# **数仓分层设计文档**

## **一、数仓 ODS 层**

### **1.1 ODS 层数据同步整体思路**

ODS 层作为数据仓库的入口，需实现从 MySQL 到 Hive/HDFS 的原始数据同步，核心目标是完整保留原始数据形态，为后续分层处理提供基础。结合脚本配置与业务需求，同步思路可分为 “数据生成与前置准备”“同步脚本配置解析”“同步后存储管理” 三部分。

### **1.2 数据生成与前置准备**

#### **1.2.1 数据生成**

按业务规则通过脚本模拟生成电商核心数据（用户、店铺、商品、行为日志），并以 CSV 格式存储（文件路径：/opt/module/ecommerce\_data/，包括 users.csv、shops.csv、products.csv、behaviors.csv）。  
目的：CSV 格式作为通用文本格式，可最大程度保留原始数据结构（如字段分隔符、空值标识），符合 ODS 层 “原始数据存储” 的定位。

#### **1.2.2 导入 MySQL**

将 CSV 文件批量导入 MySQL 数据库，形成业务源表（如 users、shops、products、behaviors）。  
作用：MySQL 作为中间存储，提供结构化查询能力，便于同步脚本通过 JDBC 连接高效读取数据，同时支持数据临时校验（如重复值、格式错误排查）。

### **1.3 同步脚本配置解析（核心步骤）**

脚本通过 “读取 MySQL 数据→轻量转换→写入 Hive/HDFS” 实现同步，各模块配置与 ODS 层需求的匹配逻辑如下：

#### **1.3.1 环境配置（env）**

env {{

execution.parallelism = 1 # 并行度设为1，确保小批量原始数据同步的顺序性，避免字段错乱

job.mode = "BATCH" # 批处理模式，适配每日全量同步场景（ODS层按日分区存储）}}

设计逻辑：ODS 层需每日全量覆盖原始数据（按 ds 分区），批处理模式可一次性读取 MySQL 全表数据，保证数据完整性；低并行度避免原始数据字段顺序混乱。

#### **1.3.2 数据源配置（source）**

source {{

Jdbc {{

url = "jdbc:mysql://{mysql\_host}:{mysql\_port}/{database}" # MySQL连接地址

driver = "com.mysql.jdbc.Driver" # JDBC驱动

connection\_check\_timeout\_sec = 100 # 连接超时校验，避免同步中断

user = "{mysql\_user}"

password = "{mysql\_password}"

table\_path = "{database}.{table}" # 待同步的MySQL表（如{database}.users）

query = "{query}" # 可选：通过SQL筛选需同步的数据（如按日分区字段过滤）

}}}}

设计逻辑：通过 JDBC 直接读取 MySQL 业务表，确保获取的是 “未经加工的原始数据”（与 ODS 层 “保留原始形态” 要求一致）；支持通过 query 参数添加过滤条件（如同步当日数据 where ds = '20250731'），适配按日分区需求。

#### **1.3.3 数据转换（transform）**

transform {{

# 空转换：仅保留原始字段，不做清洗或衍生（符合ODS层“不处理原始数据”的原则）}}

设计逻辑：ODS 层的核心职责是存储原始数据，而非加工处理。因此转换环节为空，避免修改数据原始形态（如不处理空值、不转换字段格式），确保后续分层处理可基于最原始的数据进行校验和清洗。

