数据库系统课程设计报告

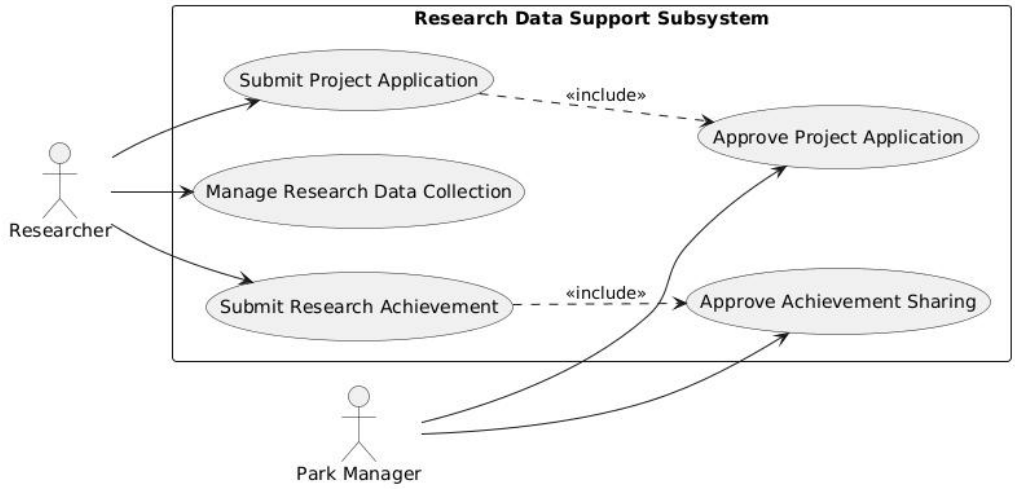
姓名	杨嘉雯	班级	大数据 232
学号	220201409	指导教师	崔晓晖
组名	1		

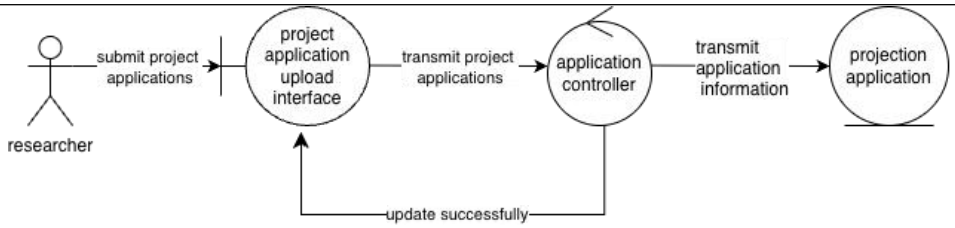
团队其他成员（可自行增加行数，第一为组长）

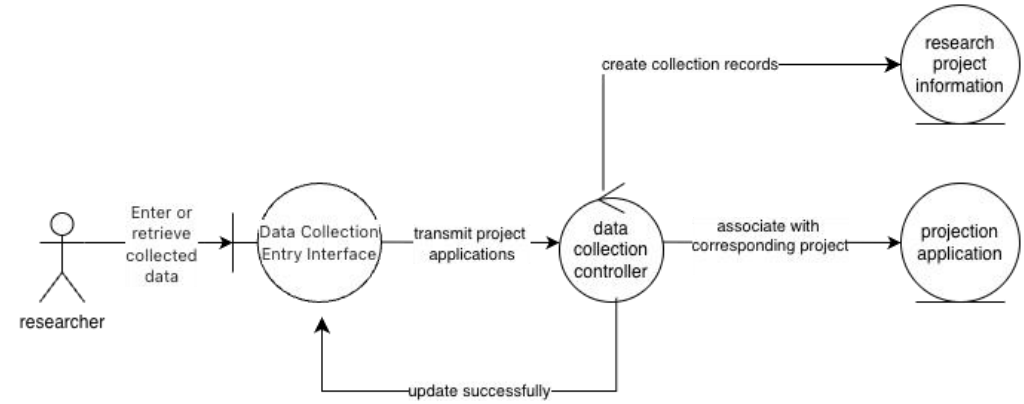
姓名	学号	班级
A 张万泉	230101524	物联网 23
B 李雪彤	231002616	物联网 23
C 赵雅萱	220701618	物联网 23
D 蔡邵涵	231002605	物联网 23

