）」的大部分核心功能。为了可读性和扩展性，采用相对清晰的表命名，并给出字段、类型、作用及主外键说明。实际实施时，可根据业务细节进行优化或拆分合并。

5.1、数据库概览

从功能角度出发，系统主要实体包括：

1. 用户与权限（用户、角色）
2. 灌槽（Raceway）管理（灌槽、设备、水质监测）
3. 鱼群管理（鱼苗批次、生长记录）
4. 饲料管理（饲料类型、投喂计划、投喂记录）
5. 病害与健康管理（疾病、健康记录）
6. 库存与销售（统一库存或分饲料库存/药品库存、销售订单）
7. 数据分析与报表（可根据需求汇总相关表中数据，无需独立建表，或视需求建立统计表）
8. 系统与日志（系统设置、日志）

以下设计示例中，部分冗余或组合字段可以根据需要进行进一步拆分或合并。

5.2、表结构设计

5.2.1. 用户与权限相关

5.2.1.1. `roles` (角色表)

1.1. `roles` (角色表)

字段	类型	说明
`role_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	角色ID（主键）
`role_name`	VARCHAR(50)	角色名称（如Admin、Technician）
`permissions`	TEXT （或 JSON）	权限集合，存储RBAC相关配置（可选）
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

- 若权限设计复杂，可另外建映射表（如 `role_permissions`），这里仅作示例。

5.2.1.2. `users` (用户表)

1.2. `users` (用户表)

字段	类型	说明
`user_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	用户ID（主键）
`username`	VARCHAR(50) UNIQUE	登录用户名
`password_hash`	VARCHAR(255)	存储加密后的密码哈希
`email`	VARCHAR(100)	邮箱
`phone`	VARCHAR(20)	电话（可选）
`role_id`	INT UNSIGNED	外键，关联 `roles.role_id`
`status`	TINYINT	用户状态（0=禁用,1=启用）
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (role_id) REFERENCES roles(role_id)
    ON DELETE SET NULL
    ON UPDATE CASCADE
```

#5.2.2. 灌槽（Raceway）管理

5.2.2.1. `raceways` (灌槽表)

2.1. `raceways` (灌槽表)

字段	类型	说明
`raceway_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	灌槽ID（主键）
`name`	VARCHAR(100)	灌槽名称
`location`	VARCHAR(100)	位置描述（可选）
`dimension`	VARCHAR(50)	尺寸或形状描述（如长、宽、深）
`capacity`	DECIMAL(10,2)	容量（单位可设定，如立方米）
`water_flow`	DECIMAL(10,2)	水流速（可选字段）
`status`	TINYINT	状态（0=停用,1=使用中等）
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

5.2.2.2. `equipment` (设备表)

2.2. `equipment` (设备表)

字段	类型	说明
`equip_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	设备ID (主键)
`raceway_id`	INT UNSIGNED	外键，对应 `raceways.raceway_id`
`equip_name`	VARCHAR(100)	设备名称（如增氧机、水泵、传感器等）
`equip_type`	VARCHAR(50)	设备类型（可自行定义类别）
`status`	TINYINT	状态（0=故障,1=正常,2=维修中等）
`last_maintenance`	DATETIME	上次维护日期
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (raceway_id) REFERENCES raceways(raceway_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
```

5.2.2.3. `water_quality_records` (水质监测记录表)

2.3. `water_quality_records` (水质监测记录表)

字段	类型	说明
`record_id`	BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY	记录ID (主键)
`raceway_id`	INT UNSIGNED	外键, 对应 `raceways.raceway_id`
`ph`	DECIMAL(3,2)	PH 值
`dissolved_oxygen`	DECIMAL(5,2)	溶解氧 (mg/L)
`ammonia`	DECIMAL(5,2)	氨氮 (mg/L)
`nitrite`	DECIMAL(5,2)	亚硝酸盐 (mg/L)
`temperature`	DECIMAL(5,2)	水温 (℃)
`recorded_at`	DATETIME	监测数据实际采集时间
`created_by`	INT UNSIGNED	记录采集人 (外键, 对应 `users.user_id`)
`note`	VARCHAR(255)	备注 (可选)
`created_at`	DATETIME	创建时间