#### **1.3.4 数据写入（sink）**

sink {{

Hive {{

table\_name = "{database}.ods\_{table}" # Hive ODS表名（如ods\_users）

metastore\_uri = "{hive\_metastore\_uri}" # Hive元数据地址

hive.hadoop.conf-path = "/etc/hadoop/conf" # Hadoop配置路径，关联HDFS

save\_mode = "overwrite" # 按日全量覆盖分区数据

partition\_by = ["ds"] # 按ds（日期，如20250731）分区，适配每日同步

dynamic\_partition = true # 开启动态分区，自动匹配MySQL中的ds字段

file\_format = "orc" # 存储格式为ORC（相比textfile，更适合后续大规模查询）

orc\_compress = "SNAPPY" # 压缩格式为SNAPPY，与ODS层压缩策略一致

tbl\_properties = {{

"external.table.purge" = "true" # 外部表删除时同时清理HDFS数据

}}

fields = {column\_names} # 字段映射：与MySQL表字段完全一致，保留原始结构

}}}}

设计逻辑：

* 存储格式与压缩：采用 ORC+SNAPPY 组合，既满足 ODS 层对压缩效率（SNAPPY 低 CPU 消耗、高速度）的要求，又通过 ORC 的列式存储提升后续查询性能（相比 textfile 更适合大规模数据）；
* 分区策略：按 ds 动态分区，每日数据写入对应日期分区（如 ds=20250731），便于按时间维度管理和查询；
* 字段映射：fields 与 MySQL 表字段完全一致，确保原始数据结构无丢失，符合 ODS 层 “完整保留原始数据” 的核心目标。

## **二、数仓 DIM 层**

### **2.1 设计目标：支撑业务分析的 “原子化维度”**

以业务可复用性和分析便捷性为核心，将高频使用的 “实体属性”（如商品、用户、店铺）和 “分类标准”（如行为类型、页面类型）抽象为独立维度表，实现：

* 统一维度定义（如城市级别、年龄分组规则标准化），避免分析口径不一致；
* 减少数据冗余（通过关联维度表而非重复存储属性），降低维护成本；
* 支持多场景复用（同一商品维度可支撑销量分析、库存分析等）。

### **2.2 核心设计原则**

#### **2.2.1 维度建模导向**

基于 “业务实体” 和 “分类标准” 拆分维度，每个维度表聚焦单一主题：

* 实体维度：商品（dim\_products）、店铺（dim\_shops）、用户（dim\_users）—— 存储静态属性（如商品名称、店铺城市）和动态指标（如销量、粉丝数）；
* 分类维度：行为类型（dim\_behavior\_types）、页面类型（dim\_page\_types）、平台（dim\_platforms）—— 标准化业务分类（如 “点击”“购买”），支撑多维度下钻分析。

#### **2.2.2 缓慢变化维（SCD）处理**

采用按日分区快照策略（通过 ds 字段），每日覆盖更新维度数据：

* 适配电商业务中属性动态变化（如商品价格调整、店铺等级升级）；
* 保留历史快照，支持 “按过去某时间点” 分析（如 “20250731 时的商品价格分布”）。

#### **2.2.3 数据质量保障**

* 缺失值处理：用 nvl 填充 unknown（如缺失的分类名称），避免关联失效；
* 格式校验：通过正则匹配 + 默认值（如价格非法时置为 0.00），保证字段合规；
* 标准化转换：将业务码值（如 is\_hot='1'）转为语义明确的类型（is\_hot=true），提升易用性。

### **2.3 各维度表设计思路拆解**

#### **2.3.1 实体维度表（以商品 / 用户为例）**

* 商品维度（dim\_products）：整合商品全量属性，提取基础字段（product\_id、name），补充衍生属性（category\_name、sub\_category\_name），清洗转换数值字段（价格、库存），将布尔型字段从字符串转为 BOOLEAN 类型。
* 用户维度（dim\_users）：整合 “用户属性 + 行为标签”，计算 age\_group（如 18-25 岁）、city\_level，左连接行为表计算 is\_active（近 30 天是否活跃）、user\_age\_days（账龄）。

#### **2.3.2 分类维度表（以行为类型 / 平台为例）**

* 行为类型维度（dim\_behavior\_types）：标准化行为定义，提取 behavior\_type 去重值，补充语义描述（如 “用户点击商品 / 页面”）和价值权重（如购买 = 5、点击 = 1），支撑行为价值分析。
* 平台维度（dim\_platforms）：统一多端分类，将平台标识（如移动网页、PC）映射为规范名称（platform\_name）和设备归类（device\_type），支持跨平台对比。

### **2.4 落地关键：从 ODS 到 DIM 的加工逻辑**

* 数据源依赖：基于 ODS 层业务表（ods\_products、ods\_behaviors 等），通过 WHERE ds=' 当日 ' 提取最新数据；
* 加工方式：用 INSERT OVERWRITE 按日分区覆盖写入，确保每日维度数据为最新快照；
* 关联逻辑：实体维度表提取 ODS 字段 + 清洗转换，分类维度表通过 DISTINCT 提取分类值 + 规则映射。

