项目报告

1 数据存储及优化对比

1.1 关系型数据库

1.1.1 适用查询范围

关系型数据库在**高效管理与查询结构化数据**时表现良好，可以应对需要复杂查询与严格处理的场景，适用于好确保数据一致性与完整性的应用。

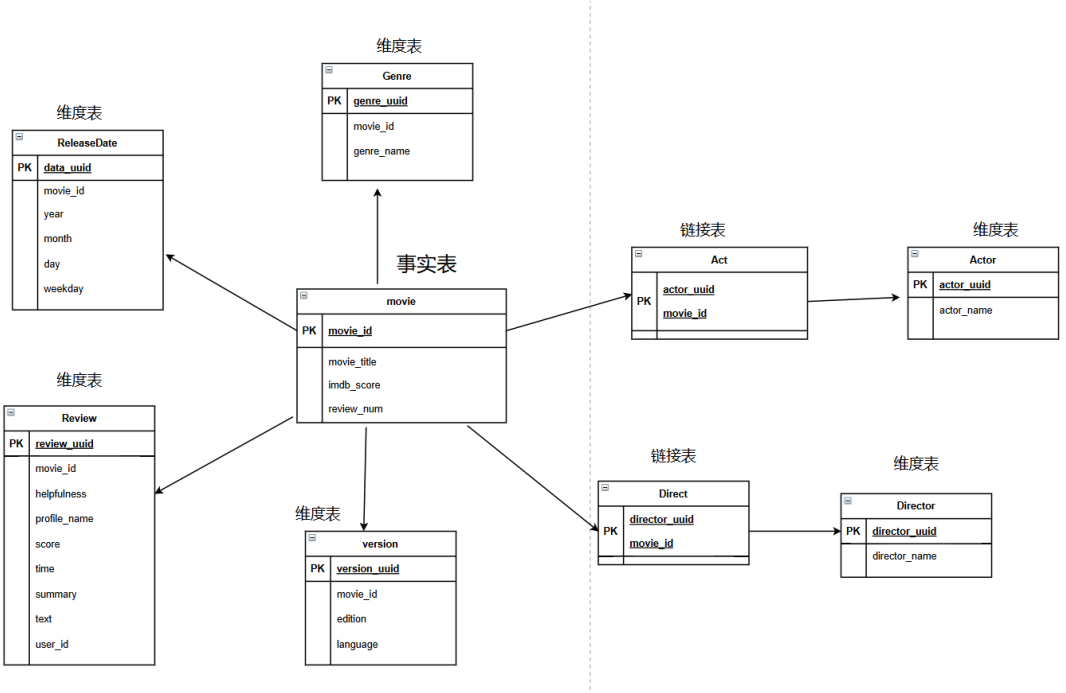
- 关系型数据库所管理的数据通常以表格形式存在，这类数据具有固定结构和明确字段定义，高度结构化。每一行代表一个记录，每列代表一种属性，这种清晰的数据组织方式使得关系型数据库能够高效地存储和检索信息。但是使用时需要定义好表与表中字段，数据结构固定不易调整。本次项目中由于需要进行存储优化，需要经常调整数据库结构，也因此需要更新或重新导入相关数据，在处理一些数据库表时较为麻烦。
- 关系型数据库在执行复杂的SQL查询方面表现出色，尤其是在涉及多个表之间的联接操作时。此外，关系型数据库还允许编写嵌套的子查询来实现复杂的业务逻辑，能够通过聚合函数对数据进行统计分析。配合上 `GROUP BY` 语句，还可以实现对数据的分组统计。这些功能可以较为灵活简单地完成数据分析等工作。
- 关系型数据库还提供了 `ACID` 事务特性，保证了数据的完整性与一致性。在进行复杂查询或事务处理时，可以确保数据结构的清晰性与可维护性。适合于金融交易、电子商务等对数据准确性和安全性有极高要求的领域，确保了即使在复杂的业务环境中也能保持数据的完整性和可靠性。

1.1.2 存储优化工作

在本次项目中，我们通过对物理模型的反规范化、预聚合以及设置冗余存储等操作对存储进行了优化。

1. 星型模型

本次项目采用星型模型结构，在数据存储上我们通过引入一定的数据冗余，换取了更好的查询性能。通过围绕事实表 `Movie` 建立一系列的维度表，将 `Movie` 的属性拆分到不同维度表中，简化了查询的复杂度，优化了查询的性能。



