# **电商购物平台 - 数据库设计说明书**

## **1. 引言 (Introduction)**

### **1.1 目的 (Purpose)**

本文档旨在详细说明“电商购物平台”项目的数据库设计方案。它是基于《电商购物平台 - 软件需求规格说明书 (SRS)》和《电商购物平台 - 软件系统设计 (SDD)》中定义的需求和总体设计进行的详细数据库结构设计。本文档旨在确保数据的一致性、完整性、可扩展性和性能，并作为数据库创建、开发团队理解数据结构以及后续维护的主要依据。

### **1.2 范围 (Scope)**

本文档涵盖了电商平台核心业务功能所需的数据库表结构设计，包括用户、地址、商品、分类、订单等模块。

* **包含:** 核心业务实体的数据表结构、详细的字段定义（含义、类型、约束、示例）、索引策略及原理、关系完整性规则、以及相关的设计考量。
* **不包含:** 缓存（如 Redis）中存储的数据结构设计（例如购物车）、详细的数据字典（本文档已包含大部分信息）、数据库部署/监控/运维细节、数据迁移方案、数据库用户权限管理等。

### **1.3 参考文献 (References)**

* 《电商购物平台 - 软件需求规格说明书 (SRS)》
* 《电商购物平台 - 软件系统设计 (SDD)》

## **2. 概念结构设计 (Conceptual Structure Design)**

本阶段识别核心业务实体及其相互关系，构建 E-R 模型。

### **2.1 核心实体 (Entities)**

* 用户 (User)
* 地址 (Address)
* 商品 (Product)
* 商品分类 (Category)
* 订单 (Order)
* 订单项 (OrderItem)

*(注：根据系统设计，购物车(ShoppingCart)和购物车项(CartItem)采用 Redis 存储，不设计为数据库实体)。*

### **2.2 实体间关系 (Relationships)**

* 一个 **用户** 可以拥有多个 **地址** (1:N)。
* 一个 **用户** 可以创建多个 **订单** (1:N)。
* 一个 **商品分类** 可以包含多个 **商品** (1:N)。
* 一个 **商品** 属于一个 **商品分类** (N:1)。
* 一个 **商品分类** 可以有父分类，也可以包含子分类 (1:N, 自引用)。
* 一个 **订单** 包含多个 **订单项** (1:N, 组成关系)。
* 一个 **商品** 可以出现在多个 **订单项** 中 (1:N) (订单项引用下单时的商品快照)。

*(对应的 E-R 图见附录)*

## **3. 逻辑结构设计 (Logical Structure Design)**

本阶段将概念模型转换为关系数据库模式，定义表、列、数据类型和约束。

### **3.1 关系模式 (Tables)**

#### **3.1.1 用户表 (t\_user)**

**描述:** 存储平台注册用户的基本信息和状态。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 用户唯一标识，自增 | 1, 2, 3... |
| username | VARCHAR(50) | Unique, Not Null | 用户名，用于登录 | "john\_doe", "user123" |
| password\_hash | VARCHAR(60) | Not Null | BCrypt等哈希算法处理后的密码 | "2a10$..." |
| email | VARCHAR(100) | Unique, Not Null | 电子邮箱，可用于登录或找回密码 | "john.doe@example.com" |
| phone\_number | VARCHAR(20) | Unique, Nullable | 手机号码，可用于登录或接收通知 | "13800138000" |
| status | TINYINT | Not Null, Default: 1 | 账户状态 | 1: Active, 0: Inactive/Disabled |
| created\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP | 记录创建时间 | "2025-04-22 10:00:00" |
| updated\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP (on update) | 记录最后更新时间 | "2025-04-22 10:05:00" |

