1. 用户故事优先级分级

P0：必须实现（核心功能）

- **US-001：创建智能体** - 核心基础功能
- **US-002：查看智能体列表** - 基本管理功能
- **US-003：编辑智能体** - 基本管理功能
- **US-004：测试智能体** - 核心调试功能
- **US-006：上传文档** - 知识库核心功能
- **US-012：发布智能体** - 核心部署功能
- **US-013：智能体对话** - 核心使用功能

P1：重要功能

- **US-005：创建知识库** - 知识库管理基础
- **US-007：查看知识库文档** - 知识库管理基础
- **US-009：创建工作流** - 工作流核心功能
- **US-010：执行工作流** - 工作流核心功能
- **US-016：复制智能体** - 提高效率的重要功能

P2：增强功能

- **US-008：关联知识库到智能体** - 功能集成增强
- **US-011：关联工作流到智能体** - 功能集成增强
- **US-014：嵌入智能体到网站** - 部署方式扩展
- **US-015：查看使用统计** - 数据分析增强
- **US-017：使用模板创建智能体** - 用户体验增强

2. 数据库设计映射

核心实体表

用户与权限表

```
-- 用户表
CREATE TABLE users (
    id VARCHAR(50) PRIMARY KEY,
    username VARCHAR(100) UNIQUE NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash VARCHAR(255) NOT NULL,
    status ENUM('active', 'inactive', 'suspended') DEFAULT 'active'
```

```
    status ENUM('active', 'inactive', 'suspended') DEFAULT 'active',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
);

-- 角色表
CREATE TABLE roles (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- 权限表
CREATE TABLE permissions (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    resource_type ENUM('agent', 'knowledge_base', 'workflow', 'system') NOT NULL,
    action ENUM('create', 'read', 'update', 'delete', 'execute', 'publish') NOT NULL,
    description TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- 角色权限关联表
CREATE TABLE role_permissions (
    role_id VARCHAR(50),
    permission_id VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (role_id, permission_id),
    FOREIGN KEY (role_id) REFERENCES roles(id),
    FOREIGN KEY (permission_id) REFERENCES permissions(id)
);

-- 用户角色关联表
CREATE TABLE user_roles (
    user_id VARCHAR(50),
    role_id VARCHAR(50,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (user_id, role_id),
    FOREIGN KEY (user_id) REFERENCES users(id),
    FOREIGN KEY (role_id) REFERENCES roles(id)
);
```

智能体相关表

```
-- 智能体表 (US-001, US-002, US-003, US-012)
CREATE TABLE agents (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
```

```
    prompt TEXT NOT NULL,
    model_config JSON, -- 存储模型类型、参数等配置
    status ENUM('draft', 'published', 'archived') DEFAULT 'draft',
    user_id VARCHAR(50) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

-- 智能体对话记录表 (US-004, US-013)
CREATE TABLE agent_conversations (
    id VARCHAR(50) PRIMARY KEY,
    agent_id VARCHAR(50) NOT NULL,
    session_id VARCHAR(100), -- 会话ID，用于多轮对话
    user_input TEXT NOT NULL,
    agent_response TEXT NOT NULL,
    metadata JSON, -- 存储对话的元数据，如token使用量、处理时间等
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (agent_id) REFERENCES agents(id)
);
```

知识库相关表

```
-- 知识库表 (US-005)
CREATE TABLE knowledge_bases (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    user_id VARCHAR(50) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

-- 文档表 (US-006, US-007)
CREATE TABLE documents (
    id VARCHAR(50) PRIMARY KEY,
    knowledge_base_id VARCHAR(50) NOT NULL,
    file_name VARCHAR(255) NOT NULL,
    file_path VARCHAR(500), -- 文件存储路径
    file_size BIGINT,
    file_type VARCHAR(50),
    content_text LONGTEXT, -- 提取的文本内容
    status ENUM('uploading', 'processing', 'completed', 'failed') DEFAULT 'uploading',
    chunk_count INT DEFAULT 0, -- 分块数量
    error_message TEXT,
    uploaded_by VARCHAR(50) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (knowledge_base_id) REFERENCES knowledge_bases(id),
    FOREIGN KEY (uploaded_by) REFERENCES users(id)
);

-- 文档分块表 (向量化存储)
CREATE TABLE document_chunks (
    id VARCHAR(50) PRIMARY KEY,
    document_id VARCHAR(50) NOT NULL,
    chunk_index INT NOT NULL,
    content TEXT NOT NULL,
    content_vector LONGBLOB, -- 向量化后的数据
    token_count INT,
    metadata JSON,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (document_id) REFERENCES documents(id)
);

-- 智能体-知识库关联表 (US-008)
CREATE TABLE agent_knowledge_bases (
    agent_id VARCHAR(50),
    knowledge_base_id VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (agent_id, knowledge_base_id),
    FOREIGN KEY (agent_id) REFERENCES agents(id),
    FOREIGN KEY (knowledge_base_id) REFERENCES knowledge_bases(id)
);
```

