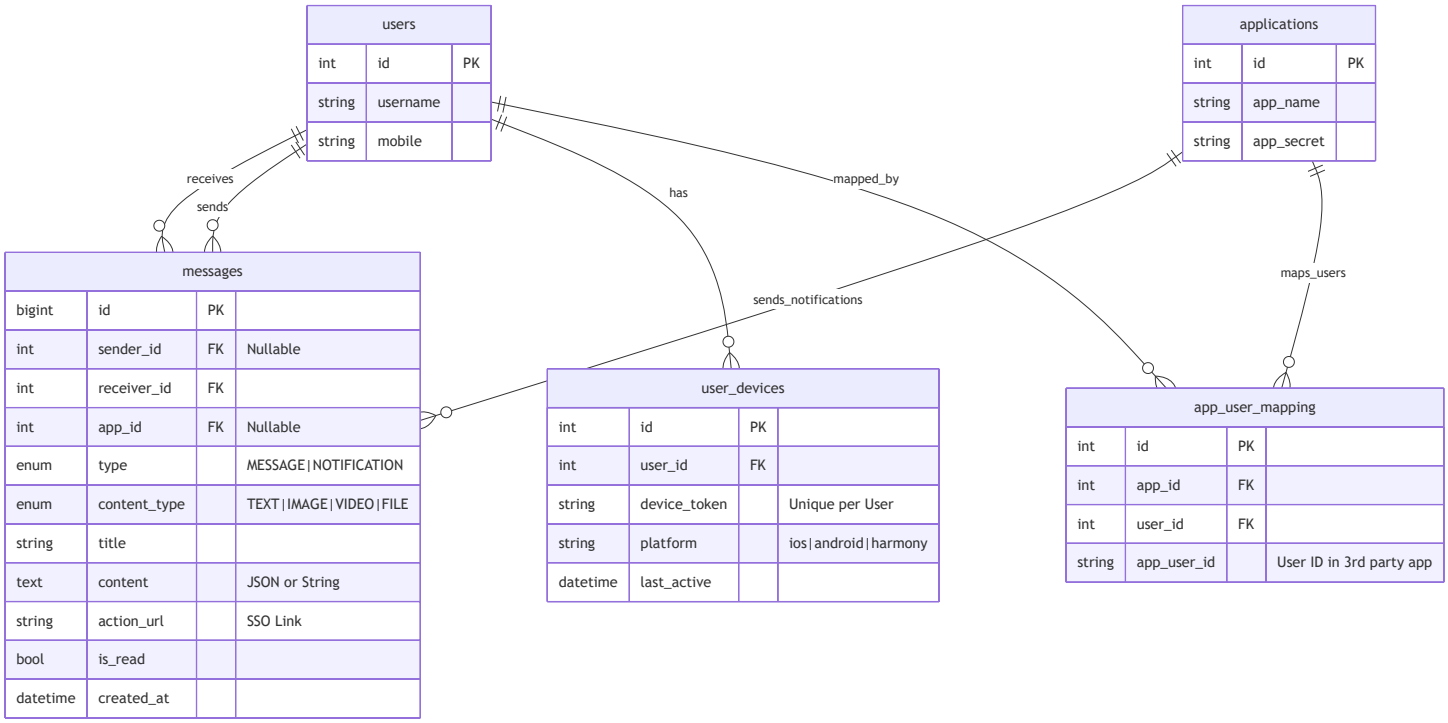
统一消息系统数据库与接口文档

版本: V1.1
日期: 2026-02-23
状态: 已发布

本文档详细描述了统一消息平台的数据库设计（Schema）、API 接口规范以及权限控制策略。

1. 数据库设计 (Database Schema)

1.1 ER 图 (Entity-Relationship Diagram)



1.2 表结构详解

1.2.1 消息表 (`messages`)