#### **3.1.2 地址表 (t\_address)**

**描述:** 存储用户的收货地址信息。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 地址唯一标识，自增 | 101, 102... |
| user\_id | BIGINT | FK -> t\_user.id, Not Null | 关联的用户 ID | 1 |
| receiver\_name | VARCHAR(50) | Not Null | 收货人姓名 | "张三" |
| phone\_number | VARCHAR(20) | Not Null | 收货人联系电话 | "13912345678" |
| province | VARCHAR(50) | Not Null | 省份 | "广东省" |
| city | VARCHAR(50) | Not Null | 城市 | "深圳市" |
| district | VARCHAR(50) | Not Null | 区/县 | "南山区" |
| detailed\_address | VARCHAR(255) | Not Null | 详细街道门牌号 | "科技园路1号" |
| is\_default | TINYINT(1) | Not Null, Default: 0 | 是否为默认地址 | 1: 是, 0: 否 |
| created\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP | 记录创建时间 | "2025-04-22 10:10:00" |
| updated\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP (on update) | 记录最后更新时间 | "2025-04-22 10:10:00" |

**外键:** user\_id 参照 t\_user(id)，建议设置 ON DELETE CASCADE，即用户注销时，其关联的地址信息也一并删除。

#### **3.1.3 商品分类表 (t\_category)**

**描述:** 存储商品的分类信息，支持层级结构。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 分类唯一标识，自增 | 1, 2, 10... |
| name | VARCHAR(100) | Not Null | 分类名称 | "手机", "笔记本电脑", "男装" |
| parent\_id | BIGINT | FK -> t\_category.id, Nullable | 父分类 ID，顶级分类此字段为 NULL | NULL (顶级), 1 (子分类) |
| sort\_order | INT | Default: 0 | 同级分类下的排序值，越小越靠前 | 0, 10, 20... |
| created\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP | 记录创建时间 | "2025-04-22 09:00:00" |
| updated\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP (on update) | 记录最后更新时间 | "2025-04-22 09:05:00" |

**外键:** parent\_id 参照 t\_category(id)，建议设置 ON DELETE SET NULL，即父分类被删除时，其子分类的 parent\_id 设为 NULL，使其成为顶级分类（或根据业务逻辑选择 RESTRICT 或 CASCADE）。

#### **3.1.4 商品表 (t\_product)**

**描述:** 存储在售商品的核心信息。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 商品唯一标识，自增 | 1001, 1002... |
| name | VARCHAR(255) | Not Null | 商品名称/标题 | "新款智能手机 Pro Max" |
| description | TEXT | Nullable | 商品详细描述（可能包含HTML） | "6.7英寸超视网膜XDR显示屏..." |
| category\_id | BIGINT | FK -> t\_category.id, Not Null | 商品所属分类 ID | 1 (手机分类) |
| price | DECIMAL(10, 2) | Not Null, CHECK (price > 0) | 商品销售价格 | 8999.00 |
| stock\_quantity | INT | Not Null, Default: 0, CHECK (stock\_quantity >= 0) | 当前库存数量 | 100 |
| image\_url | VARCHAR(512) | Nullable | 商品主图片 URL | "<https://img.example.com/>..." |
| status | TINYINT | Not Null, Default: 1 | 商品状态 | 1: On Sale, 0: Off Sale, 2: Deleted |
| created\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP | 记录创建时间 | "2025-04-22 11:00:00" |
| updated\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP (on update) | 记录最后更新时间 | "2025-04-22 11:05:00" |

**外键:** category\_id 参照 t\_category(id)，建议设置 ON DELETE RESTRICT，即不允许删除仍包含商品的分类（或根据业务设为 SET NULL）。 **检查约束:** 确保价格大于0，库存不为负数。

