Guided Project: Answering Business Questions Using SQL

This is a guided project for the Dataquest Certificate:
Intermediate SQL for Data Analysis

Database

The database being analysed is the Chinook sample database for learning SQL, especially SQLite
https://www.sqlitetutorial.net/sqlite-sample-database/

Connect to Database

In [1]:

```
%%capture
%load_ext sql
%sql sqlite:///chinook.db
```

Out[1]:

```
'Connected: None@chinook.db'
```

In [3]:

```
%%sql
SELECT
    name,
    type
FROM sqlite_master
WHERE type in ('table', 'view');
```

```
Done.
```

Out[3]:

name	type
album	table
artist	table
customer	table
employee	table
genre	table
invoice	table
invoice_line	table
media_type	table
playlist	table
playlist_track	table
track	table

Select New Albums to Add to the Store

There are four possible new albums to add to the store:
Artist/Genre
Regal : Hiphop
Red Tone : Punk

Meteor and the Girls : Pop
Slim Jim Bites : Blues

Three of these will be offered for sale. In order to decide which 3, an analysis of which genres sell best in the USA will be conducted

Write a query that returns for each genre:

absolute sales

sales percentage

In [27]:

```
%%sql
WITH track_info AS
    (
    SELECT
        t.name AS track_name,
        g.name AS genre_name,
        SUM(il.quantity) AS tracks_sold
    FROM track t
    LEFT JOIN invoice_line il ON il.track_id = t.track_id
    LEFT JOIN genre g ON g.genre_id = t.genre_id
    GROUP BY t.track_id
    ),

    genre_info AS
    (
    SELECT
        t.genre_name,
        SUM(t.tracks_sold) as tracks_sold
    FROM track_info t
    GROUP BY t.genre_name
    )


SELECT
    g.genre_name,
    g.tracks_sold,
    CAST(g.tracks_sold AS FLOAT)
         / (SELECT
                SUM(g.tracks_sold)
            FROM genre_info g) * 100
        AS sold_percentage

FROM genre_info g
ORDER BY sold_percentage DESC;
```

Done.

Out[27]:

genre_name	tracks_sold	sold_percentage
Rock	2635	55.3920538154299
Metal	619	13.012402774858103
Alternative & Punk	492	10.342652932520496
Latin	167	3.510615934412445
R&B/Soul	159	3.3424427159974774
Blues	124	2.606684885431995
Jazz	121	2.543619928526382
Alternative	117	2.4595333193188984
Easy Listening	74	1.5556022703384484

Pop	63	1.3243640950178683
Electronica/Dance	55	1.156190876602901
Classical	47	0.9880176581879335
Reggae	35	0.7357578305654824
Hip Hop/Rap	33	0.6937145259617407
Heavy Metal	8	0.16817321841496743
Soundtrack	5	0.10510826150935462
TV Shows	2	0.04204330460374186
Drama	1	0.02102165230187093
Bossa Nova	None	None
Comedy	None	None
Opera	None	None
Rock And Roll	None	None
Sci Fi & Fantasy	None	None
Science Fiction	None	None
World	None	None

Ordered by sales percentage, the four albums are:

Red Tone : Punk (10.34%)
Meteor and the Girls : Pop (1.32%)
Slim Jim Bits : Blues (2.61%)
Regal : Hiphop (0.69%)

Based off how popular each of the genres are, the 3 albums I would recommend to add to the store are:
Red Tone's
Meteor and the Girls'
Slim Jim Bits'

This means I would not recommend stocking Regal's album

Analyse Sales Support Agents

The store has several Sales Support Agents, who assist customers with sales. How do they compare to each other?

Write a query that finds the total dollar amount of sales assigned to each sales support agent within the company. Add any extra attribute you find are relevant

In [46]:

```
%%sql
SELECT
    e.first_name || ' ' || e.last_name AS employee_name,
    e.title,
    e2.first_name || ' ' || e2.last_name AS supervisor_name
FROM employee e
LEFT JOIN employee e2 ON e.reports_to = e2.employee_id
ORDER BY supervisor_name
```

Done.

Out[46]:

employee_name	title	supervisor_name
Andrew Adams	General Manager	None

Nancy Edwards	Sales Manager	Andrew Adams
Michael Mitchell	IT Manager	Andrew Adams
Robert King	IT Staff	Michael Mitchell
Laura Callahan	IT Staff	Michael Mitchell
Jane Peacock	Sales Support Agent	Nancy Edwards
Margaret Park	Sales Support Agent	Nancy Edwards
Steve Johnson	Sales Support Agent	Nancy Edwards

In [44]:

```
%%sql
WITH customer_info AS
    (SELECT
        c.customer_id,
        c.support_rep_id,
        SUM(i.total) AS total_purchases
    FROM invoice i
    INNER JOIN customer c ON c.customer_id = i.customer_id
    GROUP BY c.customer_id
    ),

    employee_info AS
    (SELECT
        e.employee_id,
        SUM(c.total_purchases) AS total_sales
     FROM employee e
     LEFT JOIN customer_info c ON c.support_rep_id = e.employee_id
     GROUP BY e.employee_id
    )

SELECT
    e.first_name || ' ' || e.last_name AS employee_name,
    ei.total_sales,
    e.title,
    e.hire_date,
    e2.first_name || ' ' || e2.last_name AS supervisor_name
FROM employee e
LEFT JOIN employee_info ei ON ei.employee_id = e.employee_id
LEFT JOIN employee e2 ON e.reports_to = e2.employee_id
WHERE e.title = 'Sales Support Agent'
ORDER BY e.hire_date
```

Done.

