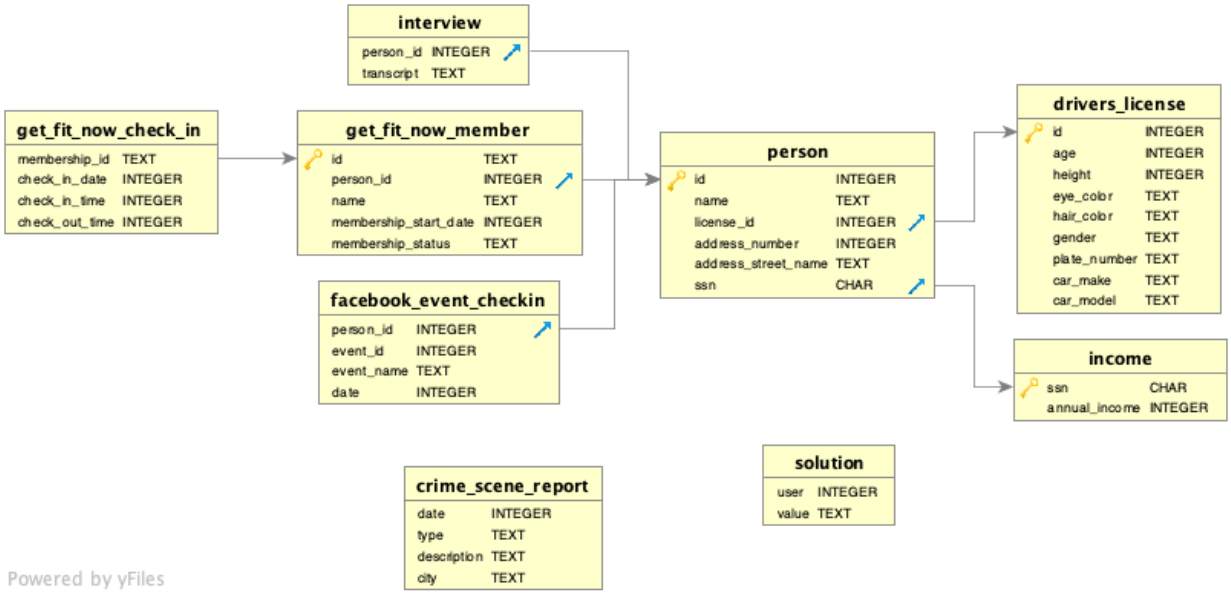
THIS WALKTHROUGH HAS SPOILERS FOR THE GAME SQL MURDER MYSTERY

The following code was generated by Robert Marston for a game found online titled "SQL Murder Mystery." This game can be found at: https://mystery.knightlab.com/
The purpose of this report is to showcase my ability to structure SQL queries using SQLite and show off how my brain approached this question with code.
The schema diagram below is provided in the exercise.



The Scene:

A crime has taken place and the detective needs your help. The detective gave you the crime scene report, but you somehow lost it. You vaguely remember that the crime was a **murder** that occurred sometime on **Jan.15, 2018** and that it took place in **SQL City**. Start by retrieving the corresponding crime scene report from the police department's database.

Next Step:
With this data we know we can query the crime_scene_report table to find our lost report so we can pick up the investigation. We can infer that **murder** is a crime type but let's do a quick check.

Input:

```
SELECT DISTINCT(type)
FROM crime_scene_report;
```

Output:

type				
robbery	theft	arson	assault	blackmail
murder	fraud	bribery	smuggling	

Next Step:
Proceed with fetching our crime scene report.

Input:
```
SELECT date,
       type,
       description,
       city
FROM crime_scene_report
WHERE date = '20180115'
      AND city = 'SQL City'
       AND type = 'murder';
```
Output:

date	type	description	city
20180115	murder	Security footage shows that there were 2 witnesses. The first witness lives at the last house on "Northwestern Dr". The second witness, named Annabel, lives somewhere on "Franklin Ave".	SQL City

Next Step:
Using this description, we can assume last address on Northwestern Drive would be the highest number and that home should be the first witness, by using MAX on address number where the street value is Northwestern Dr we can find out who that owner is and their exact address. We need the person ID to look up interview transcripts, so we fetch id in the SELECT.

