# 我去恶趣味

# 基于Hadoop平台的数据迁移中间件系统研究与实现

## 摘要

## Abstract

## 第1章 绪论

### 1.1 研究背景与意义

为什么是基于Hadoop平台的？为什么要进行数据迁移？进行数据迁移的数据不能为非常重要的数据，因为在迁移过程中有数据丢失。

### 1.2 国内外研究现状

1.Hadoop的发展现状（Hadoop生态系统及Hadoop主要技术）

2.HBase的发展现状（分析传统关系型数据库与NoSQL数据库之间的区别及特点）

3.数据迁移中间件发展现状（Sqoop、DBInputformat，kellte等）

### 1.3 研究内容

### 1.4 论文组织结构

### 1.5 本章小结

## 第2章 系统相关技术介绍

### 2.1 Hadoop简介

### 2.2 MapReduce关键技术

### 2.2.1 MapReduce并行计算

### 2.2.2 MapReduce容错处理

### 2.2.3 MapReduce作业调度

### 2.2.1 MapReduce编程模型

### 2.2.2 MapReduce处理过程

### 2.2.3 MapReduce容错机制

### 2.3 Hbase

### 2.3.1 CAP原理

### 2.3.2数据模型

### 2.3.3 数据版本控制

### 2.3.4 数据查询

### 2.3.5 基于MapReduce的数据处理

### 2.3.6 HBase与关系型数据库比较

### 2.3.1 HBase技术特点

### 2.3.2 HBase体系结构

### 2.3.3 HBase数据模型

### 2.3.4 数据查询流程分析

### 2.3.1 HBase架构

### 2.3.2 HBase数据模型

### 2.3.3 HBase API接口访问方式

### 2.4 迁移调度工具或算法

(1)XML相关技术

(2)数据迁移方法：分为同构数据库之间的迁移和异构数据库之间的迁移。

### 2.5 本章小结

## 第3章 数据迁移调度算法研究

## 第4章 数据迁移中间件系统设计

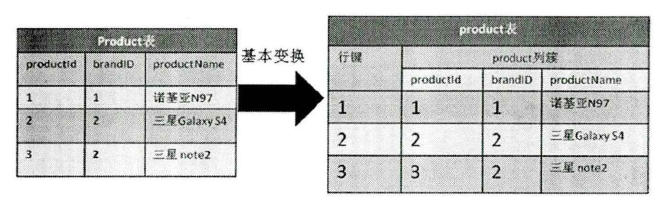
### 4.1 系统设计需求

### 4.2 系统架构设计

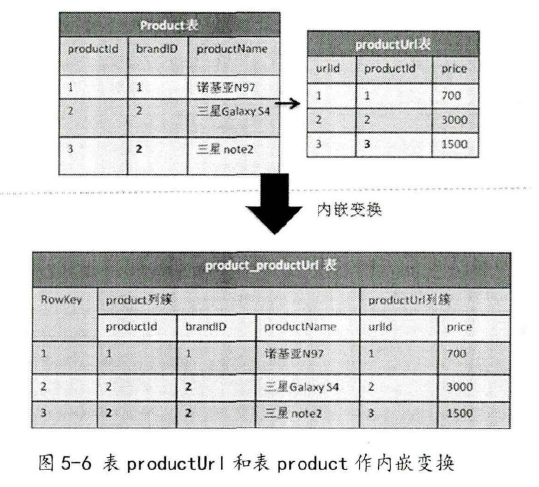
## 第5章 系统方案设计

### 5.1 数据表模式转换

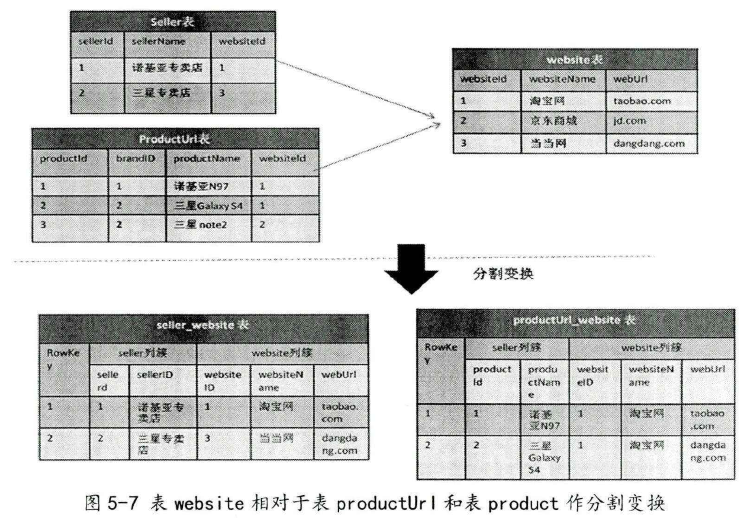
1. 基本变换：直接迁移，不做任何变化



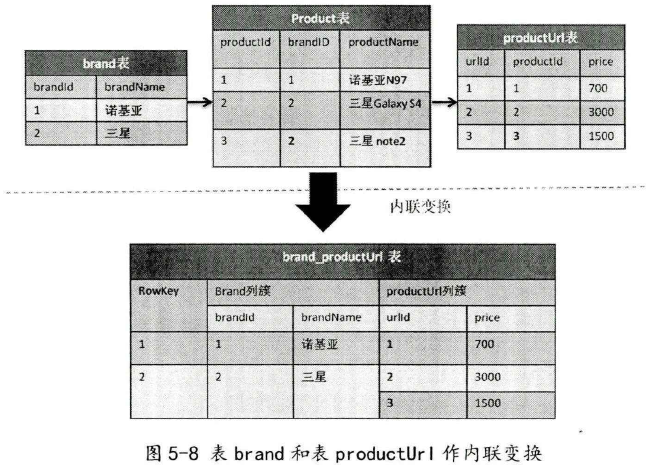
1. 内**嵌**变换：数据表B的主键是A表中的外键情况，1对1，1对N或N对N。将B表中与A表相关的列加入到主表A中实现联表查询。



1. 分割变换：数据表A与表B存在一对一、一对多或多对多的关系，A表和B表中均有指向数据表C的引用，即表C的主键分别在A和B中作为外键存在。将表C中与A相关的列加入到A中，C中与B相关的列加入到B中。根据其他需求决定是否删除C。



1. 内**联**变化：A与B之间存在1对1，1对N或N对N关系，B与C之间也存在1对1，1对N或N对N关系。即A中有指向B的引用，B中有指向C的引用。A中有外键是B的主键，B中有外键是C的主键。



### 5.2 数据迁移方法

#### 5.2.1 HBase API

#### 5.2.2 MapReduce

## 第6章 系统实现与测试

### 6.1 系统功能模块

#### 6.1.1 XML管理器

RDBMS到XML表模式

#### 6.1.2 源数据库管理器

XML到HDFS

#### 6.1.3 表模式映射管理模块

HDFS到HBase表结构

#### 6.1.4 数据类型映射模块

#### 6.1.5 数据导入模块

### 6.2大数据平台搭建

### 6.3 数据迁移测试

### 6.4 迁移性能分析

6.4.1 数据完整性

6.4.2 数据查询性能(与Sqoop或Kellte比较)

## 第7章 总结与展望