
ORACLE®
ACADEMY

Solve It With SQL

Lesson 3
Introduction to SQL: Chapter 2

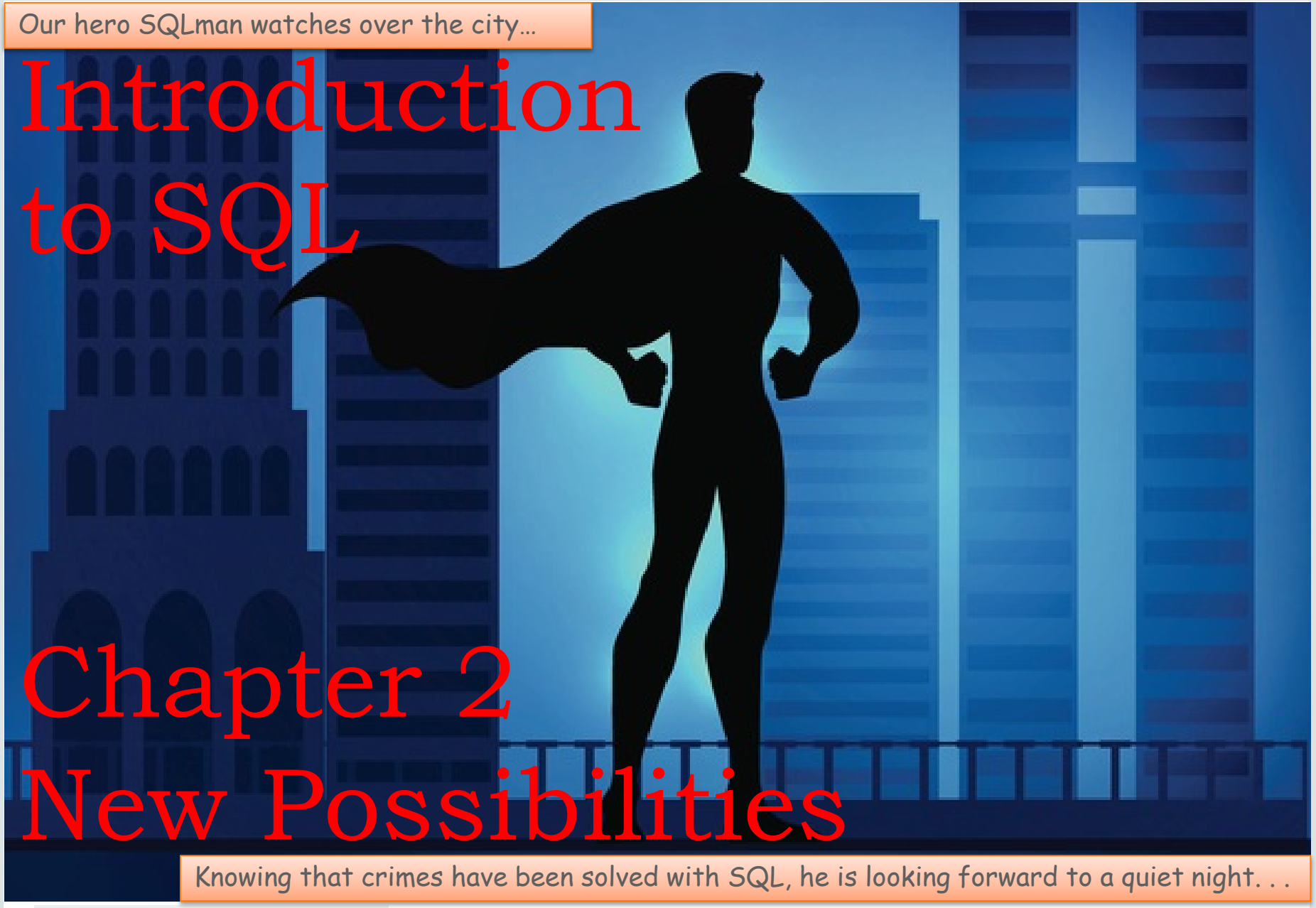
ORACLE ACADEMY

Overview

This lesson covers the following topics:

- Running a script
- Natural Joins
- Using Two tables in a query
- Order By
- Group By and Count

