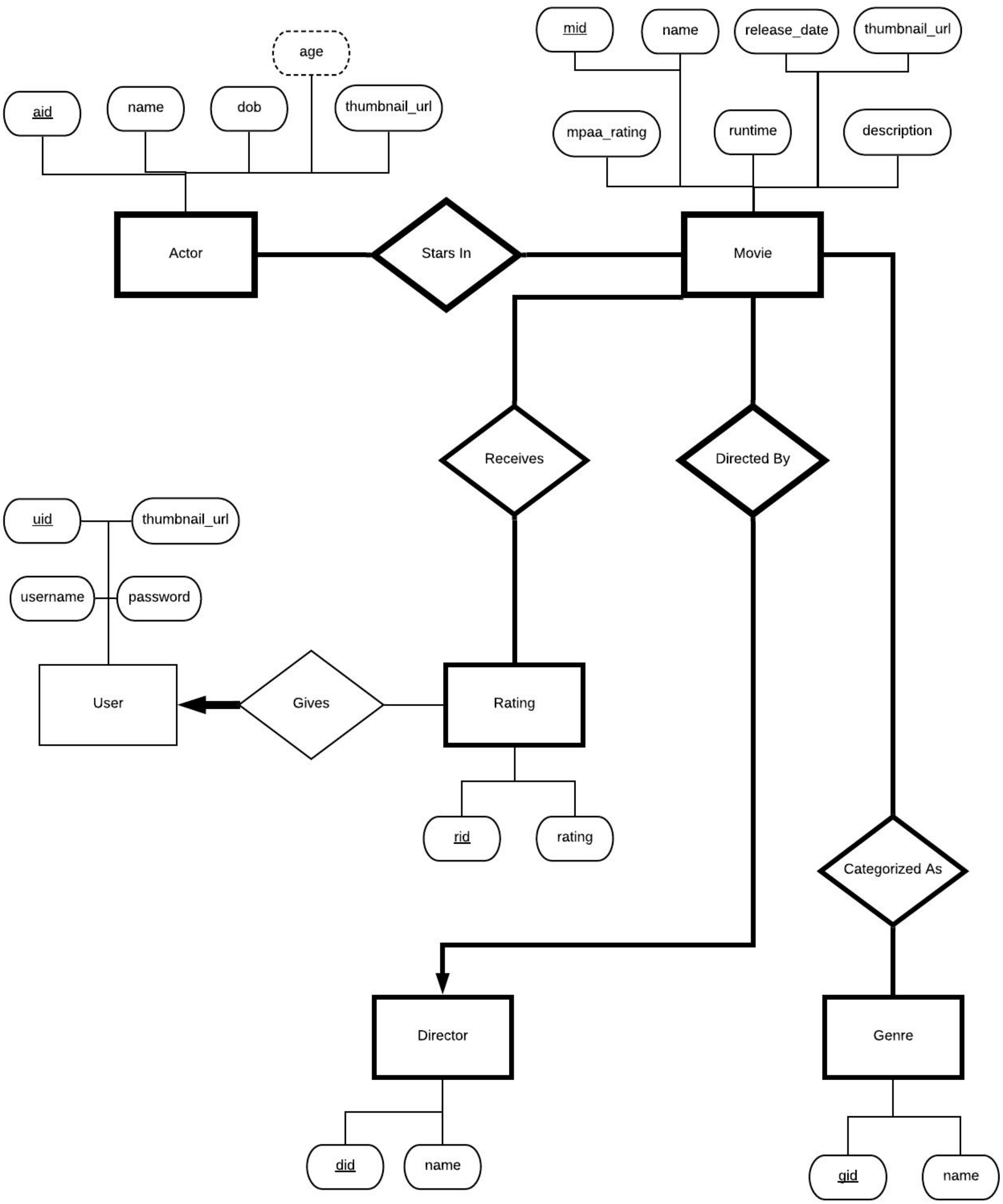
Movie Rank
Milestone 2
Database Model

CSC 675-03
Team 02

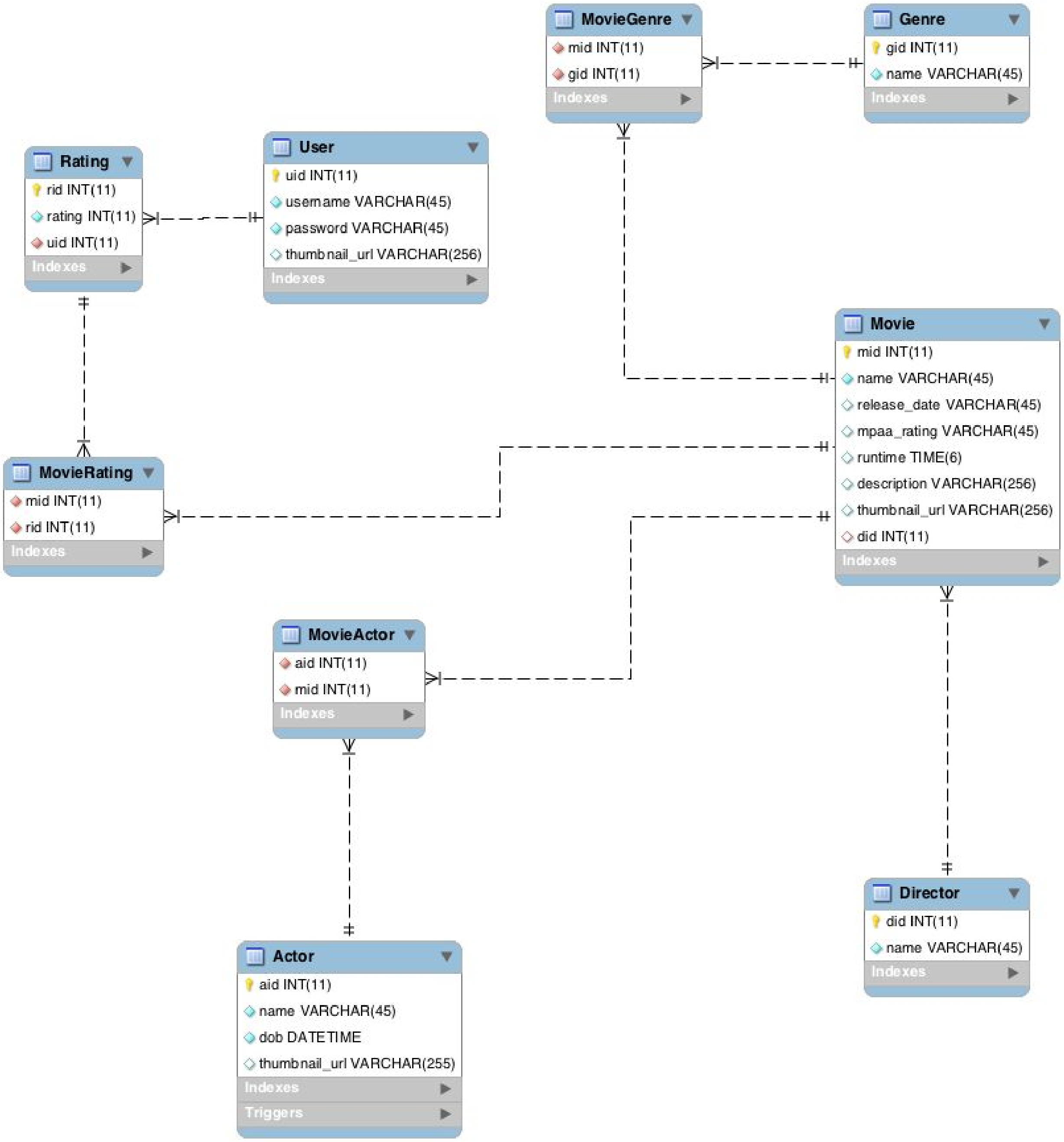
Zac Henney
Ahmad Rangeen
Bingcheng Wu
Sandesh Basnet
Tharmika Thiyagarajah
Tuan Le

Table of Contents

ERD


age
aid
name
dob
thumbnail_url
mid
name
release_date
thumbnail_url
mpaa_rating
runtime
description
Actor
Stars In
Movie
Receives
Directed By
uid
thumbnail_url
username
password
User
Gives
Rating
rid
rating
Categorized As
Director
Genre
did
name
gid
name