### **2.5 价值：为后续分析 “搭积木”**

维度表是数仓的 “基础积木”：

* 下游 DWD 明细层可通过 product\_id 关联 dim\_products 补充商品属性；
* DWS 汇总层可基于维度表做分组统计（如 “按 city\_level 统计各城市销量”）；
* 支撑业务报表（如 “不同平台 + 不同行为类型的转化漏斗”），实现多维度交叉分析。

## **三、数仓 DWD 层**

### **3.1 文档概述**

本层围绕三张核心事实表展开：dwd\_user\_behavior\_detail（用户行为明细事实表）、dwd\_order\_detail（订单明细事实表）、dwd\_payment\_detail（支付明细事实表）。通过清洗 ODS 层数据、关联维度表，为 DWS 层汇总分析提供基础明细数据。

### **3.2 设计目标**

1. 数据整合与清洗：整合分散的行为、商品、用户数据，清洗脏数据（如格式不规范的时间、空值），统一格式；
2. 维度关联：关联各类维度表，补充丰富属性，提供全面分析视角；
3. 业务语义明确：通过字段命名和注释清晰表达业务含义，支撑用户行为分析、订单支付链路分析等场景。

### **3.3 表结构设计**

#### **3.3.1 dwd\_user\_behavior\_detail（用户行为明细事实表）**

| **基础信息** | **说明** |
| --- | --- |
| 表名 | dwd\_user\_behavior\_detail |
| 注释 | 存储用户在平台上的各类行为明细数据，用于分析用户行为路径、偏好等 |
| 存储格式 | ORC（高效压缩和查询，提升性能） |
| 分区字段 | ds（字符串类型，按日期分区） |

| **字段名** | **类型** | **注释说明** |
| --- | --- | --- |
| behavior\_id | STRING | 行为唯一 ID，通过 CONCAT (ob.user\_id, '', ob.behavior\_time, '', ob.behavior\_type) 生成 |
| user\_id | STRING | 用户 ID，关联 dim\_users，空值填充 unknown |
| product\_id | STRING | 商品 ID，关联 dim\_products，无商品关联时标记为 none |
| shop\_id | STRING | 店铺 ID，从商品维度表关联获取，无关联时填充 none |
| behavior\_type | STRING | 行为类型编码，取自 ods\_behaviors，关联 dim\_behavior\_types |
| behavior\_name | STRING | 行为类型名称，来自 dim\_behavior\_types，无关联时填充 unknown |
| behavior\_value | INT | 行为价值权重，来自 dim\_behavior\_types，无关联时置 0 |
| page\_type | STRING | 页面类型编码，取自 ods\_behaviors，关联 dim\_page\_types |
| page\_module | STRING | 页面所属模块，来自 dim\_page\_types，无关联时填充 unknown |
| platform | STRING | 平台编码，取自 ods\_behaviors，关联 dim\_platforms |
| device\_type | STRING | 设备类型，来自 dim\_platforms，无关联时填充 unknown |
| user\_city\_level | STRING | 用户城市级别，来自 dim\_users，无关联时填充 unknown |
| behavior\_time | TIMESTAMP | 行为发生时间，标准化为 yyyy-MM-dd HH:mm:ss 格式，无效值置 NULL |
| stay\_duration | INT | 页面停留时长（秒），字符串转 INT，无效值置 0 |
| is\_login | BOOLEAN | 是否登录状态，暂硬编码为 false |
| behavior\_extra | STRING | 额外信息（JSON 格式），无有效字段时填充 unknown |
| is\_valid | BOOLEAN | 是否有效行为，含 test 则无效，默认 true |

* 数据来源：ods\_behaviors，关联 dim\_users、dim\_products 等维度表；
* 处理逻辑：空值处理、标准化时间格式、关联维度表补充属性。