2. 字段设置

在 `Review` 表中，我们将记录评论时间信息的 `time` 时间戳字段的类型设计为了 `bigint`，而不是常见的 `datetime` 类型。这一决策主要基于以下考量：

- **平台兼容性**：采用 `bigint` 类型确保了数据能在不同数据库系统与编程语言之间保持一致性与互操作性。
- **空间效率**：相较于 `datetime` 类型，`bigint` 所需的存储空间更小，这对于规模较大的评论表 `Review` 而言尤为重要，能够有效降低存储成本。

而在 `Movie` 表中，我们将评分相关的 `imdb_score` 字段设计为了 `float` 类型，这样的选择主要出于如下考量：

- **节省资源**：相较与 `double` 类型，`float` 所占用的存储空间更小，有助于优化存储资源的利用。
- **精度适配**：考虑到电影评分并不需要较高的数值精度，`float` 类型足以满足需求，也可以避免不必要的复杂性。

3. 冗余存储

在电影表中，我们添加了 `review_num` 评论数量字段，通过添加这一冗余存储，避免在查询最受欢迎的演员组合时与评论表进行联接操作，从而提升查询性能。

此外我们将电影的发布时间单独抽象为一张表，将电影的发布时间拆分为多个字段，而不是直接存储电影发布时间的时间戳，这样在分别根据年、月、日查询电影时，可以减少对数据的处理，简化了操作流程，也提高了查询的性能。

4. 建立索引

除了主键和外键等数据库自动建立的索引外，对常用查询字段建立单列索引和组合索引，以加速查询操作。

- **单列索引**
  - 演员表：为 `演员姓名` 建立单列索引。
  - 导演表：为 `导演姓名` 建立单列索引。
  - 风格表：为 `电影风格` 建立单列索引。
  - 时间表：为 `年份` 建立单列索引。
  - 版本表：为 `电影语言` 和 `电影格式` 分别建立单列索引。
- **组合索引**
  - 时间表：
    - 年和月建立组合索引。
    - 年和季度建立组合索引。
    - 工作日建立单列索引。

5. 建立视图

- **演员合作视图**
  建立演员与演员之间合作的视图，字段包括：
  - 演员1的 `ID`
  - 演员2的 `ID`
  - 合作电影的 `ID`
- **演员与导演合作视图**
  建立演员与导演之间合作的视图，字段包括：

- 演员的 `ID`
- 导演的 `ID`
- 合作电影的 `ID`

6. PreAggregation 表设计

为了进一步提升查询性能，我们设计了一些 `PreAggregation` 表，将耗时的计算过程放在服务器负载较小时进行。

在 `MovieGenreStatistics` 表中，我们提前汇总了各电影类别的电影总数，以加速查询某类别的电影统计数据。

字段名	数据类型	说明
`genre`	VARCHAR(100)	电影类别
`total_movies`	INT	该类别电影的总数

在 `MovieActorDirectorCollaborations` 表中，我们提前计算了演员与导演之间的合作次数，以便快速查询哪些演员和导演经常合作。

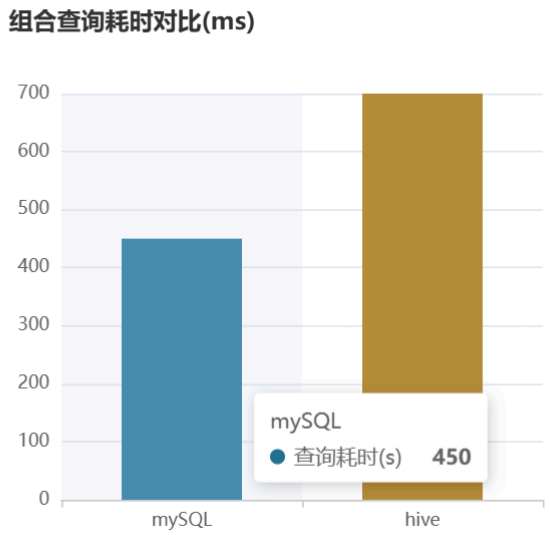
字段名	数据类型	说明
`actor_id`	BINARY(16) (PK)	演员ID
`actor_name`	VARCHAR(511)	演员姓名
`director_id`	BINARY(16) (PK)	导演ID
`director_name`	VARCHAR(511)	导演姓名
`collaborations`	INT	演员与导演合作的次数

