SQL as a Second Language

John HR Schuster

Version 2.1f, 12/28/2018

Table of Contents

SQL as a Second Language (SASL) was a course that I taught many moons ago. It is being revisited now to help answer some SQL questions brought up by those whose are taking their SQL Skills to the next level.

A PDF version of this web site is available at this Link

1. Introduction

This SQL As a Second Language version will use the Teradata SQL syntax.

Examples

> A teacher, a really good teacher, is never a giver of truth; he is a guide, a pointer to truth
>
> — Bruce Lee

1.1. Chinook Database

The training database will be the Chinook database on music record sales.

Reference: https://github.com/lerocha/chinook-database

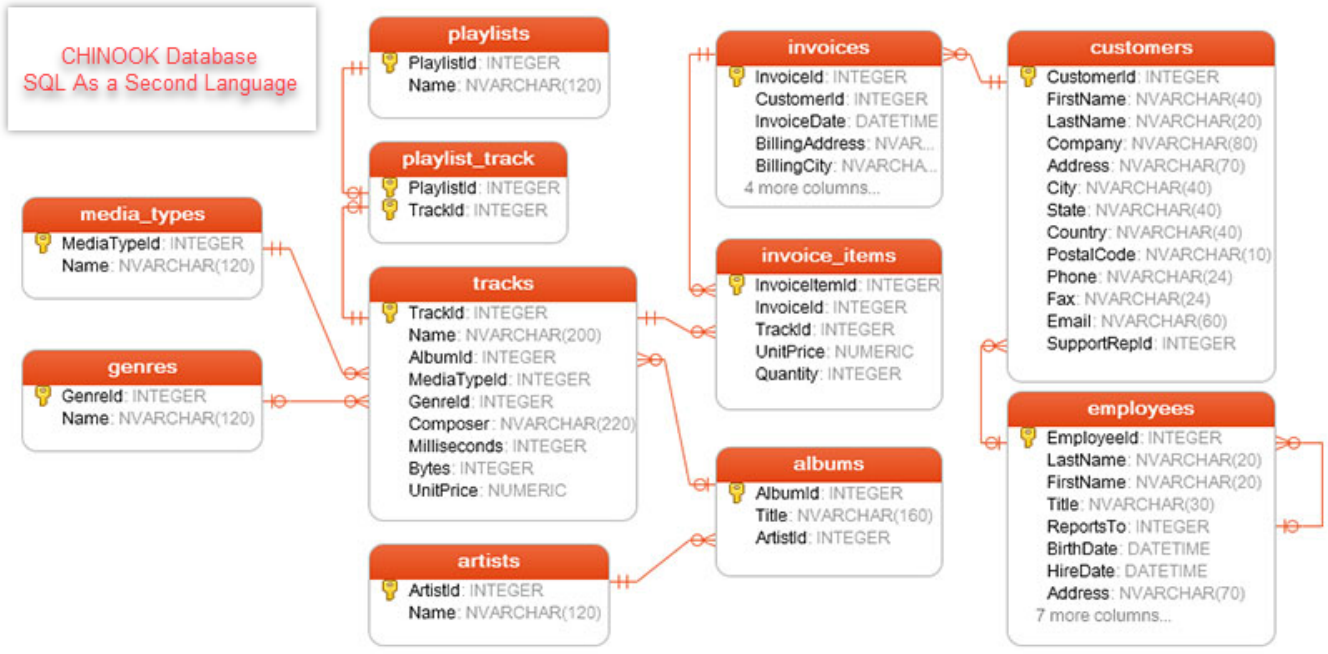

Figure 1. Chinook Database Diagram

2. Questions

To ask a **New question** to be added to this list, please email it to john.schuster@PhoenixWorkgroup.com.

How can I get a count of the number of rows in some of the tables of a database?

View this query tableCounts

How can a get a list of the objects (Tables, View, Procedures) for a specific database?

View This query dbObjects

How can I find out the columns and metadata about a specific object (Table, View)?

View this query tableColumns

How can I find where a specific column exist in a database?

View this query findColumn

How do I Join two tables together?