Input:
```
SELECT name,
       Id,
       MAX(address_number) AS max_addy_number,
       address_street_name AS street_name
FROM person
WHERE street_name = 'Northwestern Dr';
```

OUTPUT:

name	id	max_addy_number	street_name
Morty Schapiro	14887	4919	Northwestern Dr

Next Step:
The description stated the second witness was named Annabel and that she lived somewhere on “Franklin Ave”. We can use a similar query but this time order by name, she should be near the top if we order ascending. But we can limit the search to the top 5 to start. Again, we need the person ID to look up interview transcripts, so we fetch id in the SELECT.

Input:

```
SELECT name,
        id,
        address_number,
        address_street_name
FROM person
WHERE address_street_name ='Franklin Ave'
ORDER BY name
LIMIT 5.
```

OUTPUT:

name	id	address_number	address_street_name
Amado Mattan	33793	99	Franklin Ave
Annabel Miller	16371	103	Franklin Ave
Bev Billiter	46827	2316	Franklin Ave
Blake Chrones	78658	2014	Franklin Ave
Cameron Dilick	97913	2954	Franklin Ave

Next Step:

Now that we have Identified the two witnesses, we can review the interviews the officers had with them after the crime to look for more clues. ID maps to person_id so we can fetch the transcripts of our two witnesses with what we have. Let's JOIN the name from the person table to our query so we can see the name of the person and their transcript and not just their ID.

Input:

```
SELECT name,
        person_id,
        transcript
FROM interview AS i
    LEFT JOIN person AS p
        ON i.person_id=p.id
WHERE person_id ='14887'
    OR person_id='16371';
```

Output:

name	person_id	transcript
Morty Schapiro	14887	I heard a gunshot and then saw a man run out. He had a "Get Fit Now Gym" bag. The membership number on the bag started with "48Z". Only gold members have those bags. The man got into a car with a plate that included "H42W".
Annabel Miller	16371	I saw the murder happen, and I recognized the killer from my gym when I was working out last week on January the 9th.

Review data:
Killer was at the gym on 01/09/2018 (data is stored in 'YYYYMMDD')
Killer was likely a gold member.
Killer membership number starts with "48Z".
Killer was Male and entered a car with plates that include "H42W".

Next Step:
Let's write a query to get a list of members who checked into the gym on Jan 9th 2018 that have memberships that start with "48Z" we need data from two tables and they both happen to have the longest names in our schema so let's make use of aliases to lighten our workload and clean up the code a bit.

Input:

```
SELECT
        gfnm.name AS name,
        person_id,
        gfnc.membership_id,
        gfnm.membership_status,
        gfnc.check_in_date AS date
FROM get_fit_now_check_in AS gfnc
        LEFT JOIN get_fit_now_member AS gfnm
                ON gfnc.membership_id = gfnm.id
WHERE gfnc.check_in_date = '20180109'
        AND gfnc.membership_id LIKE '48z%'
ORDER BY gfnm.name;
```

OUTPUT:

name	person_id	membership_id	membership_status	date
Jeremy Bowers	67318	48Z55	gold	20180109
Joe Germuska	28819	48Z7A	gold	20180109

Next Step:
We are getting somewhere here, let's see if we can get a match on a license plate, since that is the last bit of evidence we have not addressed.

Input:

```
SELECT p.name,
        dl.id,
        plate_number,
        p.license_id
FROM drivers_license AS dl
        LEFT JOIN person AS p ON dl.id = p.license_id
WHERE plate_number LIKE '%H42W%'
        AND gender = 'male';
```

Output:

name	id	plate_number	license_id
Jeremy Bowers	423327	0H42W2	423327
Tushar Chandra	664760	4H42WR	664760

Next Step:
Okay, there are a lot of signs here that Jeremy Bowers may have done a bad. I am calling it and accusing Jeremy. There is an entry on the webpage for us to check our answer by entering the name of the person we think is guilty.

