Homework 8 : Query Plans

We have a movie database with the following tables and columns:

- `Movies(MovieID, MovieName, MovieGenre)`
- `DirectedMovie(MovieID, DirectorID)`
- `Directors(DirectorID, DirectorName, DirectorBirthDate)`
- `Actors(ActorID, ActorName, ActorBirthdate)`
- `PlayedIn(MovieID, ActorID)`

The semantics of tables and columns should be obvious from the names. Assume that each ID (for movies, directors, and actors) consumes 8 bytes of storage, each name (for movies, directors, and actors) consumes 100 bytes of storage, and each birth date (for actors and directors) consumes 16 bytes. The movie genre is represented by 8 bytes.
The table cardinalities are:

- `Movies` - 200,000
- `DirectedMovie` - 200,000
- `Directors` - 2000
- `PlayedIn` - 2,000,000
- `Actors` - 400,000

Assume that each director directed the same number of movies and that each actor played in the same number of movies. 5 % of the directors are at the same time actors themselves. 10 % of all movies are action movies.
We want to determine the optimal plan to answer the following query:

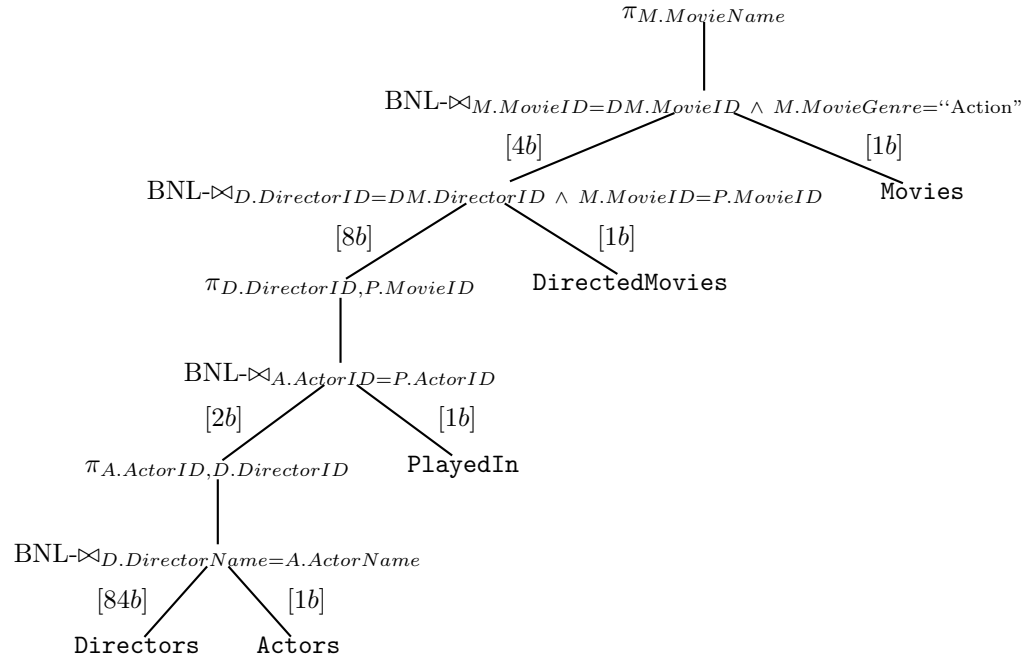
```
SELECT MovieName FROM Movies, DirectedMovie, Directors, Actors, PlayedIn
WHERE Movies.MovieGenre = "Action" AND
Movies.MovieID = DirectedMovie.MovieID AND
Directors.DirectorID = DirectedMovie.DirectorID AND
Movies.MovieID = PlayedIn.MovieID AND
Actors.ActorID = PlayedIn.ActorID AND
Actor.ActorName = Director.DirectorName
```

This query asks for the names of all action movies where the director was at the same time an actor. This query is executed on a system with around 100 buffer pages. The page size is 1024 bytes. The `Actors` and `PlayedIn` tables are both stored in the order of the `ActorID` column. The operators are to be selected from the list of operators discussed in the lecture (engine.pdf slide set).
Your answer should contain the following:

- a graphical representation of the query plan that you believe to be optimal for the given query in terms of execution time. Use the graphical notation for query plans that was introduced in the lecture, specifying join order, join operator names, position of selection and projection operations, and number of buffer pages. Use rules of thumb to decide how many buffer pages to assign to each operation; it is not required to find the optimal buffer page assignment by mathematical analysis.
- for the optimal plan, calculate the cardinalities and byte sizes of all intermediate results and the cost of that plan by drawing a table as seen in the lecture. Assume for the cost calculations that one seek operation takes 10 msecs and one page I/O operation takes 0.1 msecs.
- a text justifying thoroughly why you think that the given plan is optimal: justify the overall shape of the plan tree, the join order, the selection of operators, and the positioning of selection and projection operations. Also show in your text that you considered reasonable alternatives, for instance alternative join operators or alternative join orders, and justify why the alternatives are worse than the plan that you finally selected.

Solution: In general, we push select and project operation down as much as possible to reduce the size of intermediate results and the associated processing cost. We start by considering left-deep plans as efficient plans often have that shape (already because they allow pipelining). No indices are defined which excludes the use of index nested loop joins. Hash joins require writing data to disc. We can however find a join order that allows to keep all intermediate results stored in memory; this makes the choice of BNL joins preferable as they only require to scan the outer relation

once if the inner relation fits into memory. Our join order is chosen to minimize the size of intermediate results. This is the optimal plan:



Node	tp size	#tps/pg	#tps	#pgs	I/O pgs	#seeks
D	124	8	2,000	250	250	3
A	124	8	400,000	50,000	-	-
$D \bowtie A$	248	4	100	25	3 * 50,000	3
$\pi(DA)$	16	64	100	2	0	0
P	16	64	2,000,000	31,250	-	-
$DA \bowtie P$	32	32	500	16	1 * 31,250	1
$\pi(DAP)$	16	64	500	8	0	0
Dm	16	64	200,000	3,125	0	0
$DAP \bowtie Dm$	32	32	500	16	1 * 3,125	1
$\pi(DAPDm)$	8	128	500	4	0	0
M	116	8	200,000	25,000	-	-
$DAPDm \bowtie M$	32	32	50	7	1 * 25,000	1
$\pi(DAPDmM)$	100	10	50	4	0	0
(total)					**209,625**	**9**