View this query joinQuery Part B

How do I restrict the results by a column that `contains` some string?

View this query joinQuery Part C

How do I sort the results by multiple columns?

View this query joinQuery Part C

How can I best use "Prompts" in SQL Assistant?

unlinked

3. Queries

3.1. tableCounts

SQL Example

```
/*
=== tableCounts - Count rows in all Chinook tables

==== TOPICS

* Aggragate function count.
* UNION multiple queries.
* Sorting aka order by.

==== TIPS

* Use single quote (') to denote text.
* Use doble-quotes for renaming objects.
* Indent and one column per row.
* Put commas at the begining of the 2nd column.
* UNION allows joining of multiple queries, same number and data type of column.
* First query in union determine sizes and names of columns.
* Start small and build on to query.
* Order by can be by the ordinal number of the column
  (Column 1 is tableName, Column 2 is Rows)
*/

Select
    'Album          ' as tableName   ①
    , count(*) as "Rows" from Album  ②

UNION
Select
    'Artist' as tableName
    , count(*) from Artist

UNION
Select
    'Customer' as tableName
    , count(*) from Customer

UNION
Select
    'Employee' as tableName
    , count(*) from Employee

UNION
Select
    'Genre' as tableName
    , count(*) from Genre

UNION
```

```
Select
    'Invoice' as tableName
    , count(*) from Invoice

UNION
Select
    'Invoice Line' as tableName
    , count(*) from InvoiceLine

UNION
Select
    'Media Type' as tableName
    , count(*) from MediaType

UNION
Select
    'Playlist' as tableName
    , count(*) from Playlist

UNION
Select
    'Playlist Track' as tableName
    , count(*) from PlaylistTrack

UNION
Select
    'Track' as tableName
    , count(*) from Track

Order by 2 desc  ③
```

① First query in union determine sizes and names of columns.

② count(*) is an aggregate function

③ Order by can be by the ordinal number of the column (Column 1 is tableName, Column 2 is Rows)

SQL download link click here

Table 1. Results

tableName	Rows
Playlist Track	8715
Track	3503
Invoice Line	2240
Invoice	412
Album	347
Artist	275
Customer	59

tableName	Rows
Genre	25
Playlist	18
Employee	8
Media Type	5

Back to Introduction.

3.2. dbObjects

SQL Example

```
/*
=== dbObjects - Get a list of all the objects in a specific database

==== TOPICS

* This is a Teradata specific command

==== TIPS

* Change the database name from `chinook` to the database you are interested in
*/

help database chinook
```

SQL download link click here

Table 2. Results

Table/View/Macro name	Kind	Comment	Protection	Creator Name	Commit Option	Transaction Log	Table/View/Macro Dictionary Name	Table/View/Macro SQL Name	Table/View/Macro Name UEscape	Creator Dictionary Name	Creator SQL Name	Creator Name UEscape
Album	T	NULL	N	WINDUSER	N	Y	Album	Album		WINDUSER	WINDUSER	
Artist	T	NULL	N	WINDUSER	N	Y	Artist	Artist		WINDUSER	WINDUSER	
Customer	T	NULL	N	WINDUSER	N	Y	Customer	Customer		WINDUSER	WINDUSER	
Employee	T	NULL	N	WINDUSER	N	Y	Employee	Employee		WINDUSER	WINDUSER	
Genre	T	NULL	N	WINDUSER	N	Y	Genre	Genre		WINDUSER	WINDUSER	
Invoice	T	NULL	N	WINDUSER	N	Y	Invoice	Invoice		WINDUSER	WINDUSER	
InvoiceLine	T	NULL	N	WINDUSER	N	Y	InvoiceLine	InvoiceLine		WINDUSER	WINDUSER	
mComedyList	M	NULL	F	DBC	N	Y	mComedyList	mComedyList		DBC	DBC	