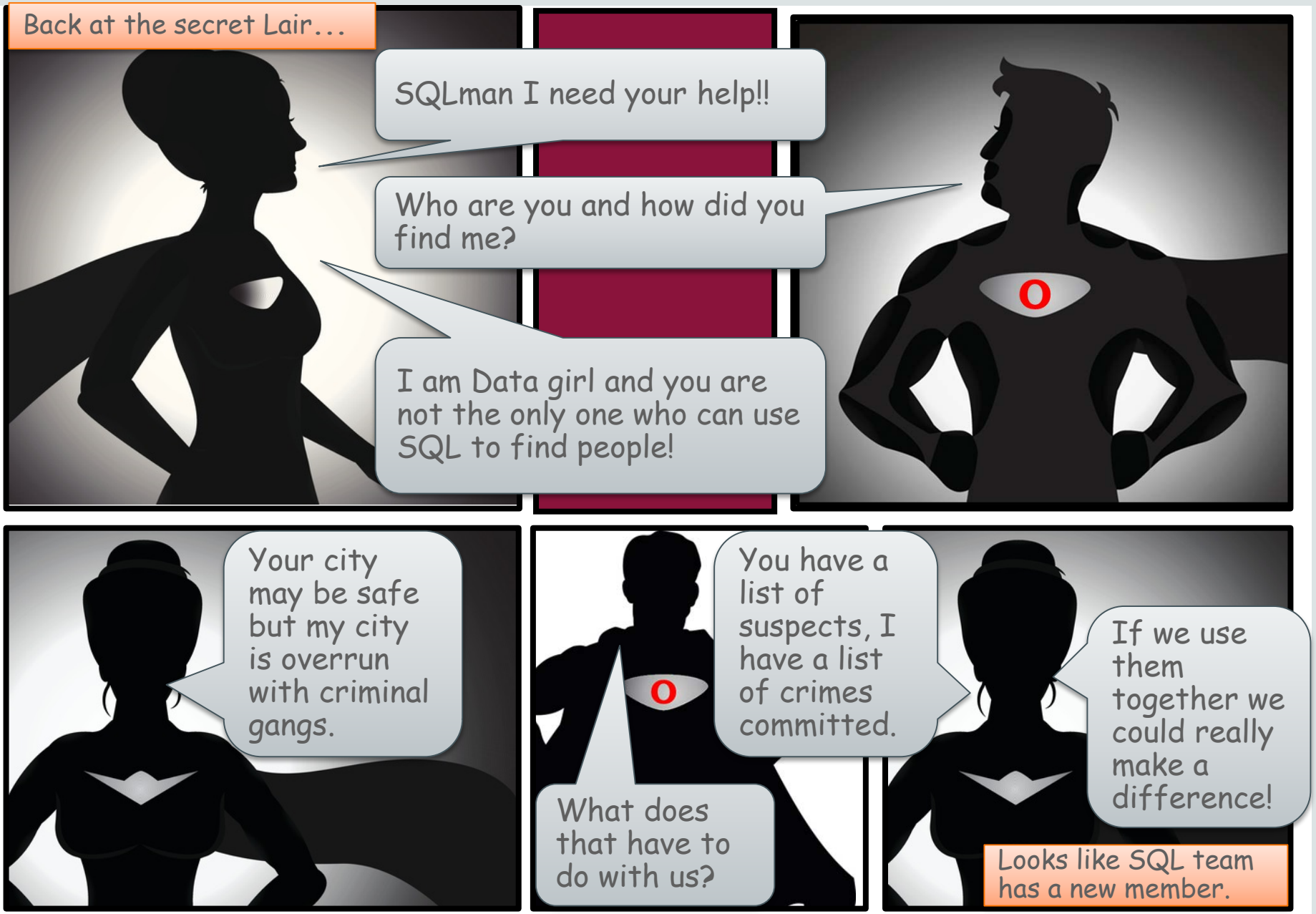
Our hero SQLman watches over the city…



Introduction to SQL

Chapter 2 New Possibilities

Knowing that crimes have been solved with SQL, he is looking forward to a quiet night. . .



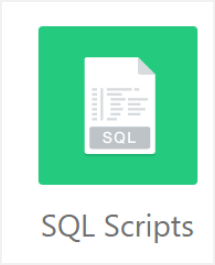
Solve It With SQL

Lesson 2
Introduction to SQL

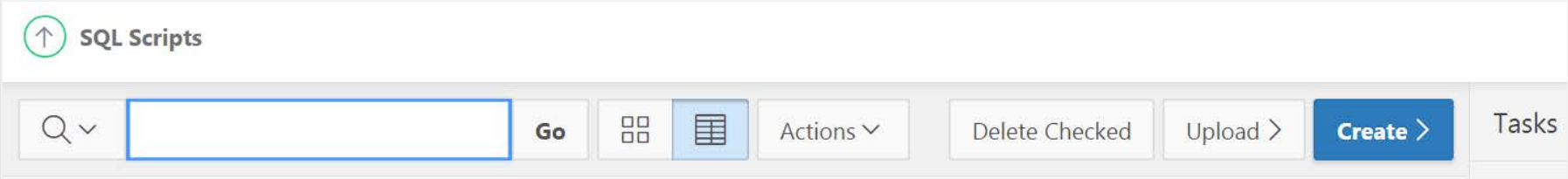
Part 1 – Running a Script

Running a script revisited

From the APEX menu click on the SQL Scripts option

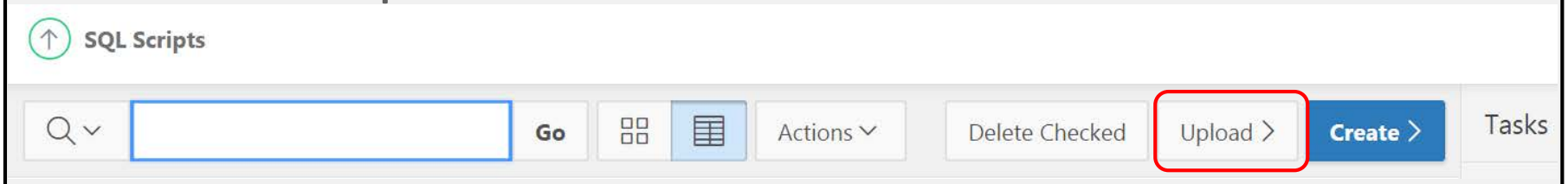


You will see the following interface:

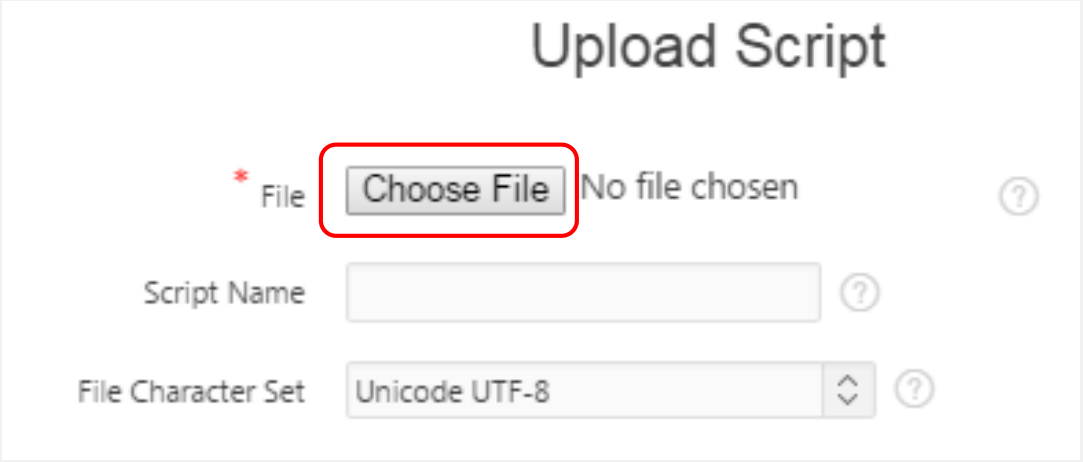


Running a script revisited cont..

Click on the Upload button

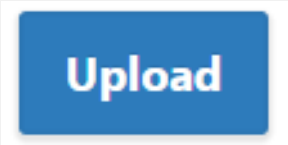


Click the Choose File button



Running a script revisited cont..

Browse to where you saved the Crimes.sql file and click ok.

Click the 

button

You will see the following information about your script:

Name	Created	Updated By	Updated	Bytes	Results	Run
Crimes.sql	2 seconds ago	Your details here	2 seconds ago	10,404	0	
Suspects.sql	58 minutes ago	Your details here	58 minutes ago	62,137	1	

Click the run  button to execute the script!

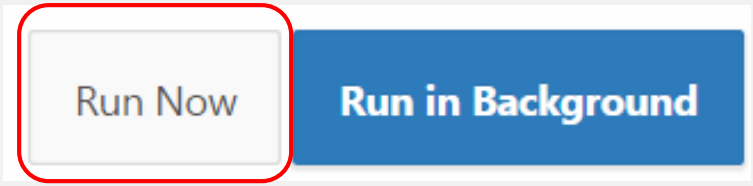
Running a script revisited cont..

