Milestone 3

Implementation

Marcus Antonelli

A Quick Review

Summary

A comprehensive discographic relational database/catalog, adapted from the business partner's local storage and Excel spreadsheets. Tracking individual artists, sessions, attachments, and more.

Core Business Rules

1. Maintain discographic format
2. Support file attachments
3. Scalable, while maintaining fine data granularity

Some Minor Changes

1. Added a new entity/table for tracks, which is a join table for artist and session.
2. Changed approach for attachments, using only paths rather than blobs or physical files.
3. Some more redefinitions...

Updated ERD (Implemented in Access)

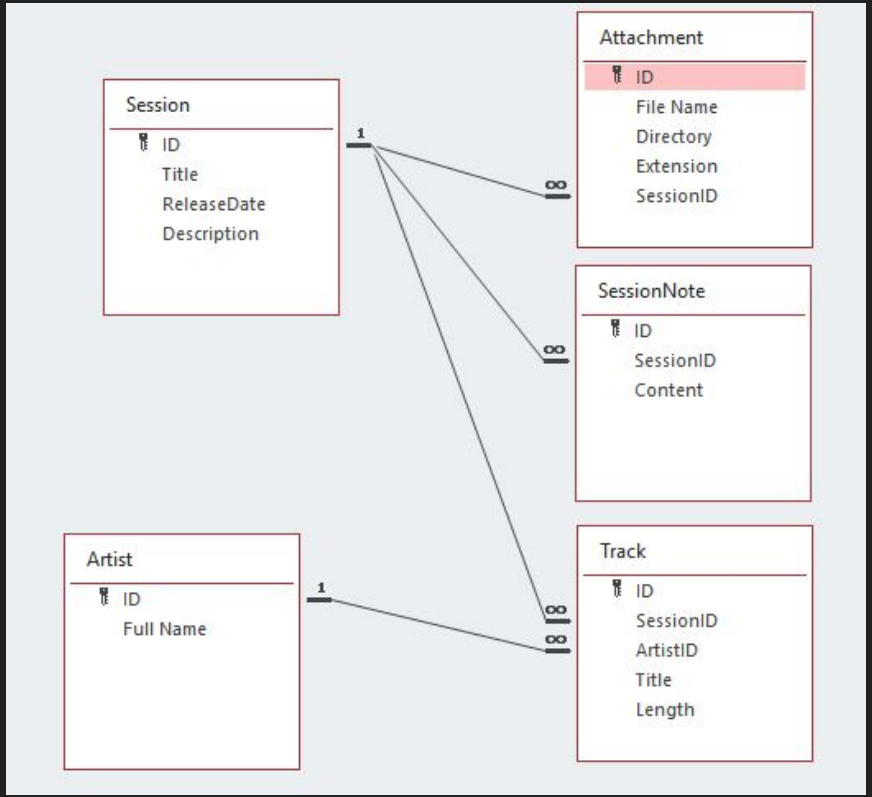

Table 1: Session

ID	Title	Release Date	Description
1	Live in Boston	1979-10-16	Firsthand recordings, excellent quality
2	Rumours	1977-02-04	Digitized vinyl - original
3	Elton John 1970 World Tour	1970-10-30	Sharp audio - Boston show
4	Live in Tivoli Gardens	1969-11-02	Duke live in Denmark
5	Live in Het Concertgebouw Amsterdam	1958-11-02	Duke Ellington and his orchestra live
6	Woodstock Festival 1969 Day 1	1969-08-15	Various performances
7	Woodstock Festival 1969 Day 2	1969-08-16	Various performances
8	Isle of Wight Festival 1970 Day 3	1970-08-28	Classic performances

Table 2: Artist

ID	Full Name
1	Elton John
2	David Bowie
3	Bob Dylan
4	The Who
5	Led Zeppelin
6	Duke Ellington
7	Fleetwood Mac
8	Elvis Presley
9	Meatloaf
10	Ray Charles
11	Miles Davis
12	Jimi Hendrix