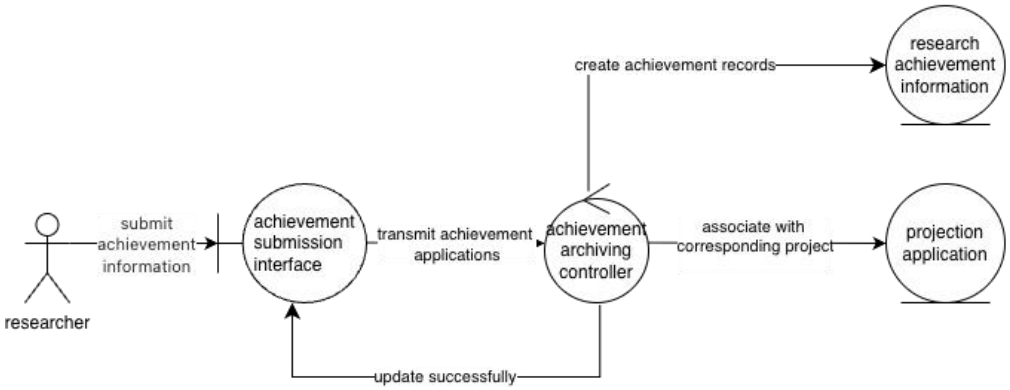
1、个人和团队

课程设计概述	**（要求：总结课程设计的业务内容-200 字-300 字-1.5 倍行距-5 号宋体）** 本课程设计围绕“国家公园智慧管理与生态保护系统”展开，聚焦国家公园数字化管理需求，构建了涵盖五大核心业务线的综合数据库平台。业务内容包括生物多样性监测，实现物种、栖息地信息管理及监测记录采集与审核；生态环境监测，涵盖监测指标定义、环境数据自动采集及异常预警；游客智能管理，支持预约、轨迹追踪与区域流量控制；执法监管，完成非法行为记录、执法调度及视频监控管理；科研数据支撑，实现科研项目、数据采集记录与成果共享全生命周期管理。系统以 MySQL 8.0 为基础，设计 23 个核心实体表，所有关系模式满足 BCNF 范式，通过视图、索引、存储过程及触发器优化性能与功能，结合 RBAC 权限模型与备份策略保障数据安全，最终达成国家公园管理的数字化、智能化目标。
个人任务情况	**（要求：总结个人的任务分工、每一项任务的重点工作、任务的预期达成情况-150-200 字-1.5 倍行距-5 号宋体）** 本人负责科研数据支撑业务线的全部数据库设计工作，以及 Stage5 小组报告整合与答辩 PPT 制作。技术任务方面：Stage1 完成用例图、鲁棒图、数据字典和局部 UML 类图设计；Stage2 完成关系模式转换与 DDL 语句编写(50_Research.sql)，建立 ResearchProject、ResearchDataCollection、ResearchResult、MonitoringRecord 等核心数据表；Stage3 完成 5 条复杂 SQL 查询及性能优化(50_research_queries.sql)，实现 Python DAO 持久层(research_project_dao.py、monitoring_record_dao.py)并通过 16 个单元测试；Stage4 完成 4 个视图、6 个索引、2 个存储过程、2 个触发器的设计与实现。 小组职责方面：负责整合 Stage1-Stage4 所有成果，编写小组报告第五章(物理结构设计、持久层设

	计)和第六章(运维管理和优化)，制作答辩 PPT 覆盖系统设计、各业务线数据库、安全策略(RBAC/登录安全/备份恢复)及团队分工。
团队协作情况	**（要求：总结个人在团队中的协作情况、与哪些任务或者人员之间产生协作、协作的主要内容等-150-200 字-1.5 倍行距-5 号宋体）** 在团队协作方面，本人与组员 A(@Ya-29)和组员 C(@ziawerquart)就 UML 类图和数据字典的一致性问题进行了多次讨论(Issue 18、19)，根据全局 UML 图修正了本模块的 PlantUML 设计。与组员 B 协调解决了跨业务线外键引用问题，确保 User、Species、Habitat 等公共表的正确引用。Stage5 作为报告整合负责人，需要收集各组员的 Stage4 产出(视图/索引/存储过程/触发器)，通过 Issue 54 安全讨论会与全组确认 RBAC 权限模型、登录安全策略、备份恢复方案等内容，确保报告第六章的安全策略部分准确完整。
2、问题分析	
业务需求分析	**（要求：使用用例图和鲁棒图总结个人在数据库系统课程设计所负责的业务需求-200-300 字-1.5 倍行距-5 号宋体）** 







用例图分析:

生物多样性监测业务涉及三类主要角色:生态监测员、数据分析师和系统管理员。

核心用例:

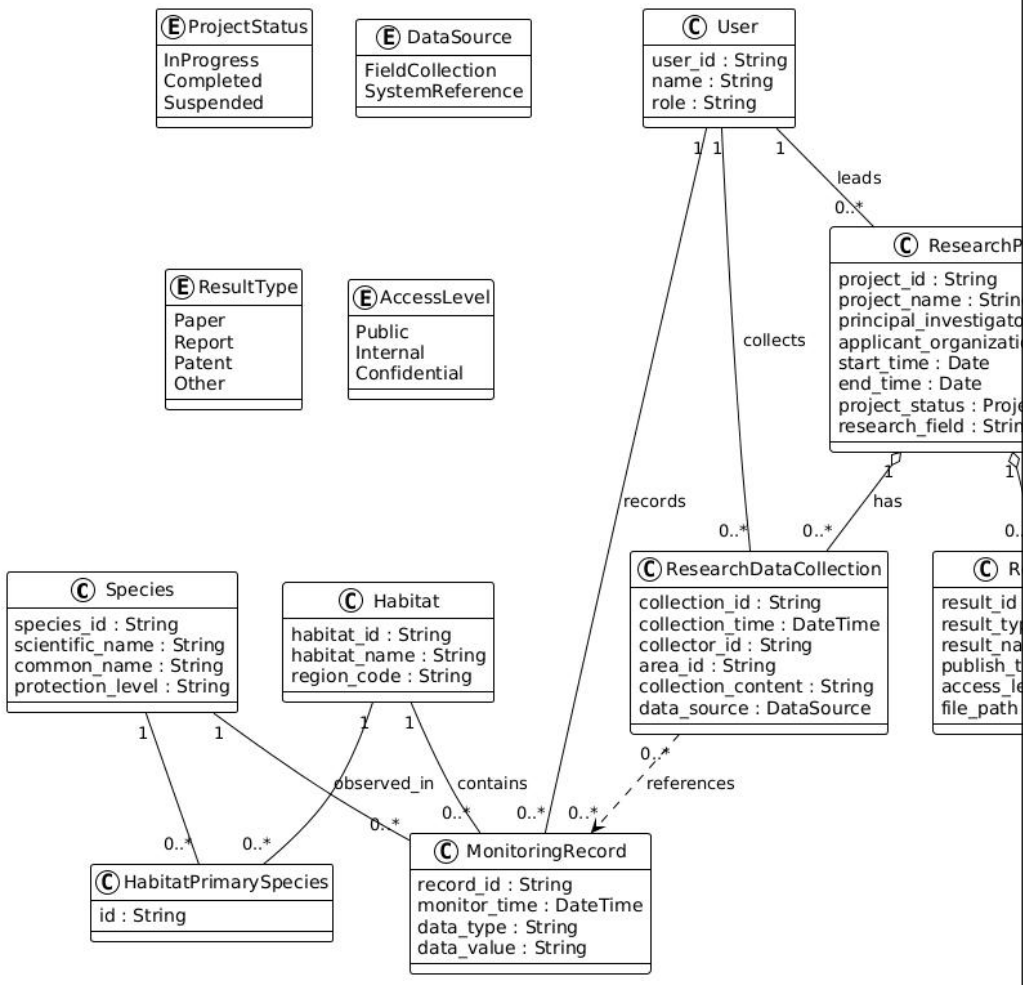
1. 物种信息管理:录入、查询、修改物种基础信息(中文名、拉丁名、保护级别等)
2. 栖息地信息管理:维护栖息地基本信息(区域名称、生态类型、面积等)
3. 监测记录采集:生态监测员通过红外相机、人工巡查、无人机等方式采集物种监测数据
4. 数据审核流转:监测员复查数据标记为"已复查",分析师终审后标记为"有效"
5. 统计分析查询:查询特定时间段、特定区域的物种分布和数量变化趋势