You will see a confirmation message

Check the details are correct

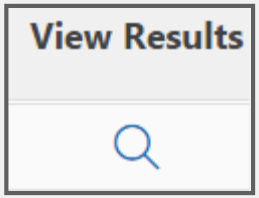
Script Name	**Crimes.sql**

Click the Run Now button



Running a script revisited cont..

When the script runs you will see a view results option:



Click on the Magnifying glass  and scroll to the bottom, you should see the following message:

Once you have completed this return to the SQL commands environment within the SQL workshop in APEX.

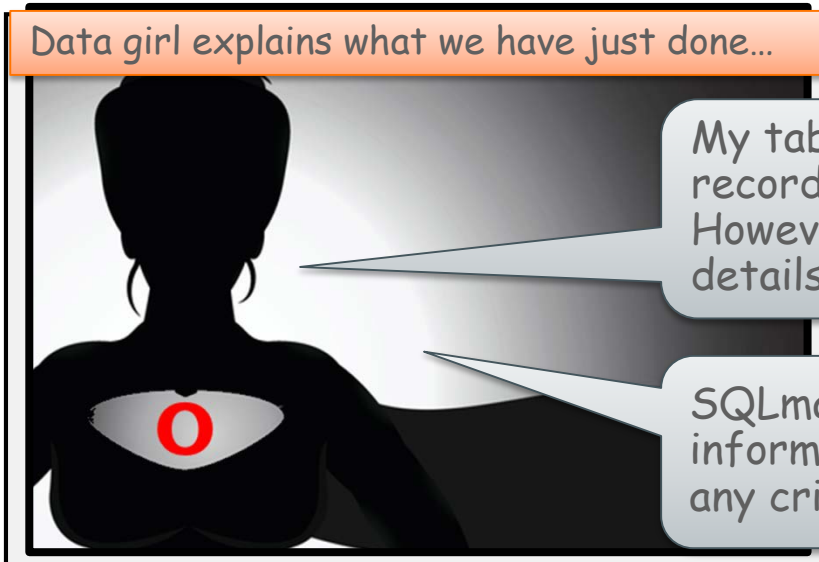
Solve It With SQL

Lesson 2
Introduction to SQL

Part 2 – Natural Joins

Why do we need two tables?

Data girl explains what we have just done…



My table has a list of crimes that have been recorded in my city with the punishment received. However it only lists the suspect id without any details of the criminals.

SQLman, your table holds all of the suspect information but you have no way of knowing what if any crimes your suspects have committed before.

So, with the tables joined together we can use the clues that we are given about the suspects and the information about their criminal history to return who is the most likely suspect to have committed that type of crime?

But how does it work? How are the tables joined?

How are the tables joined?

Data girl explains the theory behind joins..

Both of our tables have a field named suspect_id which have been defined with the same data type. This creates a **natural** join between the tables.

Because of these fields we can create a relationship between the tables by creating something known as a foreign key.

A foreign key in one table is linked to the matching primary key in another.

suspect_id in the crimes table is linked to the suspect_id field in the suspects table!

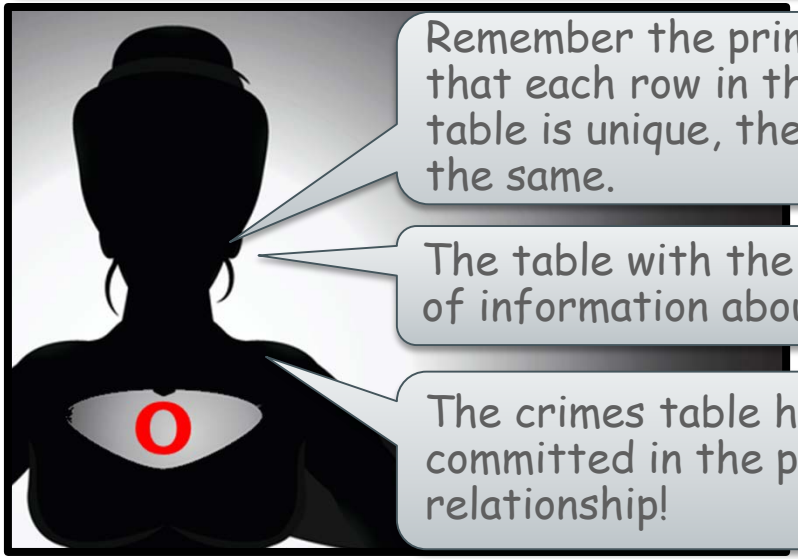
Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
CRIMES	CRIME_ID	NUMBER	-	6	0	1	-
	SUSPECT_ID	NUMBER	-	6	0	-	-
	CRIME_TYPE	VARCHAR2	30	-	-	-	-
	COMMITTED_DATE	DATE	7	-	-	-	-
	PRISON	VARCHAR2	20	-	-	-	-
	BEHAVIOR	VARCHAR2	6	-	-	-	✓
	START_DATE	DATE	7	-	-	-	-
	END_DATE	DATE	7	-	-	-	-

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable
SUSPECTS	SUSPECT_ID	NUMBER	-	6	0	1	-
	NAME	VARCHAR2	30	-	-	-	-
	SEX	VARCHAR2	12	-	-	-	-
	AGE	NUMBER	-	3	0	-	-
	HEIGHT	VARCHAR2	6	-	-	-	✓
	HAIR_COLOR	VARCHAR2	10	-	-	-	✓
	EYE_COLOR	VARCHAR2	10	-	-	-	✓
	FACIAL_HAIR	VARCHAR2	3	-	-	-	✓
	TATTOOS	VARCHAR2	3	-	-	-	✓
	GLASSES	VARCHAR2	3	-	-	-	✓
	SCARS	VARCHAR2	3	-	-	-	✓
	FEET_SIZE	VARCHAR2	6	-	-	-	✓

What does this let us do?

SQLman is still not sure what this means..

What data can I get from the tables once the primary/foreign key relationship is established?



Remember the primary key means that each row in the suspects table is unique, there can't be two the same.

The table with the foreign key allows us to store multiple pieces of information about that unique row (a single suspect).

The crimes table holds all of the crimes that the suspects have committed in the past. This is known as a one to many relationship!

What does that mean a one to many relationship?

We can track any one suspect to the many crimes they have committed!

How are the tables joined?

Time to see how it works. . .

The first thief that was caught for the bank robbery was Kasper Good. Could I see his criminal history now?

Easily, if you add the following code to the code editor in APEX you will see exactly what he has been up to!

