数据库技术第一次作业

潘亦晟
515021910384

1. 问题 1

(1) SQL 语句：SELECT name FROM people WHERE name LIKE 'John %' LIMIT 10 ;

(2) 查询结果:

John Belushi
John Gielgud
John Wayne
John Cleese
John Carpenter
John Cusack
John Denver
John Travolta
John Woo
John Barry

2. 问题 2

(1) SQL 语句：select title,year from movies where year is not null order by year limit 5;

(2) 查询结果:

Miss Jerry|1894
The Corbett-Fitzsimmons Fight|1897
Reproduction of the Corbett and Fitzsimmons Fight|1897
O Campo Grande|1898
O Carnaval em Lisboa|1898

3. 问题 3

(1) SQL 语句：select avg(runtime) from movies where year = 1963;

(2) 查询结果: 90.1671807526218

4. 问题 4

(1) SQL 语句：

```
select title from movies,ratings
    where movies.id = ratings.movie_id
    and ratings.rating = (select max(rating) from ratings) LIMIT 10;
```

(2) 查询结果:

The Maltese Phallus
Girls Loving Girls
Rocky n Rolly: Suntok sabay takbo
Havina Hede
Polar
Welcome to Punjab

The Buck-Tick Syndrome I
Battlefield of the Mind
Old Man Football
The Poodles: In the Flesh

5. 问题 5

(1) SQL 语句：

解法一：（运行速度较慢）

```
select title from movies,cast_members,people
    where people.name = 'Daniel Craig'
    and people.id = cast_members.person_id
    and cast_members.movie_id = movies. id LIMIT 10;
```

解法二：（运行速度较快）

```
SELECT title FROM movies,cast_members
    WHERE id=movie_id AND person_id IN
        (SELECT id FROM people
            WHERE name LIKE 'daniel craig')
LIMIT 10;
```

(2) 查询结果:

Obsession
Love Is the Devil: Study for a Portrait of Francis Bacon
The Trench
Hotel Splendide
Love & Rage
Some Voices
The Mother
Sylvia
The Jacket
Enduring Love

6. 问题 6

(1) SQL 语句：

```
SELECT avg(runtime) FROM movies,ratings
    WHERE movies.id = ratings.movie_id and ratings.rating > 9.0;
```

(2) 查询结果: 82.7180722891566

7. 问题 7

(1) SQL 语句：

```
SELECT count(person_id) FROM cast_members,movies
    WHERE cast_members.movie_id = movies.id
    and movies.runtime = (SELECT max(runtime) from movies);
```

(2) 查询结果: 4

8. 问题 8

(1) SQL 语句：

解法一：

```
select people.name,count(DISTINCT movie_id) as movie_cnt from directors,people
    where directors.person_id = people.id
        group by directors.person_id
        order by movie_cnt DESC LIMIT 1;
```

解法二：

```
WITH director_production_cnt(name, movie_cnt)
AS (SELECT
    people.name,
    count(DISTINCT directors.movie_id) AS movie_count
  FROM
    directors
    JOIN people ON directors.person_id = people.id
  GROUP BY directors.person_id)
SELECT name
FROM director_production_cnt
WHERE movie_cnt = (SELECT max(movie_cnt)
                   FROM director_production_cnt);
```

(2) 解答思路

使用 directors.person_id = people.id 和 group by directors.person_id 作为限制条件，配合 count（DISTINCT movie_id）使用来计算每个导演所导演的电影数，同时使用 as 语句为 count（DISTINCT）重命名一列 movie_cnt,然后最后使用 order by movie_cnt DESC LIMIT 1 选出导演电影数最多的导演，但是解法一有所缺陷，如果拥有导演电影数最多的导演有两个，那么该算法存在缺陷，只能选出其中一个导演；而解法二多了一层嵌套，可以选出所有拥有导演电影数的最大值的导演（们）。

(3) 查询结果: Jir? Yoshino|245

9. 问题 9

(1) SQL 语句：

```
select dir.name,actor.name from people dir , people actor
    where (dir.id,actor.id) in (select id1,id2 from (
        select id1,id2,max(movie_num) from
            (select directors.person_id id1,cast_members.person_id
            id2,count(directors.movie_id) as movie_num
            from directors,cast_members
              where directors.movie_id = cast_members.movie_id
              and directors.person_id != cast_members.person_id
            group by directors.person_id,cast_members.person_id
```

order by count(directors.movie_id) desc limit 1)));

(2) 解答思路

该题的难度主要在于如何从同一个表中 select 两个对象以及需要发现所有的导演都在演员表这个陷阱，先将 people 表具体命名为 dir，actor 两个对象，然后通过 directors.movie_id = cast_members.movie_id 表示两人合作以及 directors.person_id != cast_members.person_id 表示两人一个是导演一个是演员，最终通过 order by count(directors.movie_id) desc limit 1 来选择合作次数最多的演员和导演。

(3) 查询结果:

Sachi Hamano|Kuninori Yamazaki

10. 问题 10

(1) SQL 语句：

```
WITH actor_life(actor_id,time) AS
    (SELECT person_id, (max(year)-min(year)) as timee
    from cast_members,movies
    where cast_members.movie_id = movies.id
    group by person_id)
select name from people,actor_life
where people.id = actor_life.actor_id
and actor_life.time =
     (select max(time) from actor_life);
```

