**《数据库系统及其应用实践》课程实验报告**

Database Systems & Applications  
Final Report:Movie Assistant

组号：\_\_\_\_\_\_\_18\_\_\_\_\_\_\_\_ 组员：\_\_张国帅、马鼎、王菁华\_\_\_\_

完成日期：\_\_\_\_2023/6/25\_\_\_\_

**# 目录**

- [[一、项目背景](#一、项目背景)]

- [[二、数据集的基本情况](#二、数据集的基本情况)]

- [[三、应用的基本功能](#三、应用的基本功能)]

- [[四、数据库设计](#四、数据库设计)]

- [[五、查询的设计和优化](#五、查询的设计和优化)]

- [[六、用户界面设计](#六、用户界面设计)]

- [[七、开发环境](#七、开发环境)]

- [[八、应用部署和测试](#八、应用部署和测试)]

**一、项目背景**

随着互联网的发展，电影作为一种重要的文化娱乐形式，越来越受到人们的喜爱和关注。然而，面对海量的电影资源，用户往往难以快速找到自己感兴趣的电影，或者了解电影的相关信息，如演员、剧情、评价等。因此，有必要提供一个方便、快捷、准确的电影查询服务，让用户能够按照自己的喜好和需求筛选和浏览电影。

本项目旨在开发一个基于IMDb的电影查询网页——Movie Assistant，利用一个权威和全面的电影数据库，为用户提供按照评分、类型、标题等条件筛选电影的功能。同时，本项目也将展示每部电影的详细信息，如演员、制作人员、评分等，让用户能够更深入地了解电影。

本项目预期的结果是一个简洁、美观、易用的电影查询和管理网页，它能够满足用户对电影查询和浏览的需求，提高用户的体验和满意度。

**二、数据集的基本情况**

本项目采用IMDb提供的数据集。IMDb数据集是一个包含了IMDb网站上的电影、电视剧、播客、视频游戏等的信息的数据集，它提供了以下几种子集：

- title.akas.tsv.gz - 包含了标题的ID、排序、本地化标题、地区、语言、类型、属性和是否为原始标题等信息

- title.basics.tsv.gz - 包含了标题的ID、类型、主要标题、原始标题、是否为成人内容、开始年份、结束年份、运行时间和类型等信息

- title.crew.tsv.gz - 包含了标题的ID和导演和编剧的ID等信息

- title.episode.tsv.gz - 包含了电视剧集的ID、父级电视剧的ID、季数和集数等信息

- title.principals.tsv.gz - 包含了标题的ID和主要演员/工作人员的ID、排序、类别、职位和角色等信息

- title.ratings.tsv.gz - 包含了标题的ID和IMDb评分和投票数等信息

- name.basics.tsv.gz - 包含了人名的ID、主要名称、出生年份、死亡年份和已知职业等信息

这些数据集都是以gzip压缩的TSV格式文件存储的，每个文件的第一行包含了每一列的描述，每个文件都使用UTF-8字符集编码。如果某个字段缺失或为空，就用‘\N’来表示。这些数据集可以从<https://datasets.imdbws.com/>下载，每天更新。