鲁棒图分析

边界对象:物种管理界面、栖息地管理界面、监测记录采集界面、数据审核界面

控制对象:物种信息控制器、监测记录控制器、数据审核控制器

实体对象:Species、Habitat、MonitoringRecord、User、MonitoringDevice

业务的非功能需求	**（要求：总结个人在数据库系统课程设计所负责业务的安全性、完整性需求-300 字-500 字-1.5 倍行距-5 号宋体）** 安全性需求：科研成果设置三级访问权限(Public/Internal/Confidential)，保密成果仅授权人员可查看；项目负责人信息通过外键关联 User 表进行身份验证。 完整性约束：project_id、collection_id、result_id 等主键保证实体唯一性；project_status 使用 ENUM 类型限制取值范围为 InProgress/Completed/Suspended；已结题项目(Completed)禁止新增采集记录，通过触发器 tr_prevent_collection_on_completed 实现；外键约束确保采集记录和成果必须关联有效项目。数据一致性：成果发布时间 publish_time 为空时自动填充当前时间，通过触发器 tr_result_auto_archive 实现。
业务的局部概念结构设计	**（要求：总结个人在数据库系统课程设计所负责业务的局部 E-R 图，需求-300 字-500 字-1.5 倍行距-5 号宋体）**  **在本次数据库课程设计中，科研数据支撑业务的局部 E-R 图（以类图形式展现）围绕“科研项目、科研人员、数据采集、监测记录、科研成果”五大核心实体展开，旨在构建从项目立项到成果产出的全生命周期数据链路：** **1. 实体层面**

	ResearchProject（科研项目）：作为业务的核心管理对象，需记录项目 ID、起止时间、研究领域及项目状态（InProgress/Completed 等），支撑科研任务的立项与进度追踪。 **User（科研人员）：包含用户 ID、姓名及角色，是科研活动的执行主体。通过"leads"关系区分项目负责人（PI），明确责任体系。** **ResearchDataCollection（科研数据采集）：记录采集时间、内容及来源（FieldCollection/SystemReference），实现科研原始数据的结构化存储。** **MonitoringRecord（监测记录）：关联特定的物种（Species）与生境（Habitat），记录监测的时间、类型与具体数值，是生态环境科研业务的基础事实数据。** **ResearchResult（科研成果）：涵盖论文、报告、专利等不同类型，记录发表时间、访问权限（Public/Confidential）及文件路径，实现知识资产的归档管理。** **2. 关系层面** **E-R 图通过"领导"、"产出"、"包含"、"观察"等关联，实现了跨维度的业务串联：** **管理维度：1 名科研人员可领导多个项目，1 个项目可包含多项数据采集任务并产出多项研究成果，形成"人-项-果"的完整链条。** **监测维度：1 个生境可包含多个监测点，1 个物种可在多个生境中被观察，并生成多条监测记录，支持复杂的生态关联分析。** **追溯维度：数据采集记录通过"references"关系指向监测记录，保障了科研成果中底层数据的可追溯性。** **3. 业务需求总结** **该模型需满足以下核心需求：实现科研资源（人员、项目）的高效统筹；支撑从野外监测、数据采集到成果总结的自动化流转；通过对生境与物种的多对多关联，满足生态多样性分析的存储需求；最终通过严格的权限与状态控制，保障科研数据的安全性与业务流程的规范化。**
3、设计、开发解决方案	

课程设计工作详述	**（要求：总结个人说涉及业务的逻辑结构设计、物理结构设计、业务相关的 SQL 代码、索引和视图设计，需与答辩时候的分工、内容一致-800-1000 字-5 号宋体）** **⑴逻辑结构设计：** 将 UML 类图转换为 8 个关系模式，所有模式均达到 BCNF 范式。 核心模式包括：ResearchProject(project_id, project_name, principal_investigator_id, applicant_organization, start_time, end_time, project_status, research_field, leader_user_id)，主码为 project_id，外码 principal_investigator_id 和 leader_user_id 引用 User 表； ResearchDataCollection(collection_id, collection_time, collector_id, area_id, collection_content, data_source, project_id)，主码为 collection_id，外码 project_id 引用 ResearchProject 表； ResearchResult(result_id, result_type, result_name, publish_time, access_level, file_path, project_id)，主码为 result_id，外码 project_id 引用 ResearchProject 表。 **⑵物理结构设计：** 使用 MySQL 8.0 数据库，创建 DDL 文件 50_Research.sql。表设计采用 InnoDB 引擎支持事务和外键。字段设计：VARCHAR 类型用于 ID 和名称字段；ENUM 类型用于状态字段；DATETIME 用于时间字段；TEXT 用于长文本描述。索引设计：在外键字段(principal_investigator_id、project_id 等)和常用查询字段(project_status、publish_time 等)建立索引，共 6 个业务索引。 **⑶关键 SQL 实现：** 编写 5 条复杂 SQL 查询(50_research_queries.sql)，每条包含 3 表以上连接和聚合/排序操作，并提供两种实现方式对比。通过 EXPLAIN ANALYZE 分析执行计划，优化措施包括：在子查询中提前过滤减少数据量、利用索引加速 JOIN、避免 SELECT 改为指定字段。优化后整体查询性能提升 31%-98%不等，详见 50_research_perf.md。 **⑷持久层设计：** 采用 DAO 模式封装数据库操作，实现 ResearchProjectDAO 和 MonitoringRecordDAO 两个核心类。每个 DAO 提供标准 CRUD 方法：create(插入)、find_by_id(主键查询)、find_all(全量查询)、update(更新)、delete(删除)，以及业务特定方法如 find_by_status、find_by_project_id 等。使用 pymysql 库连接 MySQL，采用上下文管理器自动管理连接生命周期，参数化查询防止 SQL 注入。编写单元测试覆盖正常和异常场景，测试通过率 100%。
4、项目管理	