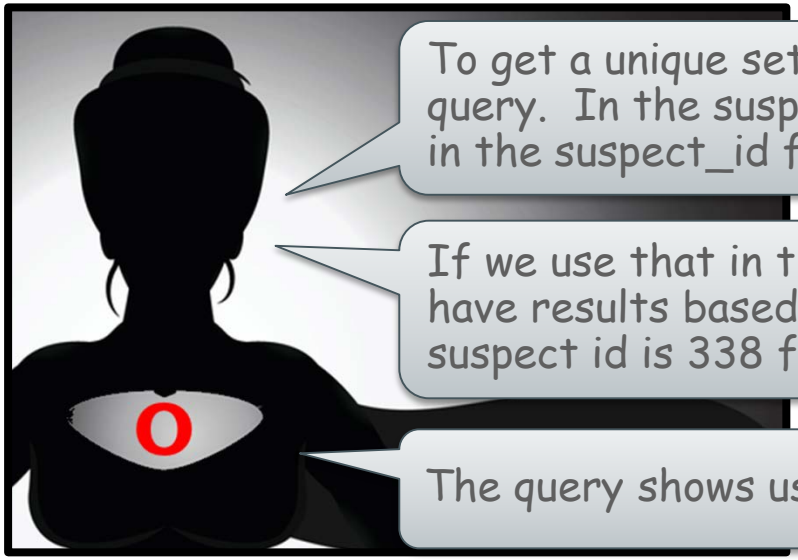
```
SELECT suspect_id, name crime_id, crime_type, committed_date,
       prison, start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE name = 'Kasper Good';
```

In this example the suspects_id field is common to both tables. The name field comes from the suspects table whereas everything else comes from the crimes table.

The trouble with this query is that if there is more than one suspect named Kasper Good all of their details will be returned.

Can you think of anything else we could use in the where clause instead of the name?

What does this let us do?

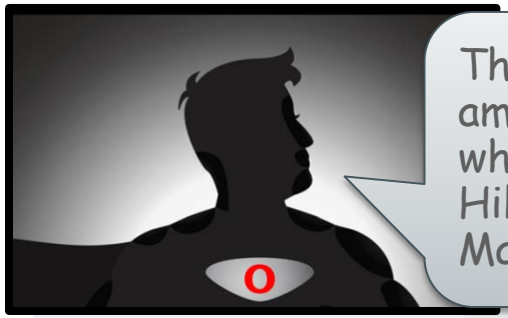


To get a unique set of results we need to use a unique value in our query. In the suspect table the unique value (primary key) is held in the suspect_id field.

If we use that in the search criteria we can guarantee that we have results based on that one suspect. We know Kasper Good's suspect id is 338 from our previous query.

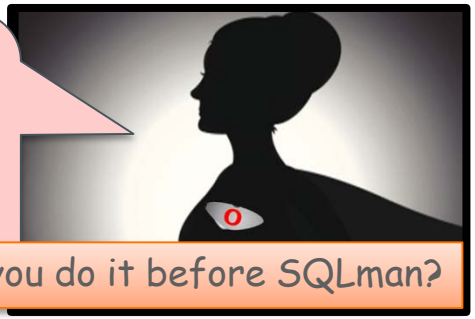
The query shows us that he has been convicted 8 times.

```
SELECT name crime_id, crime_type, committed_date, prison,
       start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 338;
```



That's amazing what about Hilary Mayer?

Challenge
How would you change the query to see all of the crimes that Hilary Mayer committed?



Can you do it before SQLman?

Challenge! Find Hilary Mayer's criminal past

Did you solve it the same way SQLman did?

I don't want to use his name in the query in case there is more than one Hilary Mayer in the system.

I'll run a query to get the suspect_id first based on what I know about him and then use the suspect_id to search his criminal past.

Query #1 :
Get the suspect_id

```
SELECT suspect_id
FROM suspects
WHERE name = 'Hilary Mayer' AND hair_color = 'Red'
AND eye_color= 'Blue' AND feet_size= 'Small'
AND tattoos = 'Yes' AND age BETWEEN 35 AND 40;
```

suspect_id: 100

Query #2 :
Get the criminal history using suspect_id 100.

```
SELECT name, crime_id, crime_type, committed_date, prison,
            start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 100;
```

Changing how our results are displayed?

Data girl is impressed with how you are picking this up. . .

That's excellent work you can see that he has been in prison 11 times already.

When you search a database table the results aren't returned in a specific order unless we specify it.

To display the results in a certain order we can use an **ORDER BY** clause.

NAME	CRIME_ID	CRIME_TYPE	COMMITTED_DATE	PRISON	START_DATE	END_DATE	BEHAVIOR
Hilary Mayer	1002	Robbery	01/12/2016	Stone House	02/15/2016	07/02/2016	Good
Hilary Mayer	802	Robbery	01/01/2013	Stone House	04/09/2013	10/24/2015	Bad
Hilary Mayer	182	Robbery	12/22/2003	Stone House	02/24/2004	01/02/2005	Bad
Hilary Mayer	162	Robbery	01/26/2003	Pull Mount	04/01/2003	11/21/2003	Bad
Hilary Mayer	122	Robbery	03/12/2002	Stone House	06/08/2002	01/24/2003	Bad
Hilary Mayer	602	Robbery	07/12/2010	Stone House	08/19/2010	06/05/2012	Bad
Hilary Mayer	402	Robbery	07/01/2008	Stone House	10/16/2008	05/24/2012	Bad
Hilary Mayer	202	Robbery	07/12/2005	Stone House	09/01/2005	10/04/2007	Bad
Hilary Mayer	82	Robbery	07/12/2001	Iron Gates	11/01/2001	02/24/2002	Bad
Hilary Mayer	62	Car Theft	07/12/1999	Iron Gates	11/05/1999	12/23/2000	Bad
Hilary Mayer	12	Pickpocket	07/12/1996	The Pokey	11/01/1996	02/11/1997	Good

11 rows returned in 0.01 seconds Download

Using ORDER BY in a query.

Let's find out how to get the data in the order that we want. . .

The ORDER BY clause comes at the very end of the query and allows you to specify what column the data is to be sorted by.

The default order is ascending (numeric and alphabetic). You can reverse the order by including the descending (DESC) operator.