#### **3.3.2 dwd\_order\_detail（订单明细事实表）**

| **基础信息** | **说明** |
| --- | --- |
| 表名 | dwd\_order\_detail |
| 注释 | 基于用户购买行为生成，记录订单详细信息，用于订单分析、交易链路追踪等 |
| 存储格式 | ORC |
| 分区字段 | ds（字符串类型，按订单行为日期分区） |

| **字段名** | **类型** | **注释说明** |
| --- | --- | --- |
| order\_id | STRING | 订单唯一 ID，通过 CONCAT (ob.behavior\_id, '\_', ob.user\_id) 生成 |
| user\_id | STRING | 下单用户 ID，取自 ods\_behaviors，空值填充 unknown |
| product\_id | STRING | 商品 ID，取自 ods\_behaviors |
| shop\_id | STRING | 店铺 ID，关联 ods\_products，无关联时填充 none |
| order\_time | TIMESTAMP | 下单时间，标准化为 yyyy-MM-dd HH:mm:ss 格式，无效值置 NULL |
| product\_num | INT | 商品数量，默认置 1，需结合完善数据源补充 |
| product\_price | DECIMAL(10,2) | 商品单价，关联 ods\_products 获取 |
| total\_amount | DECIMAL(10,2) | 订单总金额，通过 product\_num \* product\_price 计算 |
| order\_status | STRING | 订单状态，标记为 “已购买”，可扩展为 “待支付”“已取消” 等 |
| is\_valid | BOOLEAN | 是否有效订单，含 test 则无效，默认 true |

* 数据来源：ods\_behaviors（筛选 behavior\_type=' 购买 '），关联 ods\_products；
* 处理逻辑：筛选购买行为、关联商品表计算金额、标准化时间格式。

#### **3.3.3 dwd\_payment\_detail（支付明细事实表）**

| **基础信息** | **说明** |
| --- | --- |
| 表名 | dwd\_payment\_detail |
| 注释 | 基于订单购买行为生成，记录支付详细信息，用于分析支付流程、交易成功情况等 |
| 存储格式 | ORC |
| 分区字段 | ds（字符串类型，按支付行为日期分区） |

| **字段名** | **类型** | **注释说明** |
| --- | --- | --- |
| payment\_id | STRING | 支付唯一 ID，通过 CONCAT (ob.behavior\_id, '', CONCAT(ob.behavior\_id, '', ob.user\_id)) 生成 |
| order\_id | STRING | 关联订单 ID，与 dwd\_order\_detail 的 order\_id 一致 |
| user\_id | STRING | 支付用户 ID，取自 ods\_behaviors |
| payment\_time | TIMESTAMP | 支付时间，标准化为 yyyy-MM-dd HH:mm:ss 格式，无效值置 NULL |
| payment\_amount | DECIMAL(10,2) | 支付金额，关联 dwd\_order\_detail 的 total\_amount |
| payment\_status | STRING | 支付状态，标记为 “成功”，可结合实际支付结果扩展 |
| is\_valid | BOOLEAN | 是否有效支付，含 test 则无效，默认 true |

* 数据来源：ods\_behaviors（筛选 behavior\_type=' 购买 '），关联 dwd\_order\_detail；
* 处理逻辑：筛选购买行为、关联订单表获取金额、标准化时间格式。

### **3.4 维度表关联说明**

1. 通用关联逻辑：通过字段匹配关联（如 user\_id 关联 dim\_users），结合分区字段 ds 确保数据时效性；
2. 维度表作用：补充用户、商品、行为类型等属性，支撑多维度分析。

### **3.5 数据流向与依赖关系**

* 数据流向：ODS 层→DWD 层（清洗、关联维度表）→DWS 层；
* 依赖关系：dwd\_order\_detail 依赖 ods\_behaviors 和 ods\_products，dwd\_payment\_detail 依赖 dwd\_order\_detail，执行顺序为 dwd\_user\_behavior\_detail→dwd\_order\_detail→dwd\_payment\_detail。

### **3.6 业务应用场景**

1. 用户行为分析：分析行为路径、偏好，优化页面布局和商品推荐；
2. 订单分析：分析订单数量、金额、商品构成，支持库存管理；
3. 支付分析：分析支付成功率、金额分布，优化支付流程；
4. 跨表关联分析：挖掘用户行为到订单、支付的完整链路，识别高价值行为模式。