**三、应用的基本功能**

1.用户可以查询和浏览电影的信息，包括评分、类型、标题、演员、制作人员等。

2.用户可以通过输入自己感兴趣的条件，如评分、类型、标题等，来筛选出符合自己喜好的电影。

3.管理员用户可以增加、删除每部电影的详细信息，对数据库进行调整。

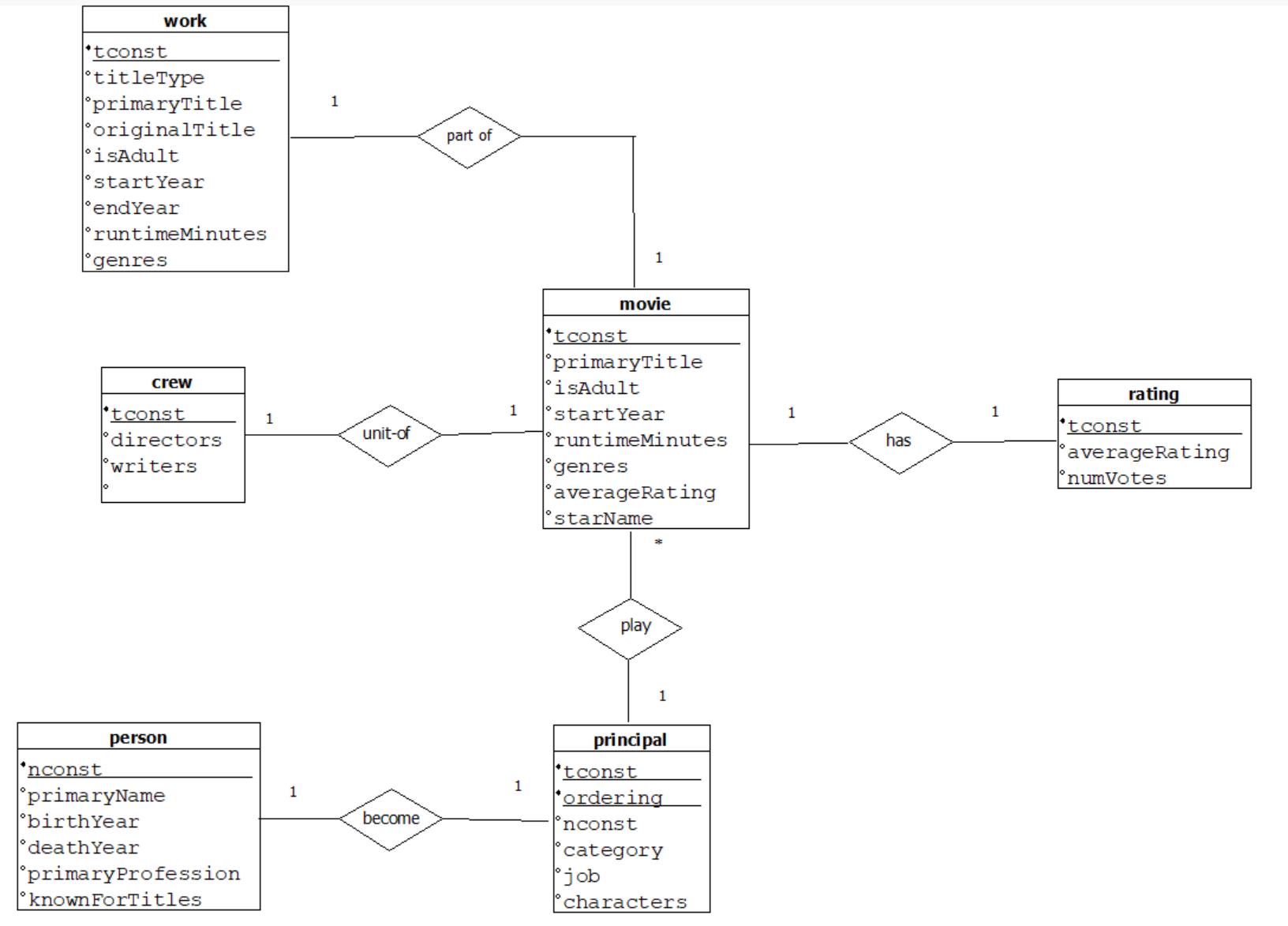
4.用户登录和注册功能

5.分页功能

6.收藏电影

**四、数据库设计**

E-R图设计：



数据库主要表的设计：

//原数据库中的作品表，包括电影、短剧等多种类型

**work**(tconst,titleType,primaryTitle,originalTitle,isAdult,startYear,endYear,runtimeMinutes,genres)

PK:tconst

//电影表，包含部分电影信息，是主要使用的表

**movie**(tconst,primaryTitle,isAdult,startYear,runtimeMinutes,genres,averageRating,starName)

PK:tconst FK:tconst->work

//工作人员表，储存导演和作者信息

**crew**(tconst,directors,writers)

PK:tconst FK:tconst->movie

//评分表，储存评分信息

**rating**(tconst,averageRating,numVotes)

PK:tconst FK:tconst->movie

//演员表，储存一部作品的多个演员信息

**principal**(tconst,ordering,nconst,category,job,characters)

PK:(tconst,ordering) FK:tconst->movie,nconst->person

//人员表，储存人员信息

**person**(nconst,primaryName,birthYear,deathYear,primaryProfession,knowForTitles)

PK:nconst

//（额外功能表）用户表，储存用户信息

**users**(uid,nickname,password,tconst,msg) PK:uid

//（额外功能表）收藏表

**star**(tconst)

**五、查询的设计和优化**

**1. 基础功能查询**

//打开数据库，显示某一页（页数，初始值为1）评分最高的前‘每页数量(100)’条电影

select tconst,primaryTitle,starName,genres,startYear,averageRating from movie order by averageRating DESC,startYear DESC limit 100

LIMIT (页数-1)\*每页数量,页数\*每页数量-1;

//查询基础信息，按演员、类型、标题等查询电影可以基于该查询修改

SELECT primaryTitle AS Title , startYear AS Year, genres AS Type, averageRating AS Rating, numVotes AS Vote\_num

FROM movie NATURAL JOIN rating

//电影中的演员

SELECT primaryTitle AS Title, primaryName AS actor, characters AS acting\_character

FROM person NATURAL JOIN principal NATURAL JOIN movie

WHERE characters <> '\N';