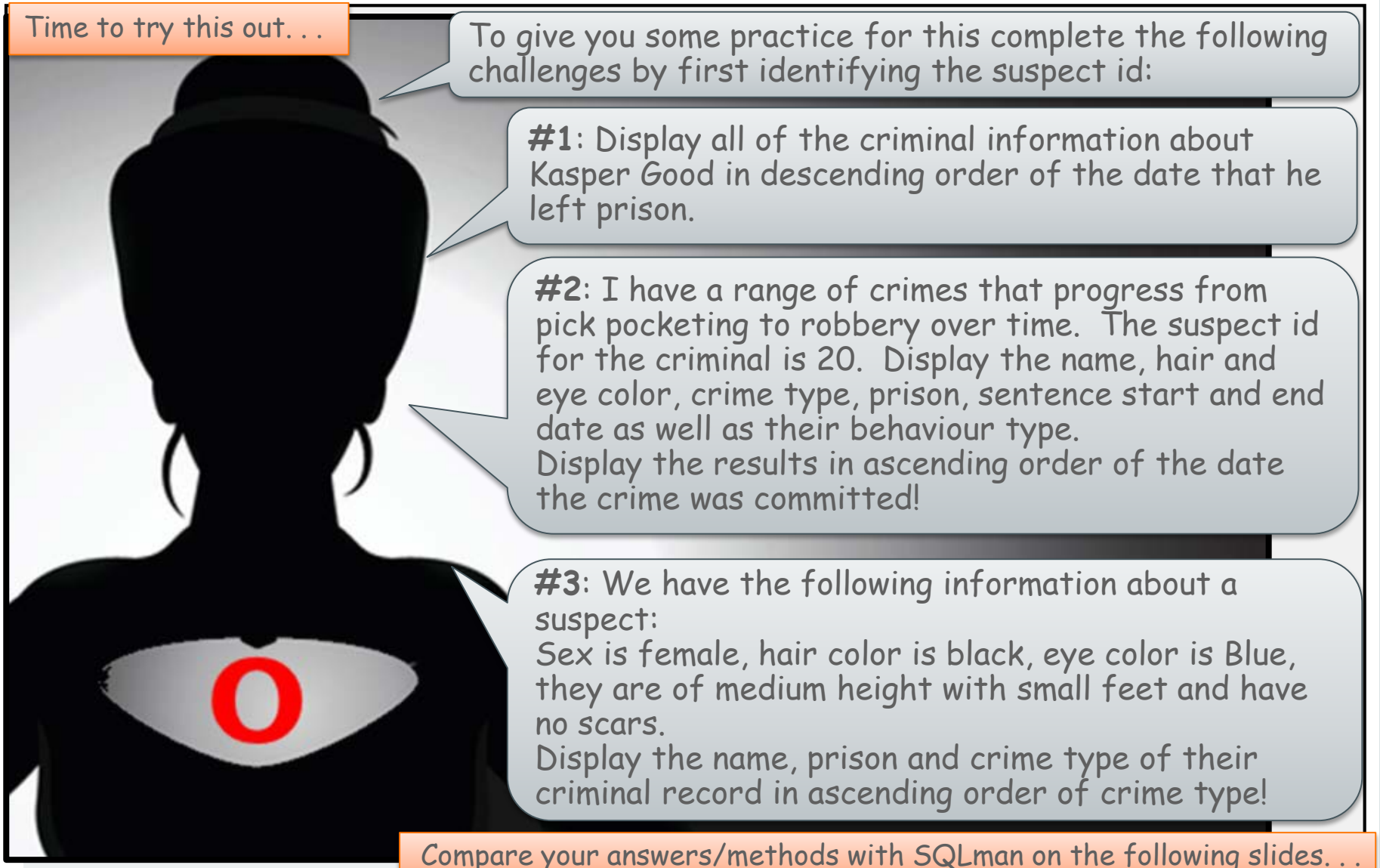
To see the crimes in ascending order (oldest first).

```
SELECT crime_id, crime_type, committed_date,
       prison,start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 100
ORDER BY committed_date;
```

To see the crimes in descending order (newest first).

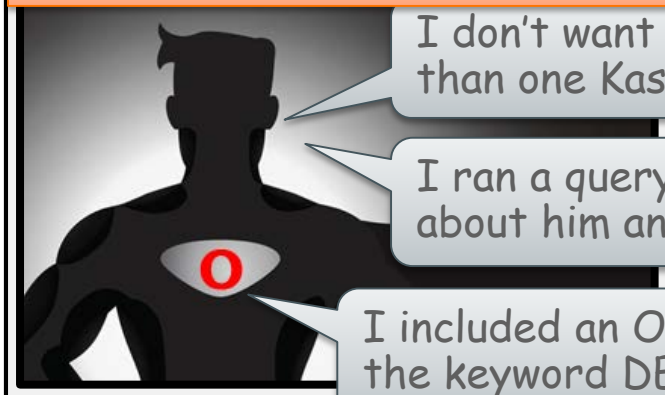
```
SELECT crime_id, crime_type, committed_date,
       prison,start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 100
ORDER BY committed_date DESC;
```

Some challenges using ORDER BY



Challenge Results - SQLman's Answers!

Did you solve challenge 1 the same way SQLman did?



I don't want to use his name in the query in case there is more than one Kasper Good in the system.

I ran a query to get the suspect_id first based on what I know about him and then use the suspect_id to search his criminal past.

I included an ORDER BY clause using the end_date field and using the keyword DESC to display the end_dates from newest to oldest.

Challenge #1a :
Get the suspect_id

suspect_id: 338

```
SELECT suspect_id
FROM suspects
WHERE name = 'Kasper Good' ;
```

Challenge #1b :
Get the criminal history using suspect_id 338.

```
SELECT name, crime_id, crime_type, committed_date, prison,
            start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 338
ORDER BY end_date DESC;
```

Challenge Results - SQLman's Answers!

Did you solve challenge 2 the same way SQLman did?

For challenge 2 I declared the column heading in the SELECT statement that matches the information required.

Although the date the crime was committed is not part of the SELECT statement I used it to specify the order that the results were returned in.

Challenge 2:
For suspect id 20 display the name, hair and eye color, crime type, prison, sentence start and end date as well as their behavior type.
Display the results in ascending order of the date the crime was committed

```
SELECT name, hair_color, eye_color, crime_type, prison,
            start_date, end_date, behavior
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 20
ORDER BY committed_date;
```

Challenge Results - SQLman's Answers!

Did you solve challenge 3 the same way SQLman did?

I need to find the suspect id based on their sex, hair color, eye color, height, feet size and scar data.

I included an ORDER BY clause using the crime type field to display the crimes in alphabetical order.