Table 3: Track

ID	SessionID	ArtistID	Title	Length
1	1	1	Your Song	05:07
2	1	1	Sixty Years On	05:47
3	1	1	Daniel	04:36
4	1	1	Skyline Pigeon	03:50
5	1	1	Take Me To The Pilot	07:32
6	2	7	Dreams	04:17
7	2	7	Go Your Own Way	03:40
8	2	7	The Chain	04:30
9	4	6	C Jam Blues	03:53
10	4	6	Rockin' In The Rhythm	05:56
11	4	6	Four-Thirty p.m.	04:44
12	4	6	Take The 'A' Train	05:46
13	6	4	Sparks	05:12
14	6	4	Shakin' All Over	04:41

Table 4: Attachment

ID	SessionID	Directory	File Name	Extension
1	2	album_photos	fm_rumors_album_cover	.png
2	6	woodstock/journals	day1	.pdf
3	3	concert_photos	elton_john1	.jpeg
4	3	concert_photos	elton_john2	.jpeg
5	3	concert_photos	elton_john3	.jpeg
6	1	audio	live_in_boston1	.flac
7	1	audio	live_in_boston2	.flac
8	8	videos	iow_festival	.mov
9	5	videos	duke_ellington992	.mp4
10	4	audio	tivoli	.wav
11	7	woodstock/journals	day2	.pdf
12	7	woodstock/journals	day2_cont	.pdf
13	5	audio	duke_in_amsterdam	.flac

Table 5: Session Notes

ID	SessionID	Content
1	1	Lorem ipsum dolor sit amet, consectetur adipiscing elit...
2	1	sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
3	1	Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris ni:
4	2	Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dol
5	3	Excepteur sint occaecat cupidatat non proident
6	4	sunt in culpa qui officia deserunt mollit anim id est laborum.
7	2	Lorem ipsum dolor sit amet occaecat cupidatat non proident
8	3	Ut enim ad minim veniam, mollit anim id est laborum.
9	4	Incididunt ut labore et dolore magna aliqua.
10	4	Dignissim suspendisse in est ante in nibh. Risus ultricies tristique nulla
11	6	Erat nam at lectus urna duis convallis convallis tellus.
12	5	Tincidunt arcu non sodales neque sodales.

Preliminary Query Information

Multiple Joins

Access requires the use of parentheses to join more than two tables (uses Jet SQL flavour/format,) whereas plain SQL would not… e.g. FROM (table 1 JOIN table 2) JOIN table 3

Parameterized Queries

Access provides parameter queries, which is denoted in the WHERE clause as a prompt surrounded in brackets… e.g. WHERE foo LIKE [this is a prompt]

String Aggregation

Access does not provide an intuitive string aggregation function (like STRING_AGG) which would've been handy – creating an aggregated string of tracks/artists that belong to the same session, to limit the amount of returned records. It *is* possible to do this with VB code.

Query 1: Getting Tracks by Session

```
1 SELECT Session.Title AS Session, Artist.[Full Name] AS Artist,
  Track.Title AS Track, Track.Length
2 FROM (Track LEFT JOIN Artist ON Track.ArtistID = Artist.ID)
3 INNER JOIN Session
4 ON Session.ID = Track.SessionID
5 WHERE Track.SessionID IN (SELECT ID FROM Session WHERE
  Session.Title LIKE [Session title:] & "*");
```

Query 1 Output

Searching with parameter “Rumours”:

Session	Artist	Track	Length
Rumours	Fleetwood Mac	Dreams	04:17
Rumours	Fleetwood Mac	Go Your Own Way	03:40
Rumours	Fleetwood Mac	The Chain	04:30

Searching with parameter “Live in”:

Session	Artist	Track	Length
Live in Boston	Elton John	Daniel	04:36
Live in Boston	Elton John	Sixty Years On	05:47
Live in Boston	Elton John	Skyline Pigeon	03:50
Live in Boston	Elton John	Take Me To The Pilot	07:32
Live in Boston	Elton John	Your Song	05:07
Live in Tivoli Gardens	Duke Ellington	C Jam Blues	03:53
Live in Tivoli Gardens	Duke Ellington	Four-Thirty p.m.	04:44
Live in Tivoli Gardens	Duke Ellington	Rockin' In The Rhythm	05:56
Live in Tivoli Gardens	Duke Ellington	Take The 'A' Train	05:46