## **四、数仓 DWS 层**

### **4.1 文档概述**

阐述两张核心宽表设计：dws\_page\_analysis\_wide（页面分析宽表）和 dws\_page\_trend\_30d（近 30 天页面趋势宽表）。基于 DWD 层明细数据聚合计算关键指标，为 ADS 层提供直接可用数据，支撑页面分析、用户行为趋势等场景。

### **4.2 设计整体目标**

1. 指标聚合：聚合用户行为、订单、支付等明细数据，生成页面访问、点击、交易转化等核心指标；
2. 维度关联：保留关键分析维度（如页面类型、平台、设备类型），满足多维度下钻分析；
3. 数据减负：预计算减少 ADS 层计算压力，提供直接可用指标；
4. 业务对齐：覆盖 “页面概览、点击分布、数据趋势、引导转化” 等核心场景。

### **4.3 具体表设计思路**

#### **4.3.1 dws\_page\_analysis\_wide（页面分析宽表）**

| **基础信息** | **说明** |
| --- | --- |
| 表名 | dws\_page\_analysis\_wide |
| 含义 | 聚合页面访问、点击、交易及商品引导等多维度指标 |
| 存储格式 | ORC（高效压缩，支持快速查询） |
| 分区字段 | ds（按日期分区，格式 yyyyMMdd，如 20250731） |

| **字段类别** | **字段名** | **含义说明** | **计算逻辑（来源）** |
| --- | --- | --- | --- |
| 维度字段 | page\_type | 页面类型（如店铺页、商品详情页） | 取自 dwd\_user\_behavior\_detail.page\_type |
| 维度字段 | page\_module | 页面所属模块 | 取自 dwd\_user\_behavior\_detail.page\_module |
| 维度字段 | platform | 平台（如 APP、H5） | 取自 dwd\_user\_behavior\_detail.platform |
| 维度字段 | device\_type | 设备类型（如手机、平板） | 取自 dwd\_user\_behavior\_detail.device\_type |
| 维度字段 | user\_city\_level | 用户城市级别（如一线、二线） | 取自 dwd\_user\_behavior\_detail.user\_city\_level |
| 流量指标 | page\_visit\_cnt | 页面访问次数（去重行为数） | COUNT(DISTINCT uv.behavior\_id) |
| 流量指标 | page\_uv | 页面访问人数（去重用户数） | COUNT(DISTINCT uv.user\_id) |
| 流量指标 | page\_click\_cnt | 页面点击次数 | COUNT (DISTINCT CASE WHEN uv.behavior\_type=' 点击 ' THEN uv.behavior\_id END) |
| 流量指标 | page\_click\_uv | 页面点击人数 | COUNT (DISTINCT CASE WHEN uv.behavior\_type=' 点击 ' THEN uv.user\_id END) |
| 板块指标 | page\_section\_click\_cnt | 页面板块点击次数 | 同 page\_click\_cnt（假设点击行为关联板块） |
| 板块指标 | page\_section\_click\_uv | 页面板块点击人数 | 同 page\_click\_uv（假设点击行为关联板块） |
| 板块指标 | page\_section\_pay\_amt | 板块引导支付金额 | SUM(CASE WHEN od.is\_valid THEN od.total\_amount ELSE 0 END) |
| 交易指标 | page\_order\_cnt | 页面下单订单数 | COUNT(DISTINCT od.order\_id) |
| 交易指标 | page\_order\_uv | 页面下单人数 | COUNT(DISTINCT od.user\_id) |
| 交易指标 | page\_pay\_cnt | 页面支付订单数 | COUNT(DISTINCT pd.payment\_id) |
| 交易指标 | page\_pay\_uv | 页面支付人数 | COUNT(DISTINCT pd.user\_id) |
| 交易指标 | page\_pay\_amt | 页面支付总金额 | SUM(pd.payment\_amount) |
| 交易指标 | page\_buy\_rate | 购买转化率（支付人数 / 访问人数） | ROUND (COUNT (DISTINCT pd.user\_id)/COUNT (DISTINCT uv.user\_id),2)（除数为 0 时置 0） |
| 引导指标 | page\_guide\_sku\_cnt | 页面引导商品数（去重商品 ID） | COUNT(DISTINCT uv.product\_id) |
| 引导指标 | page\_guide\_pay\_amt | 引导商品支付金额 | SUM(CASE WHEN pd.is\_valid THEN pd.payment\_amount ELSE 0 END) |
| 引导指标 | page\_guide\_buy\_uv | 引导商品购买人数 | COUNT(DISTINCT CASE WHEN pd.is\_valid THEN pd.user\_id END) |
| 时间辅助 | days\_ago | 统计天数（0 = 当天） | 固定为 0（单日内指标） |