Challenge #3a :
Get the suspect_id

suspect_id: 59

```
SELECT *
FROM suspects
WHERE sex = 'Female' AND hair_color = 'Black'
AND eye_color = 'Blue' AND Height = 'Medium'
AND scars = 'No' AND feet_size = 'Small';
```

Challenge #3b :
Get the criminal history using suspect_id 59.

```
SELECT name, prison, crime_type
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 59
ORDER BY crime_type;
```

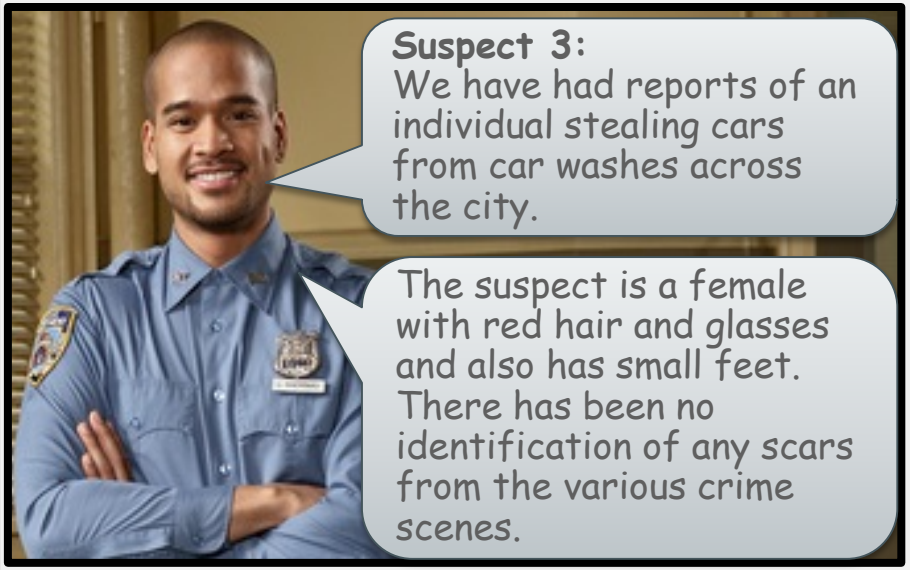
Using two tables to find our suspects!

Impressed? Wait till you see what else we can do. . .

So far we have only used the common column 'suspect_id' in both of our tables as our condition in the **WHERE** clause.

We can however use any information from both tables as part of our WHERE clause.

Think back to lesson 2, suspect 3, the police request where you could only return two suspects due to lack of information.



Suspect 3:
We have had reports of an individual stealing cars from car washes across the city.

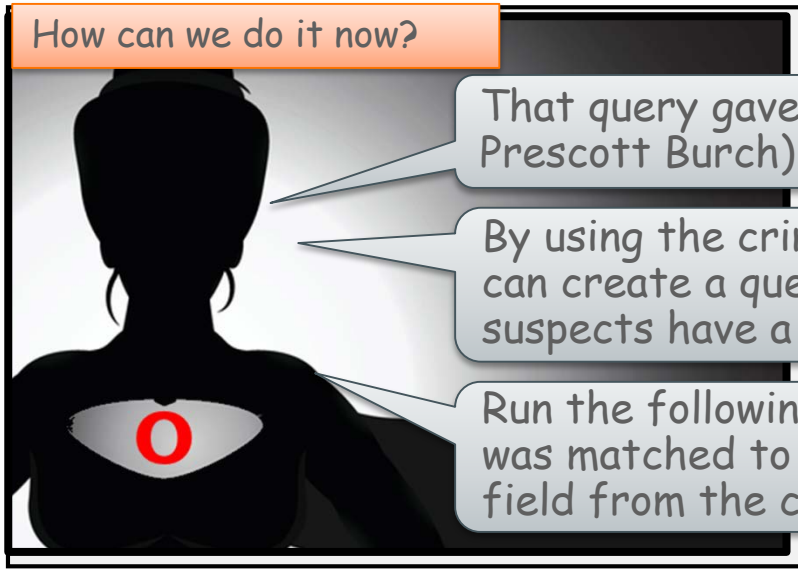
The suspect is a female with red hair and glasses and also has small feet. There has been no identification of any scars from the various crime scenes.

This was the only way you could use the information at the time

```
SELECT name, suspect_id
FROM suspects
WHERE sex        = 'Female'
AND   hair_color = 'Red'
AND   glasses    = 'Yes'
AND   scars      = 'No'
AND   feet_size  = 'Small';
```

Using two tables to find our suspects cont..

How can we do it now?



That query gave us two potential suspects (Amery Hatfield or Prescott Burch) as our result set.

By using the crimes table and the suspects table together we can create a query that will return results if any of the suspects have a history of car theft.

Run the following query that uses the original information that was matched to the suspects table as well as the crime type field from the crimes table.

```
SELECT name, suspect_id, crime_type, committed_date
FROM suspects NATURAL JOIN crimes
WHERE sex        = 'Female'
AND   hair_color = 'Red'
AND   glasses    = 'Yes'
AND   scars       = 'No'
AND   feet_size  = 'Small'
AND   crime_type = 'Car Theft'
ORDER BY committed_date DESC;
```

Using two tables to find our suspects cont..

We have narrowed the outstanding crime to a single suspect. . .

This tells us that the criminal responsible was Prescott Burch and that she has a history of car theft going back to 2007.

NAME	SUSPECT_ID	CRIME_TYPE	COMMITTED_DATE
Prescott Burch	400	Car Theft	01/12/2013
Prescott Burch	400	Car Theft	05/11/2012
Prescott Burch	400	Car Theft	09/19/2011
Prescott Burch	400	Car Theft	05/11/2010
Prescott Burch	400	Car Theft	05/11/2008
Prescott Burch	400	Car Theft	11/11/2007

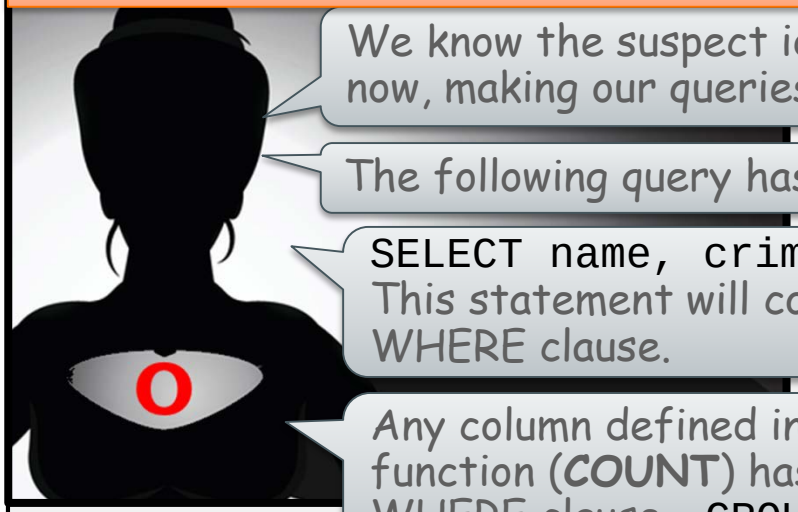
This set of results allows us to manually check how many car theft crimes this suspect has carried out (6).

SQL gives us the option to automate the task by using a **GROUP BY** clause that groups multiple records into a single summary row.

We will look at how we can use a COUNT operator to tell us how many of each type of crime has been committed.

Refining our results!

Let's have a closer look at using COUNT. . .



We know the suspect id (400) for Prescott Burch so we can use that now, making our queries much easier to read.