对于如上两张表，我们采用 MySQL 的事件进行更新，在每天凌晨时将执行更新任务，以确保不影响平时业务的执行。

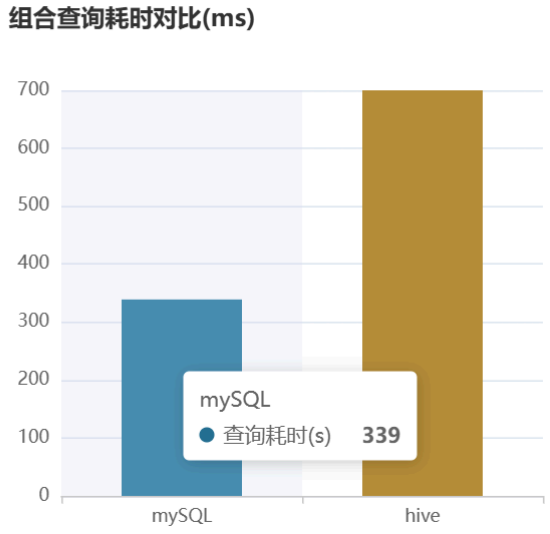
1.1.3 优化前后对比

选择了一个与类别相关的比较示例，上图为优化前为450ms，下图为优化后339ms

查询结果　耗时对比



查询结果　耗时对比



1.2 分布式数据库

1.2.1 适用查询范围

分布式数据库专注于处理大规模数据和高并发访问，适用于需要高可用性和横向扩展的场景，如大数据处理、分布式日志存储和互联网应用。

主要功能优势：

- **扩展性强**：支持水平扩展，允许数据分布在多个节点上并在多个节点并行处理查询，从而提高系统的吞吐量和性能。特别是在库表分布于不同节点时，分布式数据库能高效地处理分布式事务，确保数据的一致性和完整性。
- **负载均衡**：通过负载均衡机制动态分配查询压力，确保每个节点都在其容量范围内运行，同时优化查询执行计划以高效利用分布式环境的计算资源。
- **高效索引**：支持在不同节点上创建和维护索引，有效减少数据传输，提高查询效率。这对于分布式查询尤为重要。

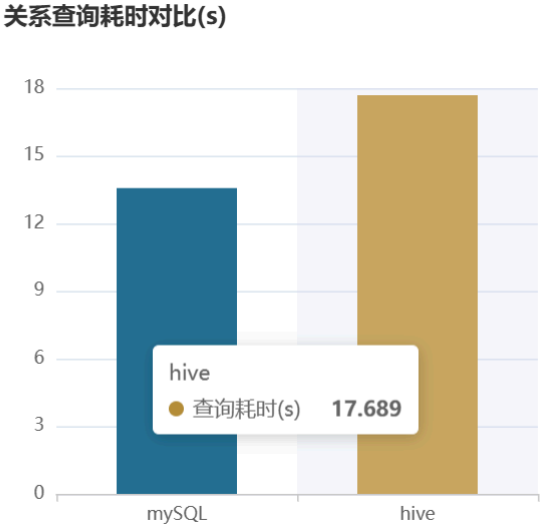
- **容错性高**：具备备容错能力，在节点故障或网络中断的情况下，系统能够继续运行。这包括数据备份和故障转移等机制，保证系统的高可用性。
- **容量弹性**：支持在不同物理设备间存储数据，通过增加节点解决容量瓶颈，实现系统的横向扩展。

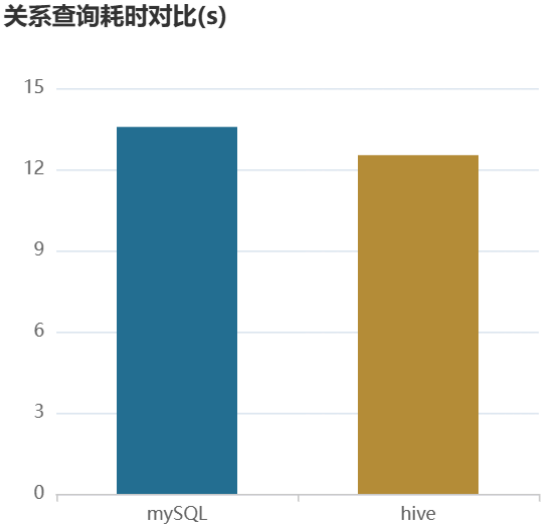
1.2.2 存储优化工作

- 将关系型数据库中的视图转为实际表，不仅复用了原有代码，还减少了演员与演员、演员与导演之间的连接操作，从而加速了查询。
- 采用与关系型数据库相同的表结构，通过引入冗余字段避免了复杂的连接操作，并设置了相同的索引以优化查询性能。
- 通过建立外部表来管理存储在外部的数据，数据库仅保存元数据，而实际数据存储在文件系统中。这种方法允许在数据存储位置直接进行查询，减少了数据的传输，提高了查询效率，避免了数据移动可能带来的性能瓶颈。

1.2.3 优化前后对比

若不将视图建立为实际的表，则在分布式数据库中会出现更多的join操作，拖慢了查询的速度，在将视图建立为实际表后，速度加快了约5秒，下图是加快后的结果，查询时间为12.523s.