Table/View/Macro name	Kind	Comment	Protection	Creator Name	Commit Option	Transaction Log	Table/View/Macro Dictionary Name	Table/View/Macro SQL Name	Table/View/Macro Name UEscape	Creator Dictionary Name	Creator SQL Name	Creator Name UEscape
MediaType	T	NULL	N	WINDUSER	N	Y	MediaType	MediaType		WINDUSER	WINDUSER	
Playlist	T	NULL	N	WINDUSER	N	Y	Playlist	Playlist		WINDUSER	WINDUSER	
PlaylistTrack	T	NULL	N	WINDUSER	N	Y	PlaylistTrack	PlaylistTrack		WINDUSER	WINDUSER	
Track	T	NULL	N	WINDUSER	N	Y	Track	Track		WINDUSER	WINDUSER	
vComedyTrack	V	NULL	F	DBC	N	Y	vComedyTrack	vComedyTrack		DBC	DBC	

Back to Introduction.

3.3. tableColumns

SQL Example

```
/*
=== tableColumns - Get a list of all the columns for a specific table or object in a
known database

==== TOPICS

* This is a Teradata specific command

==== TIPS

* Change the database name from `chinook` to the database you are interested in
*/

help table chinook.invoice
```

SQL Download link click here

Table 3. Results

ColumnName	Type	Comment	Nullable	Format	Title	MaxLength	DecimalTotalDigits	DecimalFractionalDigits	RangeLow	RangeHigh	UpperCase	Table/View?	DefaultValue	CharType	IdColType	UDTName	Temporal	ColumnDictionaryName	ColumnSQLName	ColumnNameUEscape	DictionaryTitle	SQLTitle	TitleUEscape	UDTDictionaryName	UDTSQLName	UDTNameUEscape
InvoiceId	I	NULL	N	-(10)9	NULL	4	NULL	NULL	NULL	NULL	N	T	NULL	NULL	GD	NULL	N	InvoiceId	InvoiceId		NULL	NULL	NULL	NULL	NULL	NULL
CustomerId	I	NULL	N	-(10)9	NULL	4	NULL	NULL	NULL	NULL	N	T	NULL	NULL	NULL	NULL	N	CustomerId	CustomerId		NULL	NULL	NULL	NULL	NULL	NULL

Column Name	Type	Comment	Nullable	Format	Title	MaxLength	DecimalTotalDigits	DecimalFractionalDigits	RangeLow	RangeHigh	UpperCase	Table/View?	DefaultValue	CharType	IdColType	UDTName	Temporal	ColumnDictionaryName	ColumnSQLName	ColumnNameUEscape	DictionaryTitle	SQLTitle	TitleUEscape	UDTDictionaryName	UDTSQLName	UDTNameUEscape
InvoiceDate	DA	NULL	N	yyyy-mm-dd	NULL	4	NULL	NULL	NULL	NULL	N	T	NULL	NULL	NULL	NULL	N	InvoiceDate	InvoiceDate		NULL	NULL	NULL	NULL	NULL	NULL
BillingAddress	CV	NULL	Y	X(70)	NULL	70	NULL	NULL	NULL	NULL	N	T	NULL	1	NULL	NULL	N	BillingAddress	BillingAddress		NULL	NULL	NULL	NULL	NULL	NULL
BillingCity	CV	NULL	Y	X(40)	NULL	40	NULL	NULL	NULL	NULL	N	T	NULL	1	NULL	NULL	N	BillingCity	BillingCity		NULL	NULL	NULL	NULL	NULL	NULL
BillingState	CV	NULL	Y	X(40)	NULL	40	NULL	NULL	NULL	NULL	N	T	NULL	1	NULL	NULL	N	BillingState	BillingState		NULL	NULL	NULL	NULL	NULL	NULL
BillingCountry	CV	NULL	Y	X(40)	NULL	40	NULL	NULL	NULL	NULL	N	T	NULL	1	NULL	NULL	N	BillingCountry	BillingCountry		NULL	NULL	NULL	NULL	NULL	NULL
BillingPostalCode	CV	NULL	Y	X(10)	NULL	10	NULL	NULL	NULL	NULL	N	T	NULL	1	NULL	NULL	N	BillingPostalCode	BillingPostalCode		NULL	NULL	NULL	NULL	NULL	NULL