The following query has some new syntax you haven't seen before:

`SELECT name, crime_type, COUNT(crime_type)`
This statement will count the number of crimes specified in the WHERE clause.

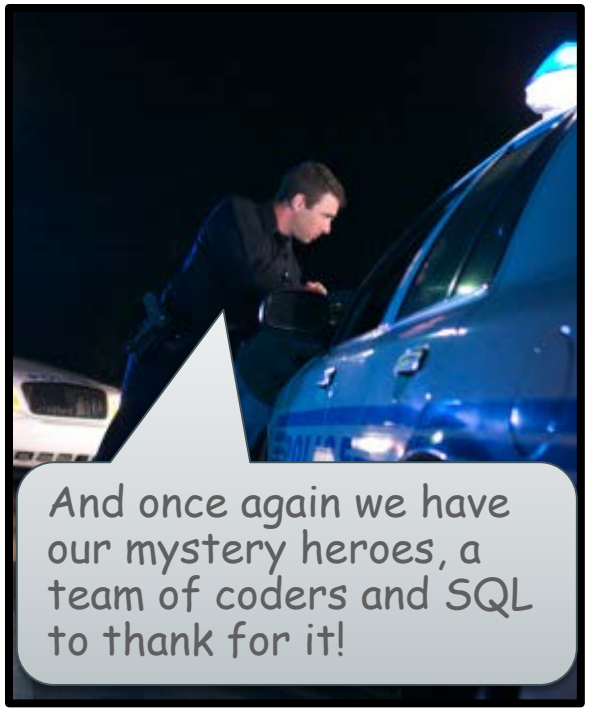
Any column defined in the SELECT statement that is not part of a function (**COUNT**) has to be placed in a **GROUP BY** clause after the WHERE clause - `GROUP BY name, crime_type;`.

```
SELECT name, crime_type, COUNT(crime_type)
FROM suspects NATURAL JOIN crimes
WHERE suspect_id = 400
AND crime_type = 'Car Theft'
GROUP BY name, crime_type;
```

Giving us these results :

NAME	CRIME_TYPE	COUNT(CRIME_TYPE)
Prescott Burch	Car Theft	6

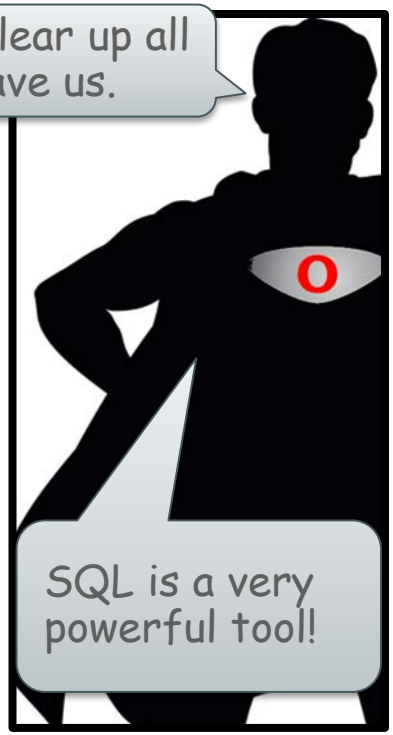
Prescott Burch, you may have thought we couldn't catch you but you were wrong!



And once again we have our mystery heroes, a team of coders and SQL to thank for it!

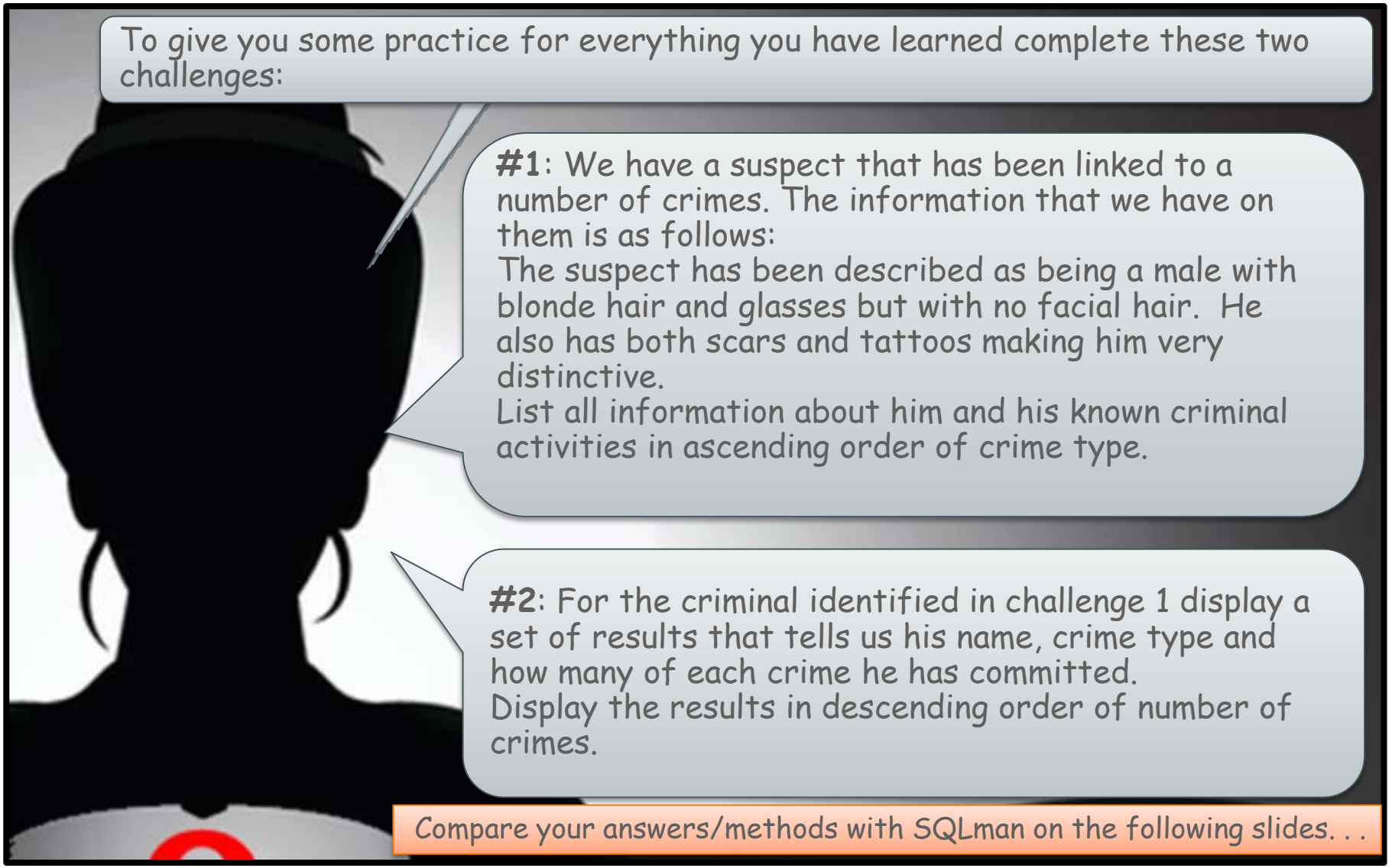
I'm glad we could clear up all of the cases you gave us.