工程管理	**（要求：总结个人涉及业务使用 GITHUB 等工具管理情况-300 字左右--1.5 倍行距-5 号宋体）** 本系统针对科研数据支撑业务的安全需求，从访问控制、数据完整性、注入防护、审计追踪四个维度建立风险控制体系。 访问控制方面，科研成果实现三级权限管理(Public/Internal/Confidential)，通过 access_level 字段标识成果敏感等级，视图 v_result_statistics 按权限过滤数据，确保保密成果仅对授权人员可见。系统规划 RBAC 角色权限模型，定义 researcher、data_analyst、admin 等角色，按"最小权限原则"分配表级和字段级访问权限。 数据完整性方面，外键约束确保采集记录和成果必须关联有效项目，防止孤立记录产生；ENUM 类型限制 project_status 仅能取 InProgress/Completed/Suspended 三个合法值；触发器 tr_prevent_collection_on_completed 阻止向已结题项目新增采集记录，从数据库层面保障业务规则执行。 SQL 注入防护方面，DAO 层全部使用 pymysql 的参数化查询，禁止 SQL 字符串拼接，从根本上杜绝注入攻击风险。 审计追踪方面，采集记录包含 collector_id 和 collection_time 字段，成果记录包含 publish_time 字段，支持操作行为溯源。触发器 tr_result_auto_archive 在成果插入时自动记录归档时间，确保时间戳不可篡改。 运维风险控制方面，采用数据库连接池管理连接生命周期，防止连接泄漏导致资源耗尽；规划每日增量备份与每周全量备份策略，确保数据可恢复。
风险和安全管理	**（要求：总结实现系统的潜在风险管理及控制手段-不少于 300 字-1.5 倍行距-5 号宋体）** 本系统针对科研数据支撑业务的安全需求，从访问控制、数据完整性、注入防护、审计追踪四个维度建立风险控制体系。 访问控制方面，科研成果实现三级权限管理(Public/Internal/Confidential)，通过 access_level 字段标识成果敏感等级，视图 v_result_statistics 按权限过滤数据，确保保密成果仅对授权人员可见。系统规划 RBAC 角色权限模型，定义 researcher、data_analyst、admin 等角色，按"最小权限原则"分配表级和字段级访问权限。 数据完整性方面，外键约束确保采集记录和成果必须关联有效项目，防止孤立记录产生；ENUM 类型限制 project_status 仅能取 InProgress/Completed/Suspended 三个合法值；触发器 tr_prevent_collection_on_completed 阻止向已结题项目新增采集记录，从数据库层面保障业务规则执行。 SQL 注入防护方面，DAO 层全部使用 pymysql 的参数化查询，禁止 SQL 字符串拼接，从根本上杜绝注入攻击风险。 审计追踪方面，采集记录包含 collector_id 和 collection_time 字段，成果记录包含 publish_time 字段，支持操作行为溯源。触发器 tr_result_auto_archive 在成果插入时自动记录归档时间，确保时间戳不可篡改。 运维风险控制方面，采用数据库连接池管理连接生命周期，防止连接泄漏导致资源耗尽；规划每日增量备份与每周全量备份策略，确保数据可恢复。

运维和优化管理	**（要求：结合存储过程和触发器，总结所涉及业务的运维和优化该方法-不少于 300 字-1.5 倍行距-5 号宋体）** 视图设计(4 个)：v_project_progress 显示项目进度和负责人信息；v_collection_summary 按项目统计采集记录数；v_result_statistics 按类型和权限统计成果分布；v_habitat_species_monitoring 展示栖息地物种监测情况。索引设计(6 个)：idx_project_status 加速状态查询；idx_project_dates 支持时间范围检索；idx_collection_project 优化项目关联查询；idx_result_access 加速权限过滤；idx_monitor_species 和 idx_monitor_habitat 优化监测记录查询。存储过程(2 个)：sp_check_project_completion 检查项目是否满足结题条件；sp_export_results_by_access 按权限导出成果清单。触发器(2 个)：tr_prevent_collection_on_completed 阻止向已结题项目新增采集记录；tr_result_auto_archive 自动填充成果发布时间。