Column Name	Type	Comment	Nullable	Format	Title	MaxLength	Decimal Total Digits	Decimal Fractional Digits	RangeLow	RangeHigh	UpperCase	Table/View?	Default value	CharType	IdColType	UDTName	Temporal	Column Dictionary Name	Column SQLName	Column Name UEscape	Dictionary Title	SQLTitle	Title UEscape	UDT Dictionary Name	UDTSQLName	UDTName UEscape
Total	D	NULL	N	----------.99	NULL	8	10	2	NULL	NULL	N	T	NULL	NULL	NULL	NULL	N	Total	Total		NULL	NULL	NULL	NULL	NULL	NULL

ColumnType defined the data type of a column. Reference: [ColumnType]

Back to Introduction.

3.4. findColumn

SQL Example

```
/*
=== findColumn - Find out what objects (Table, View) where a specific named column
exist

==== TOPICS

* This is a Teradata specific command
* The list of columns are the ones that are the most important,
there are many other columns available. Use a single column name (*) to see them all.

==== TIPS

* CoumnTypes Reference:
http://developer.teradata.com/doc/connectivity/tdnetdp/14.00/webhelp/DataTypeMappings.
html
* Replace `CustomerID` with the column you are interested in

*/

select
    ColumnName
    ,DatabaseName
    ,TableName
    ,ColumnFormat
    ,ColumnType
    ,ColumnLength
from dbc.columnsX
where ColumnName = 'CustomerID'
```

SQL Download link click here

Table 4. Results

ColumnName	DatabaseName	TableName	ColumnFormat	ColumnType	ColumnLength
CustomerId	Chinook	Invoice	-(10)9	I	4
CustomerId	Chinook	Customer	-(10)9	I	4

The ColumnType identifies the data type of the column. Reference: Column Type

Back to Introduction.

3.5. joinQuery Part A

Objective: Get a list of tracks from the **Artist** `Aerosmith` where the **Composer** is `Joe Perry`.

Approach: Begin with a simple query to get one element of the objective. In this query we want to get the **ArtistID** for `Aerosmith`

SQL Example

```
/*
=== joinQuery - Multiple Table join with result restriction

==== TOPICS

* This query joins multiple table together

==== TIPS

* Table alias shorten the typing needed to complete the query
1-2 character alias recommended.
* Column alias help to give meaningful, non conflicting names.
* Results being restricted with a a `where` clause
*/

Select
   AR.ArtistID
   ,AR.name as artistName  ①
From Artist AR  ②
where artistName = 'Aerosmith'
```

① Column alias, notice name is `artistName`

② Table alias, use alias on every instance of columns from that table

SQL Download link click here

Table 5. Results

ArtistId	artistName
3	Aerosmith

Back to Introduction.

3.6. joinQuery Part B

Objective: Get a list of tracks from the **Artist** `Aerosmith` where the **Composer** is `Joe Perry`.

Approach: Add on to the initial query to get a list of all the **Albums** for the **Artist** `Aerosmith`

SQL Example

```
/*
=== joinQuery - Multiple Table join with result restriction

Part B - Add the second table for the Join

==== TOPICS

* This query joins multiple table together

==== TIPS

* Table alias shorten the typing needed to complete the query
1-2 character alias recommended.
* Column alias help to give meaningful, non conflicting names.
* Coulmns for join must match in data type and size
* Good practice is that System Generated Primary Key column name be named with `ID` at
the end
* Primary Key and Foreign Key matching columns should usew the same name

*/

Select
   AR.ArtistID
   ,AL.AlbumID
   ,AR.name as artistName  ①
   ,AL.AlbumTitle
From Artist AR  ②

inner Join Album AL ②
on AR.ArtistID = AL.ArtistID  ③

where artistName = 'Aerosmith'
```

① Column alias, notice name is `artistName`

② Table alias, use alias on every instance of columns from that table

③ The common column used to join the two tables together

SQL Download link click here

Table 6. Results

ArtistId	AlbumId	artistName	AlbumTitle
3	5	Aerosmith	Big Ones

Back to Introduction.