#### **3.1.5 订单表 (t\_order)**

**描述:** 存储用户提交的订单主体信息。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 订单唯一标识，自增 | 5001, 5002... |
| order\_no | VARCHAR(32) | Unique, Not Null | 业务订单号，系统生成，全局唯一 | "2025042214300012345" |
| user\_id | BIGINT | FK -> t\_user.id, Not Null | 下单用户 ID | 1 |
| total\_amount | DECIMAL(12, 2) | Not Null | 订单中所有商品原价总和 | 9098.00 |
| payable\_amount | DECIMAL(12, 2) | Not Null | 用户实际需要支付的总金额（含运费、折扣） | 8999.00 |
| status | VARCHAR(20) | Not Null, Default: 'PENDING\_PAYMENT' | 订单当前状态 | 'PENDING\_PAYMENT', 'PAID', 'SHIPPED', 'DELIVERED', 'COMPLETED', 'CANCELLED' |
| receiver\_name | VARCHAR(50) | Not Null | 收货人姓名 (冗余快照) | "张三" |
| receiver\_phone | VARCHAR(20) | Not Null | 收货人电话 (冗余快照) | "13912345678" |
| receiver\_address | VARCHAR(500) | Not Null | 完整收货地址 (冗余快照) | "广东省深圳市南山区科技园路1号" |
| payment\_method | VARCHAR(50) | Nullable | 支付方式 | "SIMULATED\_PAY", "WECHAT\_PAY" |
| paid\_at | DATETIME | Nullable | 支付成功时间 | "2025-04-22 14:35:00" |
| shipped\_at | DATETIME | Nullable | 商家发货时间 | "2025-04-23 10:00:00" |
| completed\_at | DATETIME | Nullable | 订单完成时间（用户确认收货或超时） | "2025-04-25 15:00:00" |
| created\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP | 订单创建时间 | "2025-04-22 14:30:00" |
| updated\_at | DATETIME | Not Null, Default: CURRENT\_TIMESTAMP (on update) | 记录最后更新时间 | "2025-04-22 14:35:00" |

**外键:** user\_id 参照 t\_user(id)，建议设置 ON DELETE RESTRICT，即通常不允许删除仍有订单的用户（或标记为注销）。 **状态枚举值:** 订单状态 (status) 的具体值应在代码层面维护一个枚举或常量类，确保一致性。

#### **3.1.6 订单项表 (t\_order\_item)**

**描述:** 存储订单中包含的每一个商品项的详细信息（快照）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **列名 (Column)** | **类型 (Type)** | **约束 (Constraints)** | **描述 (Description)** | **示例/备注 (Example/Notes)** |
| id | BIGINT | PK, Not Null | 订单项唯一标识，自增 | 10001, 10002... |
| order\_id | BIGINT | FK -> t\_order.id, Not Null | 所属订单 ID | 5001 |
| product\_id | BIGINT | FK -> t\_product.id, Not Null | 关联的商品 ID (用于追溯原始商品) | 1001 |
| product\_name | VARCHAR(255) | Not Null | 下单时的商品名称 (快照) | "新款智能手机 Pro Max" |
| product\_image\_url | VARCHAR(512) | Nullable | 下单时的商品图片 URL (快照) | "<https://img.example.com/>..." |
| unit\_price | DECIMAL(10, 2) | Not Null | 下单时的商品单价 (快照) | 8999.00 |
| quantity | INT | Not Null, CHECK (quantity > 0) | 购买数量 | 1 |
| total\_price | DECIMAL(12, 2) | Not Null | 该项商品总价 (quantity \* unit\_price) | 8999.00 |

**外键:**

* order\_id 参照 t\_order(id)，建议设置 ON DELETE CASCADE，即删除订单时，其包含的所有订单项也一并删除。
* product\_id 参照 t\_product(id)，建议设置 ON DELETE RESTRICT，即不允许删除仍被订单项引用的商品（或标记为已删除）。  
  检查约束: 确保购买数量大于0。  
  快照字段: product\_name, product\_image\_url, unit\_price 存储的是下单时刻的数据，不随原始商品信息改变而改变，保证订单历史记录的准确性。

## **4. 物理结构设计 (Physical Structure Design)**

本阶段针对 MySQL 8.x 数据库管理系统进行具体实现。

### **4.1 存储引擎**

* 所有表均选用 **InnoDB** 存储引擎，因为它提供事务支持（ACID）、行级锁定、外键约束和崩溃恢复能力，是需要高数据一致性和并发性的电商应用的标准选择。