* 数据来源：dwd\_user\_behavior\_detail（主表，别名 uv）、dwd\_order\_detail（od）、dwd\_payment\_detail（pd）；
* 筛选条件：uv.ds='20250731'，uv.behavior\_type IN (' 访问 ',' 点击 ',' 购买 ')；
* 关联逻辑：uv 与 od 通过 user\_id 关联（行为时间≥订单时间），od 与 pd 通过 order\_id 关联；
* 分组方式：按维度字段分组聚合。

#### **4.3.2 dws\_page\_trend\_30d（近 30 天页面趋势宽表）**

| **基础信息** | **说明** |
| --- | --- |
| 表名 | dws\_page\_trend\_30d |
| 含义 | 存储按日聚合的页面核心指标，支持时间趋势分析 |
| 存储格式 | ORC |
| 分区字段 | ds（按统计日期分区，格式 yyyyMMdd） |

| **字段类别** | **字段名** | **含义说明** | **计算逻辑（来源）** |
| --- | --- | --- | --- |
| 维度字段 | page\_type | 页面类型 | 取自 dwd\_user\_behavior\_detail.page\_type |
| 维度字段 | days\_ago | 距离当前天数（0 = 当天，1=1 天前...） | DATEDIFF('2025-07-31', uv.ds) |
| 趋势指标 | page\_uv | 当日页面访问人数 | COUNT(DISTINCT uv.user\_id) |
| 趋势指标 | page\_click\_uv | 当日页面点击人数 | COUNT (DISTINCT CASE WHEN uv.behavior\_type=' 点击 ' THEN uv.user\_id END) |
| 趋势指标 | page\_pay\_uv | 当日页面支付人数 | COUNT(DISTINCT CASE WHEN pd.is\_valid THEN pd.user\_id END) |
| 趋势指标 | page\_pay\_amt | 当日页面支付总金额 | SUM(CASE WHEN pd.is\_valid THEN pd.payment\_amount ELSE 0 END) |

* 数据来源：dwd\_user\_behavior\_detail（uv）、dwd\_payment\_detail（pd）；
* 筛选条件：uv.ds BETWEEN '20250701' AND '20250731'，uv.behavior\_type IN (' 访问 ',' 点击 ',' 购买 ')；
* 关联逻辑：uv 与 pd 通过 user\_id 和 ds 关联；
* 分组方式：按 page\_type 和 uv.ds 分组，聚合每日指标。

### **4.4 调度与依赖关系**

1. 执行顺序：依赖 DWD 层表，先执行 dws\_page\_analysis\_wide，再执行 dws\_page\_trend\_30d；
2. 调度周期：天级调度（每日凌晨处理前一日数据），趋势表可每日增量更新或每月全量重跑；
3. 参数配置：开启动态分区（SET hive.exec.dynamic.partition.mode=nonstrict），测试环境可开启本地模式加速处理。

### **4.5 扩展建议**

1. 维度扩展：增加 user\_type（新 / 老用户）、page\_section\_id（页面板块 ID）；
2. 指标扩展：增加 page\_avg\_stay\_duration（平均停留时长）、click\_to\_buy\_rate（点击到购买转化率）；
3. 性能优化：用窗口函数替代日期遍历，对大表分桶（CLUSTERED BY (page\_type) INTO 8 BUCKETS）。