外键：

```
FOREIGN KEY (raceway_id) REFERENCES raceways(raceway_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
FOREIGN KEY (created_by) REFERENCES users(user_id)
    ON DELETE SET NULL
    ON UPDATE CASCADE
```

#5.2.2.4. 鱼群管理

3.1. `fish_batches` (鱼苗/鱼群批次表)

字段	类型	说明
`batch_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	批次ID (主键)
`raceway_id`	INT UNSIGNED	外键, 对应 `raceways.raceway_id`
`source`	VARCHAR(100)	鱼苗来源 (如苗场名称、供应商等)
`initial_count`	INT UNSIGNED	投放鱼苗数量
`stock_date`	DATE	投放日期
`average_size`	DECIMAL(5,2)	投放时的平均规格 (可记录体长或体重,单位可自定义)
`remarks`	VARCHAR(255)	备注 (可选)
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (raceway_id) REFERENCES raceways(raceway_id)
    ON DELETE CASCADE
    ON UPDATE CASCADE
```

5.2.2.5. `growth_records` (鱼群生长与巡检记录表)

3.2. `growth_records` (鱼群生长与巡检记录表)

字段	类型	说明
`growth_id`	BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY	生长记录ID (主键)
`batch_id`	INT UNSIGNED	外键, 对应 `fish_batches.batch_id`
`measurement_date`	DATE	测量/记录日期
`average_weight`	DECIMAL(6,3)	当日平均体重 (单位g或kg, 可自行定义)
`mortality`	INT UNSIGNED	当日死亡数量 (可用于计算成活率)
`feed_used`	DECIMAL(6,2)	当日饲料使用量 (可选, 若在此处统计)
`remarks`	VARCHAR(255)	备注 (如巡查情况, 异常情况等)
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (batch_id) REFERENCES fish_batches(batch_id)
    ON DELETE CASCADE
```

ON UPDATE CASCADE

```

---

# 5.2.3. 饲料与投喂管理

## 5.2.3.1. `feed_types` (饲料类型表)

4.1. `feed_types` (饲料类型表)

| 字段 | 类型 | 说明 |
|---|---|---|
| `feed_id` | INT UNSIGNED AUTO_INCREMENT PRIMARY KEY | 饲料ID (主键) |
| `feed_name` | VARCHAR(100) | 饲料名称 (如某品牌配合饲料) |
| `feed_type` | VARCHAR(50) | 饲料类别 (如浮性饲料、沉性饲料等) |
| `protein_level` | DECIMAL(5,2) | 蛋白含量 (%), 可选 |
| `created_at` | DATETIME | 创建时间 |
| `updated_at` | DATETIME | 更新时间 |

## 5.2.3.2. `feed_inventory` (饲料库存表)

4.2. `feed_inventory` (饲料库存表)

| 字段 | 类型 | 说明 |
|---|---|---|
| `inventory_id` | INT UNSIGNED AUTO_INCREMENT PRIMARY KEY | 库存记录ID (主键) |
| `feed_id` | INT UNSIGNED | 外键, 对应 `feed_types.feed_id` |
| `batch_number` | VARCHAR(50) | 饲料的批次号(可选) |
| `quantity` | DECIMAL(10,2) | 当前库存数量 (单位: kg或吨等) |
| `unit_price` | DECIMAL(10,2) | 单价 (可用于成本核算) |
| `last_update` | DATETIME | 上一次更新库存的时间 |
| `created_at` | DATETIME | 创建时间 |
| `updated_at` | DATETIME | 更新时间 |

外键：

```

FOREIGN KEY (feed_id) REFERENCES feed_types(feed_id)