Conclusion 1:
Success! Upon tracking down the killer, we are advised that if we read the interview transcript for Jeremy, we may be able to find out who hired him. Let's see what this transcript has to say.

Input

```
SELECT name,
        person_id,
        transcript
FROM interview AS i
        LEFT JOIN person AS p
                ON i.person_id=p.id
WHERE person_id ='67318'
```

Output:

name	person_id	transcript
Jeremy Bowers	67318	I was hired by a woman with a lot of money. I don't know her name but I know she's around 5'5" (65") or 5'7" (67"). She has red hair and she drives a Tesla Model S. I know that she attended the SQL Symphony Concert 3 times in December 2017.

CHALLENGE MODE:

Next step:
Who hired Jeremy Bowers? We are asked if we can do this **in only 2 queries.**

Given the information from Jeremy's transcript let's search the drivers_license table for females who have red hair and drive a Tesla Model S. While searching that table, we will join to person so we can get the names of the women who match the description and get their SSNs so we can further ascertain who has the most wealth.

Input:

```
SELECT p.name AS name,
        dl.id,
        height,
        hair_color,
        gender,
        car_make AS make,
        car_model AS model,
        p.ssn
FROM drivers_license AS dl
          LEFT JOIN person AS p ON dl.id = p.license_id
WHERE gender = 'female'
        AND car_make = 'Tesla'
         AND hair_color = 'red';
```

OUTPUT:

name	id	height	hair_color	gender	make	model	ssn
Miranda Priestly	202298	66	red	female	Tesla	Model S	987756388
Regina George	291182	66	red	female	Tesla	Model S	337169072
Red Korb	918773	65	red	female	Tesla	Model S	961388910

Next Step:

We have three suspects, they are all around the same height, all have red hair and drive a Tesla Model S. Let's use this data to see who has the highest income and let's see if any of these ladies checked in to events on Facebook. We will use a CTE (Common Table Expression) to first select the ID of the ladies above using their SSN. The main query will then fetch the remaining data we want to see from the income and person table. We will use a CASE statement to make a column that will return "No" if the individual did not check into a Facebook event and "Yes" if they did. Normally we would use more aliases to make the query look a little appealing and to make the JOIN easier to write but for the sake of science let's write this one out and only alias our CTE and CASE WHEN statements.

Input:

```
WITH person_ids AS (
 SELECT person.id AS person_id
 FROM income
          LEFT JOIN person
                  ON income.ssn = person.ssn
 WHERE income.ssn IN ('987756388', '337169072', '961388910')
)
SELECT
   income.ssn,
   annual_income,
   person.name,
```

```
    person.id AS person_id,
    facebook_event_checkin.date,
    facebook_event_checkin.event_name,
        CASE WHEN facebook_event_checkin.person_id IS NOT NULL THEN 'Yes' ELSE 'No' END
        AS has_checked_in
FROM income
        LEFT JOIN person
                ON income.ssn = person.ssn
                LEFT JOIN facebook_event_checkin
                        ON person.id = facebook_event_checkin.person_id
WHERE person.id IN (SELECT person_id FROM person_ids);
```

OUTPUT:

ssn	annual_income	name	person_id	date	event_name	has_checked_in
961388910	278000	Red Korb	78881	null	null	No
987756388	310000	Miranda Priestly	99716	20171206	SQL Symphony Concert	Yes
987756388	310000	Miranda Priestly	99716	20171212	SQL Symphony Concert	Yes
987756388	310000	Miranda Priestly	99716	20171229	SQL Symphony Concert	Yes

Conclusion 2:

Our evidence from Jeremy Bowers states that the women that hired him was wealthy and attended three events in December of 2017 called “SQL Symphony Concert”.

Well, Hello Miranda Priestly, I think you hired Jeremy to commit Murder!

Thank you for joining me on this adventure.