## **五、数仓 ADS 层**

### **5.1 业务背景与需求**

围绕 “页面行为分析” 场景，支撑类似 “生意参谋 - 流量 - 页面分析” 的需求，解决：

* 页面运营监控：统计各页面类型的访问、跳出、点击等指标，分析流量质量；
* 商品详情页洞察：分析商品访问深度（停留时长）与转化效率；
* 页面板块优化：分析页面内不同板块的点击与转化，指导布局迭代；
* 数据趋势跟踪：监控近 30 天指标变化，分析运营活动对流量的影响。

### **5.2 表设计整体架构**

#### **5.2.1 设计原则**

* 分层定位：基于 ODS 层行为数据（ods\_behaviors）和商品数据（ods\_products），加工为面向分析的汇总指标；
* 分区策略：按 ds（日期，yyyyMMdd）分区，聚焦 ds=20250731，支持历史回溯；
* 存储优化：采用 ORC 格式，利用列式存储和压缩特性提升查询效率。

### **5.3 各表详细设计**

#### **5.3.1 页面分析汇总表（ads\_page\_analysis）**

| **业务定位** | **全局统计不同页面类型、访问平台、流量来源的核心运营指标，对比流量质量** |
| --- | --- |

| **字段名** | **含义** | **计算逻辑** |
| --- | --- | --- |
| page\_type | 页面类型（如店铺首页） | 取自 ods\_behaviors.page\_type |
| platform | 访问平台（如移动 APP） | 取自 ods\_behaviors.platform |
| referrer | 流量来源（如搜索引擎） | 取自 ods\_behaviors.referrer |
| stat\_date | 统计日期 | 固定为 20250731 |
| visit\_count | 访问次数（非跳出） | COUNT (CASE WHEN behavior\_type!=' 跳出 ' THEN 1 END) |
| bounce\_count | 跳出次数 | COUNT (CASE WHEN behavior\_type=' 跳出 ' THEN 1 END) |
| click\_count | 点击次数 | COUNT (CASE WHEN behavior\_type=' 点击 ' THEN 1 END) |
| add\_cart\_count | 加购次数 | COUNT (CASE WHEN behavior\_type=' 加购 ' THEN 1 END) |
| purchase\_count | 购买次数 | COUNT (CASE WHEN behavior\_type=' 购买 ' THEN 1 END) |
| avg\_stay\_duration | 平均停留时长（秒） | AVG(CAST(stay\_duration AS INT)) |
| visitor\_num | 访客数（去重） | COUNT(DISTINCT user\_id) |

* 数据加工流程：来源 ods\_behaviors，筛选 ds='20250731'，按 page\_type、platform、referrer 分组聚合。

#### **5.3.2 商品详情页访问表（ads\_product\_detail\_page）**

| **业务定位** | **聚焦商品详情页，分析单个商品的访问深度和转化效率** |
| --- | --- |

| **字段名** | **含义** | **计算逻辑** |
| --- | --- | --- |
| shop\_id | 店铺 ID | 取自 ods\_behaviors.shop\_id |
| product\_id | 商品 ID | 取自 ods\_behaviors.product\_id |
| stat\_date | 统计日期 | 固定为 20250731 |
| visit\_count | 详情页访问次数（非跳出） | COUNT (CASE WHEN page\_type=' 商品详情页 ' AND behavior\_type!=' 跳出 ' THEN 1 END) |
| visitor\_num | 详情页访客数（去重） | COUNT (DISTINCT CASE WHEN page\_type=' 商品详情页 ' AND behavior\_type!=' 跳出 ' THEN user\_id END) |
| avg\_stay\_duration | 平均停留时长（秒） | AVG (CAST (stay\_duration AS INT))（仅计算商品详情页） |
| purchase\_count | 购买次数 | COUNT (CASE WHEN page\_type=' 商品详情页 ' AND behavior\_type=' 购买 ' THEN 1 END) |
| purchase\_amount | 购买金额 | 关联 ods\_products.price，结合 detail 中购买数量：CAST (price AS DECIMAL)\*CAST (split (split (detail, ' 购买数量:')[1], ' ')[0] AS INT) |