1.3 图数据库

1.3.1 适用查询范围

图数据库专注于处理**图形化数据**，主要用于节点和边之间关系的高效处理，能够灵活支持复杂关系型数据的查询，尤其在面对大量、复杂、动态的网状数据时表现出色。同时，图数据库具备实时增删改查、可视化、高可用和备份恢复功能，并严格遵循 ACID 事务特性。

主要功能优势：

- **复杂关系查询**：图数据库能够快速处理复杂节点间的关系，如查找邻居节点或共同连接点。在项目中，当查询演员与导演之间的关系时，关系型数据库通常逻辑复杂且效率较低，而图数据库能以简单的查询语句快速、高效地获取节点之间的关系。
- **路径发现**：支持查找任意两个节点之间的最短路径，并集成多种图算法，能够有效分析图中关键节点、社区结构，应用于路径规划、网络分析及依赖关系分析等场景。
- **递归查询**：图数据库支持递归查询，便于追踪节点到自身的路径，这在存储和处理复杂关系数据时非常高效，传统关系型数据库难以轻松实现这一功能，尤其在查找自循环关系时，图数据库更具优势。

1.3.2 存储优化工作

1. 存储字段选择

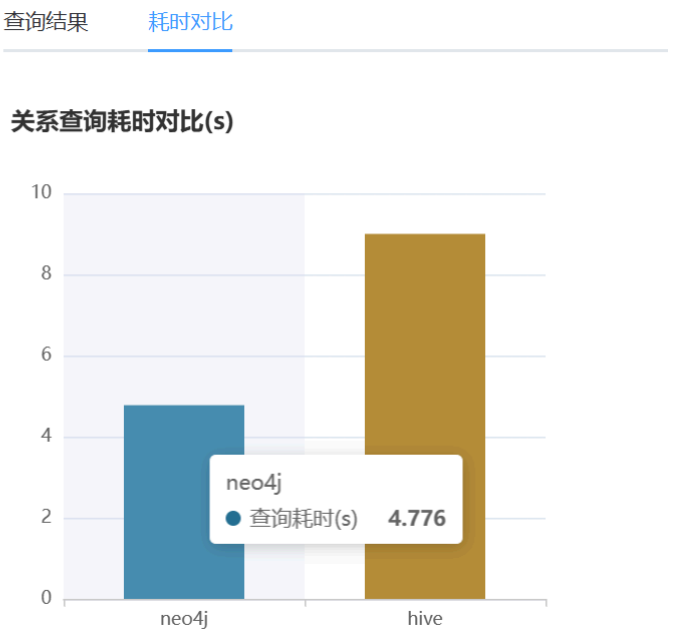
- 在数据库中只存储**4类节点**（演员、导演、电影、电影类型）以及**3种关系**（导演与电影间的执导关系、演员与电影之间的参演关系、电影类型的拥有关系），便于查询时快速进行计算。
- 在数据库中电影节点中存储电影相关的评论数，便于统计。

2. 建立索引

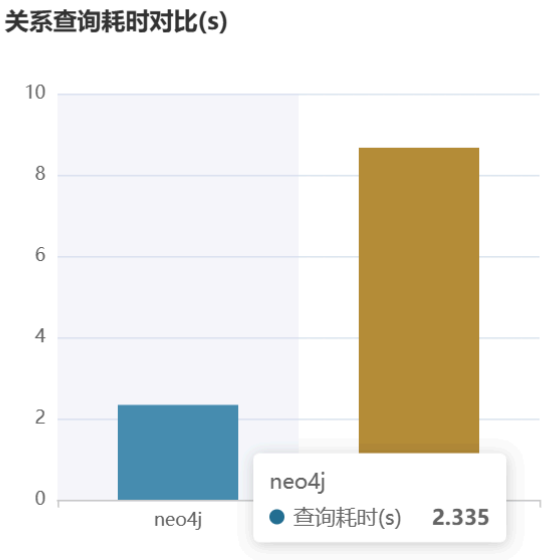
- 对查询时4类节点中将会用到的字段建立索引，分别是演员名、演员名、电影名、电影风格