Query 2: Getting Attachments by Session

```
1 SELECT Session.Title AS Session, ("/" & Attachment.Directory & "/" &
  Attachment.[File Name] & Attachment.Extension) AS Path
2 FROM Session
3 LEFT JOIN Attachment
4 ON Session.ID = Attachment.SessionID
5 WHERE Session.Title LIKE [Session title:] & “*”;
```

Query 2 Output

Searching with parameter “Rumours”:

Session	Path
Rumours	/album_photos/fm_rumors_album_cover.png

Searching with parameter “Live in”:

Session	Path
Live in Boston	/audio/live_in_boston1.flac
Live in Boston	/audio/live_in_boston2.flac
Live in Tivoli Gardens	/audio/tivoli.wav
Live in Het Concertgebouw Amsterdam	/videos/duke_ellington992.mp4
Live in Het Concertgebouw Amsterdam	/audio/duke_in_amsterdam.flac

Query 3: Getting Notes by Session

```
1 SELECT Session.Title AS Session, SessionNote.Content AS [Note]
2 FROM Session
3 INNER JOIN SessionNote
4 ON SessionNote.SessionID = Session.ID
5 WHERE Session.Title LIKE [Session title:] & “*”
6 ORDER BY Session.ID;
```

I use an order by statement here, which, in hindsight, would’ve been good to include for some of the other queries.

Query 3 Output

Searching with parameter “Woodstock”:

Session	Note
Woodstock Festival 1969 Day 1	Erat nam at lectus urna duis convallis convallis tellus.

Searching with parameter “Live in”:

Session	Note
Live in Boston	Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris ni:
Live in Boston	sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.
Live in Boston	Lorem ipsum dolor sit amet, consectetur adipiscing elit...
Live in Tivoli Gardens	Dignissim suspendisse in est ante in nibh. Risus ultricies tristique nulla
Live in Tivoli Gardens	Incididunt ut labore et dolore magna aliqua.
Live in Tivoli Gardens	sunt in culpa qui officia deserunt mollit anim id est laborum.
Live in Het Concertgebouw Amsterdam	Tincidunt arcu non sodales neque sodales.

Query 4: Getting Sessions by Year

```
1  SELECT Session.Title AS Session, Format(Session.[Release Date],
   "yyyy-mm-dd") & " - " & Session.Description AS [Date and Description],
   COUNT(Track.ID) AS [Number of Tracks]
2  FROM Session
3  LEFT JOIN Track
4  ON Session.ID = Track.SessionID
5  WHERE Year(Session.[Release Date]) = [Release year:]
6  GROUP BY Session.Title, Session.ID, Session.[Release Date],
   Session.Description;
```

The group by statement is necessary to distinguish sessions from the same year.

Query 4 Output

Searching with parameter “1979”:

Session	Date and Description	Number of Tracks
Live in Boston	1979-10-16 - Firsthand recordings, excellent quality	5

Searching with parameter “1969”:

Session	Date and Description	Number of Tracks
Live in Tivoli Gardens	1969-11-02 - Duke live in Denmark	4
Woodstock Festival 1969 Day 1	1969-08-15 - Various performances	2
Woodstock Festival 1969 Day 2	1969-08-16 - Various performances	0

Last Minute Thoughts

1. Could/Should be Added:
    - Create, update, and delete queries for every table

2. “SessionNote” Could be Renamed “Note”.

3. Parameter Queries Can be Adapted in a Scripting Language Like PHP
    - This is more or less for the long-term goal of a web interface/frontend

4. Keeping scalability in mind, some of the queries can be reworked to be more efficient with a larger dataset. (How? We’re not sure, yet.)

Thank You!

Marcus Antonelli