Thank you, SQL team. With these new skills of yours no criminal is safe.



SQL is a very powerful tool!

Some final challenges!



To give you some practice for everything you have learned complete these two challenges:

#1: We have a suspect that has been linked to a number of crimes. The information that we have on them is as follows:
The suspect has been described as being a male with blonde hair and glasses but with no facial hair. He also has both scars and tattoos making him very distinctive.
List all information about him and his known criminal activities in ascending order of crime type.

#2: For the criminal identified in challenge 1 display a set of results that tells us his name, crime type and how many of each crime he has committed.
Display the results in descending order of number of crimes.

Compare your answers/methods with SQLman on the following slides. . .

Challenge Results - SQLman's Answers!

Did you solve final challenge 1 the same way SQLman did?

For challenge 1 I used all the information given and used the * to display all information held on the suspect from both tables.

You will see in the result set that although there is a suspect_id column in both tables it is only displayed once. That is because this is the common column between the tables.

Final Challenge 1:
We have a suspect that has been linked to a number of crimes. List all information about him and his known criminal activities in ascending order of crime type.

```
SELECT *
FROM suspects NATURAL JOIN crimes
WHERE sex = 'Male'
AND     hair_color = 'Blonde'
AND     scars = 'Yes'
AND     glasses = 'Yes'
AND     facial_hair = 'No'
AND     tattoos = 'Yes'
ORDER BY crime_type;
```

Challenge Results - SQLman's Answers!

Did you solve final challenge 2 the same way SQLman did?

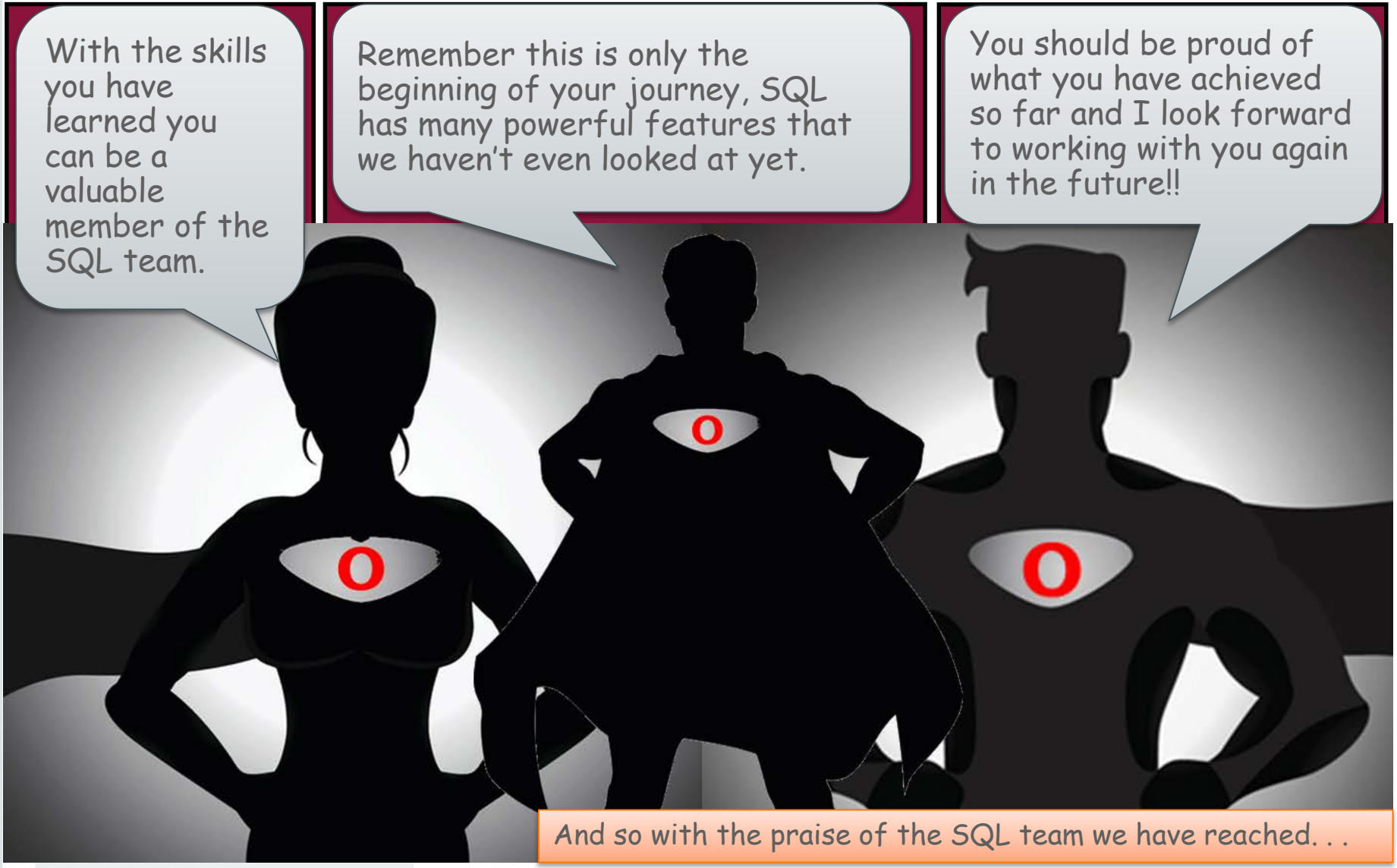
To display the information required I need to include the column names and the group function in the SELECT statement.

Remember if we want to display any column information that's not part of the group function we need to include it in a GROUP BY clause.

Final Challenge #2 : Display criminal information about a criminal known as Myles Tucker! Display his name, crime type and how many of each crime he has committed in descending order of number of crimes.

```
SELECT name, crime_type, COUNT(crime_type)
FROM suspects NATURAL JOIN crimes
WHERE sex = 'Male'
AND     hair_color = 'Blonde'
AND     scars = 'Yes'
AND     glasses = 'Yes'
AND     facial_hair = 'No'
AND     tattoos = 'Yes'
GROUP BY name, crime_type
ORDER BY COUNT(crime_type) DESC;
```

Congratulations!



Summary

In this lesson, you should have learned how to:

- Run a script
- Create Natural Joins
- Use Two tables in a query
- Use Order By
- Use Group By and Count


ORACLE
ACADEMY