1.3.3 优化前后对比

建立图数据库索引可以加速图数据库关系的查询，上图是添加索引之前，查询演员之间的关系耗时为4.776秒，下图是添加索引之后，查询演员之间的关系耗时2.335秒。



查询结果　耗时对比



2 数据质量保证

2.1 保证数据质量方法

2.1.1 数据爬取

1. 在爬取数据的过程中：
    - 为保证数据爬取的效率与便捷性，我们使用 `aiohttp` 这一 HTTP 库请求获取网页数据，并采用多线程并行爬取多个页面。
    - 为避免反爬操作，我们使用 `fake_useragent` 生成随机请求头，并引入重试机制，将失败的请求加入重试队列，确保能够成功获取数据。
    - 通过随机函数生成 `cookie`，避免同一 `cookie` 请求频率过高触发反爬机制。
    - 小组成员之间采用多种方式爬取数据，互相补充以确保完整的网页数据，避免信息遗漏。
2. 筛选出 `Movies & TV` 和 `Prime Video` 标签的数据，后续筛掉 `TV` 数据。
3. 使用 `lxml` 的 `xpath` 功能从网页中提取数据，对不同类型的网页使用不同 `xpath` 标签尝试提取数据，以保证能够从网页中正确提取出数据。

2.1.2 数据清洗

1. 将所有收集到的数据合并到一个文件中，并按 ASIN 去重，避免数据重复或遗漏。
2. 去除明显错误的数据，如演员、导演、语言等多项字段中缺失超过 2/3 以上的数据。
3. 去除人名中多余的引号等符号。
4. 去除电影数据集中的无关信息，包括电影名称中的引号与版本信息等，并将多个空格合并为一个以规范化数据。
5. 根据处理得到的电影 ASIN，筛选出评论中与电影相关的评论。
6. 对于缺失上映时间的电影，使用评论中该电影的最早评论时间作为上映时间。
7. 合并同一名字的不同表示方式，建立映射关系后，根据映射关系替换姓名，确保同一名字使用相同的表示方式。

8. 最后进行电影信息的去重，我们将电影名称相同且导演相同的认定为同一部电影，数据集中仅保留相同电影中的第一项。保留电影到最终版本的映射关系，以建立数据血缘关系。

2.2 数据质量影响因素

1. **网页结构及关键信息命名差异**
   网页结构或命名的差异可能导致爬取数据不完整或出现错误，直接影响数据质量。
2. **单一数据源的局限性**
   数据主要来源于单一平台，因平台信息本身不完整且未完全使用第三方数据源补充（如豆瓣和TMDB），受到接口请求限制或数据重复爬取的影响，导致数据质量受损。
3. **数据库设计缺陷**
   数据库设计可能存在不足，导致数据在录入时无法有效校验，出现数据重复或不完整的情况。
4. **反爬机制影响**
   Amazon 的反爬机制对爬虫工作构成阻碍，尽管已解决验证码问题，但部分页面的 detail 数据被隐藏，导致需要重新爬取，影响数据完整性。
5. **网络不稳定性**
   网络连接的不稳定或中断可能导致爬取过程中的部分数据丢失。
6. **数据模型和格式差异**
   不同数据源的数据模型和格式不一致，需要进行转换和验证，但转换过程中可能进一步影响数据质量。
7. **数据处理环节问题**
   数据清洗、转换和装载过程中，可能因逻辑混乱而引发数据丢失、重复或失真等问题。此外，在更新和重处理数据时，也可能因人为偏差造成数据质量问题。

3 数据血缘使用场景

数据血缘是一种记录和追踪数据流动路径的技术，用于了解数据在整个信息系统中的传递、转换和处理过程，广泛应用于数据治理、质量管理和风险控制等领域。

3.1 场景概述

1. **数据质量管理**
   数据血缘可用于监控和评估数据质量，通过追踪数据流动路径，快速定位数据质量问题的根本原因和发生位置，便于实时修复与改进。
2. **合规性和审计**
   在遵循法规和行业标准的场景下，数据血缘能够追踪敏感数据的处理路径，确保数据处理的合规性。同时，审计人员可借助数据血缘验证数据准确性和完整性，提升数据可信度。
3. **数据治理**
   数据血缘是数据治理中的关键部分，可帮助识别数据生成、修改和使用的全流程。通过清晰的数据追踪，支持数据治理政策、标准化规则的制定和执行。
4. **故障排查和性能优化**
   当数据处理链路中发生故障或性能下降时，数据血缘能够协助快速识别问题根源，通过追踪数据流路径，优化处理节点或排除系统故障，提高整体性能。
5. **变更管理**
   当数据架构或处理逻辑需要调整时，数据血缘可评估变更的影响范围，确保变更过程中的风险最小化。