3.7. joinQuery Part C

Objective: Get a list of tracks from the **Artist** `Aerosmith` where the **Composer** is `Joe Perry`.

Approach: Complete the request by joining the `Title` table and a row restriction using the `like` with the wildcard `%` character.

SQL Example

```
/*
=== joinQuery - Multiple Table join with result restriction

Part C - Add the third table and composer restriction

==== TOPICS

* List of all the `Aerosmith` tracks that `Joe Perry` was one of the cpmposers
* This query joins multiple tables together
* Primary and Foreign keys are system generated and always exist

==== TIPS

* Table alias shorten the typing needed to complete the query
1-3 character alias recommended.
* Column alias help to give meaningful, non conflicting names.
Good practice is to have table alias used everywhere if used
* Coulmns for join must match in data type and size
* Good practice is that System Generated Primary Key column name be named with `ID` at
the end
* Primary Key and Foreign Key matching columns should usew the same name
* `like` used to find a column that contains some string`, note the % indicating any
number of characters before/after
* Order by can use either 1. Actual column names, 2. Alias column names, 3. Ordinal
position is results list
* Alias column names use lowerUpper naming to identify them as alias
only works because original names are camelback naming.

*/

Select
   AR.ArtistID
   ,AL.AlbumID
   ,T.TrackID
   ,AR.name as artistName  ①
   ,AL.AlbumTitle
   ,T.Name as trackName    ①
   ,T.Composer
From Artist AR  ②

inner Join Album AL ②
on AR.ArtistID = AL.ArtistID  ③
```

```
inner join Track T
on AL.AlbumID = T.AlbumID

where AR.Name = 'Aerosmith'
  and T.Composer like '%Joe Perry%'  ④

order by AL.AlbumTitle
         ,trackName  ⑤
-- order by 5,6 ⑥
```

① Column alias, notice name is `artistName`

② Table alias, use alias on every instance of columns from that table

③ The common column used to join the two tables together

④ The `like` with wildcard character `%` used on `Composer` column

⑤ The order by has both actual column name `AL.AlbumTitle` and an alias column `trackName`

⑥ Shows alternative method of `order by` using ordinal column numbers

SQL Download link click here

Table 7. Results

ArtistId	AlbumId	TrackId	artistName	AlbumTitle	trackName	Composer
3	5	31	Aerosmith	Big Ones	Blind Man	Steven Tyler, Joe Perry, Taylor Rhodes
3	5	34	Aerosmith	Big Ones	Crazy	Steven Tyler, Joe Perry, Desmond Child
3	5	29	Aerosmith	Big Ones	Cryin'	Steven Tyler, Joe Perry, Taylor Rhodes
3	5	27	Aerosmith	Big Ones	Dude (Looks Like A Lady)	Steven Tyler, Joe Perry, Desmond Child
3	5	35	Aerosmith	Big Ones	Eat The Rich	Steven Tyler, Joe Perry, Jim Vallance
3	5	37	Aerosmith	Big Ones	Livin' On The Edge	Steven Tyler, Joe Perry, Mark Hudson
3	5	24	Aerosmith	Big Ones	Love In An Elevator	Steven Tyler, Joe Perry
3	5	25	Aerosmith	Big Ones	Rag Doll	Steven Tyler, Joe Perry, Jim Vallance, Holly Knight
3	5	23	Aerosmith	Big Ones	Walk On Water	Steven Tyler, Joe Perry, Jack Blades, Tommy Shaw
3	5	26	Aerosmith	Big Ones	What It Takes	Steven Tyler, Joe Perry, Desmond Child

Back to Introduction.

3.8. derivedTable Part A

Objective: Get a list by Artist that includes number of albums, number of tracks, total artist minutes and average minutes per track.

Approach: Use a set of derived queries to get the parts of the request and assemble them in the main query.

This request could be done using a single query. The derived tables are being used here, to show how they can be used to build parts of the results, that are assembled in a main query later.

This query gets the album count by artist.