**2.复杂查询测试**

/\*Enoch J. Rector导演1897年的作品

样本数据集只有这一个符合条件的数据\*/

SELECT primarytitle AS Title, startyear AS Year, genres AS Type, averageRating AS Rating

FROM movie NATURAL JOIN crew NATURAL JOIN person

WHERE primaryname = 'Enoch J. Rector' AND startYear = '1897';

/\*Blanche Bayliss的知名作品\*/

SELECT primarytitle AS Title, startyear AS Year, genres AS Type, averageRating AS Rating,starname

FROM movie

WHERE tconst IN

(SELECT person.knownfortitles FROM person

WHERE person.primaryname = 'Blanche Bayliss')

/\*Blanche Bayliss主演的Romance类型的电影\*/;

SELECT primarytitle AS Title, startyear AS Year, genres AS Type, averageRating AS Rating

FROM movie

WHERE starname = 'Blanche Bayliss' AND genres = 'Romance';

**3.查询优化**

（1）使用更合适的查询语句，如添加过滤条件（WHERE）、限制返回行数（LIMIT）、去除不必要的字段（SELECT）、去除不必要的连接（JOIN）、去除不必要的分组（GROUP BY）、去除不必要的排序（ORDER BY）等，以减少查询范围和结果集大小。

（2）为表中的关键字段创建索引，以加速扫描和连接操作；为表中的统计字段（如评分、开始时间等）创建索引，以加速排序和聚合操作；示例：

CREATE INDEX "movie\_i1" ON "public"."movie" USING btree (

"tconst",

"startyear",

"averagerating"

);

CREATE INDEX "person\_i1" ON "public"."person" USING btree (

"nconst"

);

CREATE INDEX "crew\_i1" ON "public"."crew" USING btree (

"tconst"

);

CREATE INDEX "principal\_i3" ON "public"."principal" USING btree (

"tconst",

"ordering"

);

**六、用户界面设计**

（2023/6/25）

1. 模拟登录页



1. 首页



1. 添加页



1. 详情页



**七、开发环境**

数据库环境：docker虚拟环境下的的opengauss数据库

前端： html、css和JavaScript技术

后端：采用nodejs+express框架构建本地服务器

编写和调试代码：使用Eclipse、Idea、Vscode作为集成开发环境（IDE）

**八、应用部署和测试**

**1.创建数据库，并导入数据**

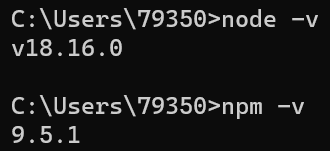
按照DDL.sql建表之后，你可以：

a.参照README中的方法，尝试导入全量数据库

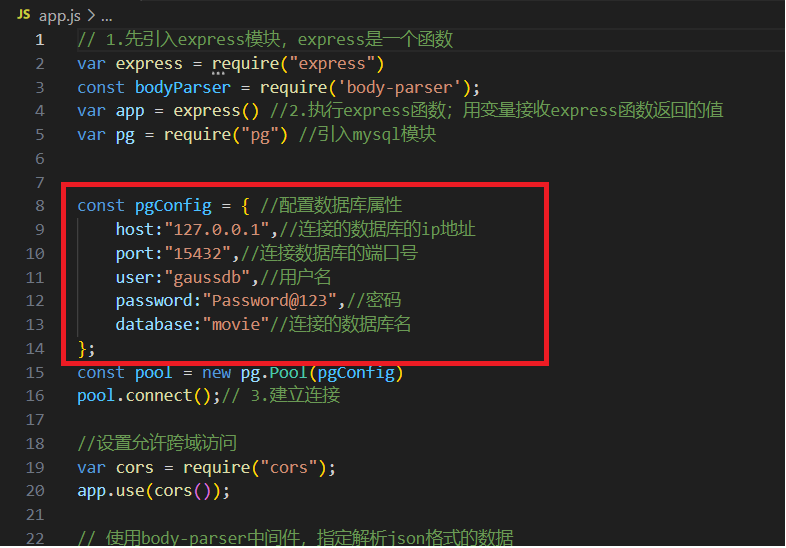
b.使用我们提供的小型数据集

**2. 安装nodejs**

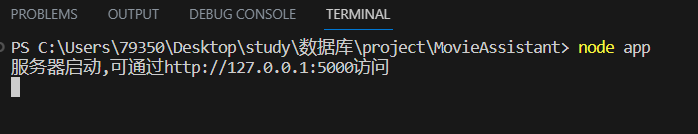
在nodejs官网下载：<https://nodejs.org/> 或 <http://www.nodejs.com.cn/>

跟着向导完成安装后，你可以检查是否成功安装node和npm****

**3.在MovieAssistant/app.js中配置数据库信息**

****

**4.在MovieAssistant目录下运行app.js**

****

**5.正常访问页面,各种功能测试正常**