ON DELETE CASCADE

ON UPDATE CASCADE

```

### 5.2.3.3. `feeding_plans` (投喂计划表)

4.3. `feeding_plans` (投喂计划表)

| 字段 | 类型 | 说明 |
|---|---|---|
| `plan_id` | INT UNSIGNED AUTO_INCREMENT PRIMARY KEY | 投喂计划ID (主键) |
| `batch_id` | INT UNSIGNED | 外键, 对应 `fish_batches.batch_id` |
| `feed_id` | INT UNSIGNED | 外键, 对应 `feed_types.feed_id` |
| `daily_amount` | DECIMAL(10,2) | 每日投喂总量 (kg等) |
| `frequency` | TINYINT | 每日投喂次数 |
| `start_date` | DATE | 计划开始日期 |
| `end_date` | DATE | 计划结束日期 (可选) |
| `notes` | VARCHAR(255) | 备注 |
| `created_at` | DATETIME | 创建时间 |
| `updated_at` | DATETIME | 更新时间 |

外键：

```

FOREIGN KEY (batch_id) REFERENCES fish_batches(batch_id),

FOREIGN KEY (feed_id) REFERENCES feed_types(feed_id)

```

### 5.2.3.4. `feeding_logs` (投喂记录表)

4.4. `feeding_logs` (投喂记录表)

| 字段 | 类型 | 说明 |
|---|---|---|
| `feeding_id` | BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY | 投喂记录ID (主键) |
| `plan_id` | INT UNSIGNED | 外键, 对应 `feeding_plans.plan_id` |
| `actual_amount` | DECIMAL(10,2) | 实际投喂量 (kg等) |
| `feeding_time` | DATETIME | 投喂时间 |
| `operator_id` | INT UNSIGNED | 记录人/投喂人员 (外键, 对应 `users.user_id`) |
| `remarks` | VARCHAR(255) | 备注 |
| `created_at` | DATETIME | 创建时间 |
```

外键：

```
FOREIGN KEY (plan_id) REFERENCES feeding_plans(plan_id)
    ON DELETE CASCADE,
FOREIGN KEY (operator_id) REFERENCES users(user_id)
    ON DELETE SET NULL
    ON UPDATE CASCADE
```

5.2.4. 健康与病害防治

5.2.4.1. `diseases` (疾病档案表)

5.1. `diseases` (疾病档案表)

字段	类型	说明
`disease_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	疾病ID (主键)
`name`	VARCHAR(100)	疾病名称
`description`	TEXT	疾病描述 (包括症状、传播途径等)
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

5.2.4.2. `health_records` (健康与病害记录表)

5.2. `health_records` (健康与病害记录表)

字段	类型	说明
`health_id`	BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY	健康记录ID (主键)
`batch_id`	INT UNSIGNED	外键，对应 `fish_batches.batch_id`
`disease_id`	INT UNSIGNED	外键，对应 `diseases.disease_id` (若为诊断出疾病)
`detection_date`	DATE	发现日期/巡检日期
`symptoms`	VARCHAR(255)	临床症状
`treatment`	VARCHAR(255)	治疗方案 (或外键关联到药品/投药记录)
`remarks`	VARCHAR(255)	备注
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (batch_id) REFERENCES fish_batches(batch_id)
    ON DELETE CASCADE,
FOREIGN KEY (disease_id) REFERENCES diseases(disease_id)
    ON DELETE SET NULL
    ON UPDATE CASCADE
```

5.2.5. 库存与销售管理

5.2.5.1. `inventory` (通用库存表，可选方案)

如果需要对除饲料外的其他物资（如药品、工具等）统一管理，可以使用一个通用的 `inventory` 表：

字段	类型	说明
`inventory_id`	INT UNSIGNED AUTO_INCREMENT PRIMARY KEY	库存ID (主键)
`item_name`	VARCHAR(100)	物品名称 (如药品名称)
`category`	VARCHAR(50)	类别 (如药品/工具/其他)
`quantity`	DECIMAL(10,2)	库存数量
`unit_price`	DECIMAL(10,2)	单价 (可选)
`last_update`	DATETIME	上一次更新的时间
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

5.2.5.2. `sales` (销售记录表)

6.2. `sales` (销售记录表)

字段	类型	说明
`sale_id`	BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY	销售ID (主键)
`batch_id`	INT UNSIGNED	外键，对应 `fish_batches.batch_id` (若单批出塘)
`sale_date`	DATE	出塘/销售日期
`quantity`	DECIMAL(10,2)	销售数量 (kg 或尾数)
`unit_price`	DECIMAL(10,2)	单价 (可选)
`total_price`	DECIMAL(10,2)	总价 (可选)
`customer_info`	VARCHAR(255)	客户名称或联系方式 (可选)
`remarks`	VARCHAR(255)	备注
`created_at`	DATETIME	创建时间
`updated_at`	DATETIME	更新时间

外键：