* 数据加工流程：ods\_behaviors 左连接 ods\_products，筛选 page\_type=' 商品详情页 ' 且 ds='20250731'，按 shop\_id、product\_id 分组聚合。

#### **5.3.3 页面点击分布表（ads\_page\_click\_distribution）**

| **业务定位** | **分析页面内不同板块的点击量、点击人数及引导支付金额，指导布局优化** |
| --- | --- |

| **字段名** | **含义** | **计算逻辑** |
| --- | --- | --- |
| page\_type | 页面类型 | 取自 ods\_behaviors.page\_type |
| detail | 板块名称（如首页推荐） | 取自 ods\_behaviors.detail（记录点击 / 购买的板块信息） |
| stat\_date | 统计日期 | 固定为 20250731 |
| click\_count | 板块点击次数 | COUNT (CASE WHEN behavior\_type=' 点击 ' THEN 1 END) |
| click\_user\_num | 板块点击人数（去重） | COUNT (DISTINCT CASE WHEN behavior\_type=' 点击 ' THEN user\_id END) |
| purchase\_amount | 板块引导支付金额 | 关联 ods\_products.price，结合 detail 中购买数量（同商品详情页逻辑） |

* 数据加工流程：ods\_behaviors 左连接 ods\_products，筛选 behavior\_type IN (' 点击 ',' 购买 ') 且 ds='20250731'，按 page\_type、detail 分组聚合。

#### **5.3.4 页面数据趋势表（ads\_page\_trend）**

| **业务定位** | **监控近 30 天（2025-07-02 至 2025-07-31）页面指标的日度变化趋势** |
| --- | --- |

| **字段名** | **含义** | **计算逻辑** |
| --- | --- | --- |
| page\_type | 页面类型 | 取自 ods\_behaviors.page\_type |
| stat\_date | 统计日期（日度） | to\_date(from\_unixtime(unix\_timestamp(behavior\_time, 'yyyy-MM-dd HH:mm:ss'))) |
| visitor\_num | 日度访客数（去重） | COUNT(DISTINCT user\_id) |
| click\_user\_num | 日度点击人数（去重） | COUNT (DISTINCT CASE WHEN behavior\_type=' 点击 ' THEN user\_id END) |
| visit\_count | 日度访问次数（非跳出） | COUNT (CASE WHEN behavior\_type!=' 跳出 ' THEN 1 END) |
| bounce\_count | 日度跳出次数 | COUNT (CASE WHEN behavior\_type=' 跳出 ' THEN 1 END) |
| avg\_stay\_duration | 日度平均停留时长（秒） | AVG(CAST(stay\_duration AS INT)) |

* 数据加工流程：来源 ods\_behaviors，时间范围 stat\_date BETWEEN '2025-07-02' AND '2025-07-31'，按 page\_type、stat\_date 分组聚合。

### **5.4 技术实现要点**

1. 时间处理函数：将 behavior\_time（yyyy-MM-dd HH:mm:ss）转换为日期：

sql

to\_date(from\_unixtime(unix\_timestamp(behavior\_time, 'yyyy-MM-dd HH:mm:ss')))

1. 字符串解析（购买数量提取）：

sql

split(split(detail, '购买数量:')[1], ' ')[0]

1. 多表关联与别名：

sql

FROM ods\_behaviors b LEFT JOIN ods\_products p ON b.product\_id = p.product\_id

1. 性能优化：通过 ds 过滤减少数据扫描，采用 ORC 格式提升聚合查询效率。

### **5.5 数据应用场景**

| **表名** | **典型分析场景** |
| --- | --- |
| ads\_page\_analysis | 对比 “APP vs PC”“搜索流量 vs 分享流量” 的转化效率，优化流量投放策略 |
| ads\_product\_detail\_page | 筛选 “访问多但购买少” 的商品，分析详情页缺陷；复制 “高转化商品” 的成功经验 |
| ads\_page\_click\_distribution | 识别 “高点击但低转化” 的板块，优化内容或位置；放大 “高转化板块” 的露出 |
| ads\_page\_trend | 分析 “活动期间” 的流量波动，评估活动效果，指导后续活动时间规划 |