存储所有用户私信和系统通知的核心表。

字段名	类型	必填	默认值	说明
`id`	BIGINT	是	Auto Inc	主键，消息唯一标识

字段名	类型	必填	默认值	说明
sender_id	INT	否	NULL	发送者 UserID (私信必填，系统通知为空)
receiver_id	INT	是	-	**接收者 UserID** (关联 users.id)
app_id	INT	否	NULL	来源应用 ID (关联 applications.id)
type	VARCHAR(20)	是	MESSAGE	消息类型: MESSAGE (私信), NOTIFICATION (通知)
content_type	VARCHAR(20)	是	TEXT	内容类型: TEXT , IMAGE , VIDEO , FILE
title	VARCHAR(255)	是	-	消息标题 (私信可与内容一致或摘要)
content	TEXT	是	-	消息内容。多媒体类型存储 JSON 字符串 (如 {"url":"...", "size":1024})
action_url	VARCHAR(1000)	否	NULL	点击跳转链接 (通常包含 SSO Ticket)
action_text	VARCHAR(50)	否	NULL	跳转按钮文案 (如"去审批")
is_read	TINYINT(1)	是	0	是否已读 (0:未读, 1:已读)
created_at	TIMESTAMP	是	NOW()	创建时间
read_at	TIMESTAMP	否	NULL	阅读时间

索引:

- idx_receiver_id : (receiver_id) - 加速“我的消息”查询
- idx_sender_id : (sender_id) - 加速“我发送的消息”查询
- idx_app_id : (app_id) - 统计应用发送量

1.2.2 用户设备表 (user_devices)

存储移动端设备的推送 Token，用于离线推送。

字段名	类型	必填	说明
id	INT	是	主键
user_id	INT	是	关联用户 ID
device_token	VARCHAR(255)	是	厂商推送 Token (如 APNs Token)
platform	VARCHAR(20)	是	平台: ios , android , harmony
device_name	VARCHAR(100)	否	设备名称 (如 "iPhone 13")
last_active	TIMESTAMP	是	最后活跃时间

约束:

- `uq_user_device`： `UNIQUE(user_id, device_token)` - 防止同一用户同一设备重复注册

2. 接口权限设计 (Authentication & Authorization)

本系统采用 **混合认证策略 (Hybrid Auth)**，同时支持用户操作和第三方系统调用。

2.1 认证方式

A. 用户认证 (User Context)

- **适用场景**: 移动端/Web端用户发送私信、查询历史消息。
- **机制**: HTTP Header `Authorization: Bearer <JWT_TOKEN>`
- **身份**: 解析 JWT 获取 `current_user` 对象。

B. 应用认证 (Application Context)

- **适用场景**: OA/ERP 等后端系统调用接口发送业务通知。
- **机制**: HTTP Headers 签名验证。
  - `X-App-Id`：应用 ID
  - `X-Timestamp`：当前时间戳 (用于防重放)
  - `X-Sign`：签名字符串
- **签名算法**: `MD5(app_id + timestamp + app_secret)`
- **身份**: 验证通过后获取 `current_app` 对象。

2.2 权限控制矩阵

操作	接口路径	用户 (User) 权限	应用 (App) 权限	说明
发送私信	`POST /messages/`	✅ 允许 (type=MESSAGE)	✅ 允许	应用可冒充系统发私信
发送通知	`POST /messages/`	❌ 禁止	✅ 允许 (type=NOTIFICATION)	只有应用能发通知
查询列表	`GET /messages/`	✅ 仅限查询自己的	❌ 禁止	应用只负责发，不负责查
会话列表	`GET /conversations`	✅ 仅限查询自己的	❌ 禁止	-

操作	接口路径	用户 (User) 权限	应用 (App) 权限	说明
标记已读	`PUT /read`	✅ 仅限操作自己的	❌ 禁止	状态由用户端触发
上传文件	`POST /upload`	✅ 允许	✅ 允许	-

3. API 接口定义 (API Reference)

Base URL: `/api/v1`

3.1 消息发送 (Send Message)

支持用户私信和应用通知的统一发送接口。

- **Endpoint**: `POST /messages/`
- **Auth**: User Token 或 App Signature
- **Request Body**:

参数名	类型	必填	说明
`receiver_id`	int	选填	接收者统一用户ID (与 `app_user_id` 二选一)
`app_id`	int	选填	应用ID (若使用 `app_user_id` 则必填)
`app_user_id`	string	选填	第三方业务系统账号 (需已建立映射)
`type`	string	是	`MESSAGE` 或 `NOTIFICATION`
`content_type`	string	是	`TEXT` , `IMAGE` , `VIDEO` , `FILE`
`title`	string	是	标题
`content`	string/json	是	文本内容或文件元数据 JSON
`action_url`	string	否	原始跳转链接
`action_text`	string	否	按钮文案
`auto_sso`	bool	否	是否自动封装 SSO 跳转 (默认为 False)
`target_url`	string	否	若 `auto_sso=true` , 此为最终业务目标 URL

- **Example Request (应用发送通知)**:

```
{
  "app_id": 101,
  "app_user_id": "zhangsan_oa",
  "type": "NOTIFICATION",
  "content_type": "TEXT",
  "title": "OA审批提醒",
  "content": "您有一条新的报销单待审批",
  "auto_sso": true,
  "target_url": "http://oa.example.com/audit/123",
  "action_text": "立即处理"
}
```

- **Response (200 OK)**:

```
{
  "id": 501,
  "type": "NOTIFICATION",
  "content": "您有一条新的报销单待审批",
  "action_url": "http://api.platform.com/api/v1/auth/sso/jump?app_id=101&redirect_to=...",
  "created_at": "2026-02-23T10:00:00"
}
```

3.2 获取会话列表 (Get Conversations)

获取当前用户的会话聚合列表（类似微信首页）。

- **Endpoint**: `GET /messages/conversations`
- **Auth**: User Token Only
- **Response**: List of Conversations

```
[
  {
    "user_id": 0,
    "username": "System",
    "full_name": "系统通知",
    "unread_count": 5,
    "last_message": "您的密码已重置",
    "last_message_type": "TEXT",
    "updated_at": "2026-02-23T10:05:00"
  },
  {
    "user_id": 102,
    "username": "13800138000",
    "full_name": "李四",
    "unread_count": 0,
    "last_message": "[IMAGE]",
    "last_message_type": "IMAGE",
    "updated_at": "2026-02-22T18:30:00"
  }
]
```


3.3 获取聊天记录 (Get History)

- **Endpoint**: `GET /messages/history/{other_user_id}`
- **Params**:
  - `other_user_id`：对方用户ID (0 代表系统通知)
  - `skip`：分页偏移 (默认0)
  - `limit`：条数 (默认50)
- **Response**: List of Messages

3.4 消息状态管理

- **获取未读数**: `GET /messages/unread-count`
- **标记单条已读**: `PUT /messages/{message_id}/read`
- **标记全部已读**: `PUT /messages/read-all`
- **删除消息**: `DELETE /messages/{message_id}`

3.5 文件上传 (Upload)

- **Endpoint**: `POST /messages/upload`
- **Content-Type**: `multipart/form-data`
- **Form Field**: `file`
- **Response**:

```
{
  "url": "messages/1/2026/02/uuid.jpg",
  "filename": "image.jpg",
  "content_type": "image/jpeg",
  "size": 50200
}
```

> **注意**: 返回的 `url` 是 MinIO 对象 Key，发送消息时应将此 JSON 作为 `content` 发送。

3.6 SSO 跳转 (SSO Jump)

- **Endpoint**: `GET /auth/sso/jump`
- **Params**:
  - `app_id`：目标应用 ID
  - `redirect_to`：最终跳转地址
- **逻辑**: 验证登录态 -> 生成 Ticket -> 302 跳转到 `AppCallbackUrl?ticket=...&next=redirect_to`。

4. WebSocket 实时接口

- **URL**: `ws://{host}/api/v1/ws/messages`
- **Query Param**: `token={JWT_TOKEN}`
- **Events**:
  - **Server -> Client**:

    ```
    {
      "type": "NEW_MESSAGE",
      "data": {
        "id": 502,
        "sender_id": 102,
        "title": "新消息",
        "content": "...",
        "content_type": "TEXT",
        "created_at": "..."
      }
    }
    ```

  - **Client -> Server**:
    - `ping`：心跳包，服务器回复 `pong`。

5. 枚举定义

MessageType

- `MESSAGE`：普通用户私信
- `NOTIFICATION`：系统/应用通知

ContentType

- `TEXT`：纯文本
- `IMAGE`：图片
- `VIDEO`：视频
- `FILE`：普通文件