**Navid Vadsariya**

**136884186**

**Lab 02 - Relational Algebra**

**(Selection, Projection, Join)**

Consider the following relations:

**MOVIES** (id:int, title:varchar(35), year:int, director:int) Id is the key

**ACTORS** (id:int, name:varchar(20), lastname:varchar(30)) Id is the key.

**CASTINGS** (movieid:int, actorid:int) (movieid and actorid) is the key

**DIRECTORS**(id:int, name:varchar(20), lastname:varchar(30)) id is the key

1. What is the result of the following queries?

It displays the director having ID 100.

* 1. **Πtitle, year(**)

It displays only the title and year of director with moive Id of 100.

* 1. **Πtitle, year,name,lastname(**

It displays title, name and lastname of movie and director having common ID.

1. Using the same schema as above, write each of the following queries as a relational algebra expression:
2. List all actors.

**σ name(ACTORS)**

1. List the name and the year of all movies.

**Πname,year(movies)**

1. Find all movies produced in 2010.

**σyear=2010(movies)**

1. List all actors in the Avatar movie.

**Πactor.name,actor.lastname(σmovie.title=Avatar(moviesmovies.id=casting.moviesid(castingcasting.actorid=actor.idactor)))**

1. Find the movie title, year, and the director name for movies produced in 2019.
2. Find movie title and the movie director’s name and last name for all movies that the actor with ID = 200 plays a character in them.

**Πmovie.title,directors.name,directors.lastname(σmovieid=(σactorid=200(casting)))**

1. Find all actors played in movies produced after 2010 and before 2018.

**σ actors=(Πactorid(Πid = >2010 && <2018(movies)))**