Problem

Given a database with the relations below, express the following queries using *SQL, relational algebra*, *relational calculus* and QBE.

```
        Film (Title, Director, Year, Company)
        Actor (SSN, Name, Birth_Date)
        Plays (SSN, Title, Earnings)
```

Q1: Retrieve the names of actors who earned more than 1000 for a single movie in 1997.

Q2: Retrieve the names of actors who played in all movies directed by John Woo.

<table>
<tr><th>Query</th><th>SQL</th><th>algebra</th><th>(tuple) calculus</th><th>QBE</th></tr>
<tr>
<td><b>Q1</b></td>
<td>Select A.name
From Actor A, Plays P, Film F
Where A.SSN=P.SSN and
P.Title=F.Title and F.year=1997
and P.Earnings>1000</td>
<td>$\pi_{sname}(\sigma_{Film.Year=1997 \text{ and } Plays.Earning > 1000}(\text{Actors JOIN}_{SSN}\text{Plays JOIN}_{Title}\text{ Film}))$</td>
<td>$\{X \,/\, \exists A \in Actors\ \exists P \in Plays\ \exists F \in Films\ (X.name = A.name \wedge A.SSN=P.SSN \wedge P.Title=F.Title \wedge F.Year = 1997 \wedge P.Earnings>1000\}$</td>
<td>
<table>
<tr><td>Actors</td><td>SSN</td><td>Name</td><td>BDate</td></tr>
<tr><td></td><td>_S</td><td>P.</td><td></td></tr>
</table>
<table>
<tr><td>Plays</td><td>SSN</td><td>Title</td><td>Earnings</td></tr>
<tr><td></td><td>_S</td><td>_T</td><td>>1000</td></tr>
</table>
<table>
<tr><td>Films</td><td>Title</td><td>Director</td><td>Year</td></tr>
<tr><td></td><td>_T</td><td></td><td>1997</td></tr>
</table>
</td>
</tr>
<tr>
<td><b>Q2</b></td>
<td>Select A.name
From Actor A
Where not exists ((
Select F.Title
From Films F
Where F.Director=Woo
Except
Select P.Title
From Plays P
Where P.SSN=A.SSN))</td>
<td>$\pi_{name}(\text{Actors JOIN } (\pi_{SSN,Title}\text{Plays} \,/\, \pi_{Title}(\sigma_{Director=Woo}\text{Films})))$</td>
<td>$\{X \,/\, \exists A \in Actors\ \wedge \forall F \in Films\ (F.Director = Woo \Rightarrow \exists P \in Plays\ (F.Title=P.Title \wedge P.SSN=A.SSN \wedge X.name=A.name))\}$</td>
<td>
<table>
<tr><td>Actors</td><td>SSN</td><td>Name</td><td>BDate</td></tr>
<tr><td></td><td>_S</td><td></td><td></td></tr>
</table>
<table>
<tr><td>Plays</td><td>SSN</td><td>Title</td><td>Earnings</td></tr>
<tr><td>~</td><td>_S</td><td>_T</td><td></td></tr>
</table>
<table>
<tr><td>Films</td><td>Title</td><td>Director</td><td>Year</td></tr>
<tr><td></td><td>_T</td><td>Woo</td><td></td></tr>
</table>
<table>
<tr><td>BadIds</td><td>SSN</td></tr>
<tr><td>I.</td><td>_S</td></tr>
</table>
<table>
<tr><td>Actors</td><td>SSN</td><td>Name</td><td>BDate</td></tr>
<tr><td></td><td>_S2</td><td>P.</td><td></td></tr>
</table>
<table>
<tr><td>BadIds</td><td>SSN</td></tr>
<tr><td>~</td><td>_S2</td></tr>
</table>
</td>
</tr>
</table>

For the following queries write only the SQL statements:

Q3: Which is the movie that spent the largest amount on actors (not on a single actor)

Q4: Display the actor names who earned in a film more money than the maximum amount that "Travolta" earned in a single movie.

Select Temp.Title From (Select Title, SUM(Earnings) AS S From Plays Group By Title) as Temp Where Temp.S = (Select MAX(S) From Temp)	Select A.name From Actor A, Plays P Where A.SSN=P.SSN and P.Earnings>(Select P2.Earnings From Plays P2, Actors A2 Where P2.SSN=A2.SSN and A2.name="Travolta")

Problem

In the example database of the previous problem assume the following sizes of each attribute:

Film (Title : 40 bytes, Director: 20 bytes, Year: 4 bytes, Company: 20 bytes)
Actor (SSN: 4 bytes, Name: 20 bytes, Date_of_Birth: 4 bytes)

There exist 30,000 films in the database and 100,000 actors. Each block/page is 512 bytes and each pointer is 6 bytes. The blocking factor of a file (*bfr*) is the number of records that fit in a page.

1] What is the blocking factor for Film relation bfr_F and bfr_A for Actor relation? bfr_F =512/84 = 6, bfr_A = 512/28 =18

2] Assuming that the Film relation is sorted on the Title and there is no index what is the cost (in terms of block reads) for:

a] finding the film with title "Titanic": the file is stored in 30,000/6=5,000 pages. Cost of binary search: $\log_2 5000$=13

b] finding all the films directed by "Tarantino": we need sequential scan since sorting is not based on director (5000 pages)

3] Assume that the Actor relation is sorted on the name and you want to create an index on SSN (each index entry has the form <SSN, pointer>).

a] What is the blocking factor for the index (single-level): bfr_{Aindex}= 512/(4+6)=51

b] How many index entries you need (explain): 100,000 we need dense index because sorting is according to name (not SSN).

c] How many number of blocks are required for these entries: 100,000/51=1961

d] What is the cost of retrieval based on a single SSN using this organization. $\log_2 1961$+1=12

e] If you convert the above index in multiple-level index, how many levels you need (assuming full blocks)?

At the next level we index 1961 pages – i.e., index contains 1961/51=39 pages. We need an additional top level with 1 page