工作流相关表

```
-- 工作流表 (US-009)
CREATE TABLE workflows (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    definition JSON NOT NULL, -- 存储工作流的JSON定义
    status ENUM('active', 'inactive') DEFAULT 'active',
    user_id VARCHAR(50) NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id)
);

-- 工作流执行记录表 (US-010)
CREATE TABLE workflow_executions (
    id VARCHAR(50) PRIMARY KEY,
    workflow_id VARCHAR(50) NOT NULL,
    input_parameters JSON,
    output_result JSON,
    status ENUM('pending', 'running', 'completed', 'failed') DEFAULT 'pending',
```

```
    started_at TIMESTAMP NULL,
    completed_at TIMESTAMP NULL,
    error_message TEXT,
    execution_log JSON, -- 存储详细的执行日志
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (workflow_id) REFERENCES workflows(id)
);

-- 智能体-工作流关联表 (US-011)
CREATE TABLE agent_workflows (
    agent_id VARCHAR(50),
    workflow_id VARCHAR(50),
    trigger_condition JSON, -- 触发条件的JSON配置
    priority INT DEFAULT 0,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (agent_id, workflow_id),
    FOREIGN KEY (agent_id) REFERENCES agents(id),
    FOREIGN KEY (workflow_id) REFERENCES workflows(id)
);
```

部署与使用统计表

```
-- 智能体部署表 (US-012, US-014)
CREATE TABLE agent_deployments (
    id VARCHAR(50) PRIMARY KEY,
    agent_id VARCHAR(50) NOT NULL,
    deployment_type ENUM('web_link', 'api', 'embed') NOT NULL,
    access_url VARCHAR(500), -- 访问链接
    api_key VARCHAR(100) UNIQUE, -- API密钥
    embed_code TEXT, -- 嵌入代码
    config JSON, -- 部署配置
    status ENUM('active', 'inactive') DEFAULT 'active',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (agent_id) REFERENCES agents(id)
);

-- 使用统计表 (US-015)
CREATE TABLE usage_statistics (
    id VARCHAR(50) PRIMARY KEY,
    agent_id VARCHAR(50) NOT NULL,
    deployment_id VARCHAR(50),
    metric_type ENUM('conversation_count', 'user_count', 'api_call') NOT NULL,
    metric_value INT NOT NULL,
    period_date DATE, -- 统计日期
    metadata JSON,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (agent_id) REFERENCES agents(id),
    FOREIGN KEY (deployment_id) REFERENCES agent_deployments(id)
);
```

模板与复制表

```
-- 智能体模板表 (US-017)
CREATE TABLE agent_templates (
    id VARCHAR(50) PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    description TEXT,
    category VARCHAR(100),
    prompt_template TEXT NOT NULL,
    default_config JSON,
    is_public BOOLEAN DEFAULT FALSE,
    created_by VARCHAR(50),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (created_by) REFERENCES users(id)
);

-- 智能体复制记录表 (US-016)
CREATE TABLE agent_copies (
    id VARCHAR(50) PRIMARY KEY,
    original_agent_id VARCHAR(50) NOT NULL,
    copied_agent_id VARCHAR(50) NOT NULL,
    copied_by VARCHAR(50) NOT NULL,
    copy_type ENUM('full', 'structure_only') DEFAULT 'full',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (original_agent_id) REFERENCES agents(id),
    FOREIGN KEY (copied_agent_id) REFERENCES agents(id),
    FOREIGN KEY (copied_by) REFERENCES users(id)
);
```

数据库设计特点

1. **功能分离**：每个模块都有独立的表结构，便于维护和扩展
2. **RBAC支持**：通过用户-角色-权限三层结构实现细粒度权限控制
3. **数据完整性**：使用外键约束确保数据一致性
4. **扩展性**：使用JSON字段存储复杂配置，便于后续功能扩展
5. **审计追踪**：包含创建时间、更新时间等审计字段
6. **状态管理**：统一的状态字段便于生命周期管理

这个数据库设计能够完整支持所有用户故事，并为未来的功能扩展预留了空间。