SQL Murder Mystery:

Experienced SQL sleuths start here

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.

Exploring the Database Structure

Experienced SQL users can often use database queries to infer the structure of a database. But each database system has different ways of managing this information. The SQL Murder Mystery is built using SQLite. Use this SQL command to find the tables in the Murder Mystery database.

Run this query to find the names of the tables in this database.

This command is specific to SQLite. For other databases, you'll have to learn their specific syntax.

```
1 SELECT name
2   FROM sqlite_master
3  where type = 'table'
```

RUN ⇩ RESET

name
crime_scene_report
drivers_license
facebook_event_checkin
interview
get_fit_now_member
get_fit_now_check_in
solution
income
person

Besides knowing the table names, you need to know how each table is structured. The way this works is also dependent upon which database technology you use. Here's how you do it with SQLite.

Run this query to find the structure of the `crime_scene_report` table

Change the value of 'name' to see the structure of the other tables you learned about with the previous query.

```
1 SELECT sql
2   FROM sqlite_master
3  where name = 'crime_scene_report'
```

RUN ⇩ RESET

sql
CREATE TABLE crime_scene_report (date integer, type text, description text, city text)

```sql
SELECT sql
  FROM sqlite_master
 where name = 'drivers_license'
```

RUN ⇩ RESET

sql
CREATE TABLE drivers_license (id integer PRIMARY KEY, age integer, height integer, eye_color text, hair_color text, gender text, plate_number text, car_make text, car_model text)

```sql
SELECT sql
  FROM sqlite_master
 where name = 'facebook_event_checkin'
```

RUN ⇩ RESET

sql
CREATE TABLE facebook_event_checkin (person_id integer, event_id integer, event_name text, date integer, FOREIGN KEY (person_id) REFERENCES person(id))

```sql
SELECT sql
  FROM sqlite_master
 where name = 'interview'
```

RUN ⇩ RESET

sql
CREATE TABLE interview (person_id integer, transcript text, FOREIGN KEY (person_id) REFERENCES person(id))

```sql
SELECT sql
  FROM sqlite_master
 where name = 'get_fit_now_member'
```

RUN ⇩ RESET

sql
CREATE TABLE get_fit_now_member (id text PRIMARY KEY, person_id integer, name text, membership_start_date integer, membership_status text, FOREIGN KEY (person_id) REFERENCES person(id))

```sql
SELECT sql
  FROM sqlite_master
 where name = 'get_fit_now_check_in'
```

RUN ⇩ RESET

sql
CREATE TABLE get_fit_now_check_in (membership_id text, check_in_date integer, check_in_time integer, check_out_time integer, FOREIGN KEY (membership_id) REFERENCES get_fit_now_member(id))

```sql
SELECT sql
  FROM sqlite_master
 where name = 'solution'
```

RUN ⇩ RESET

sql
CREATE TABLE solution (user integer, value text)

```sql
SELECT sql
  FROM sqlite_master
 where name = 'income'
```

RUN ⇩ RESET

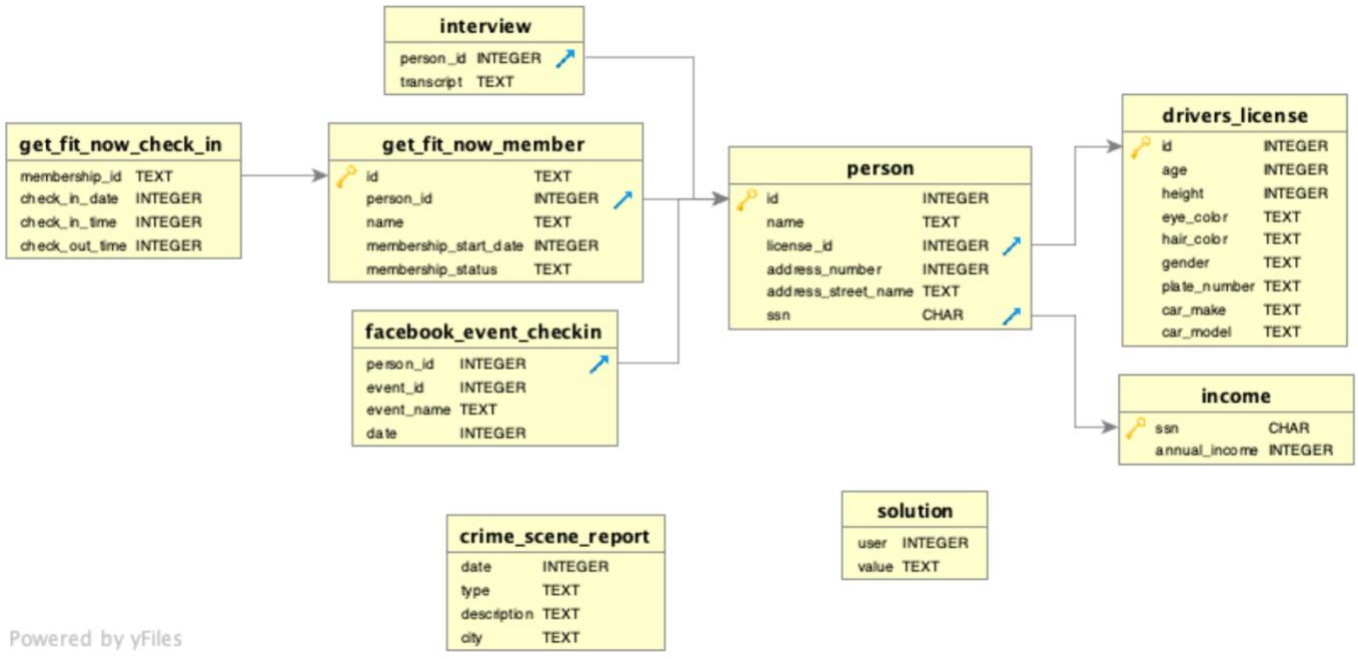
sql
CREATE TABLE income (ssn CHAR PRIMARY KEY, annual_income integer)

```sql
SELECT sql
  FROM sqlite_master
 where name = 'person'
```

RUN ⇩ RESET

sql
CREATE TABLE person (id integer PRIMARY KEY, name text, license_id integer, address_number integer, address_street_name text, ssn CHAR REFERENCES income (ssn), FOREIGN KEY (license_id) REFERENCES drivers_license (id))

But click here to show the schema diagram.