Out[44]:

employee_name	total_sales	title	hire_date	supervisor_name
Jane Peacock	1731.51	Sales Support Agent	2017-04-01 00:00:00	Nancy Edwards
Margaret Park	1584.0	Sales Support Agent	2017-05-03 00:00:00	Nancy Edwards
Steve Johnson	1393.92	Sales Support Agent	2017-10-17 00:00:00	Nancy Edwards

There are 3 sales support agents, out of 8 employees. They all work under Nancy Edwards, who is the Sales Manager.

The obvious conclusion drawn from the data is that the longer a Sales Support Agent has been at their current position, the better they are at their job, and the more sales they make.

Analyse by Country

Find, per country:

Total number of customers

Total value of sales
Average value of sales per customer
Average order value

Countries with only one customer should be collected in an 'Other' group, sorted to the bottom

In [119]:

```
%%sql
WITH customer_info AS
    (SELECT
        c.customer_id,
        CASE
             WHEN(SELECT
                     COUNT(customer_id)
                 FROM customer
                 WHERE country = c.country) = 1 THEN 'Other'
             ELSE c.country
             END AS country,
        SUM(i.total) AS total_purchases,
        COUNT(i.total) AS number_of_purchases
    FROM invoice i
    INNER JOIN customer c ON c.customer_id = i.customer_id
    GROUP BY c.customer_id
    ),

    country_info AS
    (SELECT
         c.country,
         SUM(c.total_purchases) AS total_sales,
         SUM(c.number_of_purchases) AS total_purchases,
         COUNT(c.total_purchases) AS number_of_customers,
         CASE
            WHEN c.country = 'Other' THEN 1
            ELSE 0
         END AS other
     FROM customer_info c
     GROUP BY country
    )

SELECT
    ci.country,
    ci.number_of_customers,
    ci.total_sales,
    CAST(ci.total_sales AS FLOAT) / ci.number_of_customers AS sales_per_customer,
    CAST(ci.total_sales AS FLOAT) / ci.total_purchases AS avg_order_value
FROM country_info ci
ORDER BY ci.other, total_sales DESC
```

Done.

Out[119]:

country	number_of_customers	total_sales	sales_per_customer	avg_order_value
USA	13	1040.49	80.03769230769231	7.942671755725191
Canada	8	535.59	66.94875	7.047236842105264
Brazil	5	427.67999999999995	85.53599999999999	7.011147540983606
France	5	389.07	77.814	7.7814
Germany	4	334.62	83.655	8.161463414634147
Czech Republic	2	273.24	136.62	9.108
United Kingdom	3	245.51999999999998	81.83999999999999	8.768571428571429
Portugal	2	185.13	92.565	6.3837931034482756
India	2	183.14999999999998	91.57499999999999	8.72142857142857
Other	15	1094.94	72.99600000000001	7.448571428571429

Find the amount of purchases that our album purchases vs individual track purchases

Goal:

From each invoice, compare all the tracks bought to the album of the first track. If they match, it is an album purchase.

In [227]:

```
%%sql

WITH invoice_track AS
    (
    SELECT
        il.invoice_id,
        MIN(il.track_id) AS track_id

        FROM invoice_line il
        GROUP BY il.invoice_id
    )

SELECT
    album_purchase,
    COUNT(invoice_id) AS number_of_invoices,
    CAST(COUNT(invoice_id) AS FLOAT) / (
                        SELECT COUNT(*) FROM INVOICE
                        ) * 100 AS percentage_of_invoices
FROM
    (SELECT
        i.invoice_id,
        CASE
            WHEN
                 (

                /*Get all the tracks*/
                SELECT t1.track_id FROM track t1
                /*Where the album is*/
                WHERE t1.album_id = (
                                    /*the album that*/
                                    SELECT t2.album_id from track t2
                                    /*the first track of the album belongs to*/
                                    WHERE t2.track_id = (
                                                        SELECT track_id FROM invoice_tr
ack
                                                        WHERE i.invoice_id = invoice_id
                                                        )
                                    )
                EXCEPT

                    /*Get all the tracks*/
                    SELECT il.track_id FROM invoice_line il
                    /*For this invoice*/
                    WHERE il.invoice_id = i.invoice_id

                ) IS NULL

                 AND

                (
                /*Get all the tracks*/
                SELECT il.track_id FROM invoice_line il
                /*For this invoice*/
                WHERE il.invoice_id = i.invoice_id

                 EXCEPT
```

```
                /*Get all the tracks*/
                SELECT t1.track_id FROM track t1

                /*Where the album is*/
                WHERE t1.album_id = (

                                    /*the album that*/
                                    SELECT t2.album_id from track t2

                                    /*the first track of the invoice belongs to*/
                                    WHERE t2.track_id = (
                                                        SELECT track_id FROM invoice_tr
ack
                                                        WHERE i.invoice_id = invoice_id
                                                        )
                                    )
                ) IS NULL

            THEN 'yes'
            ELSE 'no'
            END AS album_purchase

    FROM invoice i
    )
GROUP BY album_purchase
```

Done.

Out[227]:

album_purchase	number_of_invoices	percentage_of_invoices
no	500	81.43322475570032
yes	114	18.566775244299674

The store should continue to buy whole albums, as whole albums represent nearly a fifth of their sales