Computational Thinking
Solutions to Exercise 8 (Databases)

1 Database Queries

a) SELECT id, title FROM movie LIMIT 5;

b) SELECT * FROM movie ORDER BY title DESC LIMIT 2;

c) SELECT COUNT(*) FROM movie WHERE year > 2000;

d) SELECT title, tomatometer FROM movie WHERE title = 'The Matrix';

e) SELECT COUNT(*) FROM movie
   WHERE tomatometer > (SELECT tomatometer FROM movie
   WHERE title = 'The Matrix');

f) SELECT year, AVG(tomatometer) AS avg FROM movie
   GROUP BY year
   ORDER BY avg DESC LIMIT 5;

g) SELECT title FROM movie
   WHERE title LIKE 'X%'
   ORDER BY title DESC;

h) SELECT COUNT(*) FROM movie
   WHERE title LIKE '%fight%';
2 Advanced Database Queries

a) 
```
SELECT person.name, cast_info.role_id, person.gender 
FROM cast_info 
JOIN person ON person.id = cast_info.person_id 
JOIN movie ON movie.id = cast_info.movie_id 
JOIN role_type ON role_type.id = cast_info.role_id 
WHERE role_type.role = 'actress' AND movie.title = 'The Matrix';
```

b) 
```
SELECT COUNT(DISTINCT person.id) 
FROM cast_info 
JOIN role_type ON role_type.id = cast_info.role_id 
JOIN person ON person.id = cast_info.person_id 
WHERE role_type.role = 'director' AND person.gender = 'f';
```

c) 
```
SELECT DISTINCT person.name FROM cast_info 
JOIN person ON person.id = cast_info.person_id 
JOIN movie ON movie.id = cast_info.movie_id 
WHERE (cast_info.role_id = 2 or cast_info.role_id = 1) 
AND EXISTS ( 
    SELECT DISTINCT ci.person_id FROM cast_info AS ci 
    WHERE ci.role_id = 8 
    AND cast_info.person_id = ci.person_id 
    GROUP BY ci.person_id 
    HAVING COUNT(ci.person_id) > 20 
); 
```

Alternative solution:
```
SELECT DISTINCT person.name FROM person 
JOIN cast_info ON person.id = cast_info.person_id 
JOIN role_type ON cast_info.role_id=role_type.id 
WHERE role_type.role IN ('actor','actress') 
AND 20 < ( 
    SELECT COUNT(*) FROM cast_info AS ci 
    JOIN role_type AS rt ON ci.role_id=rt.id 
    WHERE ci.person_id = person.id 
    AND rt.role='director' 
); 
```

d) 
```
SELECT movie.title, COUNT(*) AS cnt 
FROM movie_keyword 
JOIN movie ON movie_keyword.movie_id = movie.id 
GROUP BY movie.id 
ORDER BY cnt DESC 
LIMIT 1;
```

e) 
```
SELECT AVG(cnt), MAX(cnt), MIN(cnt) FROM ( 
    SELECT movie.title, COUNT(*) AS cnt 
    FROM movie_keyword 
    JOIN movie ON movie_keyword.movie_id = movie.id 
    GROUP BY movie.id 
    ORDER BY cnt DESC 
    LIMIT 1; 
); 
```
f) SELECT
   person.name,
   AVG(movie.tomatometer) AS average,
   COUNT(ci.person_id) AS cnt,
   MAX(movie.year) AS maxyear
FROM cast_info AS ci
JOIN movie ON movie.id = ci.movie_id
JOIN person ON person.id = ci.person_id
WHERE ci.role_id = 1
GROUP BY person.id
HAVING AVG(movie.tomatometer) > 85 AND COUNT(ci.person_id) > 30
   AND MAX(movie.year) > 2000
ORDER BY maxyear DESC, average DESC;

g) SELECT person.name
FROM person
JOIN cast_info ON person.id = cast_info.person_id
JOIN movie ON cast_info.movie_id = movie.id
WHERE cast_info.role_id = 8 AND movie.tomatometer > 90
GROUP BY person.id
HAVING COUNT(*) > 10;