UDACITY

‹ Back to Data Foundations

Music SQL Database

REVIEW

CODE REVIEW 8

ANNOTATIONS 1

HISTORY

▾ SQL3.txt 8

```
/* Which country has the highest top ten customers and how much was spent? */
```

AWESOME

Nice work here!
I admired how organized and structured the work was. Including descriptive headings to each query provided ease of access and a clear walk through the queries and what the queries were to answer. This is great! Please keep it up.✓

Why we should comment code.

```

SELECT Customer.Country, SUM(Invoice.Total) AS Total_amount_spent
```

AWESOME

Bravo!
There is a great consistency I noticed in the provided queries. The good practice of selecting only columns that are relevant to the analysis to be made. This is excellent work due to query run times.
SQL Tutorial: How To Write Better Queries

```
FROM customer
JOIN Invoice
ON Customer.CustomerId = Invoice.CustomerId
JOIN InvoiceLine
ON InvoiceLine.InvoiceId = Invoice.InvoiceId
GROUP BY country
ORDER BY Total_amount_spent DESC
LIMIT 10;
```

AWESOME

Nice work!
Limiting the records to `10` rows, this made the visualizations very clear and visually appealing. This is awesome!✓

```



/* What is the album name that sold most in USA, in which month and how many? */

SELECT  Strftime('%m', Invoice.InvoiceDate) as Month, Album.Title as Album_name, COUNT(*) Number_sold, SUM(Invoice.Total)
FROM Invoice
JOIN InvoiceLine
```

AWESOME

JOINS and aggregations are properly included in all the queries as required. Nicely done!

```
ON InvoiceLine.InvoiceId = Invoice.InvoiceId
JOIN Album
On Album.AlbumId = Track.AlbumId
JOIN Track
ON Track.TrackId = InvoiceLine.TrackId
WHERE BillingCountry = "USA"
GROUP BY Month
ORDER BY Number_sold DESC;
```

SUGGESTION

This is good way of using `group by and order by` with column names. Also, I think another great way of doing so is by using just the column numbers. This way seems to be more efficient as it reduces query run times. Speed is a great factor of queries.💪

```



/*Which top 5 tracks is the highest sold in the album of song in the United Kingdom? */
SELECT Track.Name as Track_name, Album.Title as Album_title, (count(Quantity) * sum(InvoiceLine.UnitPrice)) as Unit_sold,
FROM Track
JOIN Album
ON Album.AlbumId = Track.AlbumId
JOIN InvoiceLine
On InvoiceLine.TrackId = Track.TrackId
Join Invoice
On Invoice.InvoiceId = InvoiceLine.InvoiceId
WHERE BillingCountry = 'United Kingdom'
GROUP BY Track_name
ORDER BY Total DESC
LIMIT 5;
```

AWESOME

Good job always using `;` to terminate a query. This is a good practice. It makes your work very professional. You did great with the queries.

The Most Effective Way to Write Effective SQL: Change Your Thinking Style

```
/*In how many states does each support agent have the highest sales?*/

WITH t1 as (SELECT e.EmployeeId, e.LastName lastname, e.FirstName firstname, c.state as State, SUM(i.Total) totalsales
            FROM Employee e
            JOIN Customer c
            ON e.EmployeeId = c.SupportRepId
            JOIN Invoice i
            ON c.CustomerId = i.CustomerId
            GROUP BY EmployeeId, 4),

     t2 as (SELECT State, MAX(totalsales) totalsales
            FROM t1
            GROUP BY State)

SELECT (t1.firstname ||" "||  t1.lastname) as SalesSupportRep, COUNT(t1.State) NumberOfStates, SUM(t1.totalsales) TotalSal
```

AWESOME

Good job using `||` to concatenate first and last name into salesuportRep. This is good skill. There is another great way of doing this using the `concat` function. Consider giving it a look at your free time.

```
FROM t1
JOIN t2
ON t1.State = t2.State AND t1.totalsales = t2.totalsales
GROUP BY EmployeeId
ORDER BY TotalSales DESC;
```

AWESOME

Brilliant work with the provided queries in this submission!
They are all well structured, neatly presented, simple to understand yet very insightful. Well done!🙏

RETURN TO PATH