6. **数据资产管理**
   数据血缘有助于全面了解数据资产的流转路径，支持资源分配优化。它可以帮助识别哪些数据资源被频繁使用，哪些资源被闲置，以实现更高效的数据管理。
7. **数据安全**
   数据血缘能追踪敏感数据的流动路径，帮助明确敏感数据的暴露风险并加强安全防护。同时，可追踪数据泄露事件的源头并快速采取应对措施。
8. **业务影响分析**
   在业务变更或系统升级时，数据血缘能够分析变更对下游业务流程的影响，支持变更方案的制定与实施，从而最大限度降低业务风险。

3.2 项目应用

在本次项目中，我们为各项数据分别建立一张表记录血缘关系。在电影合并过程中，我们将合并到一起的电影的 ASIN 记录到一张表中，以便确认最终数据的来源。

B000U788UM	
B002AEHB92	B00455KQ3C,B004802YWG
B00007KFM2	
B000IOM0XI	
B0000CEUY7	
B000KHX762	
B002AWM10S	
B00004D2YN	
B00005AXLJ	B00000K31Q,B002S8AGZG
B002T38C9C	
B0083HXK86	
B000ZMEOKU	
B00006SFNE	
B000067J0N	
B0033D2C16	
B0095D45I6	
B0001P1BT8	
B00018D4OK	
B0006Z2NTK	
6304287097	B000MV9018,B00005QBZB
B001EUSYAI	
B009AJB0YM	B009AJB43O
B00006RHSH	
B001CW7ZTY	
B0007WQH62	
B001O0TMIG	
B001G6RRB2	
B001BKV9TE	B0018SRV3W
B002LII6IW	
B001DZA49I	
B00009AV7Q	B00009AV7T
B000A7JKR2	

以电影导演为例，我们使用一张单独的映射表记录 movie 表中的导演的来源。

a colder kind of death	brad turner	B002ZRYL1A,B002ZS68BA
a convenient truth: urban	giovanni vaz del bell	B007EMSZFY
a conversation about race	craig bodeker	B0032K6HWA,B0032KBOMI
a cool dry place	john m. smith	B001NEP5IE,B001NEI5ZO
a corner of her eye: katri	konrad aderer	B001055S4I,B001059H04
a cottage on dartmoor	anthony asquith	B005DWAFPA,B004D18T7I
a country coyote goes holl	winston hibbler	B0096HIXBG
a crude awakening: the oil	basil gelpke	B003EJ7WCS
a crude awakening: the oil	basil gelpke, ray mcc	B003EJ7WAK
a cry from the streets	lewis gilbert	B004J2VBJO
a cry in the dark	fred schepisi	B001NWPDK6,B00009B0NL
a damsel in distress	george c. stevens	6304212038,B001T8E6AW
a damsel in distress	george stevens	B001T87C5I
a dangerous man	keoni waxman	B0036EHTMO
a dangerous man - lawrence	chris menaul	B0024UNHSG
a dangerous method	david cronenberg	B007OUINLC
a dangerous place	jerry p. jacobs	B002A5WNIU,B002A5WNG2
a darker reality	chris kazmier	B0076R5LVS
a date with judy	richard hope	B004VS9EPO,B004VS9JCC
a day at the races	sam wood	B001NPF5IS,B001NPJ0Y8
a day in a life	jean mercier	B000YDS690,B000YDU6CA
a day in flatbush brooklyn	greg p	B005IY4IJM,B005IY4J78
a day in the life	sticky fingaz	B002I51Z18
a day in the lifeless	john gloster	B007LGM5L8,B007LGM79I
a delicate balance	don richardson	B00028HC5G
a dennis the menace christ	rob oliver	B000YEB5JW

4 小组贡献度

学号	姓名	百分比
2253721	陈茁扬	25%
2254198	段子涛	25%
2251528	王浩然	25%
2253206	韩明洋	25%