### **4.2 字符集与排序规则**

* 选用 **utf8mb4** 字符集，以支持存储包括 Emoji 在内的 4 字节 UTF-8 字符，满足全球化和用户生成内容的需求。
* 选用 **utf8mb4\_unicode\_ci** (或 MySQL 8.0+ 的 utf8mb4\_0900\_ai\_ci) 作为默认排序规则，提供基于 Unicode 标准的不区分大小写和重音的比较，适用于大多数查询场景。

### **4.3 索引策略与原理**

合理的索引是数据库性能的关键。以下是索引设计策略及原因：

* **主键 (PK) 索引:** 自动创建，聚集索引（InnoDB），保证数据行的唯一性，并作为行的物理存储顺序（或指向）。
* **外键 (FK) 索引:** 为所有外键列创建普通索引。
  + **原因:** 加速 JOIN 操作（根据外键关联表）；加速外键约束检查；优化 ON DELETE/ON UPDATE 级联操作的性能。
* **唯一约束 (Unique) 索引:** 自动创建唯一索引。
  + **原因:** 快速检查插入或更新的数据是否违反唯一性；唯一索引也可用于查询优化。
* **业务查询优化索引:**
  + t\_user.idx\_phone\_number: 如果手机号可用于登录或频繁查询。
  + t\_address.idx\_user\_id: 快速查找某用户的所有地址。
  + t\_category.idx\_parent\_id: 快速查找某分类下的所有子分类。
  + t\_product.idx\_category\_id, t\_product.idx\_status: 支持按分类、按状态筛选商品列表的高频操作。可以考虑创建 (category\_id, status) 或 (status, category\_id) 复合索引，具体顺序取决于最常见的查询过滤顺序。
  + t\_product.idx\_name: 用于商品名称搜索。对于模糊搜索 (LIKE '%keyword%') 效果有限，若搜索是核心功能，应考虑使用**全文索引 (Full-Text Index)** 或外部搜索引擎 (如 Elasticsearch)。前缀索引 (name(191)) 是一个折中方案，适用于 LIKE 'keyword%' 查询。
  + t\_order.idx\_user\_id\_status: **覆盖索引**优化。用户查询自己订单列表时，通常会按状态过滤 (WHERE user\_id = ? AND status = ?)，此复合索引能高效定位，且如果查询仅需 id, order\_no, status 等索引包含的列，可避免回表。
  + t\_order.idx\_created\_at: 支持按订单创建时间范围查询或排序。
  + t\_order\_item.idx\_order\_id: 快速获取某订单的所有订单项。
  + t\_order\_item.idx\_product\_id: 可能用于统计某商品的销售情况或关联查询。

### **4.4 数据完整性与参照完整性**

* **实体完整性:** 通过主键约束保证每行数据的唯一性。
* **域完整性:** 通过数据类型、NOT NULL 约束、CHECK 约束（如 price > 0, quantity > 0）和 DEFAULT 约束保证列值的有效性。
* **参照完整性:** 通过外键约束维护表之间的引用关系。
  + **ON DELETE** / **ON UPDATE** 行为:
    - CASCADE: 父表记录删除/更新时，子表关联记录也随之删除/更新 (e.g., t\_address on t\_user delete, t\_order\_item on t\_order delete)。需谨慎使用，防止意外数据丢失。
    - SET NULL: 父表记录删除/更新时，子表外键列设为 NULL (e.g., t\_category.parent\_id on parent delete)。适用于允许孤立子记录的场景。
    - RESTRICT / NO ACTION (默认): 如果存在子表关联记录，则不允许删除/更新父表记录 (e.g., t\_product referenced by t\_order\_item, t\_user referenced by t\_order)。这是最安全的选择，强制先处理子表数据。
  + 本设计中已根据常见业务逻辑选择了合适的参照动作，具体见 CREATE TABLE 语句中的 FOREIGN KEY 定义。