Database Model



SQL Code

```
-- MySQL Script generated by MySQL Workbench
-- Sat Apr 20 16:58:26 2019
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

-- -----------------------------------------------------
-- Schema MovieRank
-- -----------------------------------------------------

-- -----------------------------------------------------
-- Schema MovieRank
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `MovieRank` DEFAULT CHARACTER SET utf8 ;
USE `MovieRank` ;

-- -----------------------------------------------------
-- Table `MovieRank`.`Actor`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`Actor` (
  `aid` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `dob` DATETIME NOT NULL,
  `thumbnail_url` VARCHAR(255) NULL,
  PRIMARY KEY (`aid`),
  UNIQUE INDEX `aid_UNIQUE` (`aid` ASC))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`Director`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`Director` (
  `did` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`did`),
  UNIQUE INDEX `did_UNIQUE` (`did` ASC))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`Movie`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`Movie` (
  `mid` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `release_date` VARCHAR(45) NULL,
  `mpaa_rating` VARCHAR(45) NULL,
  `runtime` TIME(6) NULL,
  `description` VARCHAR(256) NULL,
  `thumbnail_url` VARCHAR(256) NULL,
  `did` INT NULL,
  PRIMARY KEY (`mid`),
  UNIQUE INDEX `mid_UNIQUE` (`mid` ASC),
  INDEX `did_idx` (`did` ASC),
  CONSTRAINT `did`
    FOREIGN KEY (`did`)
    REFERENCES `MovieRank`.`Director` (`did`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`MovieActor`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`MovieActor` (
  `aid` INT NOT NULL,
  `mid` INT NOT NULL,
  INDEX `aid_idx` (`aid` ASC),
  INDEX `mid_idx` (`mid` ASC),
  CONSTRAINT `aid`
    FOREIGN KEY (`aid`)
    REFERENCES `MovieRank`.`Actor` (`aid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `mvid`
    FOREIGN KEY (`mid`)
    REFERENCES `MovieRank`.`Movie` (`mid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`Genre`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`Genre` (
  `gid` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`gid`),
  UNIQUE INDEX `gid_UNIQUE` (`gid` ASC))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`User`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`User` (
  `uid` INT NOT NULL AUTO_INCREMENT,
  `username` VARCHAR(45) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `thumbnail_url` VARCHAR(256) NULL,
  PRIMARY KEY (`uid`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC),
  UNIQUE INDEX `uid_UNIQUE` (`uid` ASC))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`Rating`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`Rating` (
  `rid` INT NOT NULL AUTO_INCREMENT,
  `rating` INT NOT NULL,
  `uid` INT NOT NULL,
  PRIMARY KEY (`rid`),
  UNIQUE INDEX `rid_UNIQUE` (`rid` ASC),
  INDEX `uid_idx` (`uid` ASC),
  CONSTRAINT `uid`
    FOREIGN KEY (`uid`)
    REFERENCES `MovieRank`.`User` (`uid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`MovieRating`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`MovieRating` (
  `mid` INT NOT NULL,
  `rid` INT NOT NULL,
  INDEX `rid_idx` (`rid` ASC),
  INDEX `mid_idx` (`mid` ASC),
  CONSTRAINT `rmid`
    FOREIGN KEY (`mid`)
    REFERENCES `MovieRank`.`Movie` (`mid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `rid`
    FOREIGN KEY (`rid`)
    REFERENCES `MovieRank`.`Rating` (`rid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `MovieRank`.`MovieGenre`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `MovieRank`.`MovieGenre` (
  `mid` INT NOT NULL,
  `gid` INT NOT NULL,
  INDEX `mid_idx` (`mid` ASC),
  INDEX `gid_idx` (`gid` ASC),
  CONSTRAINT `mid`
    FOREIGN KEY (`mid`)
    REFERENCES `MovieRank`.`Movie` (`mid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE,
  CONSTRAINT `gid`
```

```
    FOREIGN KEY (`gid`)
    REFERENCES `MovieRank`.`Genre` (`gid`)
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;

USE `MovieRank`;

DELIMITER $$
USE `MovieRank`$$
CREATE DEFINER = CURRENT_USER TRIGGER `MovieRank`.`Actor_AFTER_UPDATE` AFTER UPDATE ON `Actor` FOR
EACH ROW
BEGIN

END
$$


DELIMITER ;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Testing

Test #	Test Type	Entity 1	Relationship	Type Rel	Entity 2	Pass / Fail
1	DELETE	Actor				Pass
2	DELETE	Director				Pass
3	DELETE	Genre				Pass
4	DELETE	Movie				Pass
5	DELETE	Rating				Pass
6	DELETE	User				Pass
7	SELECT	Actor	MovieActor	M:M	Movie	Pass
8	SELECT	Genre	MovieGenre	M:M	Movie	Pass
9	SELECT	Movie		M:1	Director	Pass
10	SELECT	Director		1:M	Movie	Pass