Databases

Assume a simple movie database with the following schema. (You may assume that producers have a unique certification number, Cert, that is also recorded in the Movie relation as attribute prodC#; and no two movies are produced with the same title.)

Movie(title, year, length, prodC#)
StarsIn(movieTitle, movieYear, starName)
Producer(name, address, cert)
MovieStar(name, gender, birthdate)

(a) Write the following queries in SQL:

(i) Who were the male stars in the film The Red Squirrel? [1 mark]

(ii) Which movies are longer than Titanic? [2 marks]

(b) SQL has a boolean-valued operator IN such that the expression s IN R is true when s is contained in the relation R (assume for simplicity that R is a single attribute relation and hence s is a simple atomic value).

Consider the following nested SQL query that uses the IN operator:

SELECT name
FROM Producer
WHERE cert IN (SELECT prodC#
FROM Movie
WHERE title IN (SELECT movieTitle
FROM StarsIn
WHERE starName='Nancho Novo'));

(i) State concisely what this query is intended to mean. [1 mark]

(ii) Express this nested query as a single SELECT-FROM-WHERE query. [2 marks]

(iii) Is your query from part (b)(ii) always equivalent to the original query? If yes, then justify your answer; if not, then explain the difference and show how they could be made equivalent. [6 marks]

(c) SQL has a boolean-valued operator EXISTS such that EXISTS R is true if and only if R is not empty.

Show how EXISTS is, in fact, redundant by giving a simple SQL expression that is equivalent to EXISTS R but does not involve EXISTS or any cardinality operators, e.g. COUNT. [Hint: You may use the IN operator.] [8 marks]