### **4.5 潜在优化与考量**

* **分区 (Partitioning):** 对于 t\_order 和 t\_order\_item 表，随着数据量增长（年订单量预期 50 万 [Source 54]），性能可能下降。未来可考虑：
  + **按范围分区 (Range Partitioning):** 基于 created\_at 列按月或按年分区，便于按时间范围查询和归档旧数据。
  + **按哈希分区 (Hash Partitioning):** 基于 user\_id 进行哈希分区，可以将不同用户的订单分散到不同分区，均衡 I/O，但不利于按时间范围查询。
  + 分区策略的选择需根据主要查询模式和数据管理需求决定。
* **读写分离:** 针对高读取负载，可考虑设置 MySQL 主从复制，将读请求路由到从库，降低主库压力。
* **数据归档:** 对于历史订单数据（如超过2年的订单），可以考虑定期将其迁移到归档库或数据仓库，以减小主库体积，提升在线查询性能。
* **反范式设计:** 当前设计中已包含必要的反范式（如订单中的收货信息快照、订单项中的商品信息快照），以保证历史数据不变性和查询性能。未来根据性能瓶颈，可能需要考虑更多的反范式设计，但这会增加数据冗余和维护一致性的成本。

### **4.6 数据表定义 (CREATE TABLE Statements)**

-- 用户表  
CREATE TABLE `t\_user` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '用户ID',  
 `username` VARCHAR(50) NOT NULL UNIQUE COMMENT '用户名 (5-20位字母/数字/下划线)',  
 `password\_hash` VARCHAR(60) NOT NULL COMMENT 'BCrypt哈希后的密码',  
 `email` VARCHAR(100) NOT NULL UNIQUE COMMENT '邮箱 (RFC 5322)',  
 `phone\_number` VARCHAR(20) NULL UNIQUE COMMENT '手机号',  
 `status` TINYINT NOT NULL DEFAULT 1 COMMENT '状态 (1:Active, 0:Inactive)',  
 `created\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  
 `updated\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  
 INDEX `idx\_phone\_number` (`phone\_number`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='用户信息表';  
  
-- 地址表  
CREATE TABLE `t\_address` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '地址ID',  
 `user\_id` BIGINT NOT NULL COMMENT '用户ID',  
 `receiver\_name` VARCHAR(50) NOT NULL COMMENT '收货人姓名',  
 `phone\_number` VARCHAR(20) NOT NULL COMMENT '收货人手机号',  
 `province` VARCHAR(50) NOT NULL COMMENT '省份',  
 `city` VARCHAR(50) NOT NULL COMMENT '城市',  
 `district` VARCHAR(50) NOT NULL COMMENT '区/县',  
 `detailed\_address` VARCHAR(255) NOT NULL COMMENT '详细地址',  
 `is\_default` TINYINT(1) NOT NULL DEFAULT 0 COMMENT '是否默认地址 (1:是, 0:否)',  
 `created\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  
 `updated\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  
 INDEX `idx\_user\_id` (`user\_id`),  
 FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`id`) ON DELETE CASCADE  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='用户收货地址表';  
  
-- 商品分类表  
CREATE TABLE `t\_category` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '分类ID',  
 `name` VARCHAR(100) NOT NULL COMMENT '分类名称',  
 `parent\_id` BIGINT NULL COMMENT '父分类ID',  
 `sort\_order` INT DEFAULT 0 COMMENT '排序值 (越小越靠前)',  
 `created\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  
 `updated\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  
 INDEX `idx\_parent\_id` (`parent\_id`),  
 FOREIGN KEY (`parent\_id`) REFERENCES `t\_category` (`id`) ON DELETE SET NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='商品分类表';  
  