SQL Example

```
/*
=== derivedTable - Gets the list of artists and album counts

Part A - Gets the list of artists with album count
this will become one of the derived tables in the final query

==== TOPICS

* Table alias
* Column Alias
* Join objects (Table, View, Derived Table) together
* Derived Table (Sub-Query)
* Count aggregate


==== TIPS

.Table Alias
* Table alias shorten the typing needed to complete the query,
1-3 character alias recommended.
* Good practice is to have table alias used everywhere in the query.

.Column Alias
* Column alias help to give meaningful, non conflicting names.
* Good practice, alias column names use lowerUpper naming
to identify them as alias (albumCount)
* Good practic, Use the same alias across all queries.

.Naming practice
* Good practice, System Generated Primary Index column name be
include with `ID` suffix.
* Good practice, Primary Key and Foreign Key join columns
should use the same name.
```

```
.Join requirements
* Coulmns for join must match in data type and size.

.Derived Tables (Sub-Queries)
* Start with a regular query until you get the results you need.
* Enclose the regular query in `(...)` and assign a table alias name.
* Join the derived table to another.

.Aggregate Functions
* Functions used to compress or limit the number of rows into a single row.
* `Count(?)` Counts the number of rows, duplicates included in the count.

*/


select
    AR.ArtistID
    ,count(AL.AlbumID) as albumCount  ① ②
from Album AL ③

INNER JOIN Artist AR  ③
on AR.ArtistID = AL.ArtistID ④

Group by AR.ArtistID  ⑤
```

① Column Alias

② Aggregate `Count` function

③ Table alias

④ Common column to join two tables

⑤ Any non-aggregate column must be included in `group by`

SQL Download link click here

Table 8. Results

ArtistId	albumCount
223	1
265	1
19	2
122	1
80	2
244	1
202	1
101	2
59	3
141	1

ArtistId	albumCount
242	1
263	1
221	1
120	1
78	1
99	2
200	1
17	1
57	1

Only 20 rows of the result being shown. The result set has 204 rows, one for each artist.

Back to Introduction.

4. Reference

4.1. Column Type

ColumnType reference table.

Table 9. ColumnType

ColumnType Abbrev	ColumnType Description
A1	ARRAY
AN	MULTI-DIMENSIONAL ARRAY
AT	TIME
BF	BYTE
BO	BLOB
BV	VARBYTE
CF	CHARACTER
CO	CLOB
CV	VARCHAR
D	DECIMAL
DA	DATE
DH	INTERVAL DAY TO HOUR
DM	INTERVAL DAY TO MINUTE
DS	INTERVAL DAY TO SECOND
DY	INTERVAL DAY
F	FLOAT
HM	INTERVAL HOUR TO MINUTE
HS	INTERVAL HOUR TO SECOND
HR	INTERVAL HOUR
I	INTEGER
I1	BYTEINT
I2	SMALLINT
I8	BIGINT
JN	JSON
MI	INTERVAL MINUTE
MO	INTERVAL MONTH
MS	INTERVAL MINUTE TO SECOND
N	NUMBER
PD	PERIOD(DATE)
PM	PERIOD(TIMESTAMP WITH TIME ZONE)

ColumnType Abbrev	ColumnType Description
PS	PERIOD(TIMESTAMP)
PT	PERIOD(TIME)
PZ	PERIOD(TIME WITH TIME ZONE)
SC	INTERVAL SECOND
SZ	TIMESTAMP WITH TIME ZONE
TS	TIMESTAMP
TZ	TIME WITH TIME ZONE
UT	UDT Type
XM	XML
YM	INTERVAL YEAR TO MONTH
YR	INTERVAL YEAR
++	TD_ANYTYP

5. Document History

Table 10. Document History

Date	Version	Author	Description
12/28/2018	V2.1f	JHRS	Added derivedQuery from archived
12/27/2018	V2.1e	JHRS	Added joinQuery set from archive
12/21/2018	V2.1d	JHRS	Added vsCode snippet for quick query insert added Reference section
12/20/2018	V2.1c	JHRS	Attempting standard document template
12/17/2018	V2.1b	JHRS	Initial version