```
FOREIGN KEY (batch_id) REFERENCES fish_batches(batch_id)
    ON DELETE SET NULL
    ON UPDATE CASCADE
```

> 如果一次销售涉及多个批次，需增加 `sales` 与 `fish_batches` 之间的中间表（如 `sale_details`），此处给出最简化的设计。

5.2.6. 日志与系统设置

5.2.6.1. `system_settings`

用于存储全局系统配置，比如水质阈值、通知方式等。

7.1. `system_settings`
用于存储全局系统配置，比如水质阈值、通知方式等。

字段	类型	说明
`setting_key`	VARCHAR(100)	配置键（主键），如 `water_ph_min`, `water_ph_max` 等
`setting_value`	VARCHAR(255)	配置值，或使用 TEXT/JSON 存储更复杂的设置
`updated_at`	DATETIME	更新时间

> 或者设定自增主键，然后在表内用 `key` 和 `value` 字段即可。

5.2.6.2. `system_logs`

用于记录系统操作日志、错误日志等。

7.2. `system_logs`
用于记录系统操作日志、错误日志等。

字段	类型	说明
`log_id`	BIGINT UNSIGNED AUTO_INCREMENT PRIMARY KEY	日志ID (主键)
`user_id`	INT UNSIGNED	操作者的ID (可为空表示系统日志)
`action`	VARCHAR(100)	操作名称或类型 (如 CREATE, UPDATE, LOGIN等)
`description`	TEXT	具体描述，含被操作的记录ID、更改前后对比等
`ip_address`	VARCHAR(45)	记录操作IP (IPv4/IPv6)
`created_at`	DATETIME	日志产生时间

5.3. 字段与索引设计要点

5.3.1. **主键**：大部分表使用 `AUTO_INCREMENT` 整数作为主键，也可考虑使用 UUID。

5.3.2. 外键约束：

- 大多设置为 `ON DELETE CASCADE` 或 `SET NULL`，根据业务逻辑选择。
- 注意实际生产环境中，很多团队会在应用层处理外键关系，而在数据库层弱化外键以便更灵活地扩展（视团队规范而定）。

5.3.3. 索引：

- 常用的查询字段（如 `raceway_id`, `batch_id`, `created_at` 等）应建立索引或复合索引。
- 对于经常进行模糊搜索的字段（如 `name`、`location`），可视情况建普通索引。

5.3.4. 时间字段：

- 所有关键实体建议保留 `created_at`、`updated_at` 便于溯源与审计。
- 时间类型可选 `DATETIME` 或 `TIMESTAMP`，根据项目需求和时间范围决定。

5.3.5. 数据类型：

- 数值字段（如重量、数量、价格等）需合理设置精度；浮点运算可能有精度问题，可酌情使用 `DECIMAL`。

5.3.6. 字符编码：

- 建议全库使用 `utf8mb4` 或者 `utf8mb4_unicode_ci`，更好地兼容多语言及表情符号等。

5.4.扩展

- 上述设计涵盖了系统主要模块对应的数据表及字段结构；在实际开发中，可结合业务流程进一步优化。
- 针对 多对多 的业务场景（如一次销售涉及多个批次，用户可能有多个角色），需要引入中间表(Mapping Table)。
- 如果引入自动化监控(传感器数据频繁写入)，还需要考虑大规模水质记录的数据分表策略或 TSDB（时序数据库）方案，以保证效率。
- 可根据实际需求，为每个表设计触发器或存储过程来进行数据校验或自动统计，也可通过应用层进行逻辑处理。

以上就是基于「蓝洋智渔--灌槽养鱼管理系统（罗非）」的 MySQL 8 数据库设计示例，提供了核心表和字段，能够支撑主要功能模块的开发。实际项目中，可根据具体需求进一步细化或调整。祝你开发顺利！

叶梓阳 总经理

广东知周数字科技有限公司

官网 www.caffz.com

13826867328