-- 商品表  
CREATE TABLE `t\_product` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '商品ID',  
 `name` VARCHAR(255) NOT NULL COMMENT '商品名称',  
 `description` TEXT NULL COMMENT '商品描述',  
 `category\_id` BIGINT NOT NULL COMMENT '分类ID',  
 `price` DECIMAL(10, 2) NOT NULL COMMENT '价格 (需>0)',  
 `stock\_quantity` INT NOT NULL DEFAULT 0 COMMENT '库存数量 (需>=0)',  
 `image\_url` VARCHAR(512) NULL COMMENT '商品主图URL',  
 `status` TINYINT NOT NULL DEFAULT 1 COMMENT '状态 (1:On Sale, 0:Off Sale, 2:Deleted)',  
 `created\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  
 `updated\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  
 INDEX `idx\_category\_id` (`category\_id`),  
 INDEX `idx\_status` (`status`),  
 INDEX `idx\_name` (`name`(191)) COMMENT '商品名称索引，用于搜索',  
 FOREIGN KEY (`category\_id`) REFERENCES `t\_category` (`id`),  
 CHECK (`price` > 0),  
 CHECK (`stock\_quantity` >= 0)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='商品信息表';  
  
-- 订单表  
CREATE TABLE `t\_order` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '订单ID',  
 `order\_no` VARCHAR(32) NOT NULL UNIQUE COMMENT '订单号',  
 `user\_id` BIGINT NOT NULL COMMENT '用户ID',  
 `total\_amount` DECIMAL(12, 2) NOT NULL COMMENT '商品总金额',  
 `payable\_amount` DECIMAL(12, 2) NOT NULL COMMENT '应付总额',  
 `status` VARCHAR(20) NOT NULL DEFAULT 'PENDING\_PAYMENT' COMMENT '订单状态',  
 `receiver\_name` VARCHAR(50) NOT NULL COMMENT '收货人姓名 (冗余)',  
 `receiver\_phone` VARCHAR(20) NOT NULL COMMENT '收货人电话 (冗余)',  
 `receiver\_address` VARCHAR(500) NOT NULL COMMENT '收货地址 (冗余)',  
 `payment\_method` VARCHAR(50) NULL COMMENT '支付方式',  
 `paid\_at` DATETIME NULL COMMENT '支付时间',  
 `shipped\_at` DATETIME NULL COMMENT '发货时间',  
 `completed\_at` DATETIME NULL COMMENT '完成时间',  
 `created\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '创建时间',  
 `updated\_at` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '更新时间',  
 INDEX `idx\_user\_id\_status` (`user\_id`, `status`),  
 INDEX `idx\_created\_at` (`created\_at`),  
 FOREIGN KEY (`user\_id`) REFERENCES `t\_user` (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='订单信息表';  
  
-- 订单项表  
CREATE TABLE `t\_order\_item` (  
 `id` BIGINT AUTO\_INCREMENT PRIMARY KEY COMMENT '订单项ID',  
 `order\_id` BIGINT NOT NULL COMMENT '订单ID',  
 `product\_id` BIGINT NOT NULL COMMENT '商品ID (快照关联)',  
 `product\_name` VARCHAR(255) NOT NULL COMMENT '商品名称 (快照)',  
 `product\_image\_url` VARCHAR(512) NULL COMMENT '商品图片URL (快照)',  
 `unit\_price` DECIMAL(10, 2) NOT NULL COMMENT '下单时单价 (快照)',  
 `quantity` INT NOT NULL COMMENT '购买数量 (需>0)',  
 `total\_price` DECIMAL(12, 2) NOT NULL COMMENT '该项总价',  
 INDEX `idx\_order\_id` (`order\_id`),  
 INDEX `idx\_product\_id` (`product\_id`),  
 FOREIGN KEY (`order\_id`) REFERENCES `t\_order` (`id`) ON DELETE CASCADE,  
 FOREIGN KEY (`product\_id`) REFERENCES `t\_product` (`id`) ON DELETE RESTRICT,  
 CHECK (`quantity` > 0)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci COMMENT='订单项信息表';

## **5. 附录：E-R 图 (PlantUML)**