(2) 解答思路

首先通过使用 with …as 建立一个临时表 actor_life, 该临时表拥有两个属性（actor_id,time）分别表示演员的 id 和演员生涯时长，通过 max（year）-min（year）和 group by person_id 计算演员生涯时长 time。最后通过 actor_life.actor_id = people.id 和 actor_life.time = max 选出演员生涯时长最长的演员名。

(3) 查询结果: John Malkovich

11. Bonus

(1) SQL 语句：

```
CREATE TABLE bacon_degree
(
    actor_id TEXT PRIMARY KEY,
    bacon_index INT,
    FOREIGN KEY (actor_id) REFERENCES people(id)
);

INSERT INTO bacon_degree (actor_id, bacon_index) SELECT
  people.id as actor_id,
  NULL
```

```
FROM
   people;

UPDATE bacon_degree
SET bacon_index = 0
WHERE
   actor_id in (SELECT id from people WHERE name = 'Kevin Bacon');

UPDATE bacon_degree
SET bacon_index = 1
WHERE
   actor_id in (
         SELECT second.person_id AS actor_id
         FROM
            cast_members first, cast_members second,bacon_degree first_idx, bacon_degree
second_idx
         WHERE
            (first.movie_id = second.movie_id)
            AND (first_idx.actor_id = first.person_id)
            AND (second_idx.actor_id = second.person_id)
            AND (first_idx.bacon_index = 0)
            AND (second_idx.bacon_index is NULL)
   );

UPDATE bacon_degree
SET bacon_index = 2
WHERE
   actor_id in (
         SELECT second.person_id AS actor_id
FROM
   cast_members first, cast_members second,bacon_degree first_idx, bacon_degree second_idx
WHERE
   (first.movie_id = second.movie_id)
      AND (first_idx.actor_id = first.person_id)
      AND (second_idx.actor_id = second.person_id)
      AND (first_idx.bacon_index = 1)
      AND (second_idx.bacon_index is NULL)
   );

UPDATE bacon_degree
SET bacon_index = 3
WHERE
   actor_id in (
         SELECT second.person_id AS actor_id
```

```
FROM
   cast_members first, cast_members second,bacon_degree first_idx, bacon_degree second_idx
WHERE
   (first.movie_id = second.movie_id)
      AND (first_idx.actor_id = first.person_id)
      AND (second_idx.actor_id = second.person_id)
      AND (first_idx.bacon_index = 2)
      AND (second_idx.bacon_index is NULL)
   );

UPDATE bacon_degree
SET bacon_index = 4
WHERE
   actor_id in (
         SELECT second.person_id AS actor_id
FROM
   cast_members first, cast_members second,bacon_degree first_idx, bacon_degree second_idx
WHERE
   (first.movie_id = second.movie_id)
      AND (first_idx.actor_id = first.person_id)
      AND (second_idx.actor_id = second.person_id)
      AND (first_idx.bacon_index = 3)
      AND (second_idx.bacon_index is NULL)
   );

UPDATE bacon_degree
SET bacon_index = 5
WHERE
   actor_id in (
         SELECT second.person_id AS actor_id
FROM
   cast_members first, cast_members second,bacon_degree first_idx, bacon_degree second_idx
WHERE
   (first.movie_id = second.movie_id)
      AND (first_idx.actor_id = first.person_id)
      AND (second_idx.actor_id = second.person_id)
      AND (first_idx.bacon_index = 4)
      AND (second_idx.bacon_index is NULL)
   );

UPDATE bacon_degree
SET bacon_index = 6
WHERE
   actor_id in (
```

```
    SELECT second.person_id AS actor_id
    FROM
      cast_members first, cast_members second,bacon_degree first_idx, bacon_degree
second_idx
    WHERE
      (first.movie_id = second.movie_id)
        AND (first_idx.actor_id = first.person_id)
        AND (second_idx.actor_id = second.person_id)
        AND (first_idx.bacon_index = 5)
        AND (second_idx.bacon_index is NULL)
  );

SELECT
  people.name as name,
  bacon_degree.bacon_index as bacon_index
FROM
  people, bacon_degree
WHERE
  people.id = bacon_degree.actor_id
  AND people.name in ('Sean Connery', 'Humphrey Bogart', 'Shirley Temple');

DROP TABLE bacon_degree;
```

(2) 解答思路

首先分析一下 bacon_degree 的实质，可以将 bacon_degree 看成以 Kevin Bacon 为起点的一个 BFS（广度优先搜索），每个演员的 bacon_degree 即 BFS 中的搜索层数。

对于 SQL 实现，我们先建立一个新的表 bacon_degree，该表有两个属性演员 id 和演员的 bacon_degree。起始状态中，每个演员的 bacon_degree 属性都为 null。

对于第 0 层，我们将 Kevin Bacon 的 bacon_degree 设为 0.

对于第 i 层，我们将设定本身 bacon_degree 为 null 且与 bacon_degree 为 i-1 的演员同时参演的演员的 bacon_degree 为 i；

根据 6 度空间理论，我们迭代 6 次后，分别选取'Sean Connery', 'Humphrey Bogart', 'Shirley Temple'的 bacon_degree.

(3) 查询结果:

Sean Connery | 3
Humphrey Bogart | 3
Shirley Temple | 2