Primero analizo que datos tiene la tabla de reportes de escenas de crímenes:

```
SELECT *
FROM crime_scene_report
LIMIT 5;
```

RUN ⇩ RESET

date	type	description	city
20180115	robbery	A Man Dressed as Spider-Man Is on a Robbery Spree	NYC
20180115	murder	Life? Dont talk to me about life.	Albany
20180115	murder	Mama, I killed a man, put a gun against his head...	Reno
20180215	murder	REDACTED REDACTED REDACTED	SQL City
20180215	murder	Someone killed the guard! He took an arrow to the knee!	SQL City

Localizo el archivo del caso (según el tipo de crimen, la fecha en que fue cometido y la ciudad) y leo la descripción:

```
SELECT *
FROM crime_scene_report
WHERE date = '20180115' AND type = 'murder' AND city = 'SQL City';
```

RUN ⇩ RESET

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

Ahora sé que he de buscar 2 testigos.

Primero analizo los datos que contiene la tabla de ‘personas’ para después, según los datos que aparecen en el archivo del crimen, encontrar a mis dos testigos.

```sql
SELECT *
FROM person
LIMIT 5;
```

RUN ⇩ RESET

id	name	license_id	address_number	address_street_name	ssn
10000	Christoper Peteuil	993845	624	Bankhall Ave	747714076
10007	Kourtney Calderwood	861794	2791	Gustavus Blvd	477972044
10010	Muoi Cary	385336	741	Northwestern Dr	828638512
10016	Era Moselle	431897	1987	Wood Glade St	614621061
10025	Trena Hornby	550890	276	Daws Hill Way	223877684

Testigo 1:

```sql
SELECT *
FROM person
WHERE address_street_name = 'Northwestern Dr'
AND address_number = (Select MAX (address_number)
                      FROM person
                      WHERE address_street_name = 'Northwestern Dr'
                      GROUP by address_street_name);
```

RUN ⇩ RESET

id	name	license_id	address_number	address_street_name	ssn
14887	Morty Schapiro	118009	4919	Northwestern Dr	111564949

Testigo 2:

```sql
SELECT *
FROM person
WHERE address_street_name = 'Franklin Ave'
AND name LIKE '%Annabel%';
```

RUN ⇩ RESET

id	name	license_id	address_number	address_street_name	ssn
16371	Annabel Miller	490173	103	Franklin Ave	318771143

Ahora busco las entrevistas que se hicieron a los dos testigos (tabla ‘inteview’) para extraer información de sus declaraciones:

```sql
SELECT *
FROM interview
WHERE person_id = 14887 OR person_id = 16371;
```

RUN ⇩ RESET

person_id	transcript
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".
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.

Teniendo en cuenta los datos aportados por los testigos busco al asesino:

```sql
SELECT m.name, m.membership_status,
    ci.check_in_date, dl.plate_number
FROM person AS p
JOIN get_fit_now_member AS m
ON p.id = m.person_id
JOIN get_fit_now_check_in AS ci
ON m.id = ci.membership_id
JOIN drivers_license AS dl
ON p.license_id = dl.id
WHERE m.id LIKE '48Z%'
AND m.membership_status = 'gold'
AND dl.plate_number LIKE '%H42W%'
AND ci.check_in_date = 20180109;
```

RUN ⇩ RESET

name	membership_status	check_in_date	plate_number
Jeremy Bowers	gold	20180109	0H42W2

Did you find the killer?

```sql
INSERT INTO solution VALUES (1, 'Jeremy Bowers');

        SELECT value FROM solution;
```

RUN ⇩ RESET

value
Congrats, you found the murderer! But wait, there's more... If you think you're up for a challenge, try querying the interview transcript of the murderer to find the real villain behind this crime. If you feel especially confident in your SQL skills, try to complete this final step with no more than 2 queries. Use this same INSERT statement with your new suspect to check your answer.

Busco dentro de las entrevistas la del asesino (Jeremy Bowers):

```sql
SELECT *
FROM interview
WHERE person_id = (SELECT id
                   FROM person
                   WHERE name = 'Jeremy Bowers');
```

RUN ⇩ RESET

person_id	transcript
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.

Ahora busco a una mujer que cumpla con todas las características que el asesino describe:

```sql
SELECT p.name
FROM person AS p
JOIN drivers_license AS dl
ON p.license_id = dl.id
JOIN facebook_event_checkin AS f
ON p.id = f.person_id
WHERE dl.gender = 'female'
AND dl.height BETWEEN 65 AND 67
AND dl.hair_color = 'red'
AND dl.car_make = 'Tesla' AND dl.car_model = 'Model S'
AND f.event_name = 'SQL Symphony Concert'
AND f.date LIKE '201712%'
GROUP BY f.person_id;
```

RUN ⇩ RESET

name
Miranda Priestly

Did you find the killer?

```sql
INSERT INTO solution VALUES (1,'Miranda Priestly');

        SELECT value FROM solution;
```

RUN ⇩ RESET

value
Congrats, you found the brains behind the murder! Everyone in SQL City hails you as the greatest SQL detective of all time. Time to break out the champagne!